ceph

an intro to ceph for hpc

sage weil – inktank
lug – 2013.04.16

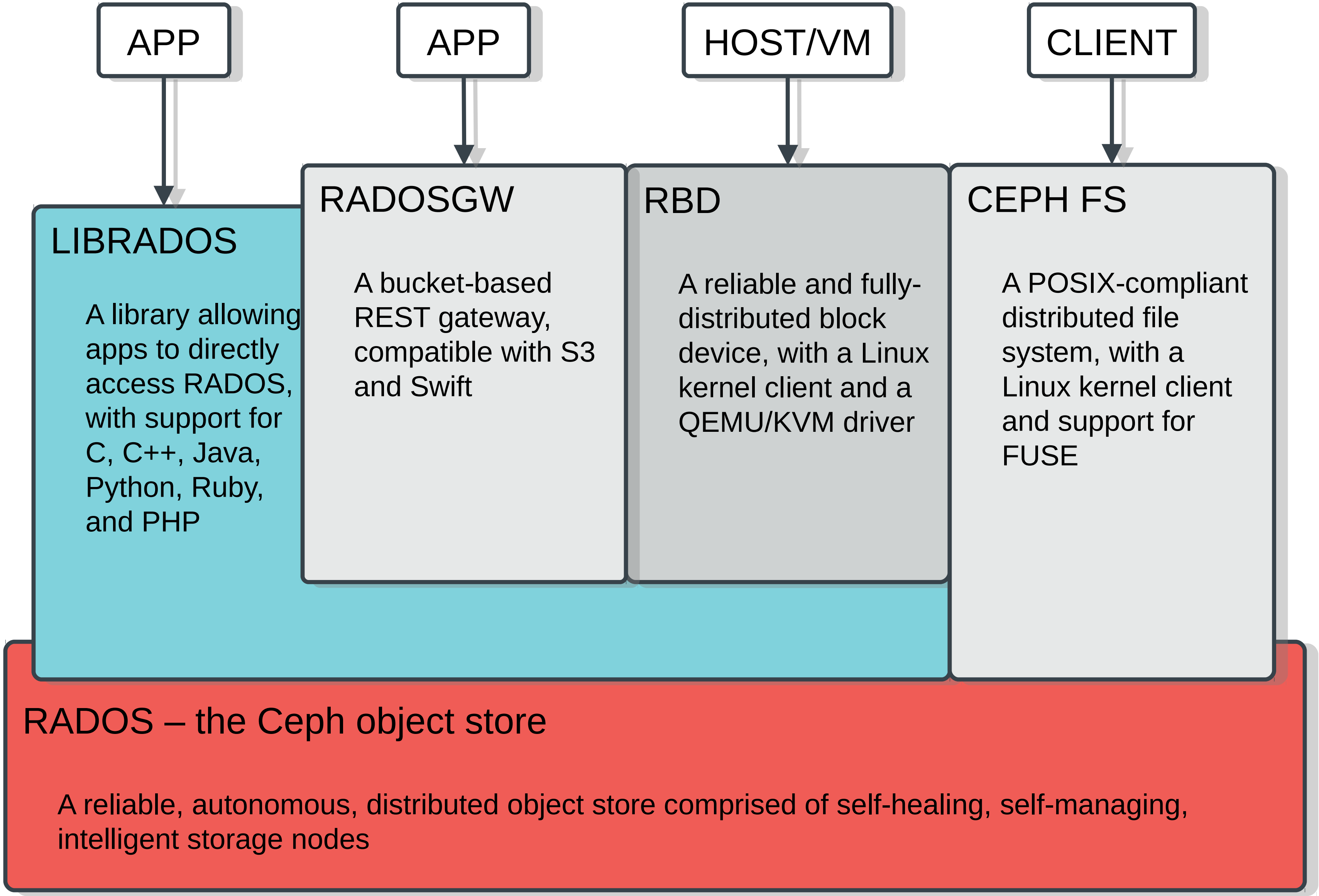
inktank

what is ceph?

- distributed storage system
  - reliable system built with unreliable components
  - fault tolerant, no SPoF
- commodity hardware
  - expensive arrays, controllers, specialized networks not required
- large scale (10s to 10,000s of nodes)
  - heterogenous hardware (no fork-lift upgrades)
  - incremental expansion (or contraction)
- dynamic cluster

what is ceph?

- unified storage platform
  - scalable object + compute storage platform
  - RESTful object storage (e.g., S3, Swift)
  - block storage
  - distributed file system
- open source
  - LGPL server-side
  - client support in mainline Linux kernel



APP

APP

HOST/VM

CLIENT

LIBRADOS

A library allowing apps to directly access RADOS, with support for C, C++, Java, Python, Ruby, and PHP

RADOSGW

A bucket-based REST gateway, compatible with S3 and Swift

RBD

A reliable and fully-distributed block device, with a Linux kernel client and a QEMU/KVM driver

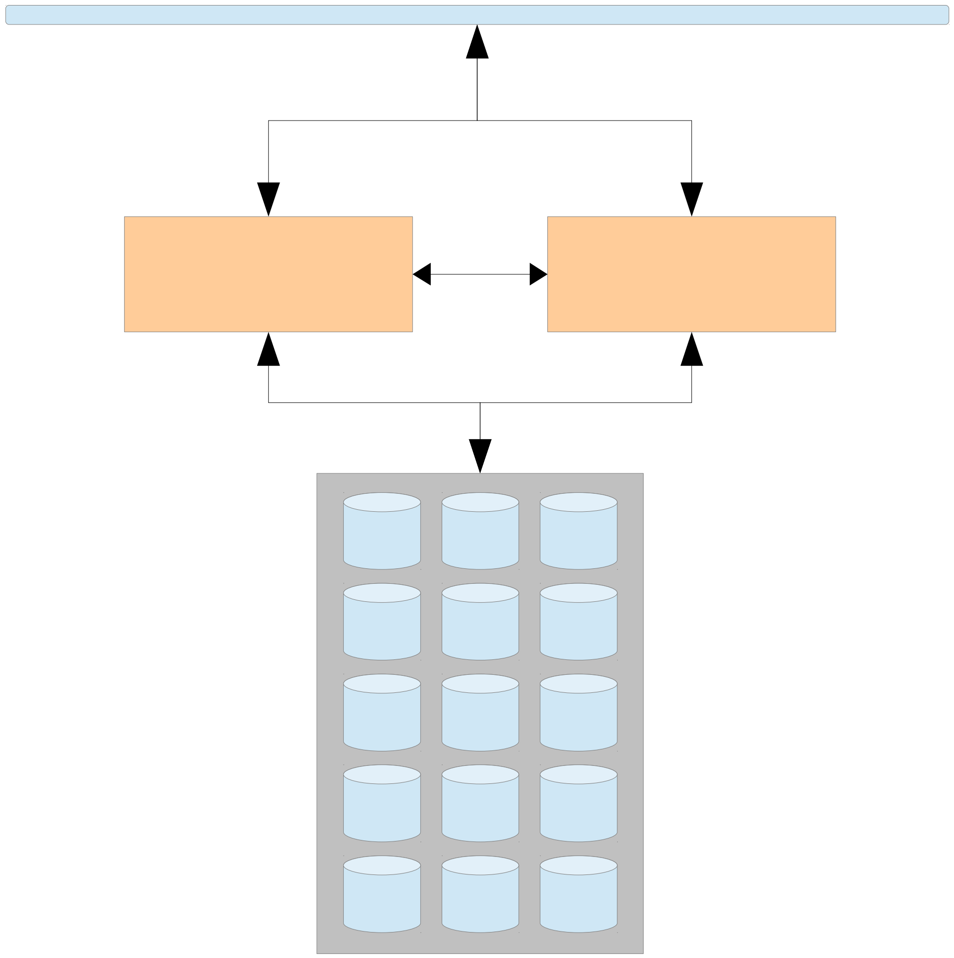
CEPH FS

A POSIX-compliant distributed file system, with a Linux kernel client and support for FUSE

RADOS – the Ceph object store

A reliable, autonomous, distributed object store comprised of self-healing, self-managing, intelligent storage nodes

conventional HA



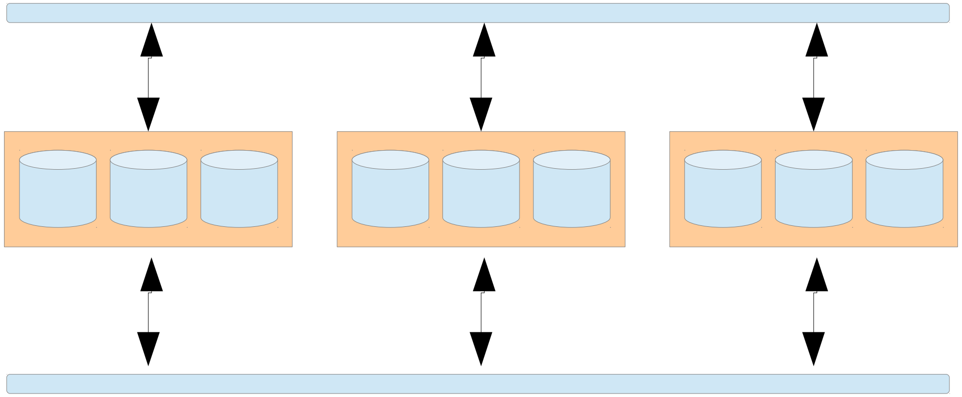
access network

redundant head

reliable disk array

“clients stripe data across reliable things”

distributed model

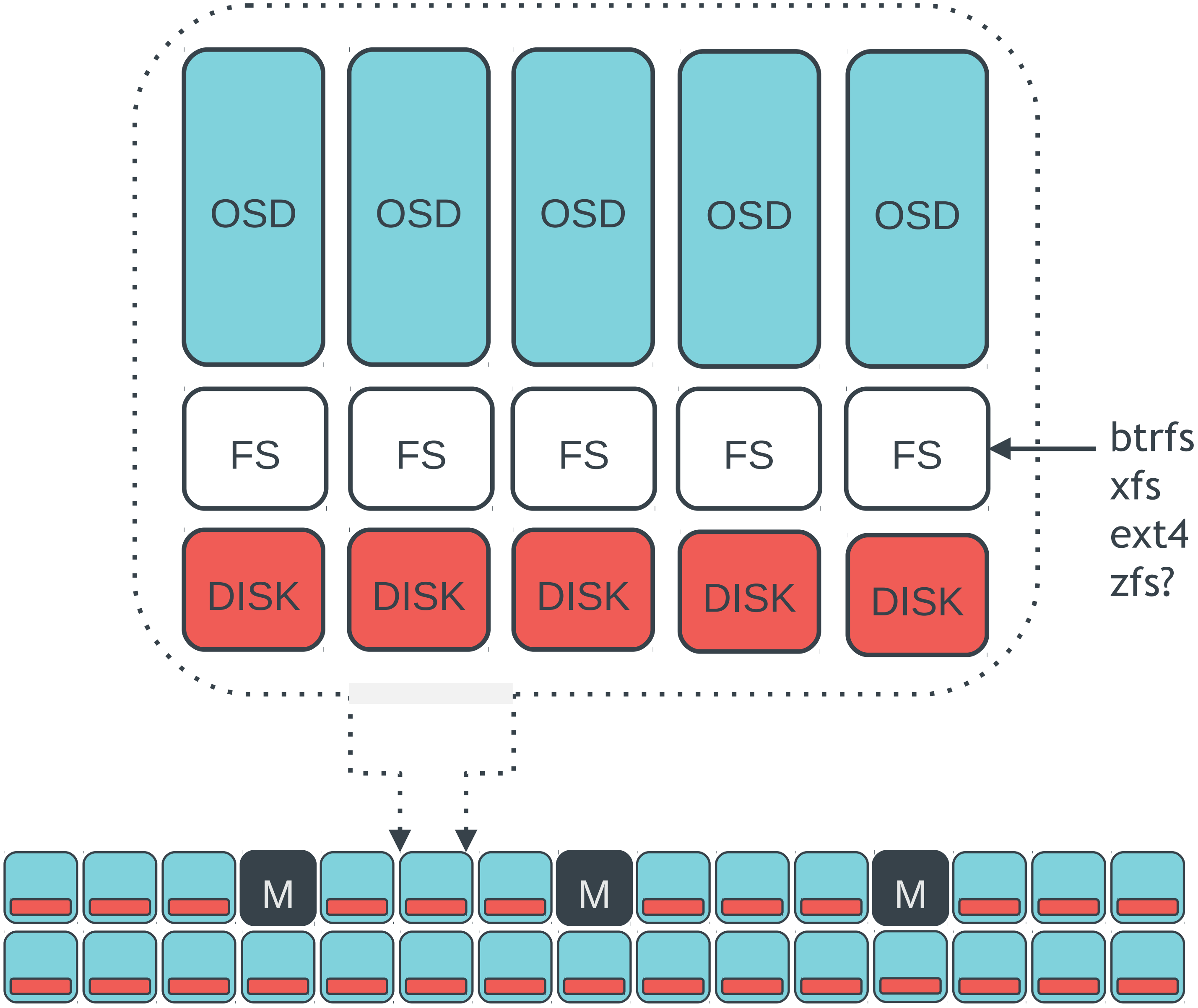


access network

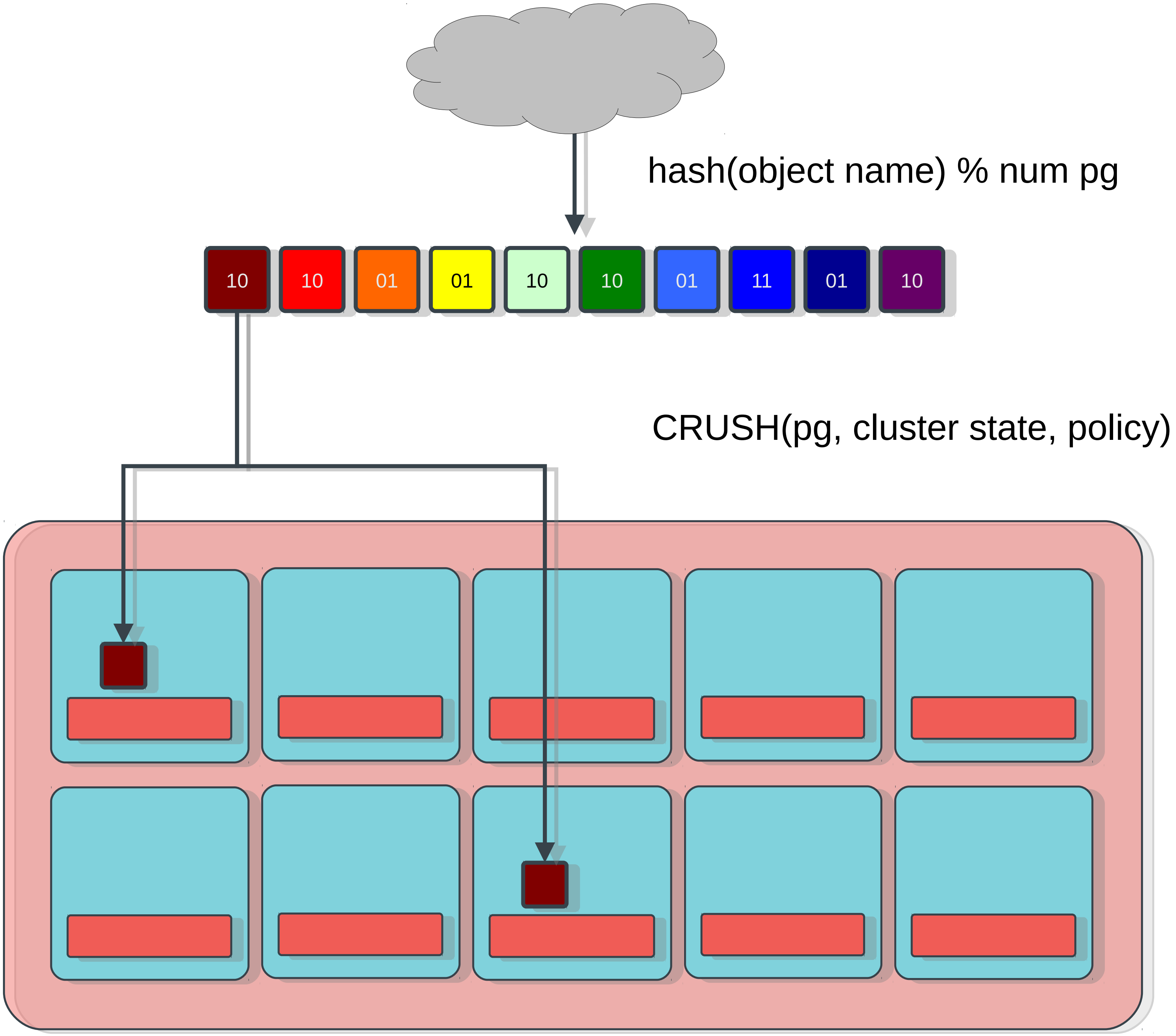
unreliable targets

backside network
(optional)

“client stripe across unreliable things”
“servers coordinate replication, recovery”

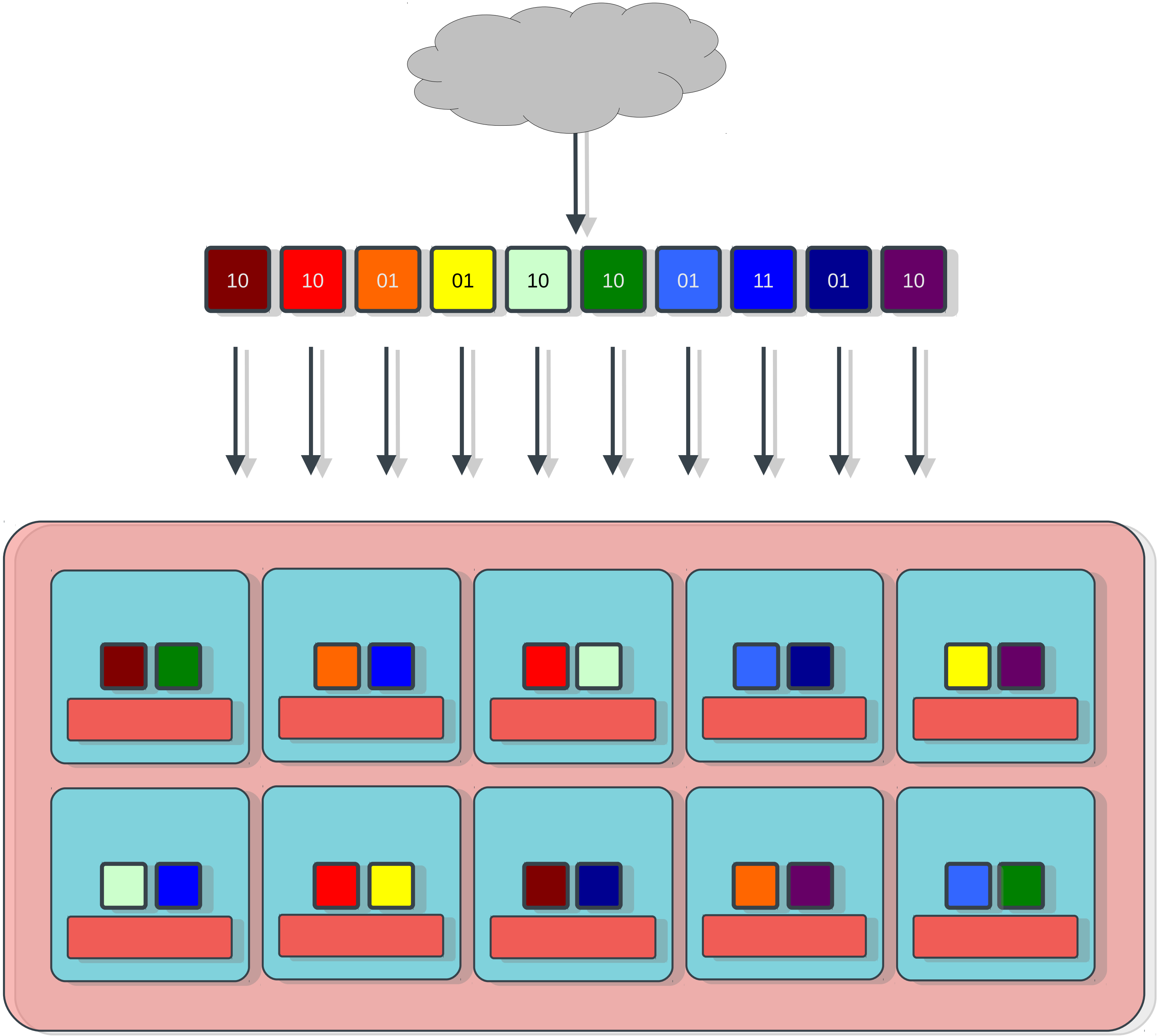


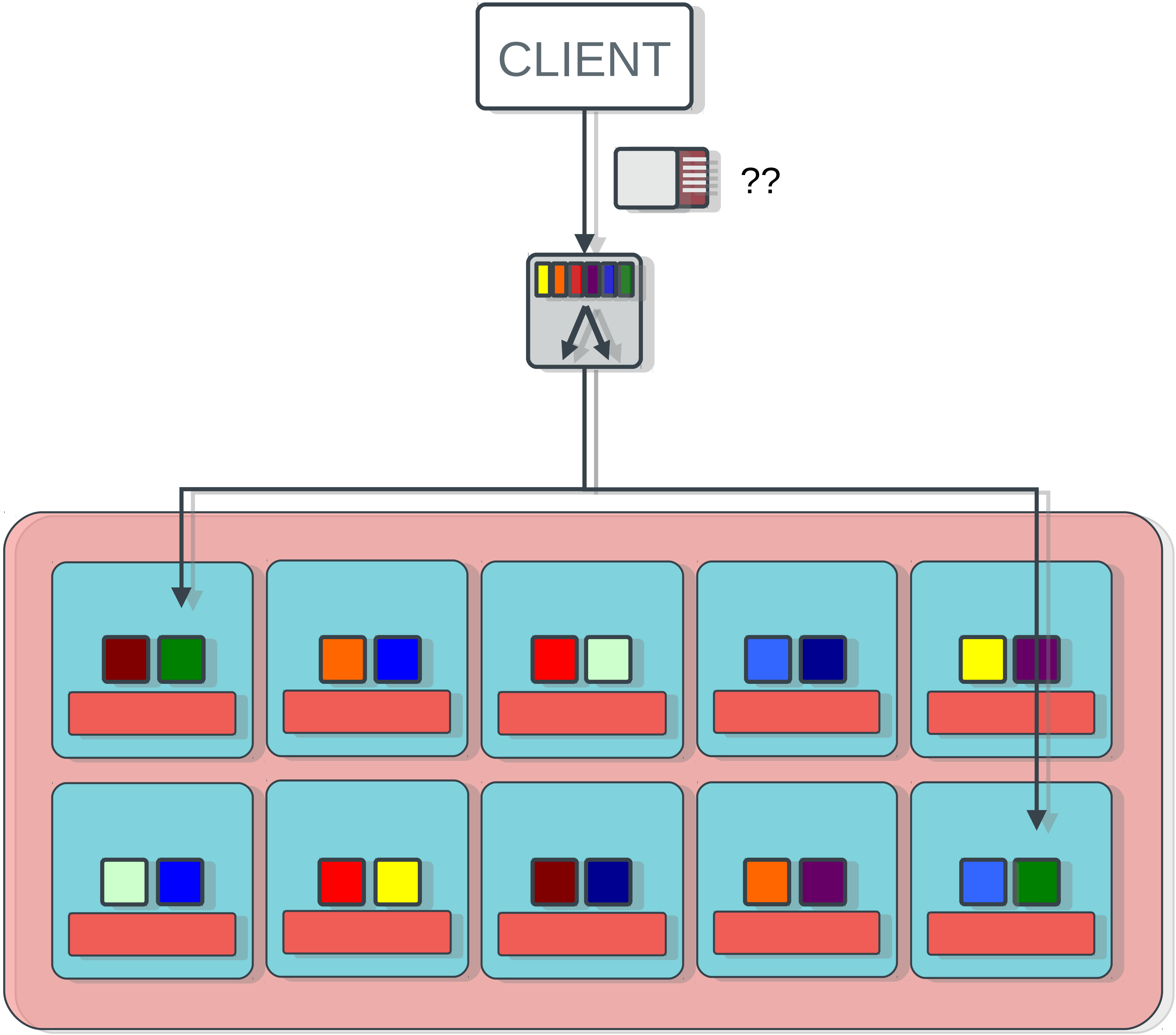
btrfs
xfs
ext4
zfs?

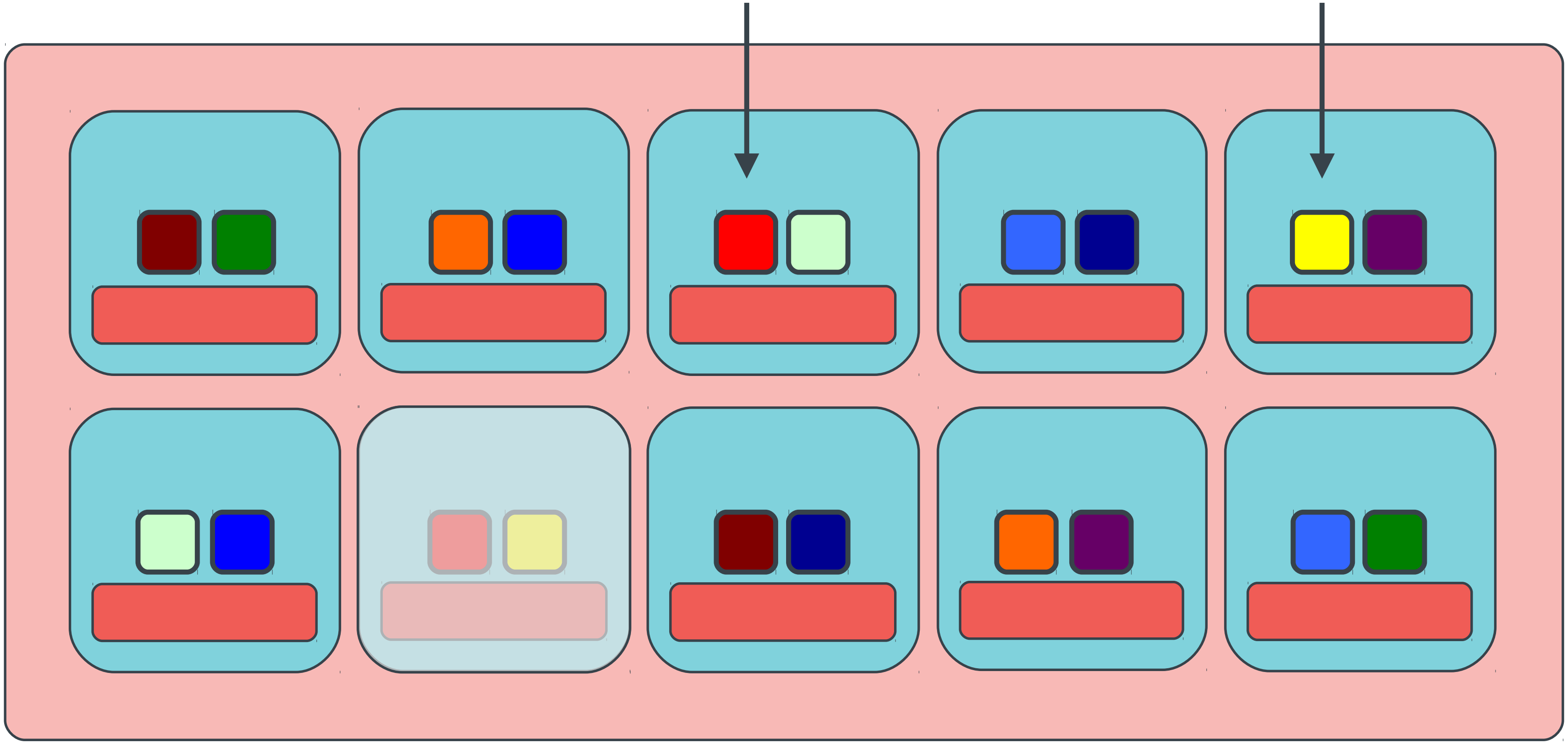


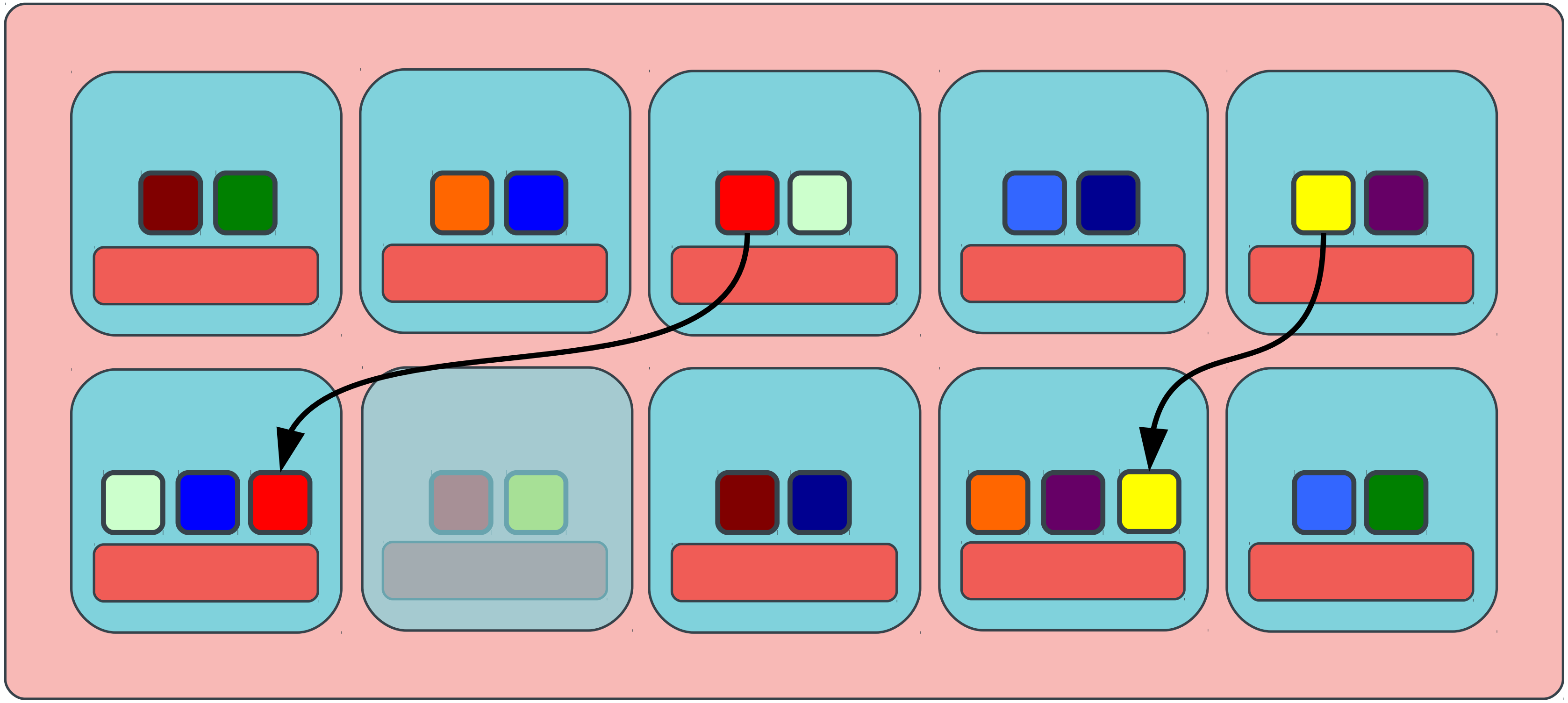
hash(object name) % num pg

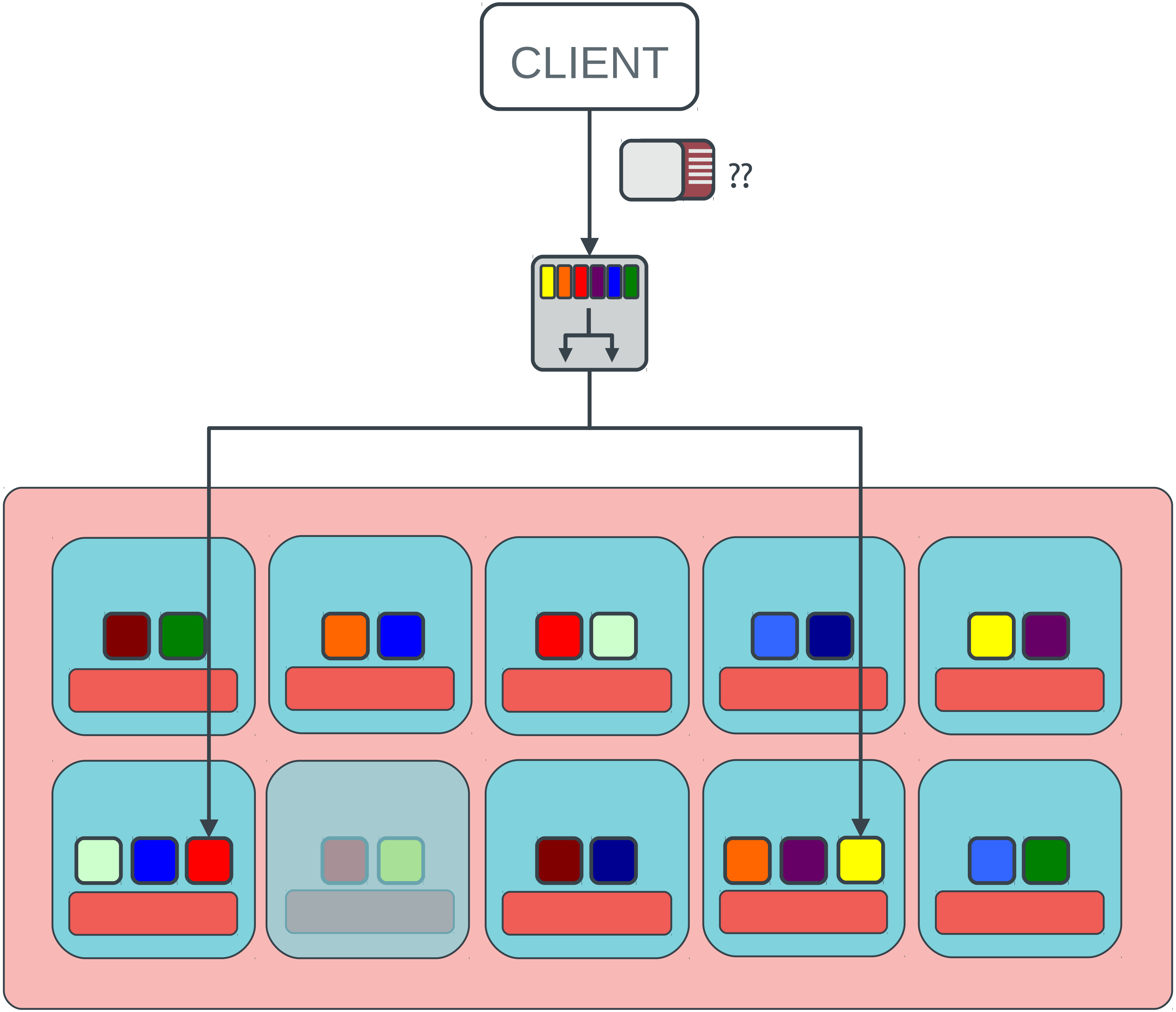
CRUSH(pg, cluster state, policy)


10
10
01
01
10
10
01
11
01
10


CLIENT
??






CLIENT
??

L

librados

- direct access to RADOS from applications
- C, C++, Python, PHP, Java, Erlang
- direct access to storage nodes
- no HTTP overhead

rich librados API

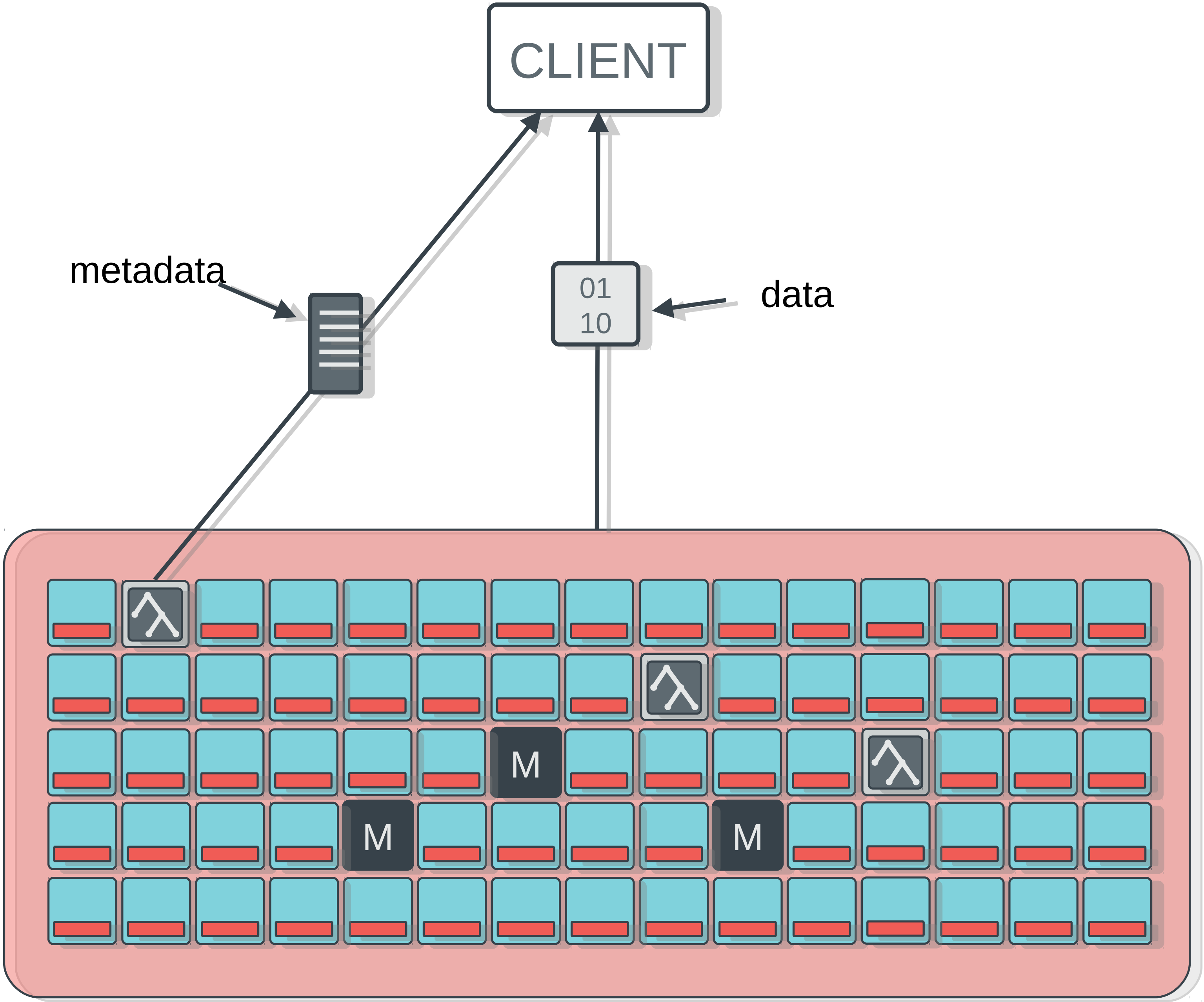
- efficient key/value storage inside an object
- atomic single-object transactions
  - update data, attr, keys together
  - atomic compare-and-swap
- object-granularity snapshot infrastructure
- embed code in ceph-osd daemon via plugin API
  - arbitrary atomic object mutations, processing
- inter-client communication via object

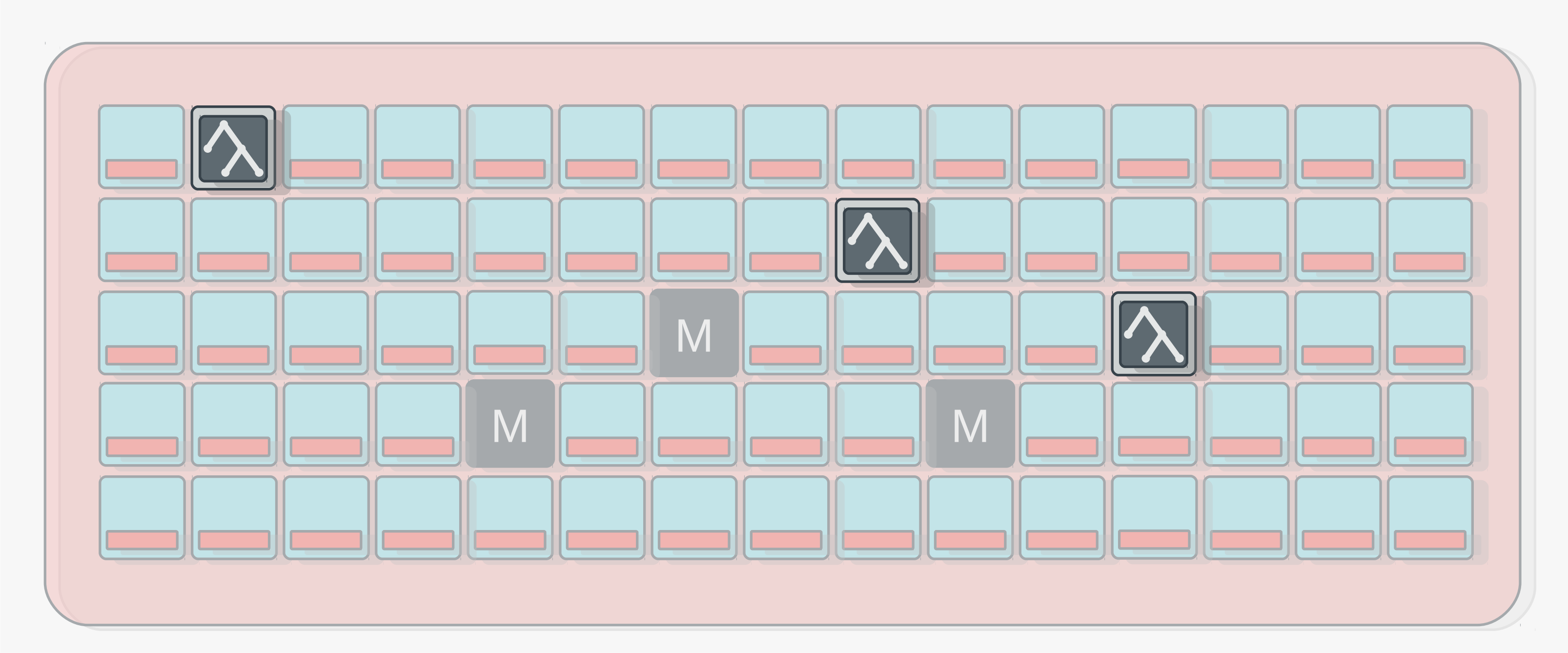
die, POSIX, die

- successful exascale architectures will replace or transcend POSIX
  - hierarchical model does not distribute
- line between compute and storage will blur
  - some processes is data-local, some is not
- fault tolerance will be first-class property of architecture
  - for both computation and storage

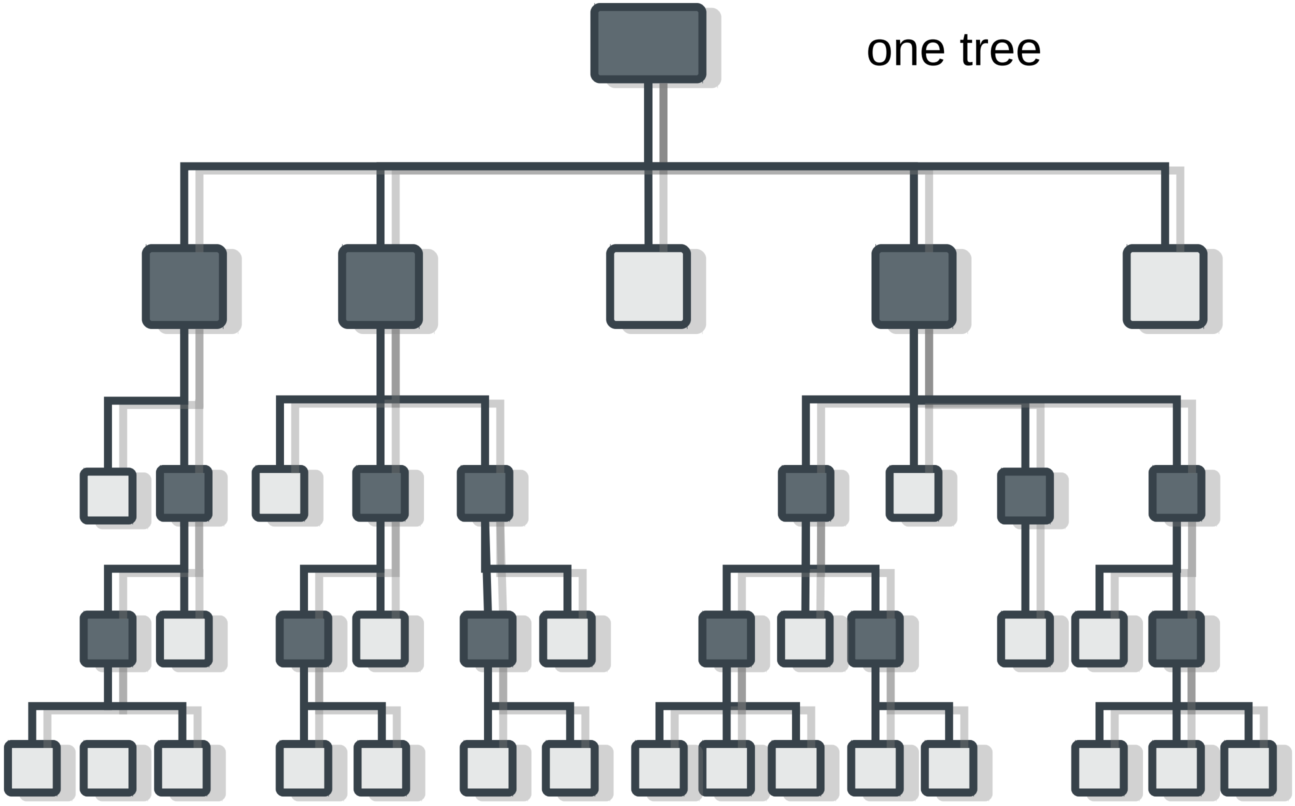
POSIX – I'm not dead yet!

- CephFS builds POSIX namespace on top of RADOS
  - metadata managed by ceph-mds daemons
  - stored in objects
- strong consistency, stateful client protocol
  - heavy prefetching, embedded inodes
- architected for HPC workloads
  - distribute namespace across cluster of MDSs
  - mitigate bursty workloads
  - adapt distribution as workloads shift over time

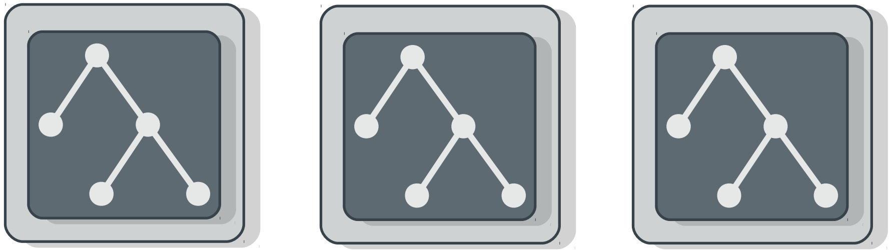

CLIENT
metadata
01
10
data
M
M
M


M
M
M

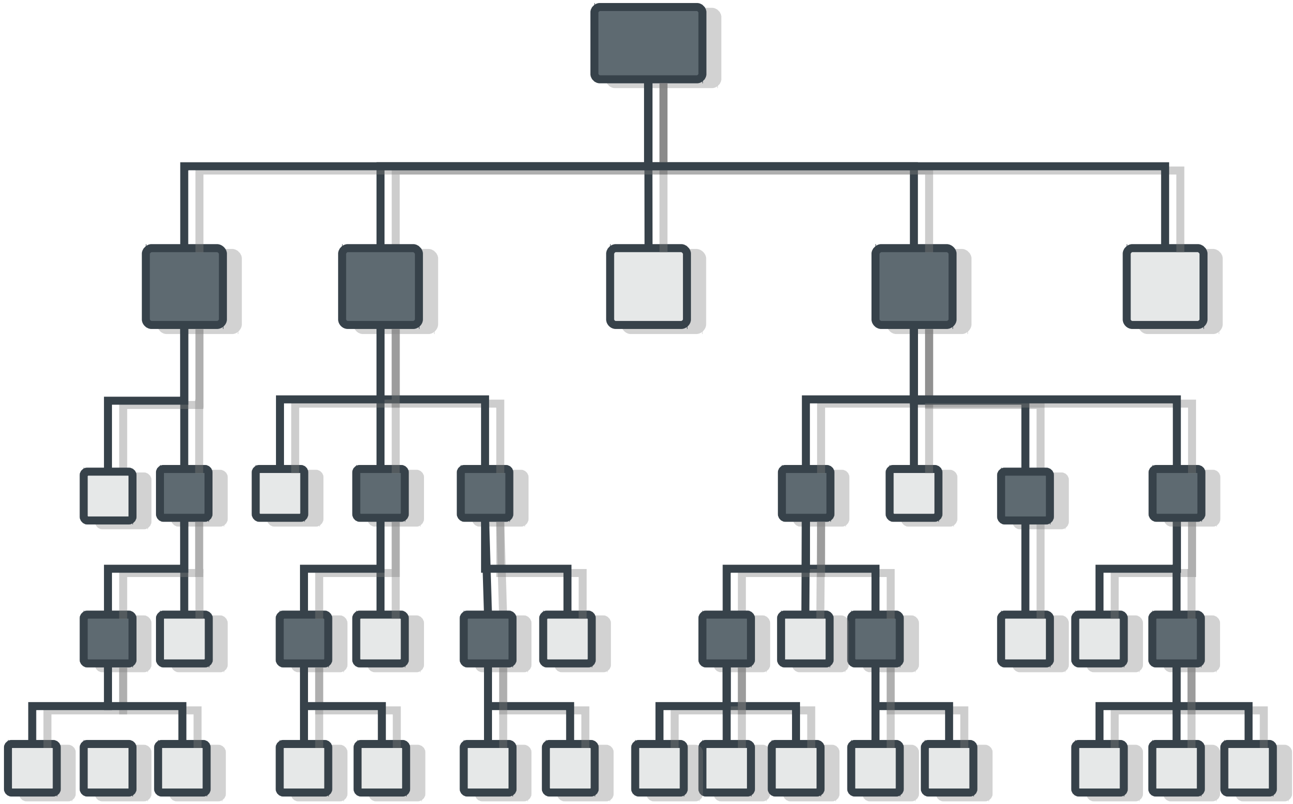
one tree

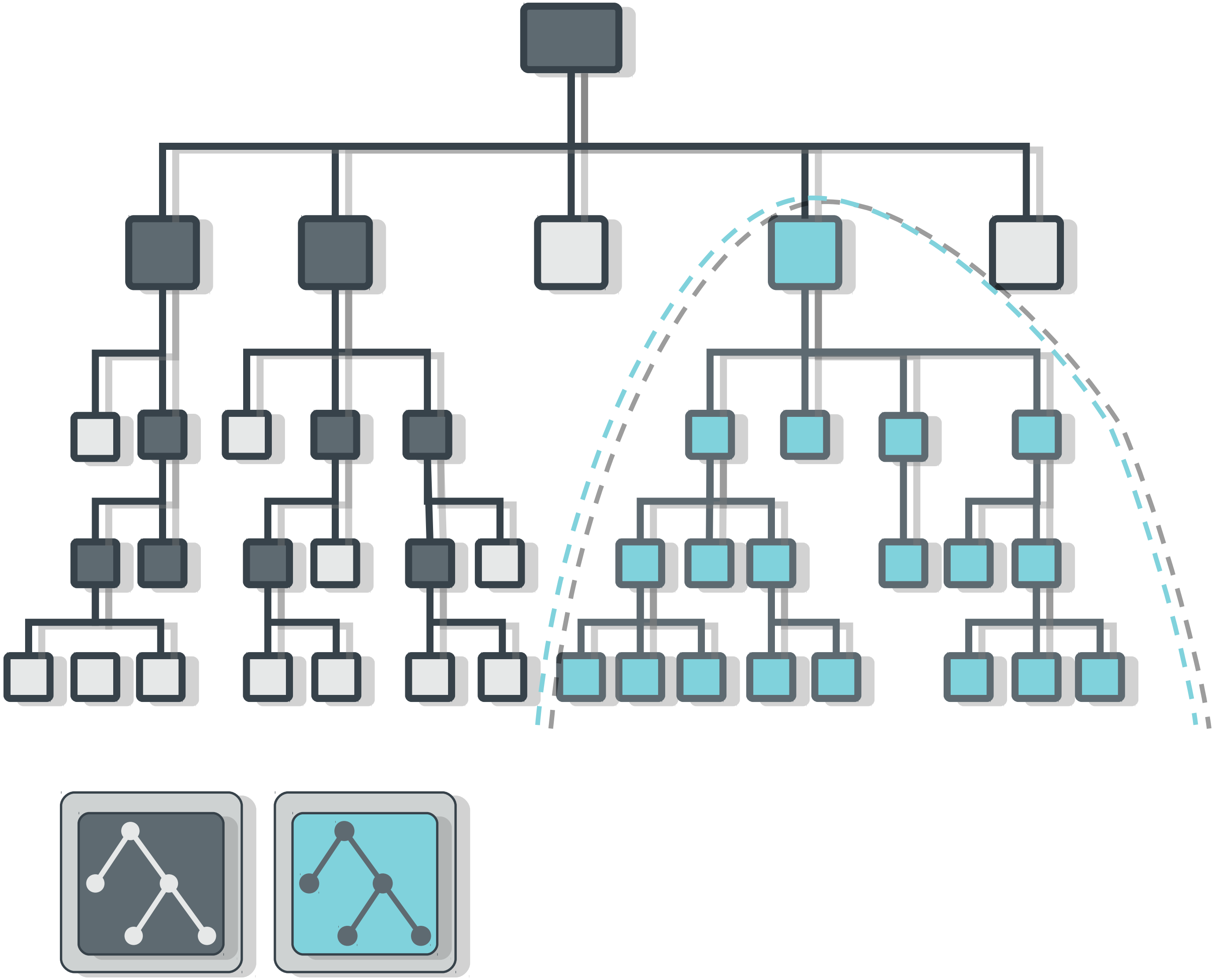


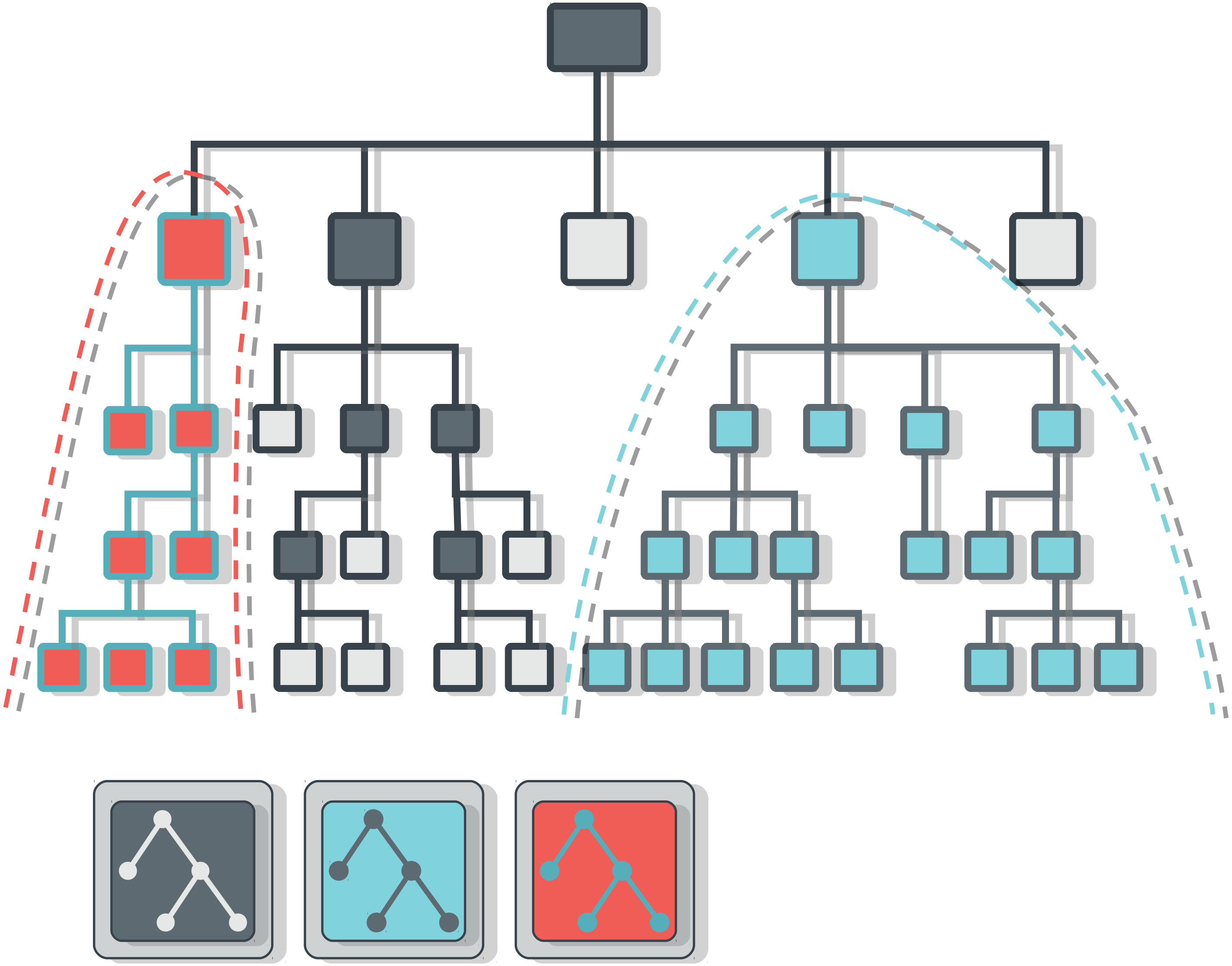
three metadata servers

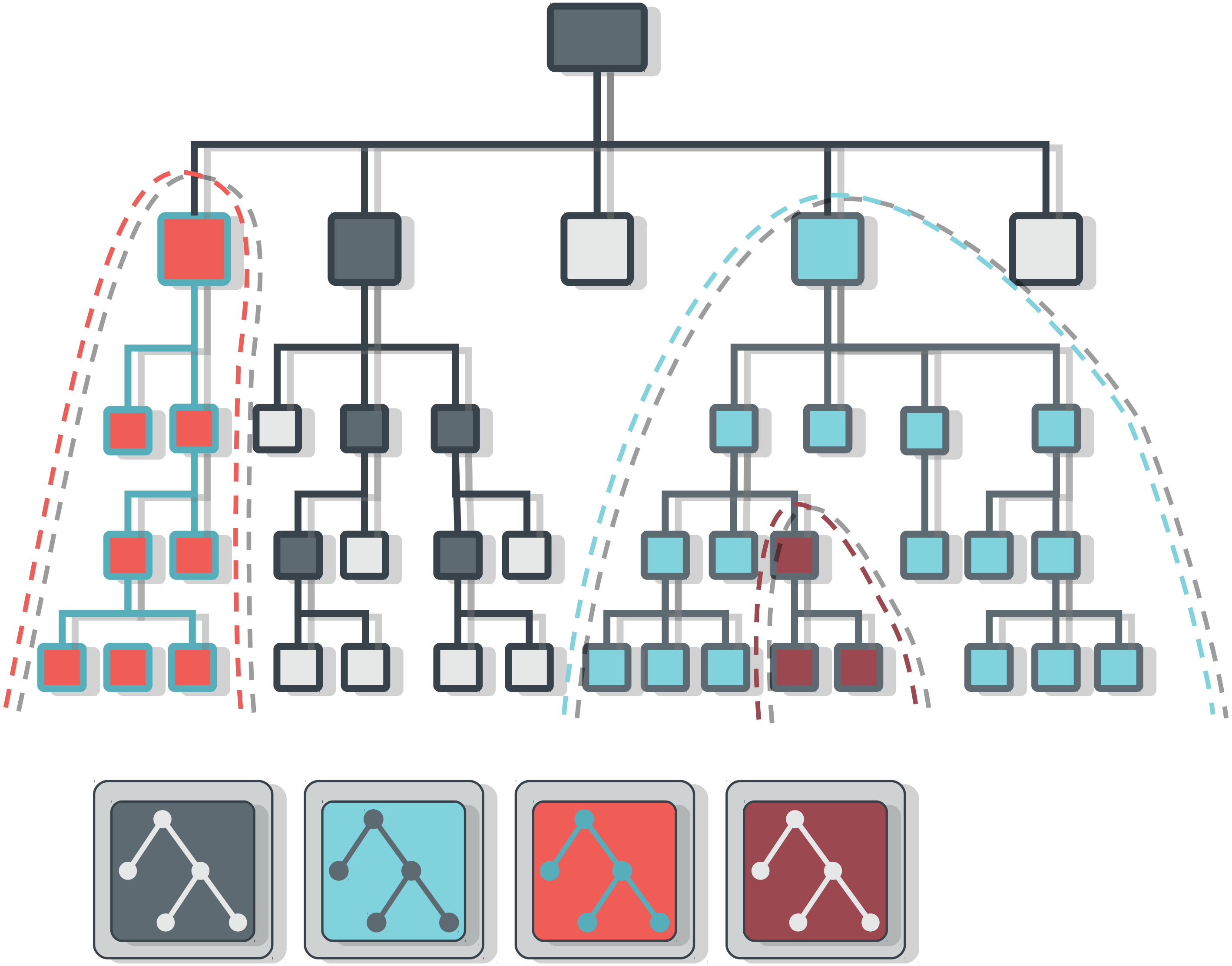


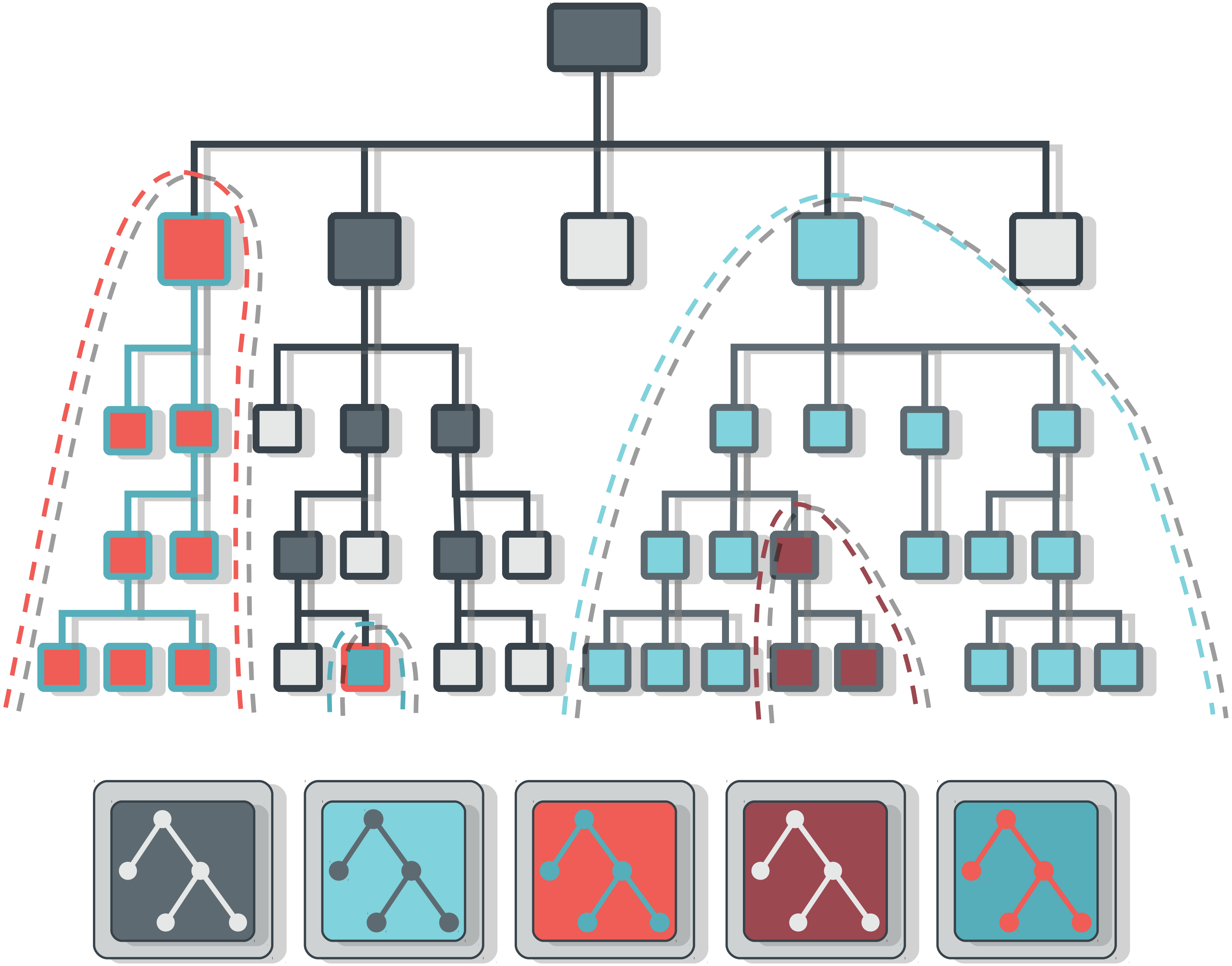
??











DYNAMIC SUBTREE PARTITIONING

recursive accounting

- ceph-mds tracks recursive directory stats
  - file sizes
  - file and directory counts
  - modification time
- efficient

```
$ ls -alSh | head
total 0
drwxr-xr-x 1 root          root          9.7T    2011-02-04 15:51 .
drwxr-xr-x 1 root          root          9.7T    2010-12-16 15:06 ..
drwxr-xr-x 1 pomceph       pg4194980     9.6T    2011-02-24 08:25 pomceph
drwxr-xr-x 1 mcg_test1     pg2419992     23G     2011-02-02 08:57 mcg_test1
drwx--x--- 1 luko          adm           19G     2011-01-21 12:17 luko
drwx--x--- 1 eest          adm           14G     2011-02-04 16:29 eest
drwxr-xr-x 1 mcg_test2     pg2419992     3.0G    2011-02-02 09:34 mcg_test2
drwx--x--- 1 fuzyceph      adm           1.5G    2011-01-18 10:46 fuzyceph
drwxr-xr-x 1 dallasceph    pg275         596M    2011-01-14 10:06 dallasceph
```

snapshots

- snapshot arbitrary subdirectories
- simple interface
  - hidden '.snap' directory
  - no special tools

```
$ mkdir foo/.snap/one    # create snapshot
$ ls foo/.snap
one
$ ls foo/bar/.snap
_one_1099511627776       # parent's snap name is mangled
$ rm foo/myfile
$ ls -F foo
bar/
$ ls -F foo/.snap/one
myfile  bar/
$ rmdir foo/.snap/one    # remove snapshot
```

running ceph in lustre environments

- it's not ideal, but it's possible
- ceph is not optimized for high end hardware
  - redundancy from expensive arrays unnecessary
  - ceph replicates *across* unreliable servers
  - more disks, cheaper hardware
- ceph utilizes flash/NVRAM directly
  - write journal/buffer
  - usually present but buried inside disk array

ORNL experiment

- tune ceph on lustre OSTs backed by DDN
- started at 100MB/sec, ended at 5.5GB/sec
  - net >11GB/sec w/ journaling
  - 12GB/sec max, so reached >90%
- double-writes
  - journal to one LUN, write to another
- IPoIB
  - no native IB support...yet

slow march to respectable

- range of issues
  - IB, IPoIB configuration
  - misc DDN/SCSI tweaks
  - data on SAS, journals on SATA
  - reorganization of DDN RAID LUNs
  - tune OSD/node ratios
  - disabled cache mirroring on DDN controllers
  - disabled TCP autotuning
  - tune readahead

how can you help?

- try ceph and tell us what you think
  - http://ceph.com/resources/downloads
- http://ceph.com/resources/mailing-list-irc/
  - ask if you need help
- ask your organization to start dedicating resources to the project http://github.com/ceph
- find a bug (http://tracker.ceph.com) and fix it
- participate in our ceph developer summit
  - http://ceph.com/events/ceph-developer-summit

questions?

-

thanks

sage weil

sage@inktank.com

@liewegas

http://github.com/ceph

http://ceph.com/