a unified distributed storage system

sage weil

ceph day – november 2, 2012

outline

- why you should care
- what is it, what it's for
- how it works
  - architecture
- how you can use it
  - librados
  - radosgw
  - RBD, the ceph block device
  - distributed file system
- roadmap
- why we do this, who we are

ceph

why should you care about another storage system?

requirements

- diverse storage needs
  - object storage
  - block devices (for VMs) with snapshots, cloning
  - shared file system with POSIX, coherent caches
  - structured data... files, block devices, or objects?
- scale
  - terabytes, petabytes, exabytes
  - heterogeneous hardware
  - reliability and fault tolerance

ceph

time

- ease of administration
- no manual data migration, load balancing
- painless scaling
  - expansion **and** contraction
  - seamless migration

ceph

cost

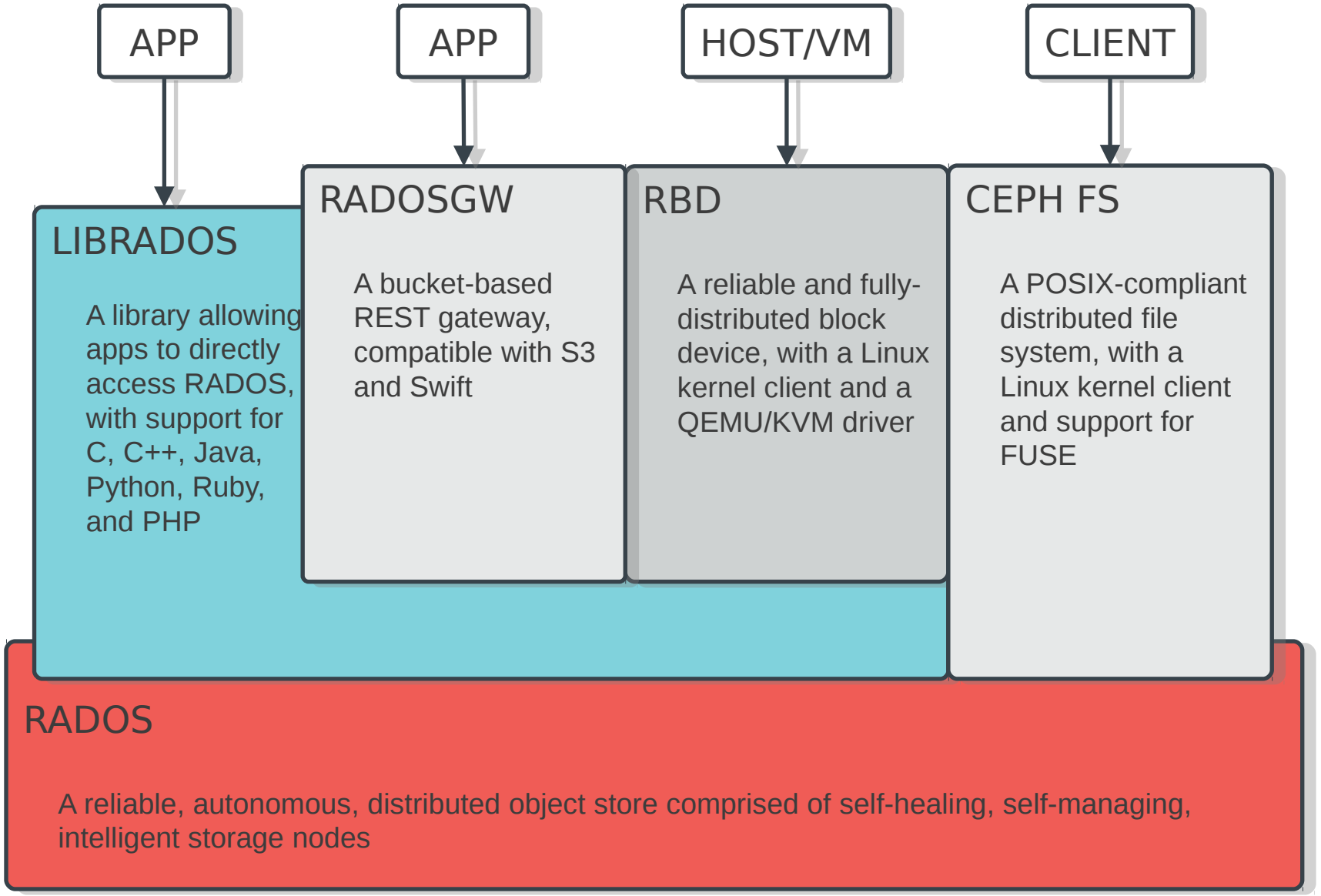
- linear function of size or performance
- incremental expansion
  - no fork-lift upgrades
- no vendor lock-in
  - choice of hardware
  - choice of software
- open

what is ceph?

unified storage system

- objects
  - native
  - RESTful
- block
  - thin provisioning, snapshots, cloning
- file
  - strong consistency, snapshots

ceph


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

ceph

distributed storage system

- data center scale
  - 10s to 10,000s of machines
  - terabytes to exabytes
- fault tolerant
  - no single point of failure
  - commodity hardware
- self-managing, self-healing

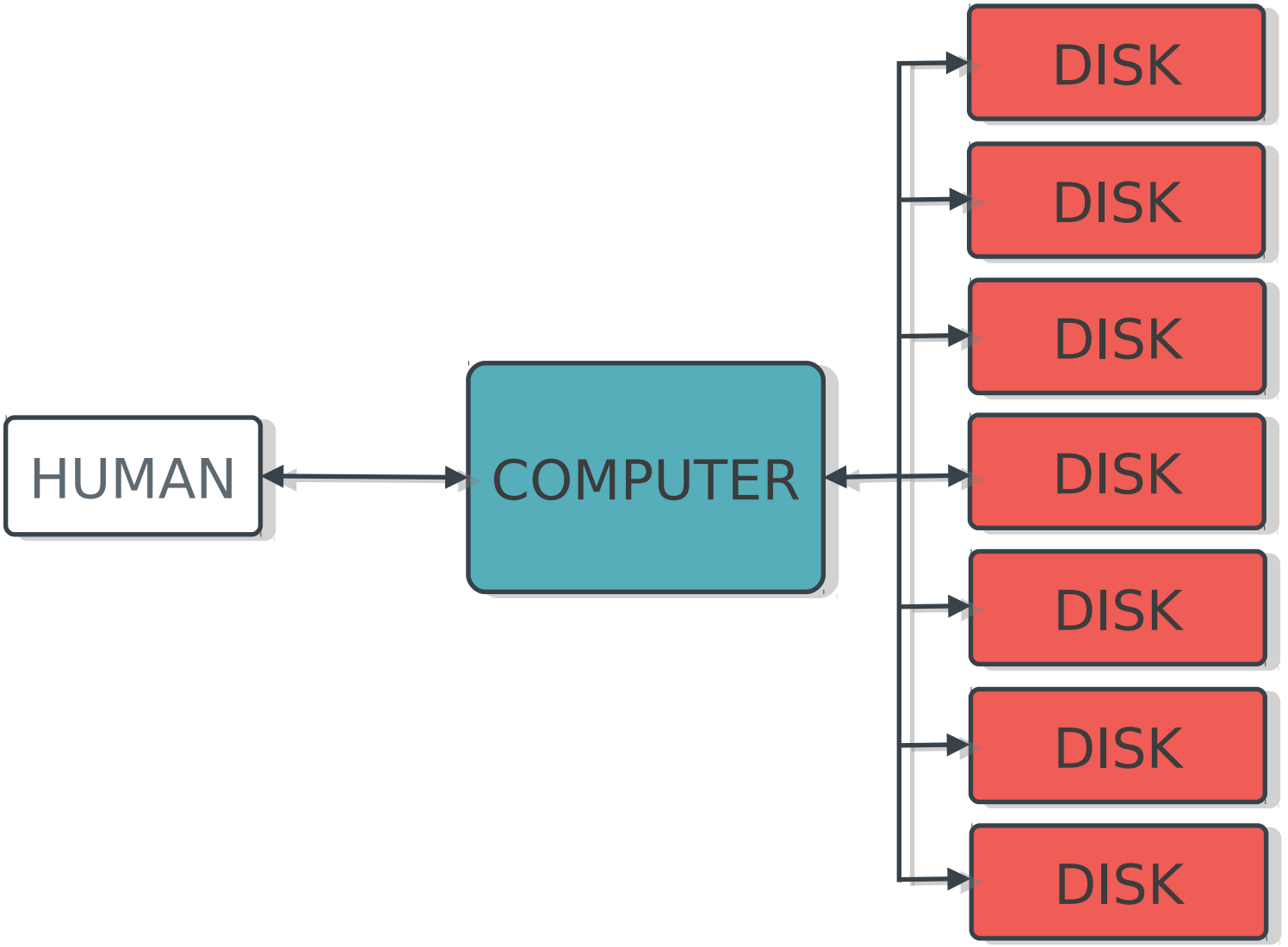
ceph

ceph object model

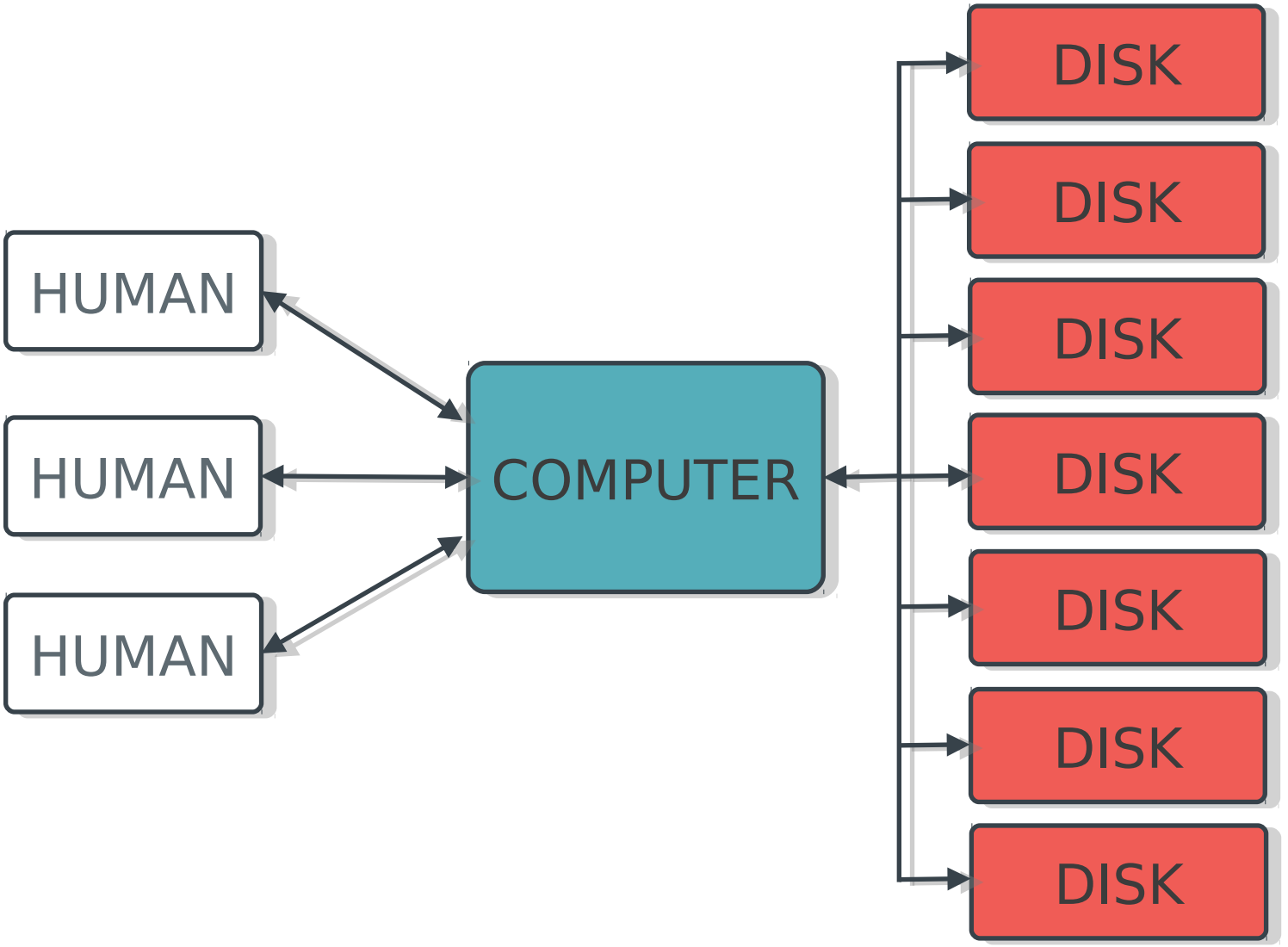
- pools
  - 1s to 100s
  - independent namespaces or object collections
  - replication level, placement policy
- objects
  - bazillions
  - blob of data (bytes to gigabytes)
  - attributes (e.g., “version=12”; bytes to kilobytes)
  - key/value bundle (bytes to gigabytes)

why start with objects?

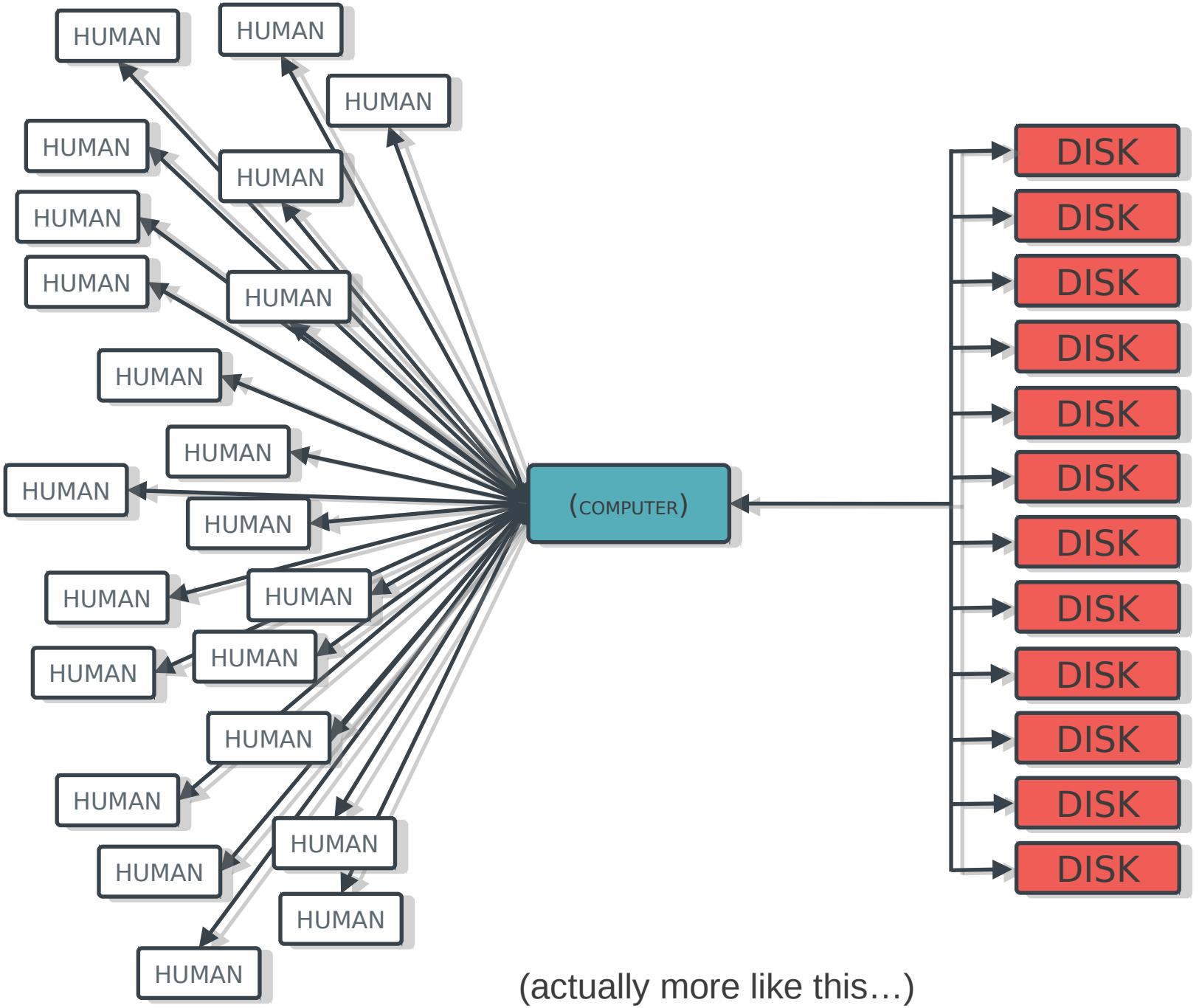
- more useful than (disk) blocks
  - names in a single flat namespace
  - variable size
  - simple API with rich semantics
- more scalable than files
  - no hard-to-distribute hierarchy
  - update semantics do not span objects
  - workload is trivially parallel



ceph

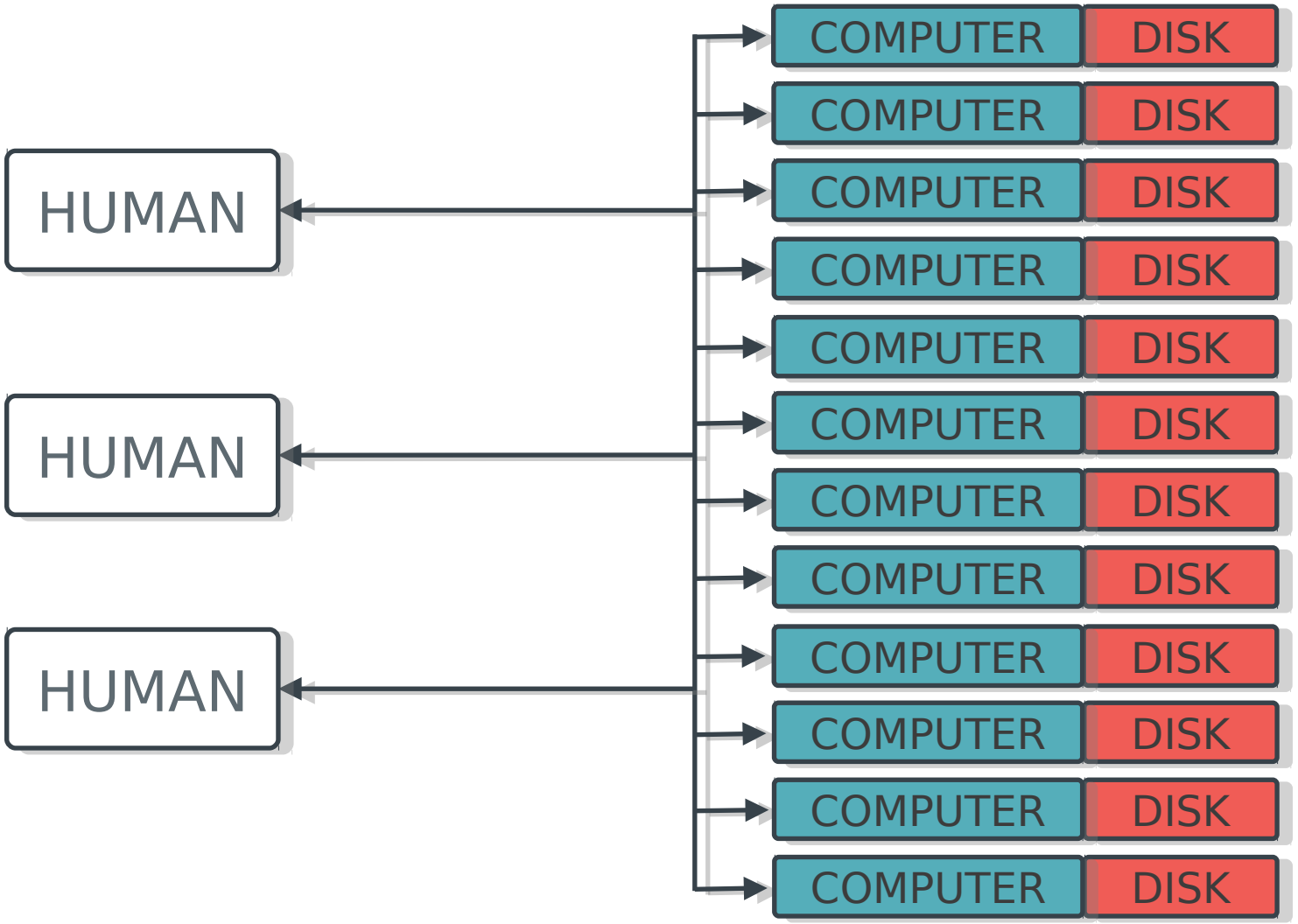

HUMAN
HUMAN
HUMAN
COMPUTER
DISK
DISK
DISK
DISK
DISK
DISK
DISK

ceph

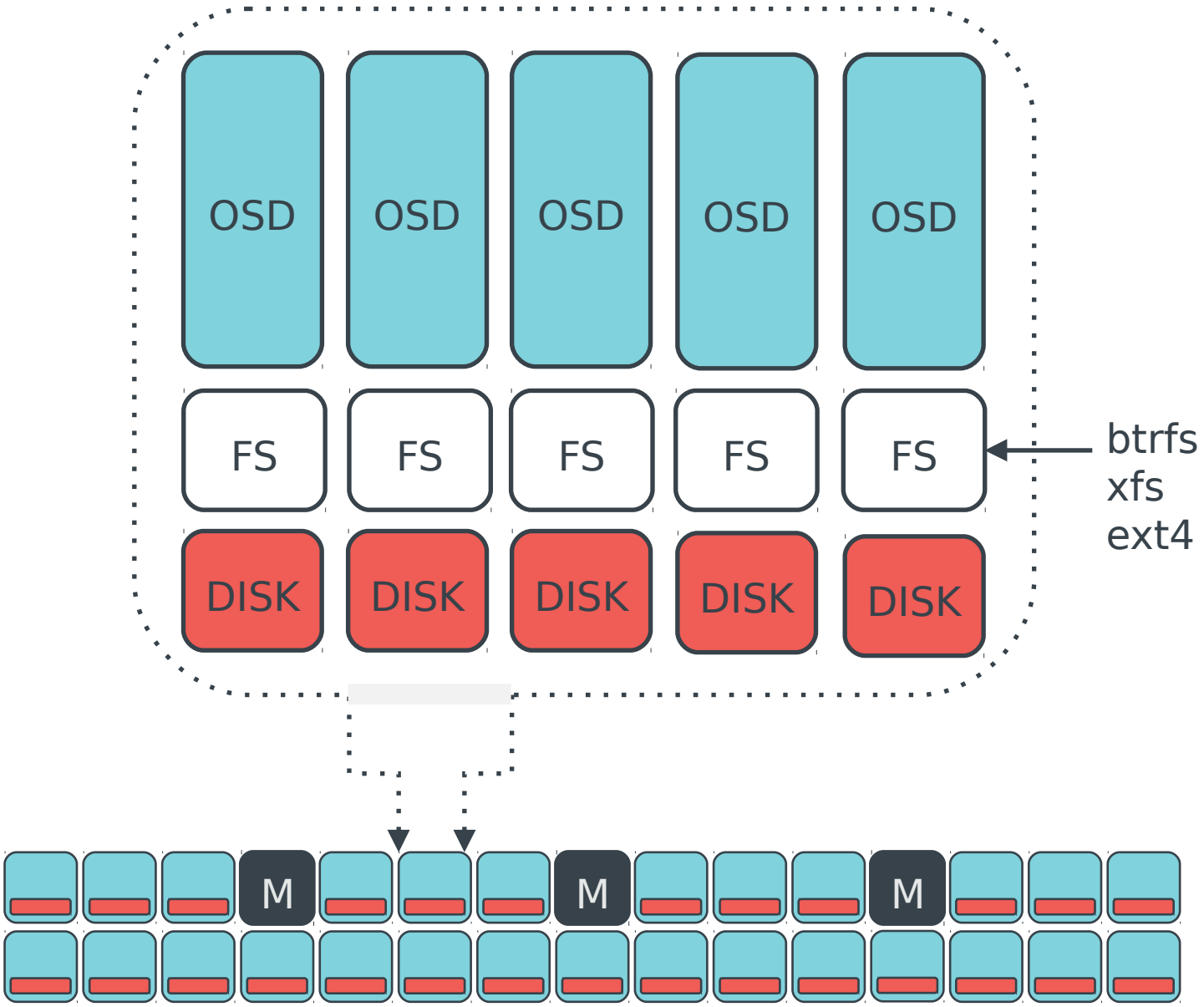


(actually more like this…)

ceph


HUMAN
HUMAN
HUMAN
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK

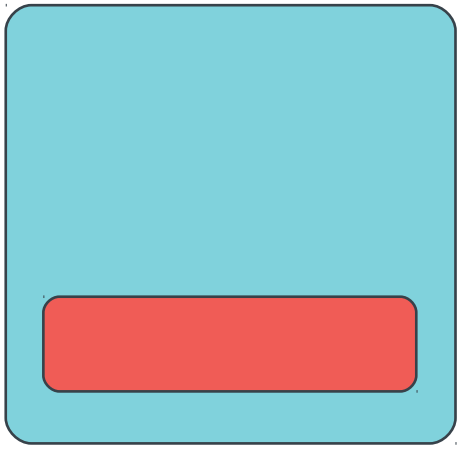
ceph


OSD
OSD
OSD
OSD
OSD
FS
FS
FS
FS
FS
DISK
DISK
DISK
DISK
DISK
btrfs
xfs
ext4
M
M
M

ceph

Monitors:

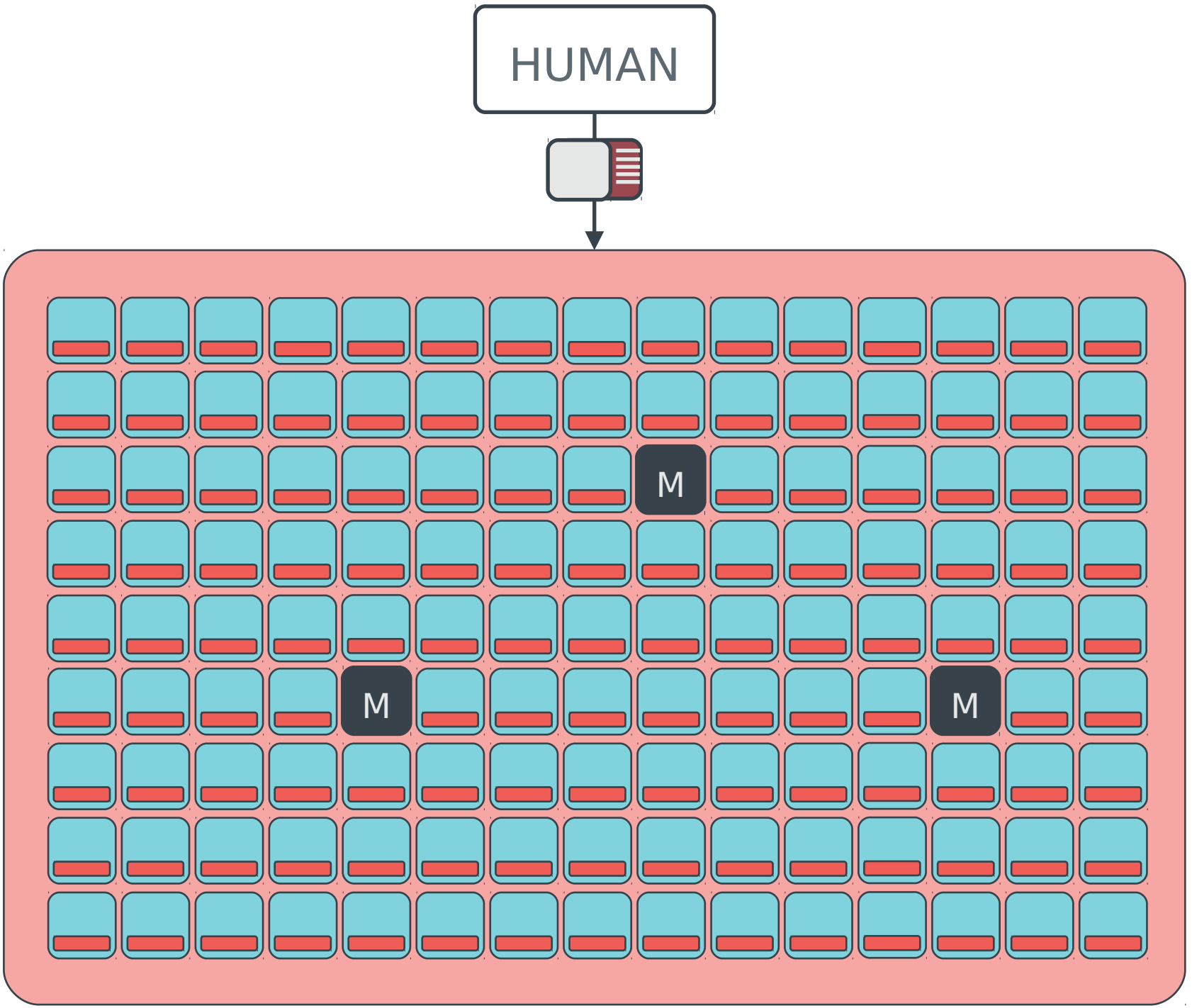
- Maintain cluster membership and state
- Provide consensus for distributed decision-making
- Small, odd number
- These do **not** serve stored objects to clients



Object Storage Daemons (OSDs):

- At least three in a cluster
- One per disk or RAID group
- Serve stored objects to clients
- Intelligently peer to perform replication tasks

ceph



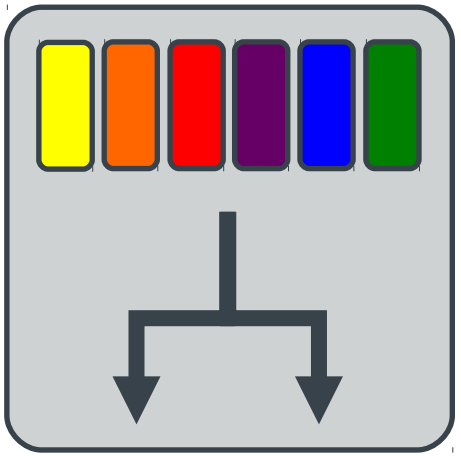
ceph

data distribution

- all objects are replicated N times
- objects are automatically placed, balanced, migrated in a dynamic cluster
- must consider physical infrastructure
  - ceph-osds on hosts in racks in rows in data centers

- three approaches
  - pick a spot; remember where you put it
  - pick a spot; write down where you put it
  - calculate where to put it, where to find it

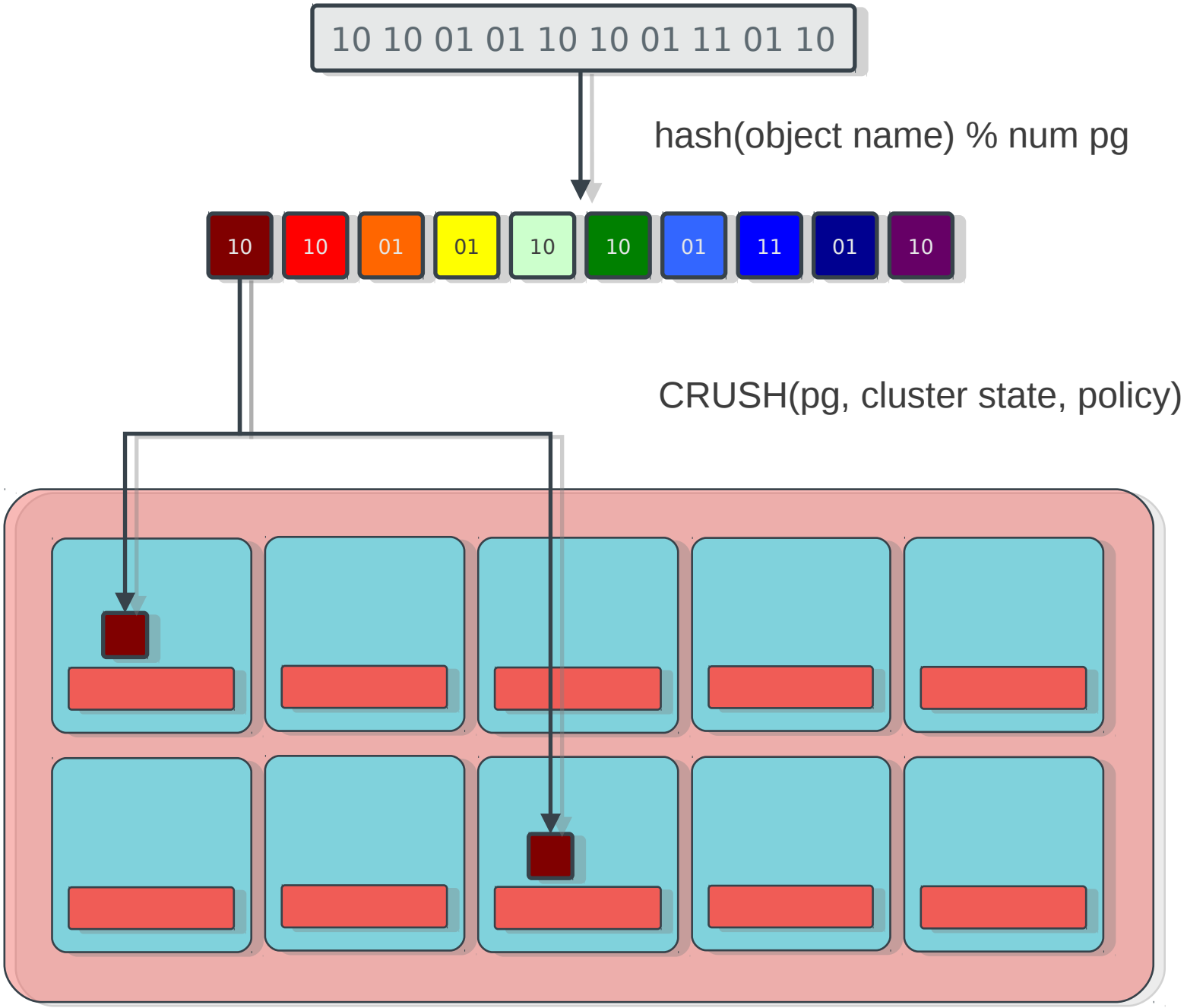
ceph



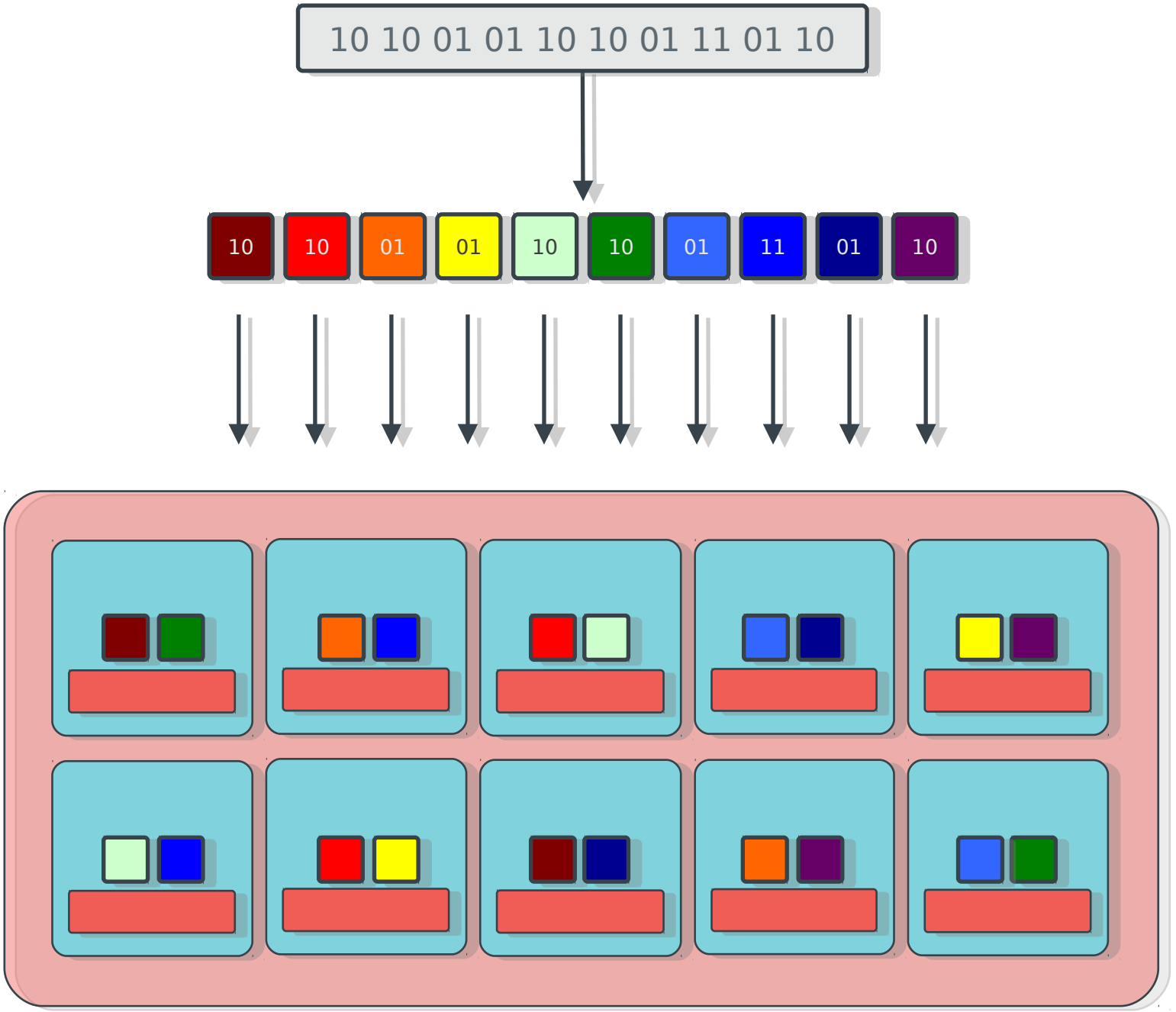
CRUSH

- Pseudo-random placement algorithm
- Fast calculation, **no lookup**
- Repeatable, deterministic
- Ensures even distribution
- Stable mapping
  - Limited data migration
- Rule-based configuration
  - specifiable replication
  - infrastructure topology aware
  - allows weighting

ceph


10 10 01 01 10 10 01 11 01 10
hash(object name) % num pg
10
10
01
01
10
10
01
11
01
10
CRUSH(pg, cluster state, policy)

ceph

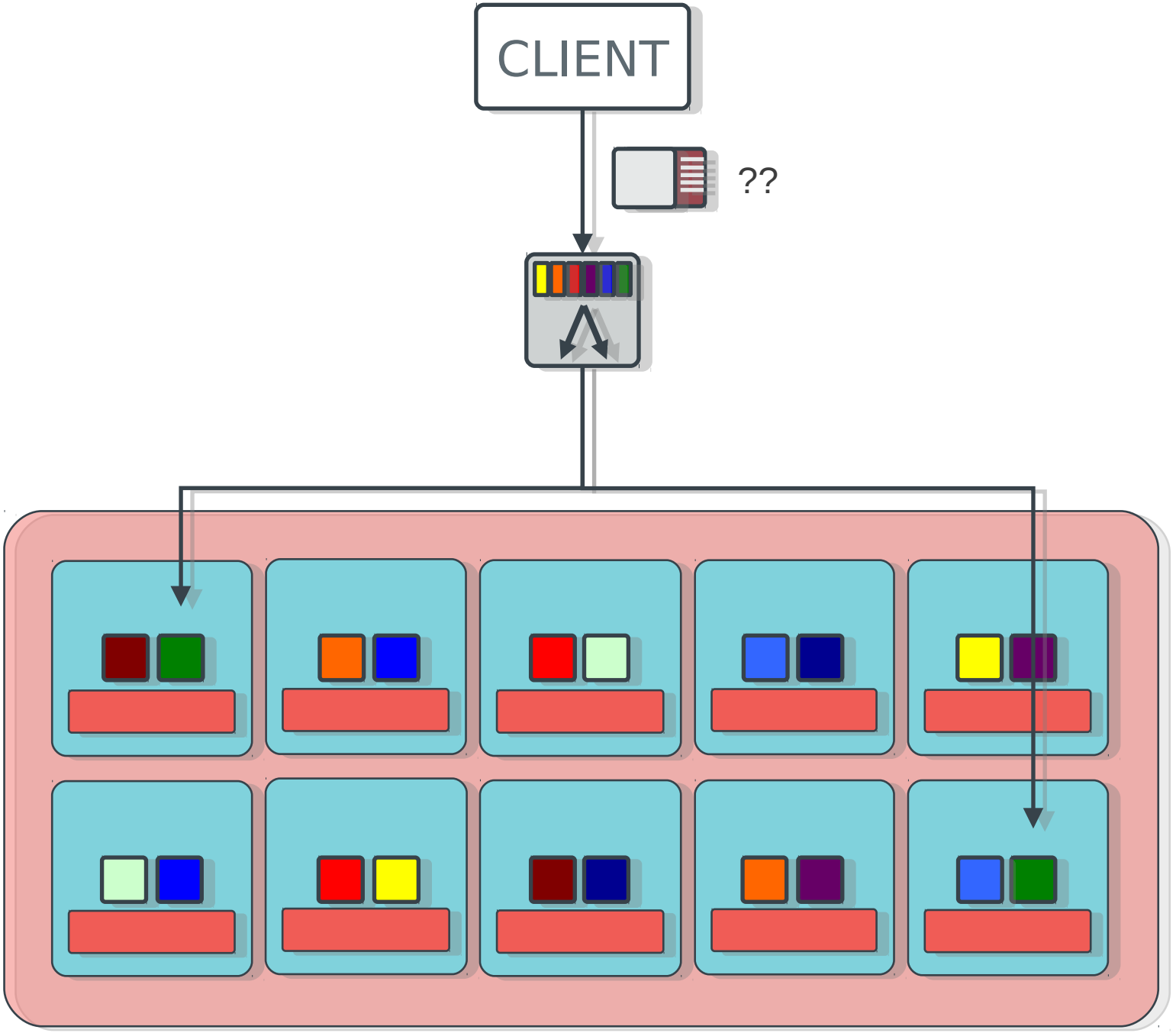


ceph

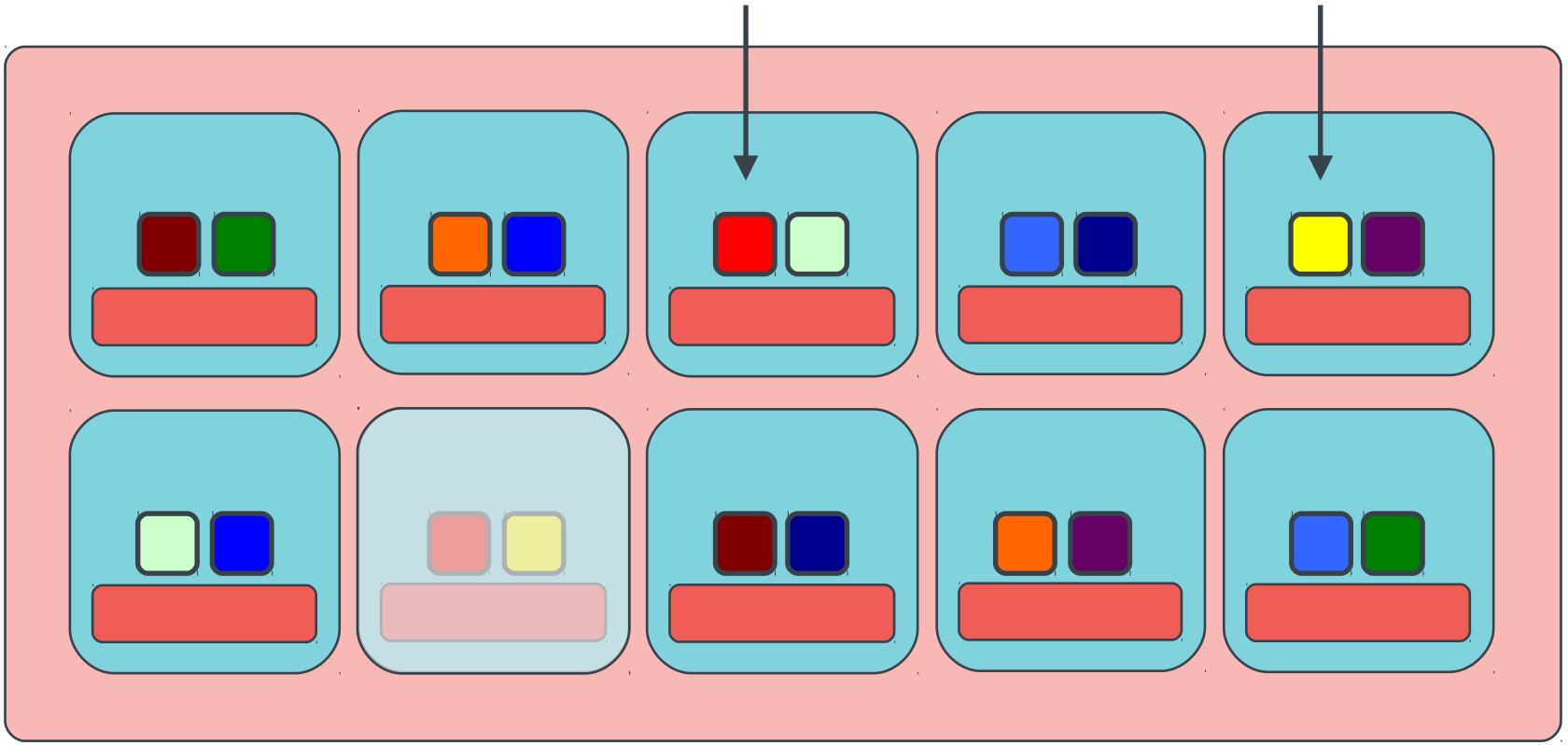
RADOS

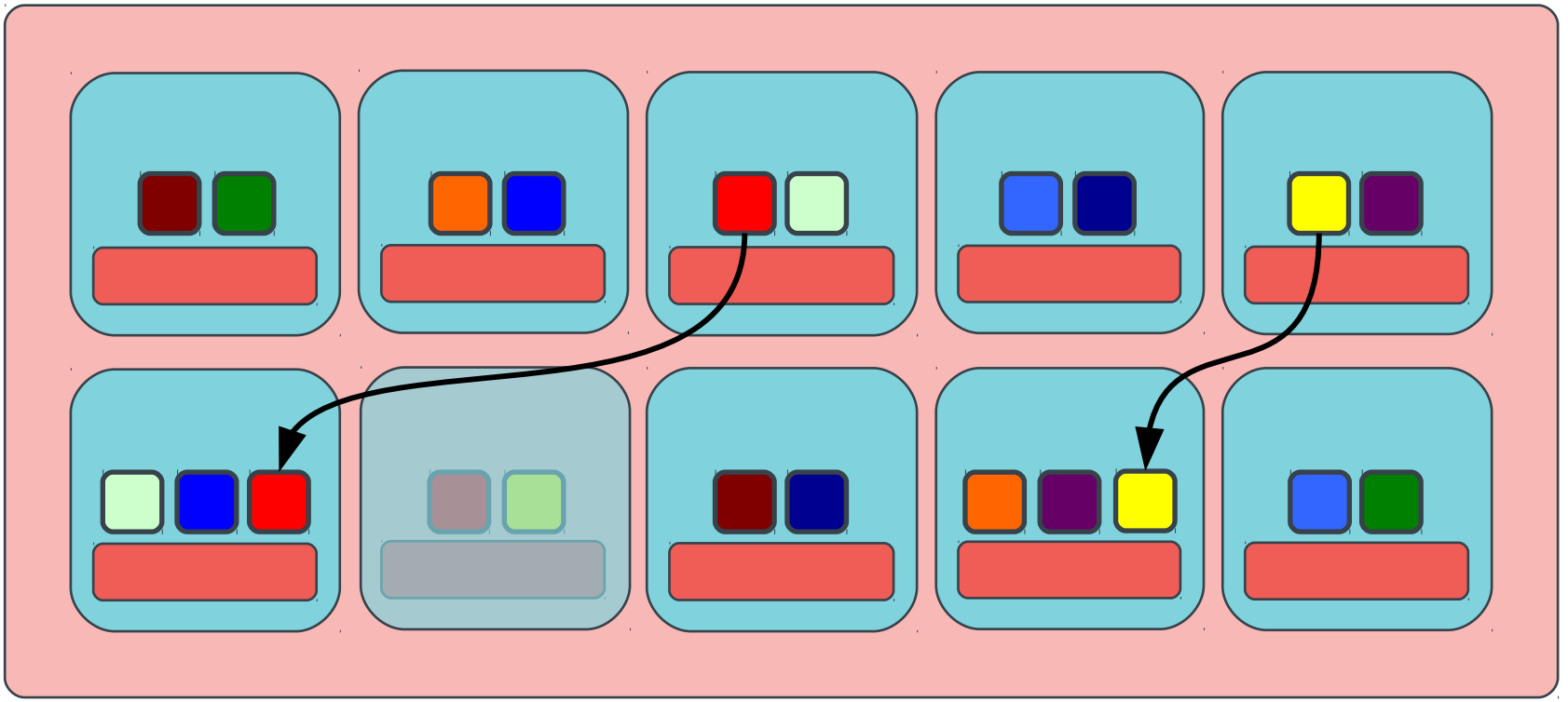
- monitors publish **osd map** that describes cluster state
  - ceph-osd node status (up/down, weight, IP)
  - CRUSH function specifying desired data distribution
- object storage daemons (OSDs)
  - safely replicate and store object
  - migrate data as the cluster changes over time
  - coordinate based on shared view of reality
- decentralized, distributed approach allows
  - massive scales (10,000s of servers or more)
  - the illusion of a single copy with consistent behavior

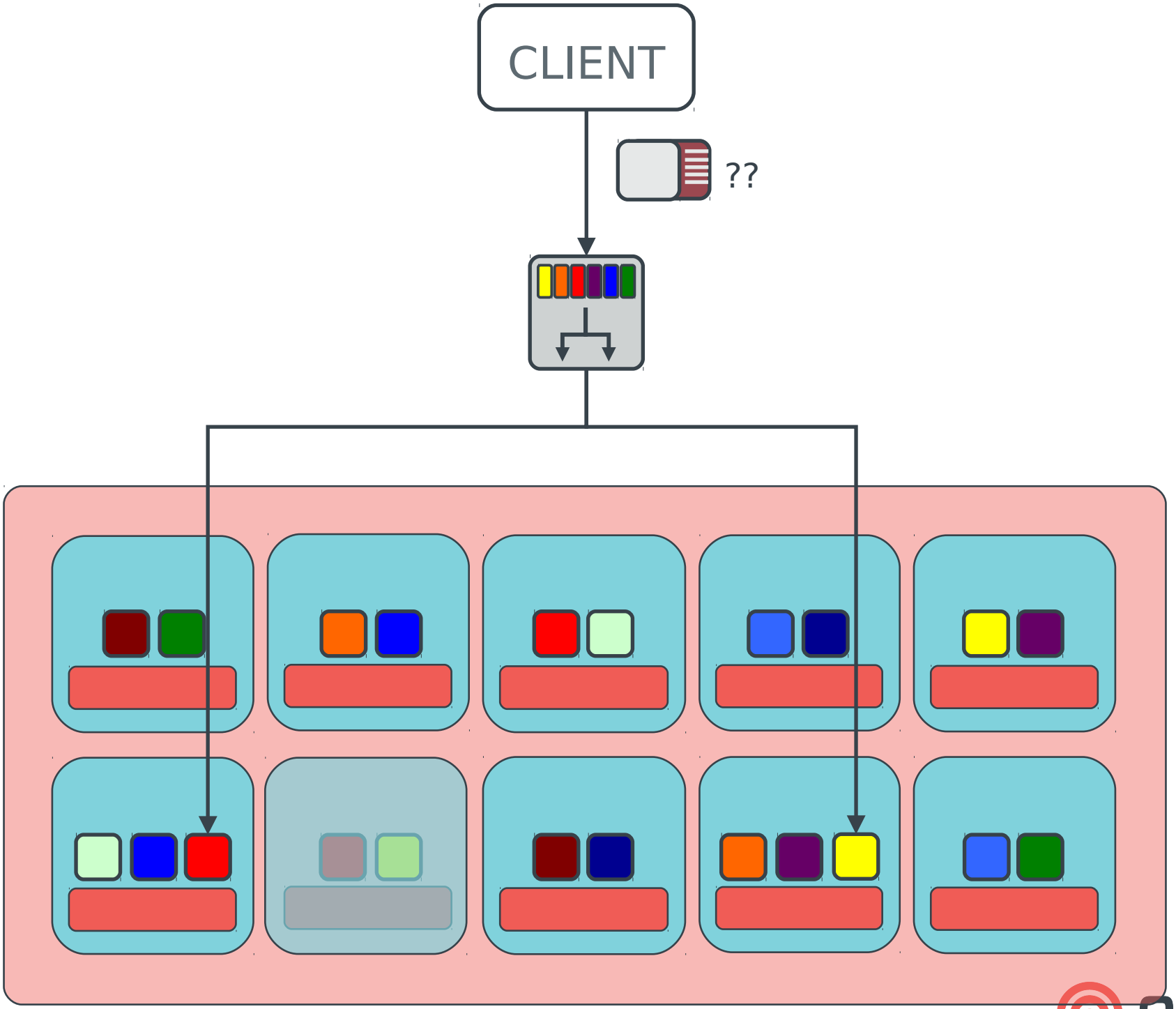
ceph



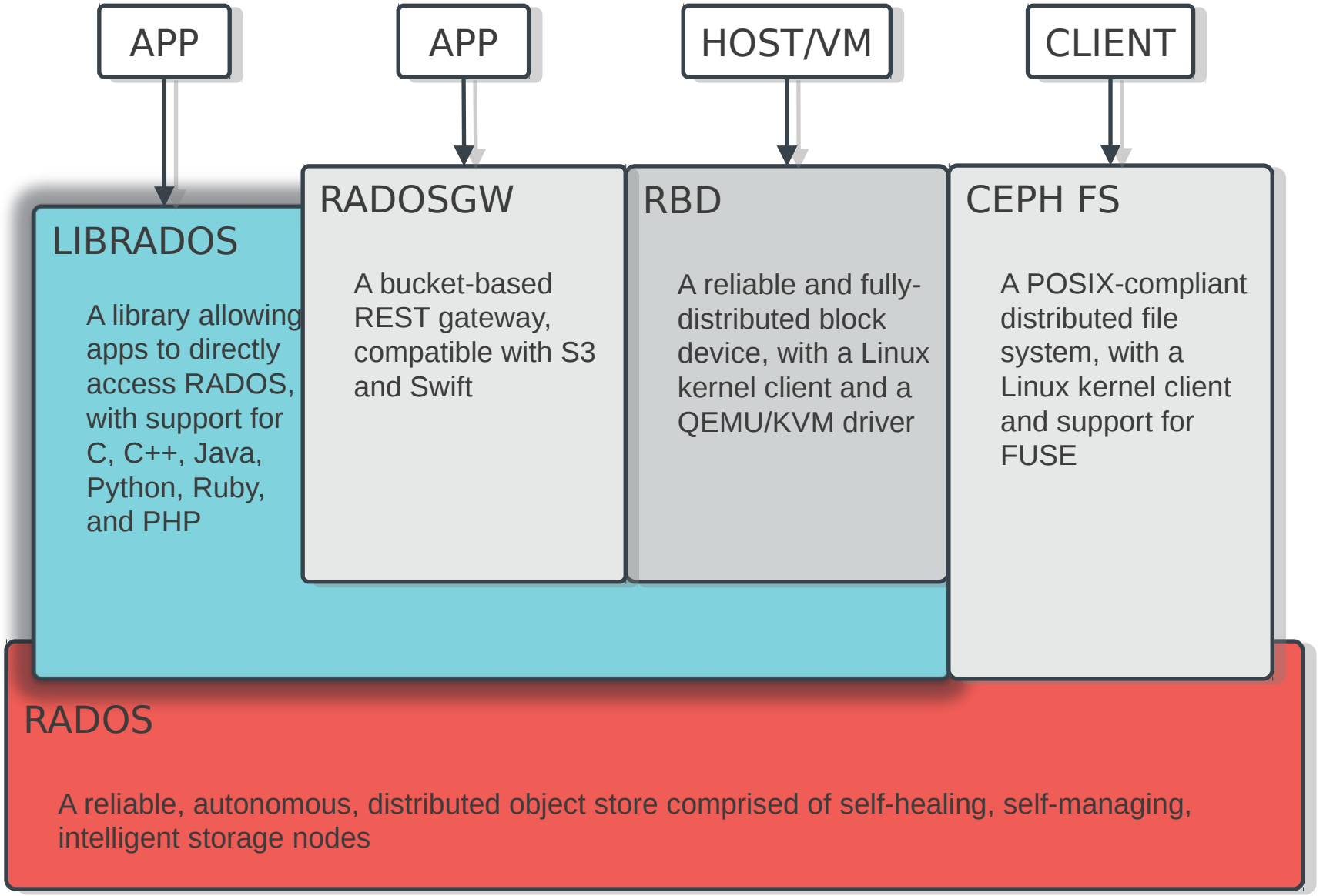
ceph



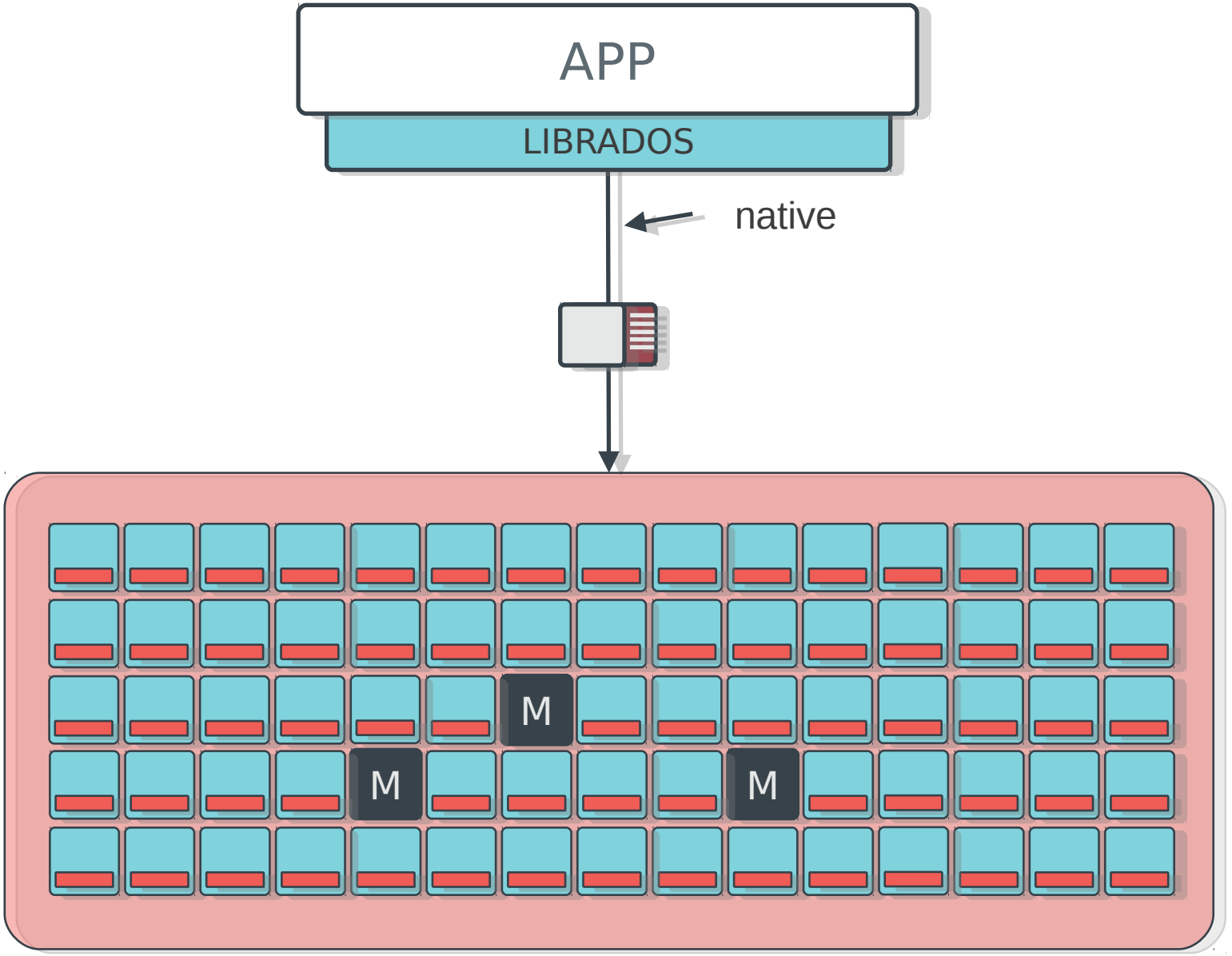




ceph


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

ceph

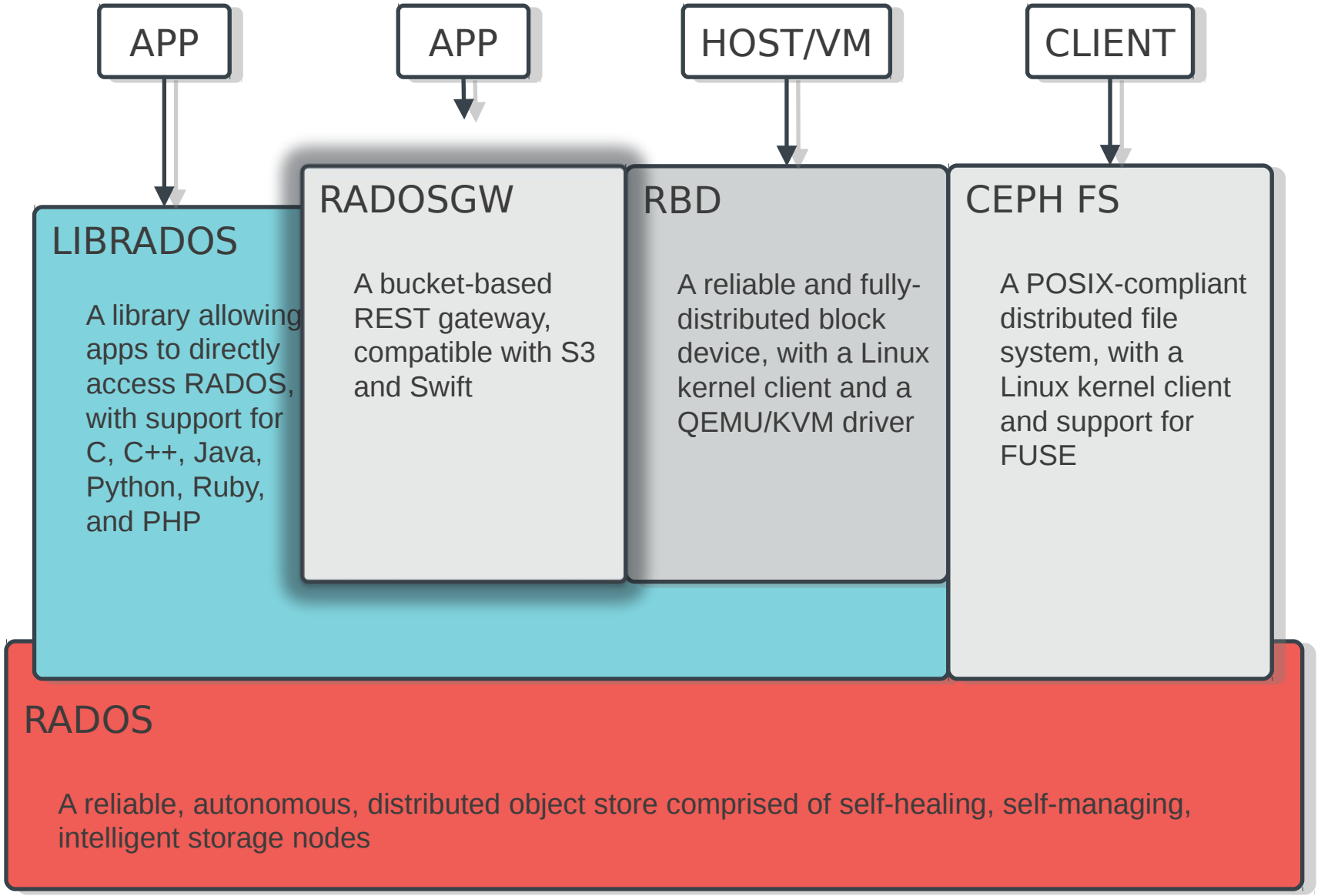


ceph

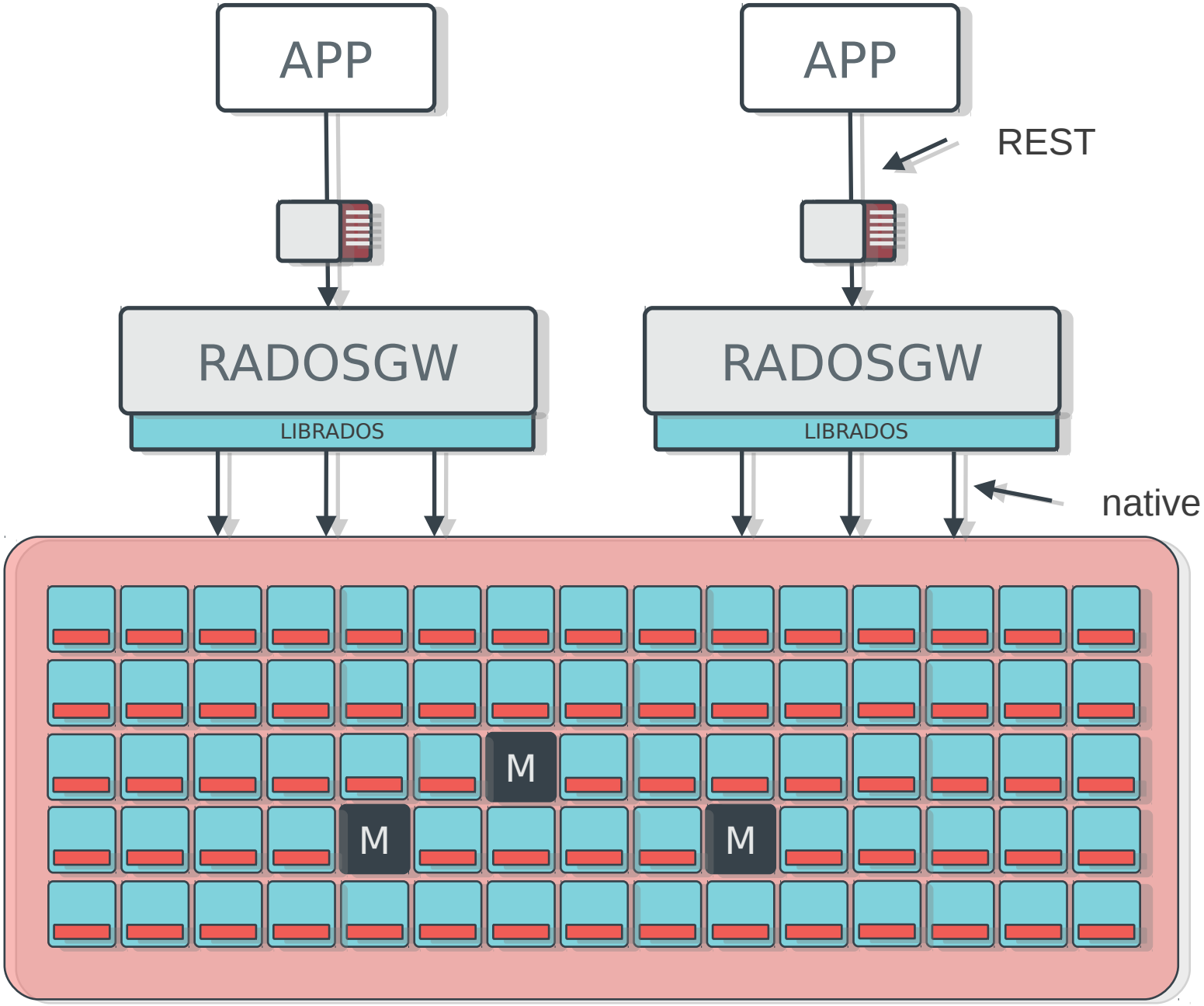
LIBRADOS

- Provides direct access to RADOS for applications
- C, C++, Python, PHP, Java
- No HTTP overhead

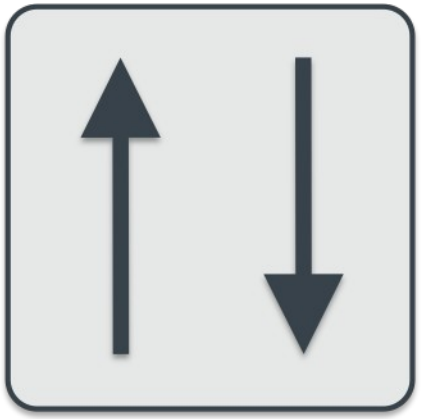
ceph


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

ceph



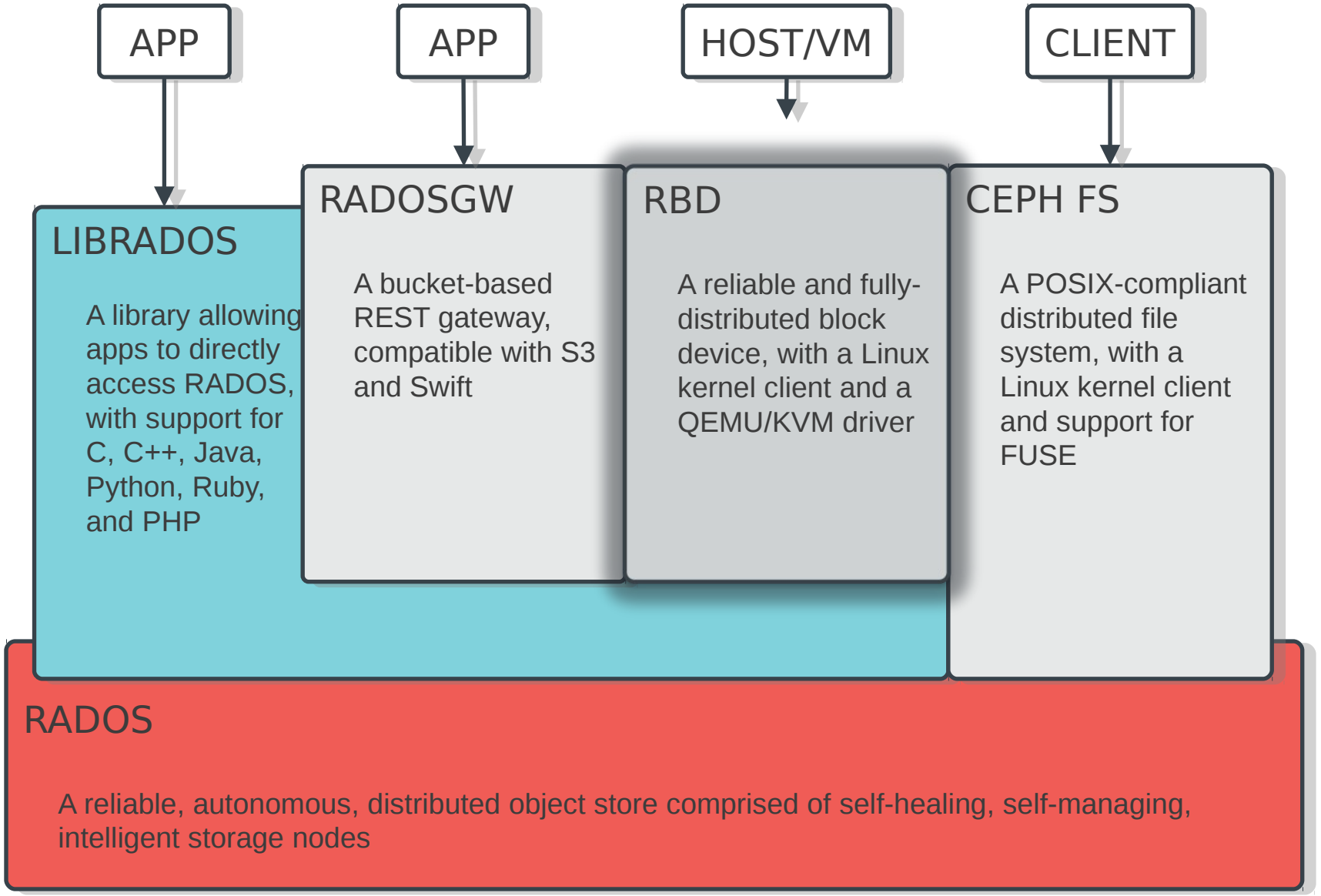
ceph



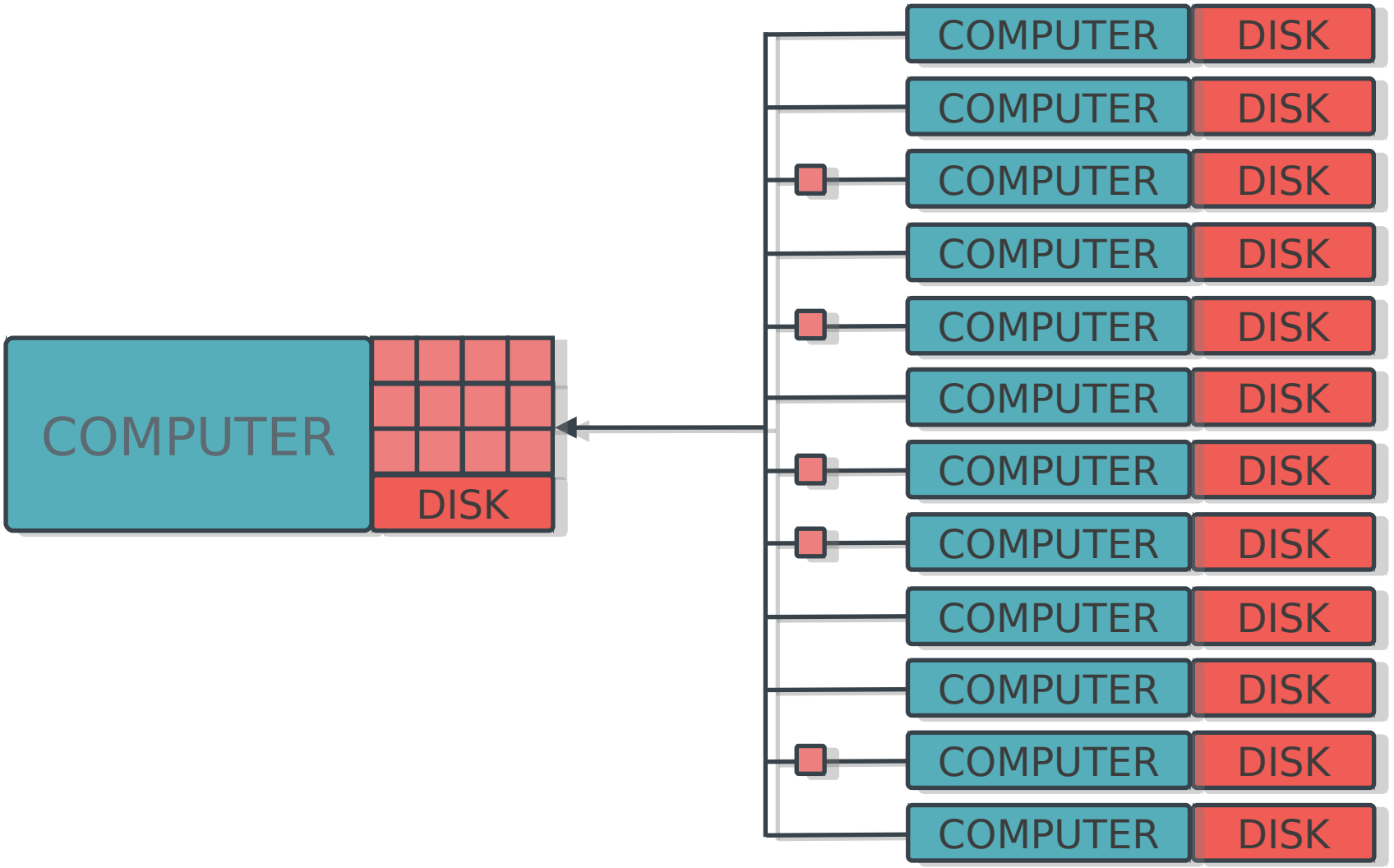
RADOS Gateway:

- REST-based interface to RADOS
- Supports buckets, accounting
- Compatible with S3 and Swift applications

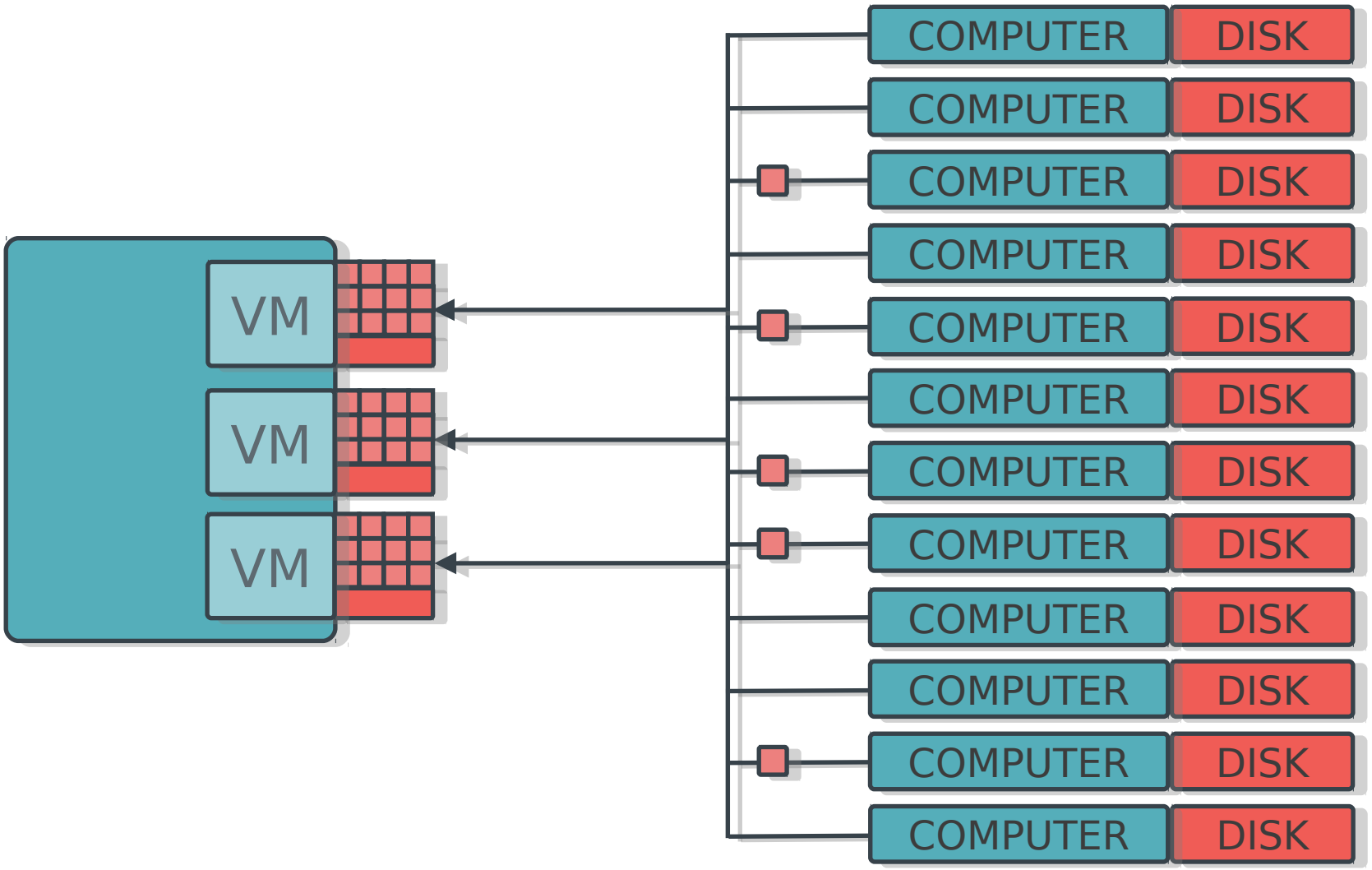
ceph


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

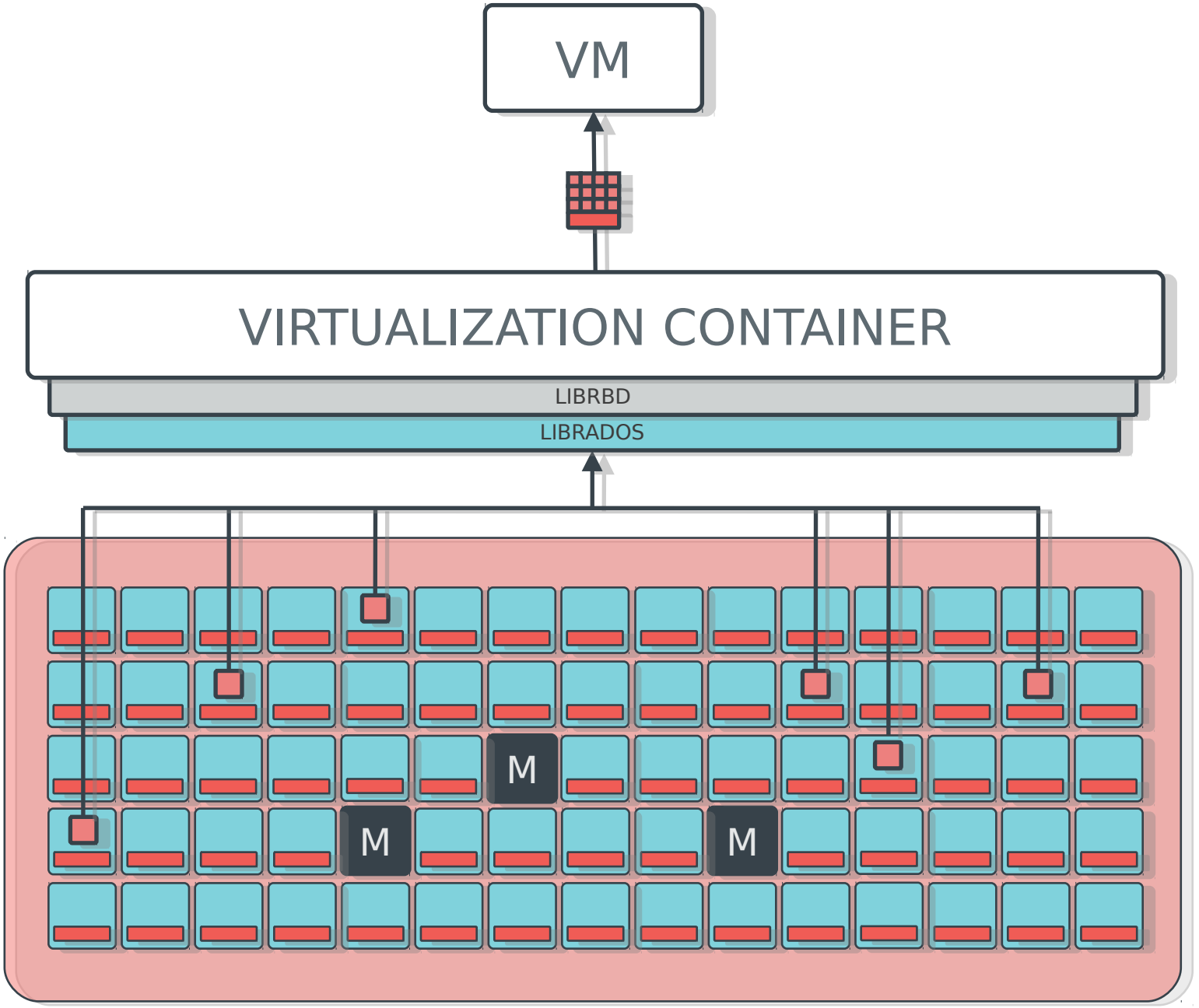
ceph


COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK

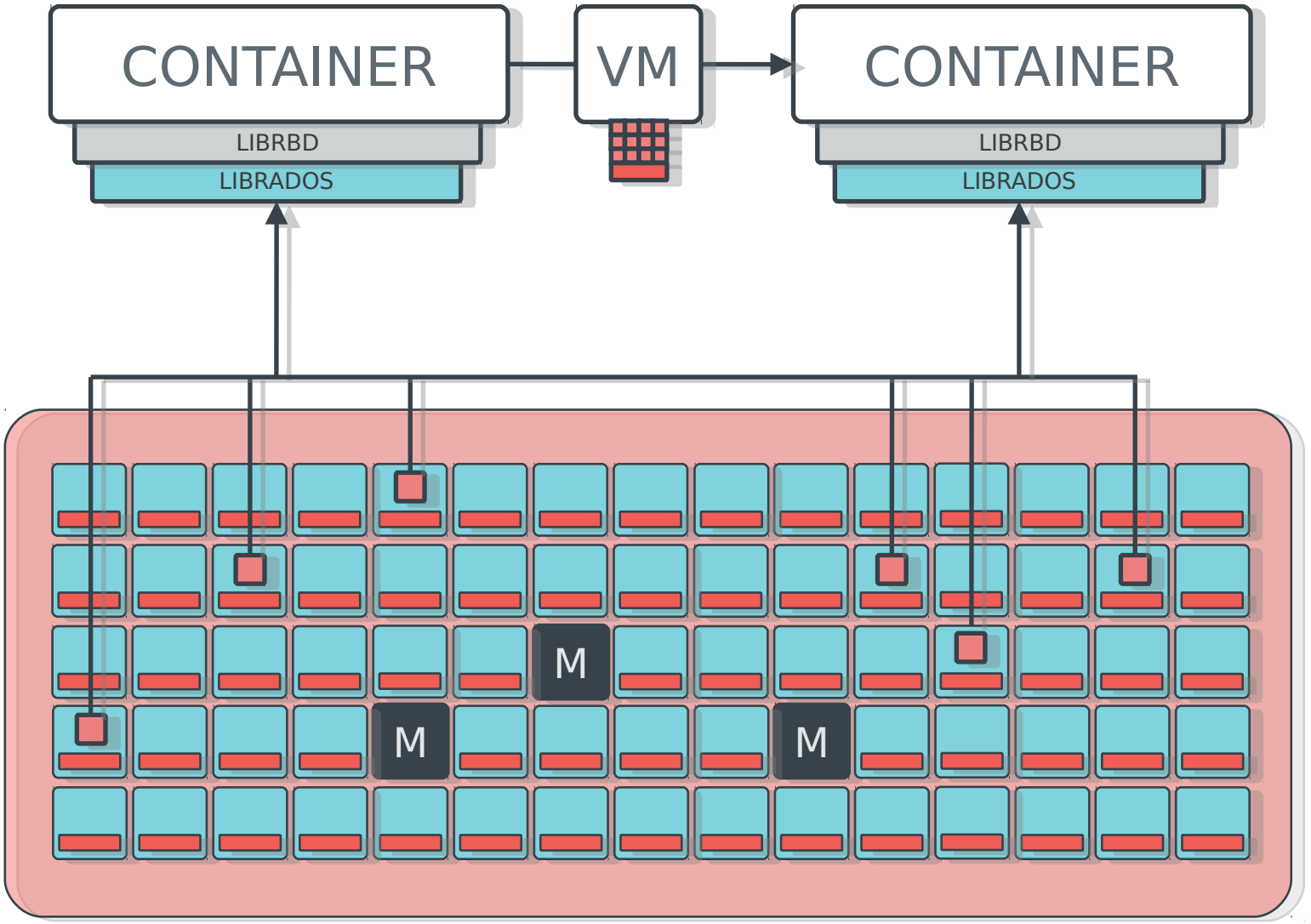
ceph


VM
VM
VM
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK
COMPUTER
DISK

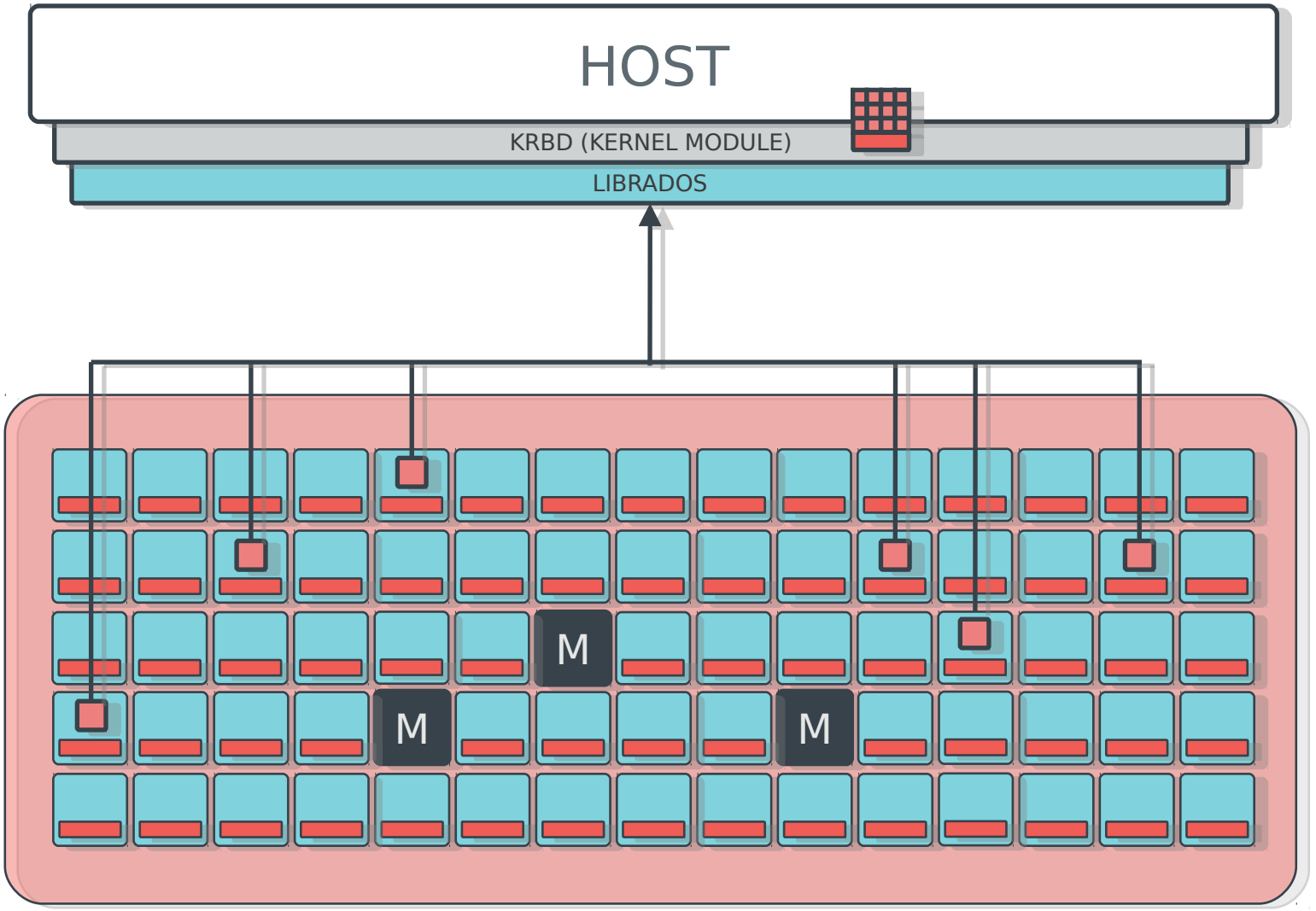
ceph



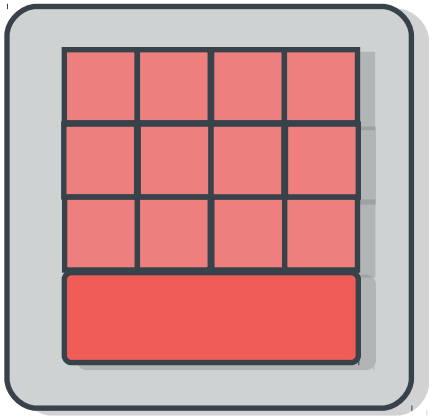
ceph


CONTAINER
VM
CONTAINER
LIBRBD
LIBRADOS
LIBRBD
LIBRADOS
M
M
M

ceph



ceph



RADOS Block Device:

- Storage of virtual disks in RADOS
- Decouples VMs and containers
  - Live migration!
- Images are striped across the cluster
- Snapshots!
- Support in
  - Qemu/KVM
  - OpenStack, CloudStack
  - Mainline Linux kernel

ceph

HOW DO YOU

SPIN UP

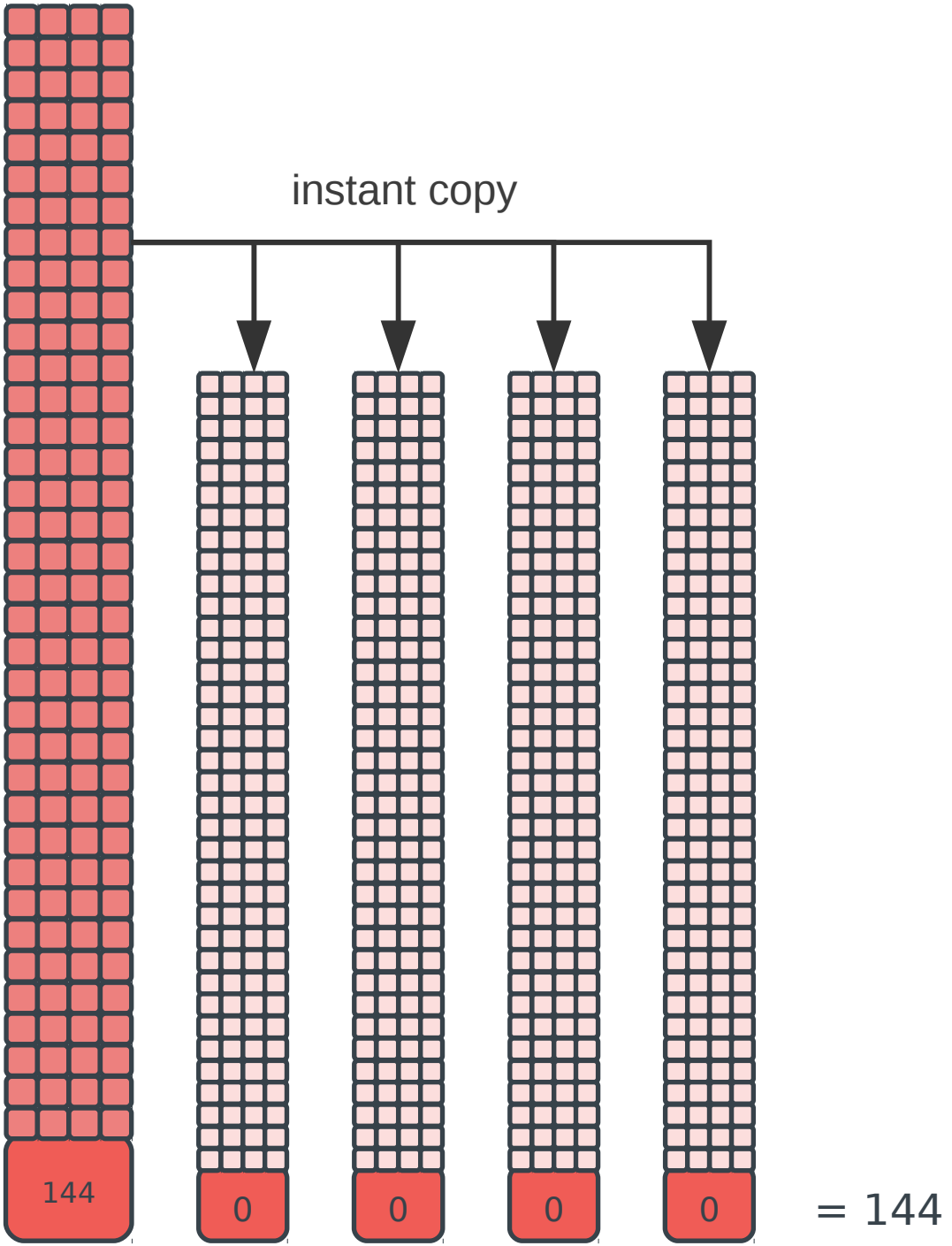
THOUSANDS OF VMs

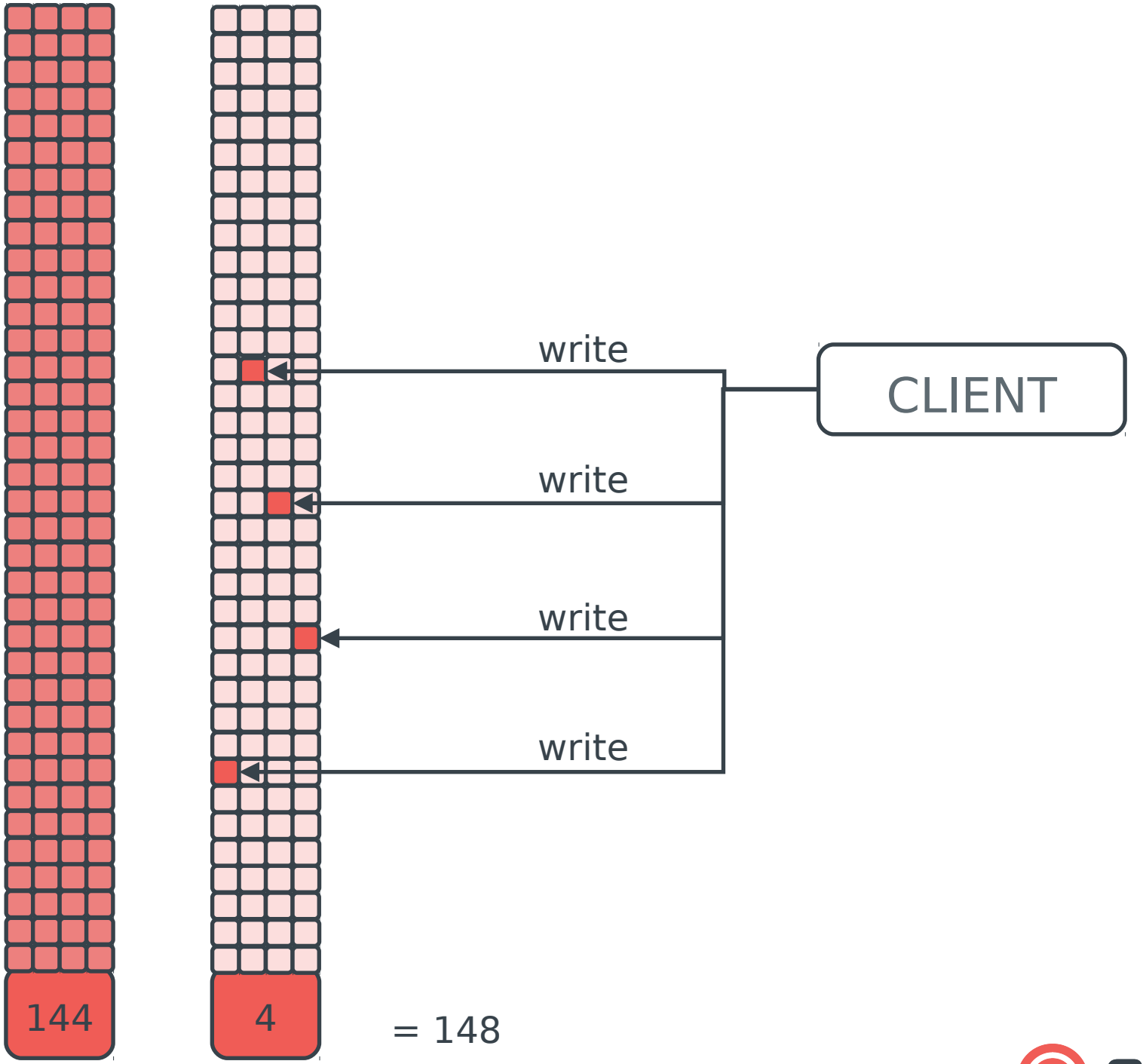
INSTANTLY

AND

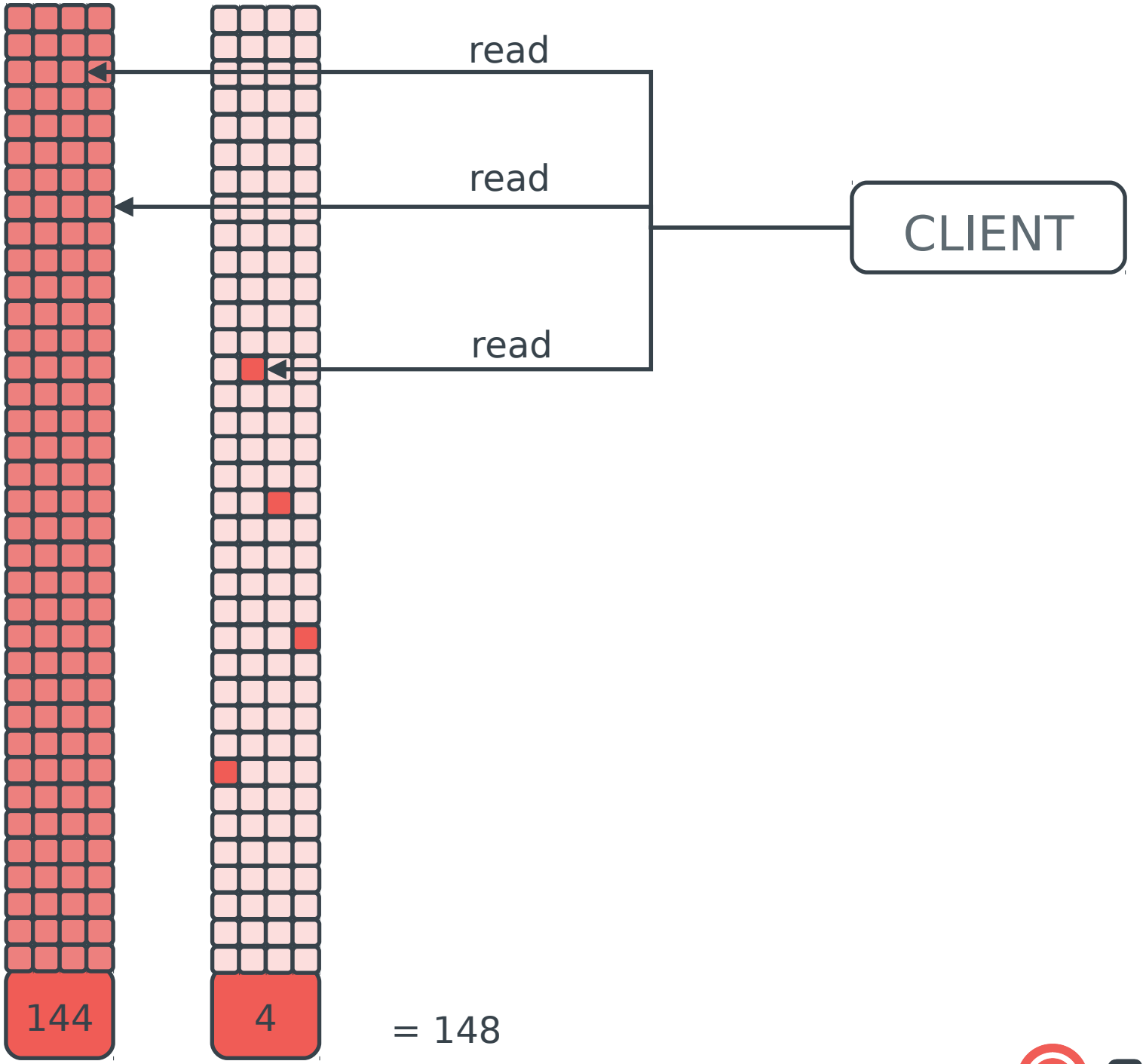
EFFICIENTLY?

ceph

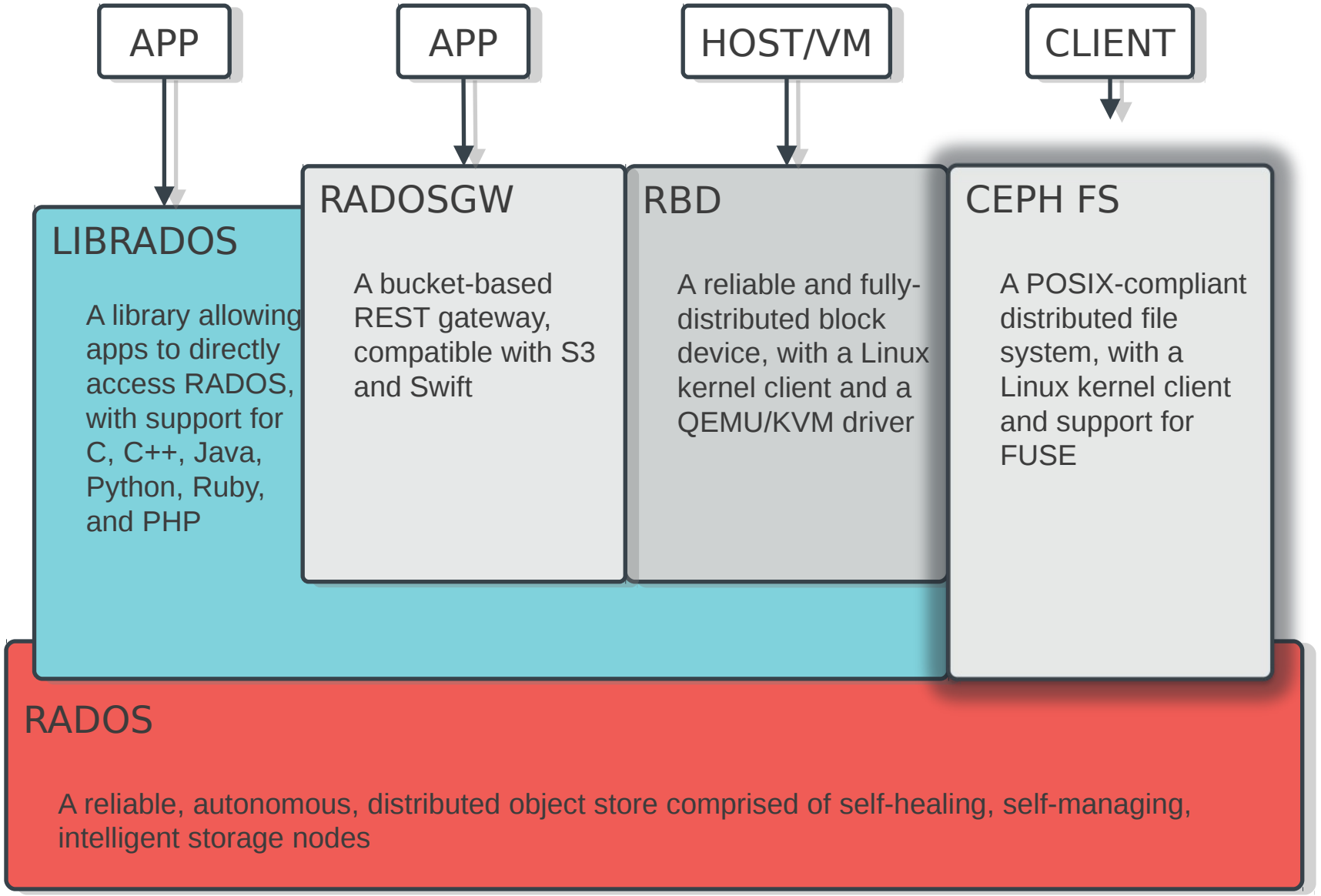

instant copy
144
0
0
0
0
= 144



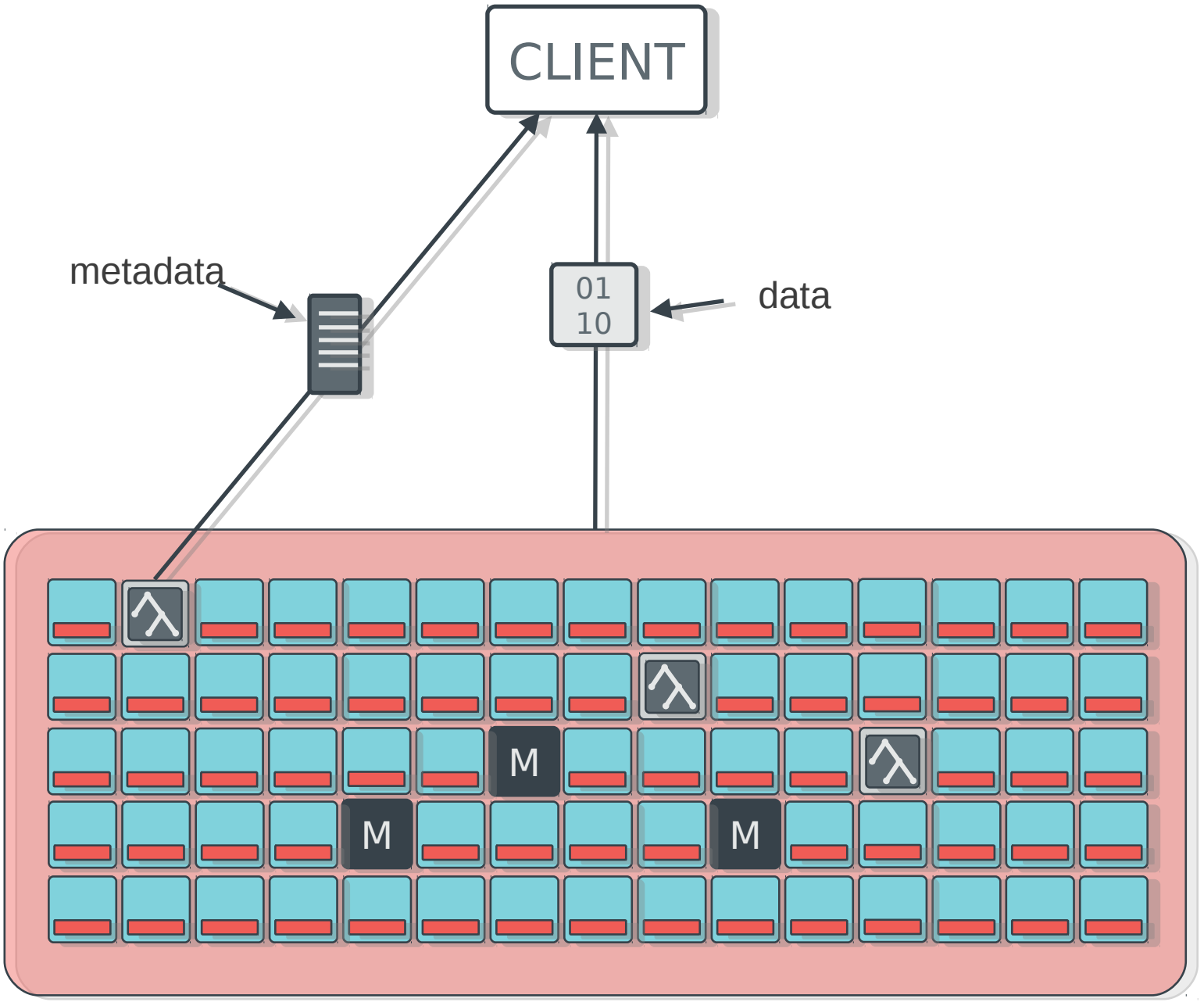
ceph



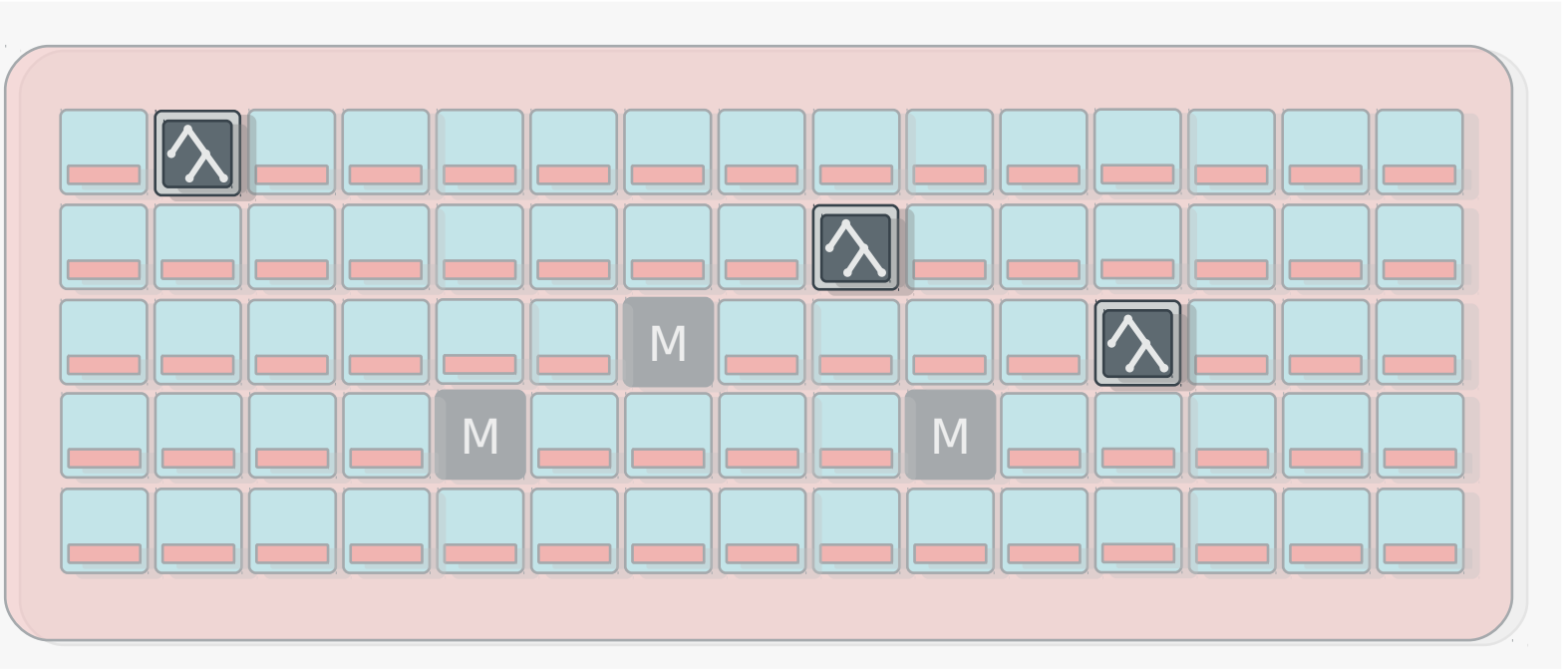
ceph


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

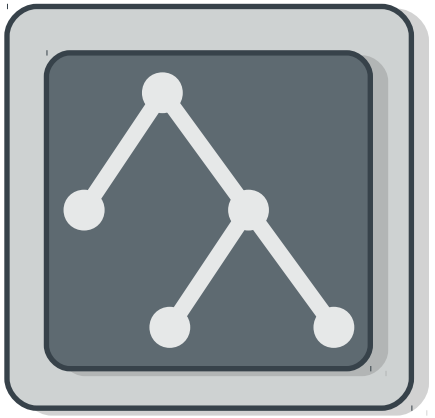
ceph



ceph

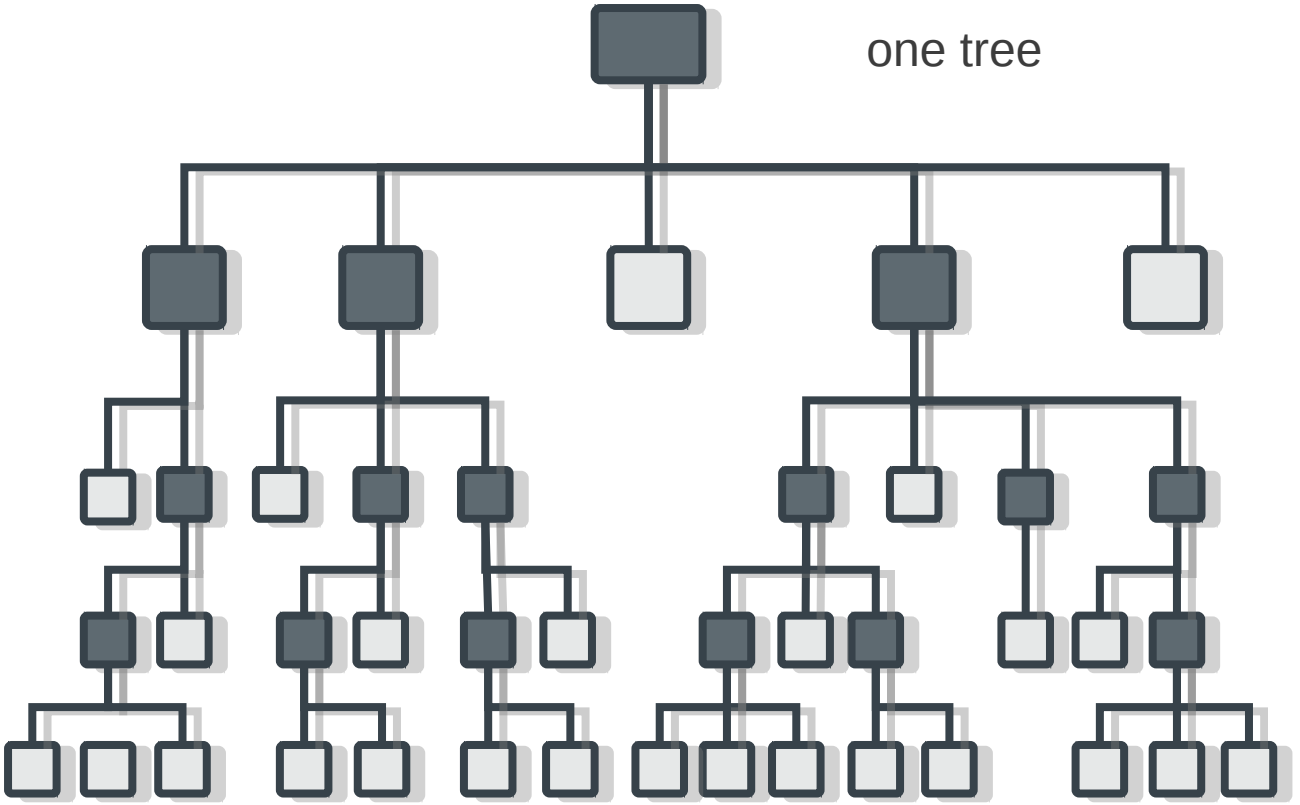


ceph

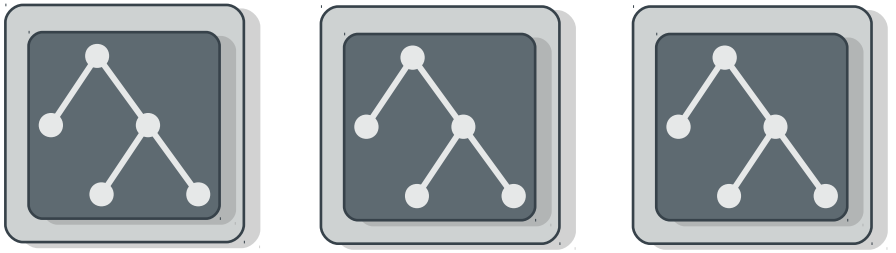


Metadata Server

- Manages metadata for a POSIX-compliant shared filesystem
  - Directory hierarchy
  - File metadata (owner, timestamps, mode, etc.)
- Stores metadata in RADOS
- Does not serve file data to clients
- Only required for shared filesystem

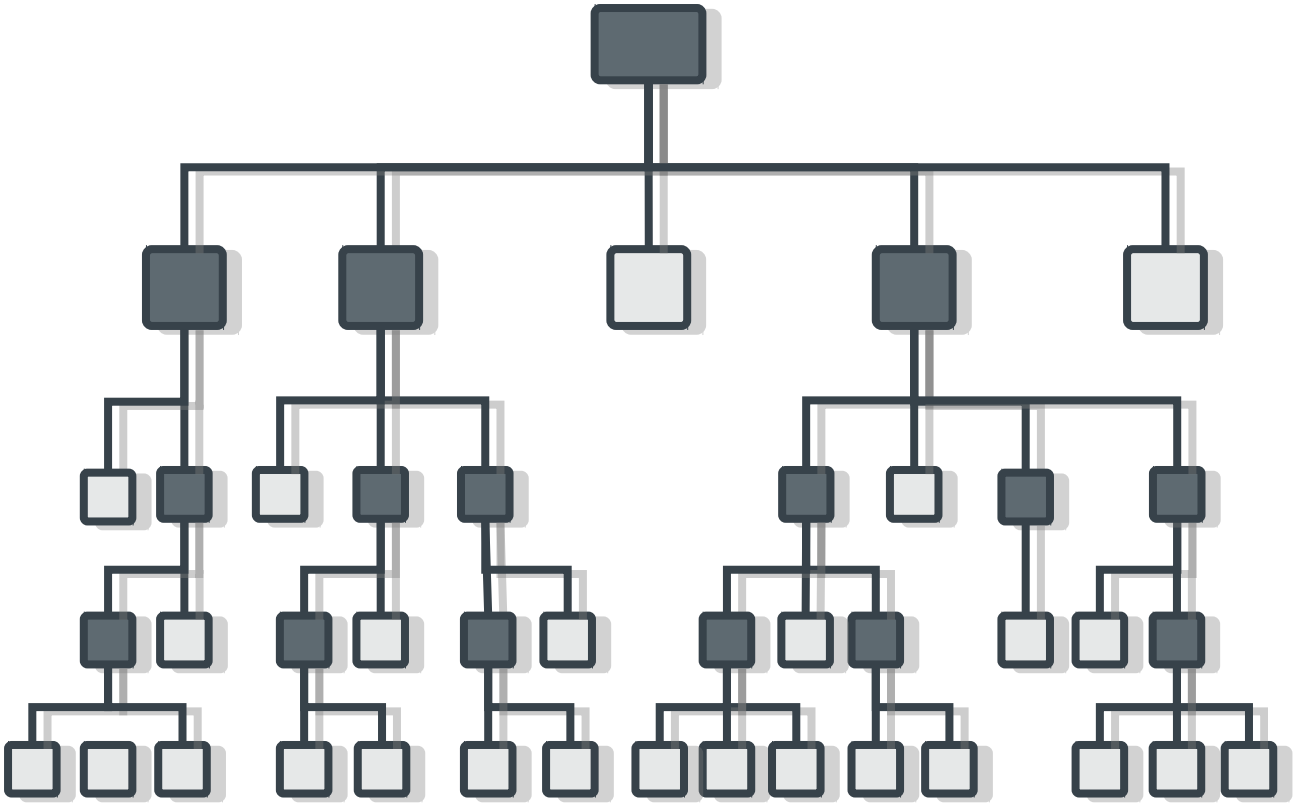


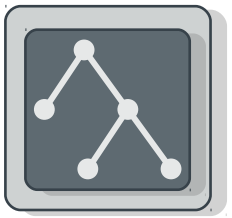
three metadata servers



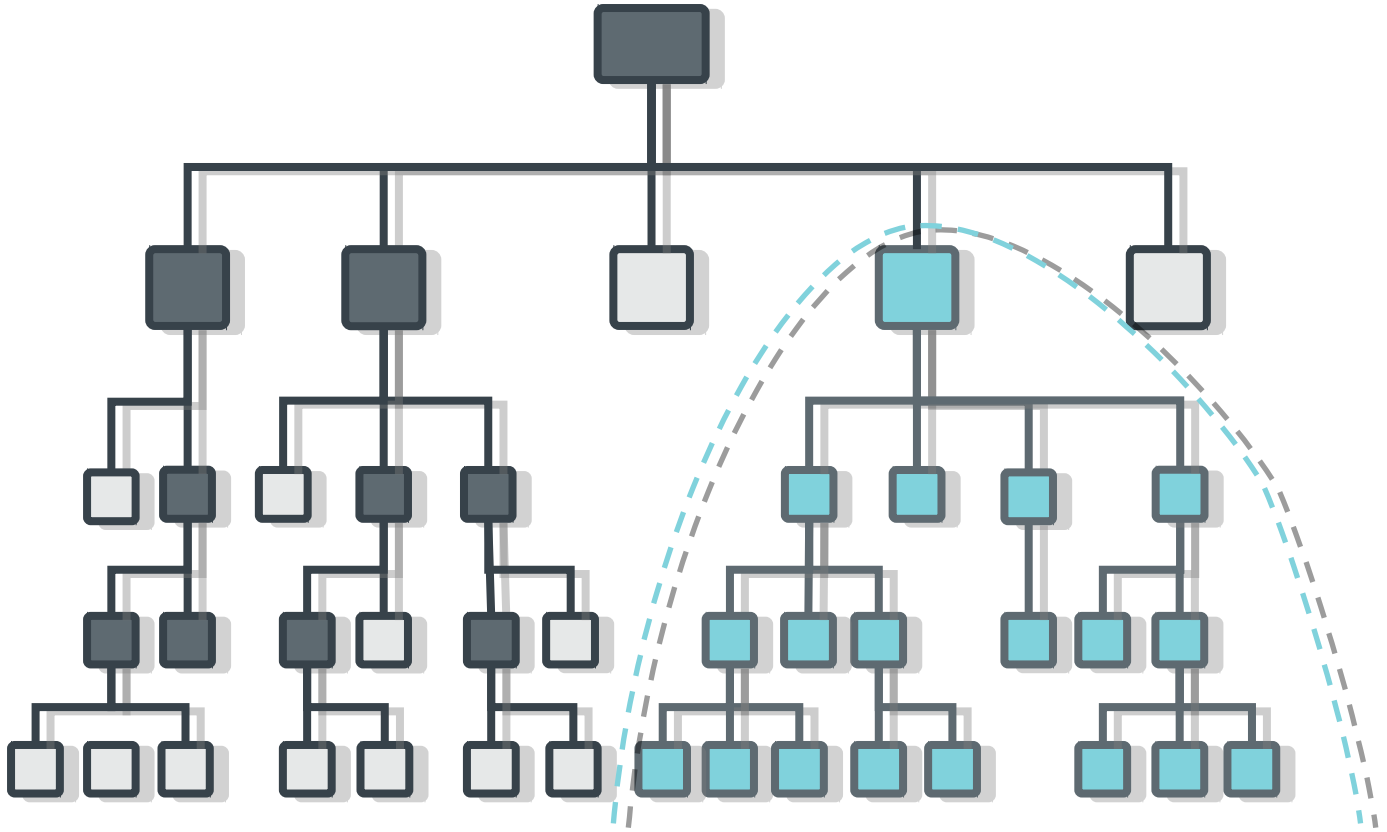
??

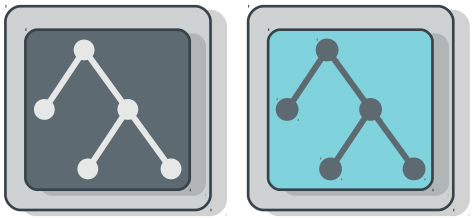
ceph



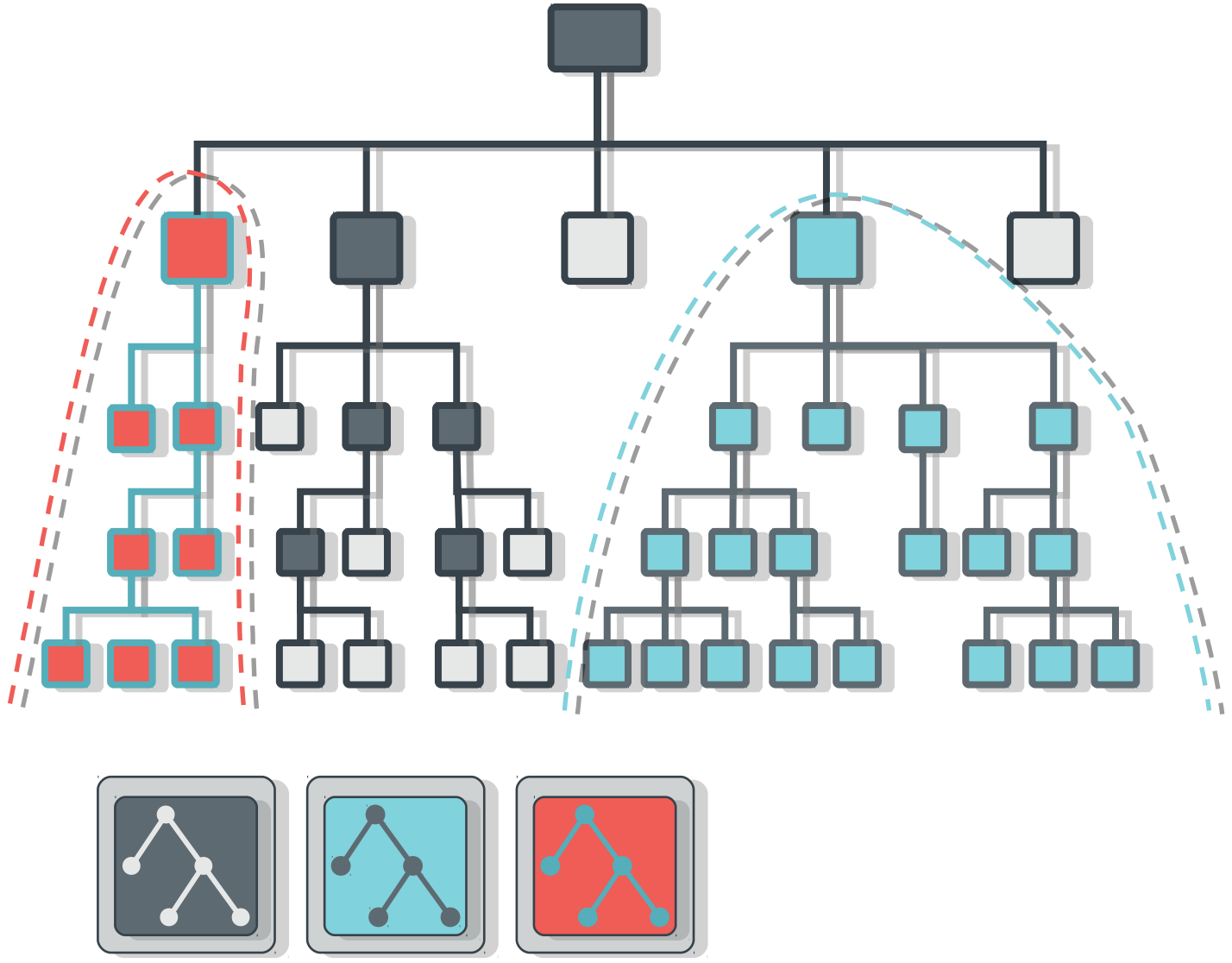


ceph

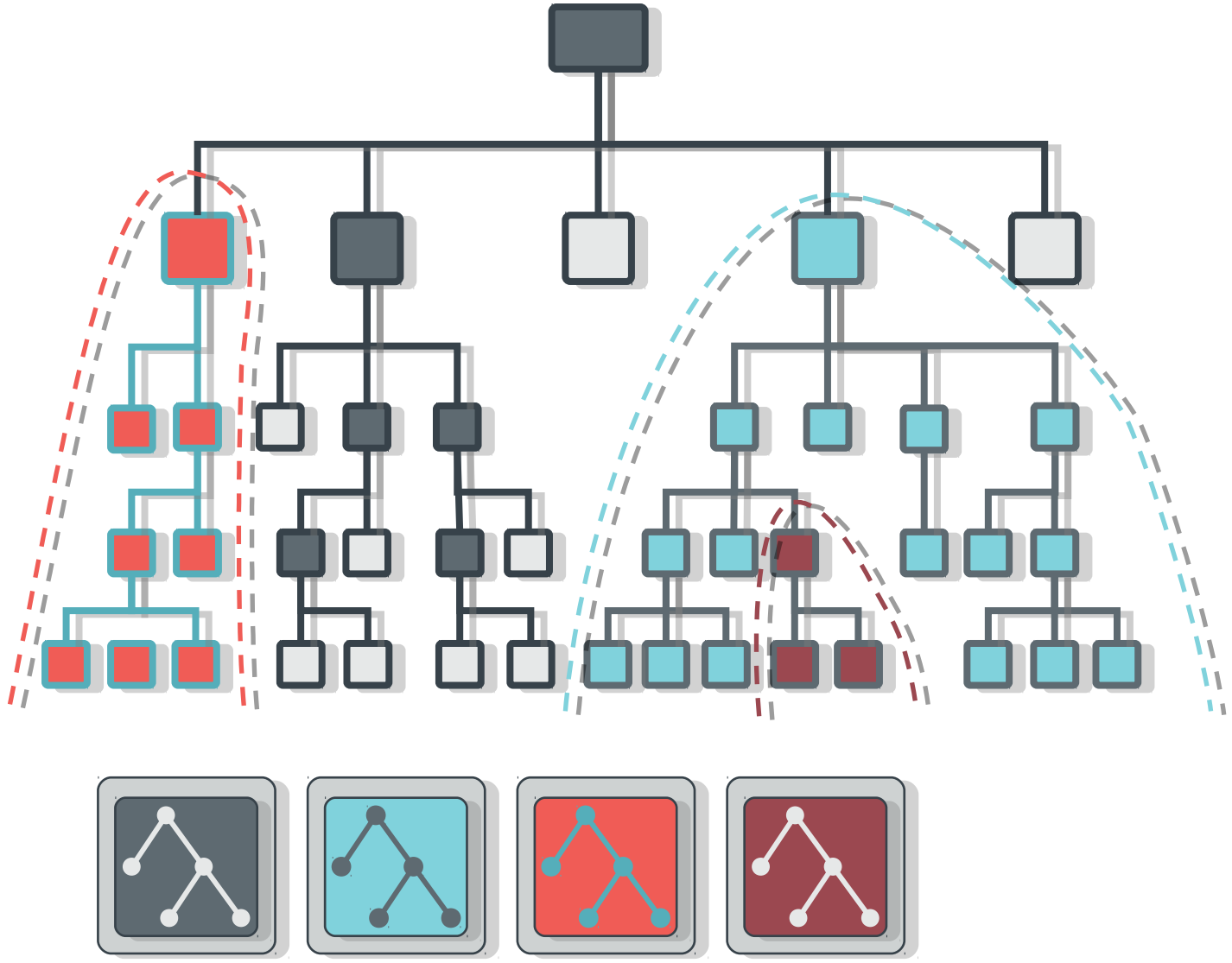




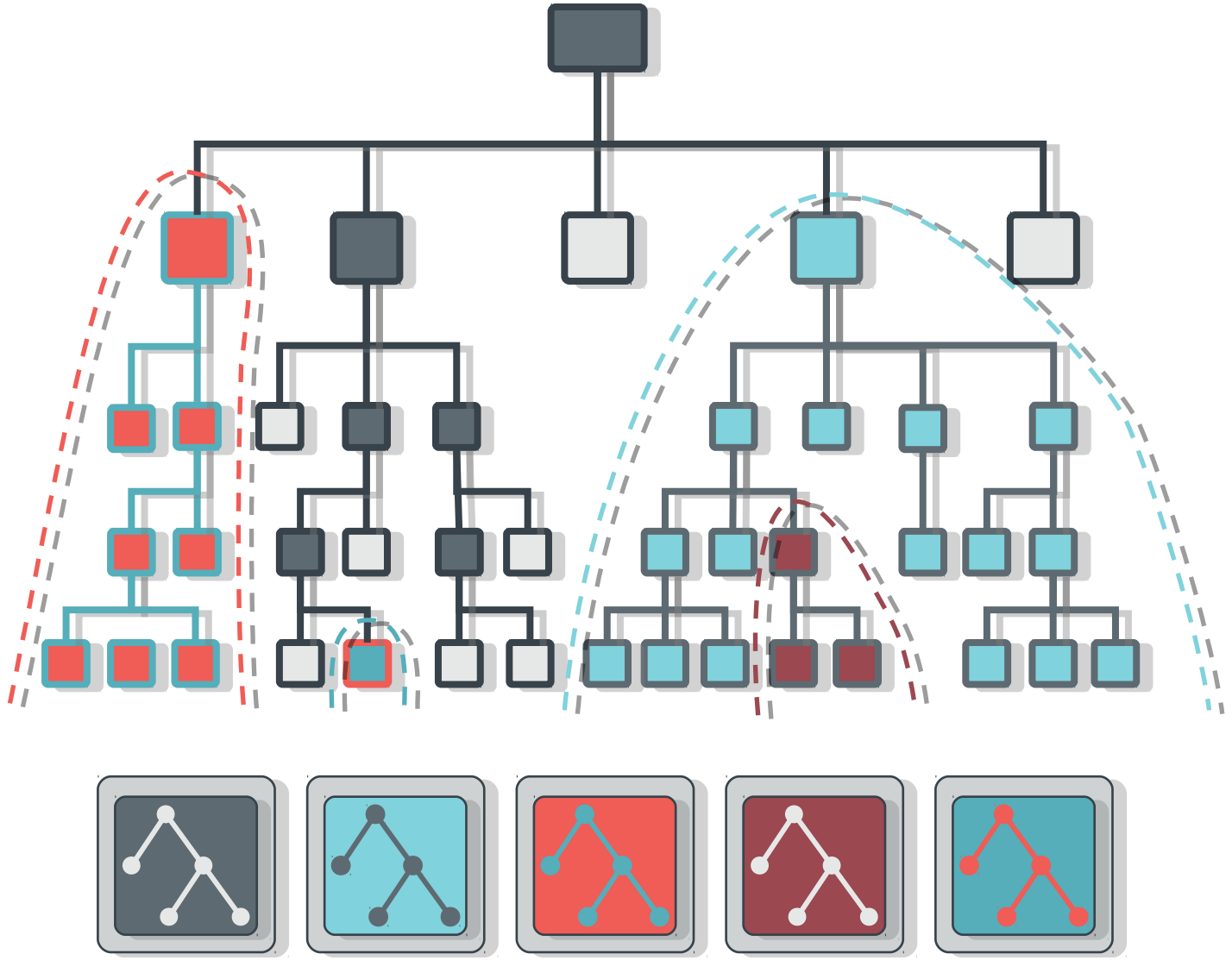
ceph



ceph



ceph



DYNAMIC SUBTREE PARTITIONING

ceph

recursive accounting

- ceph-mds tracks recursive directory stats
  - file sizes
  - file and directory counts
  - modification time
- virtual xattrs present full stats
- 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
```

ceph

snapshots

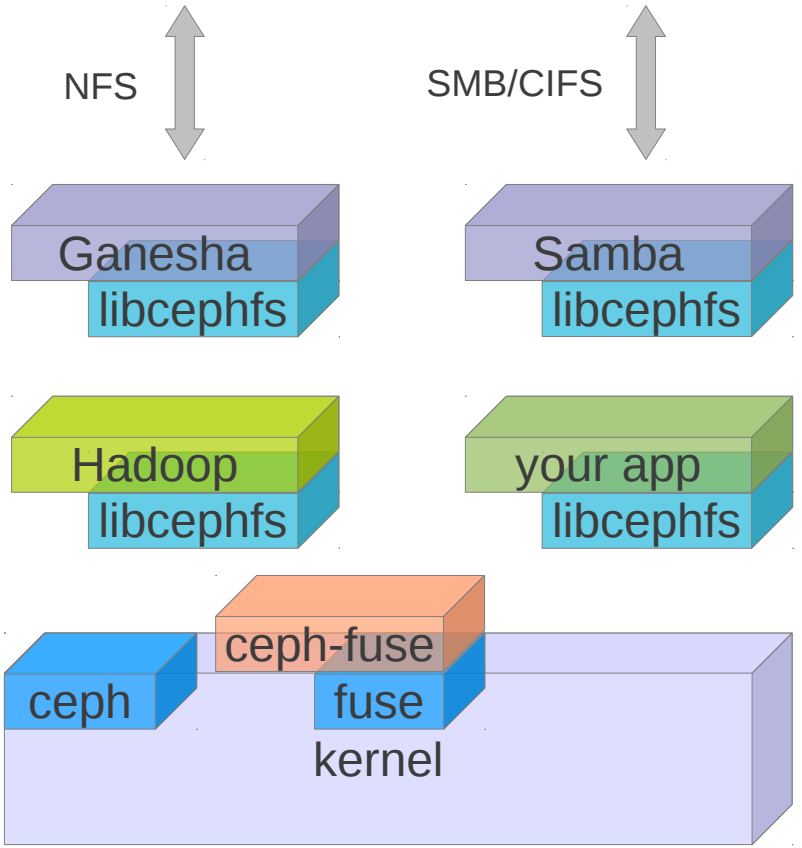
- volume or subvolume snapshots unusable at petabyte scale
  - snapshot arbitrary subdirectories
- simple interface
  - hidden '.snap' directory
  - no special tools

```
$ 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
```

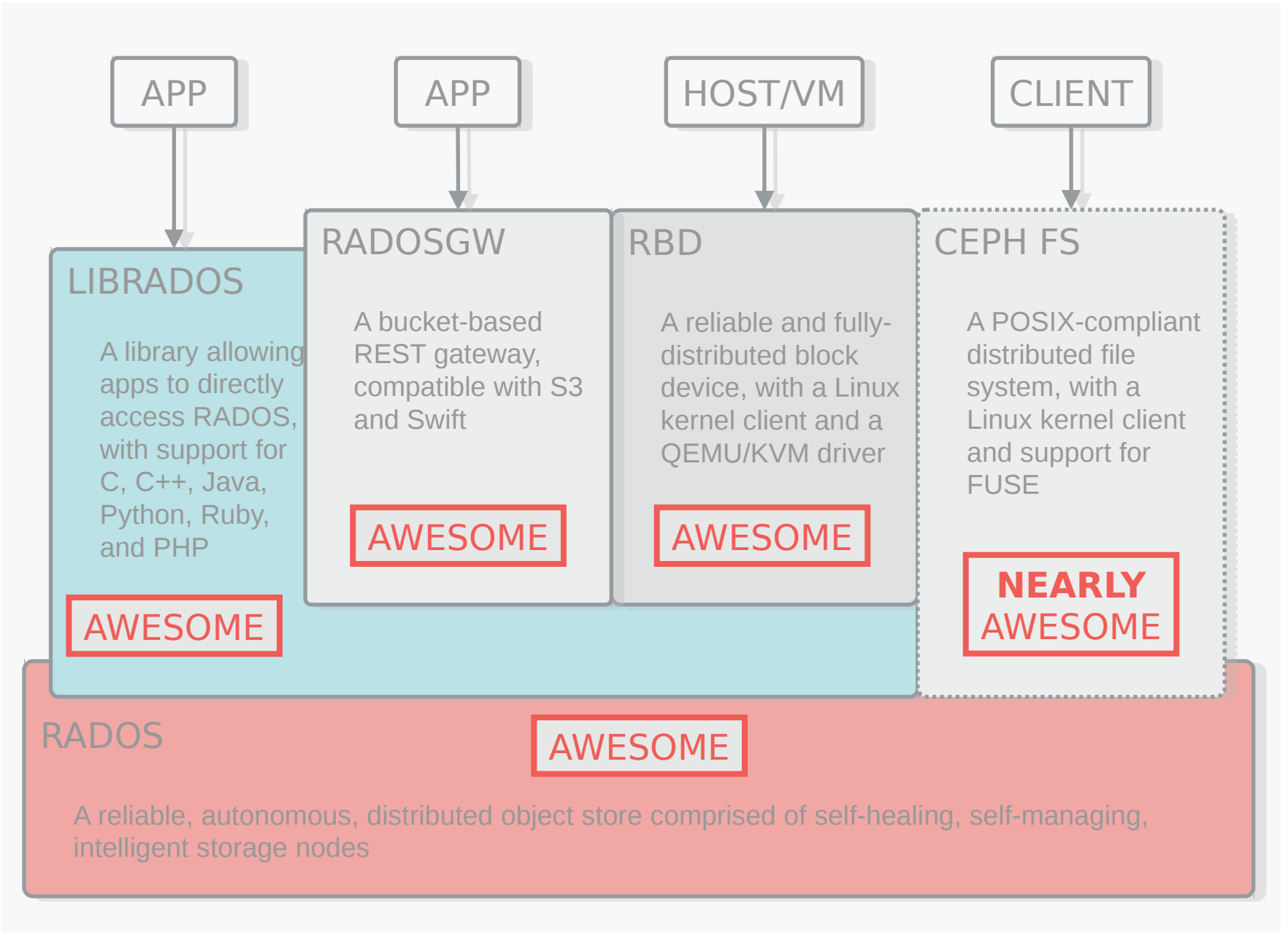
ceph

multiple protocols, implementations

- Linux kernel client
  - mount -t ceph 1.2.3.4:/ /mnt
  - export (NFS), Samba (CIFS)
- ceph-fuse
- libcephfs.so
  - your app
  - Samba (CIFS)
  - Ganesha (NFS)
  - Hadoop (map/reduce)



ceph


APP
APP
HOST/VM
CLIENT
LIBRADOS
A library allowing apps to directly access RADOS, with support for C, C++, Java, Python, Ruby, and PHP
AWESOME
RADOSGW
A bucket-based REST gateway, compatible with S3 and Swift
AWESOME
RBD
A reliable and fully-distributed block device, with a Linux kernel client and a QEMU/KVM driver
AWESOME
CEPH FS
A POSIX-compliant distributed file system, with a Linux kernel client and support for FUSE
NEARLY AWESOME
RADOS
AWESOME
A reliable, autonomous, distributed object store comprised of self-healing, self-managing, intelligent storage nodes

ceph

current status

- argonaut stable release v0.48
  - rados, RBD, radosgw
- bobtail stable release v0.55
  - RBD cloning
  - improved performance, scaling, failure behavior
  - radosgw API, performance improvements
  - freeze in ~1 week, release in ~4 weeks

ceph

roadmap

- file system
  - pivot in engineering focus
  - CIFS (Samba), NFS (Ganesha), Hadoop
- RBD
  - Xen integration, iSCSI
- radosgw
  - Keystone integration
- RADOS
  - geo-replication
  - PG split

ceph

why we do this

- limited options for scalable open source storage
- proprietary solutions
  - expensive
  - don't scale (well or out)
  - marry hardware and software
- users hungry for alternatives
  - scalability
  - cost
  - features

ceph

two fields

- green: cloud, big data
  - incumbents don't have a viable solution
  - most players can't afford to build their own
  - strong demand for open source solutions
- brown: traditional SAN, NAS; enterprise
  - incumbents struggle to scale out
  - can't compete on price with open solutions

ceph

licensing

- <yawn>
- promote adoption
- enable community development
- prevent ceph from becoming proprietary
- allow organic commercialization

ceph

ceph license

- LGPLv2
  - “copyleft”
    - free distribution
    - allow derivative works
    - changes you distribute/sell must be shared
  - ok to link to proprietary code
    - allow proprietary products to incude and build on ceph
    - does not allow proprietary derivatives of ceph

fragmented copyright

- we do not require copyright assignment from contributors
  - no single person or entity owns all of ceph
  - no single entity can make ceph proprietary
- strong community
  - many players make ceph a safe technology bet
  - project can outlive any single business

why its important

- ceph is an ingredient
  - we need to play nice in a larger ecosystem
  - community will be key to ceph's success
- truly open source solutions are disruptive
  - open is a competitive advantage
    - frictionless integration with projects, platforms, tools
    - freedom to innovate on protocols
    - leverage community testing, development resources
    - open collaboration is efficient way to build technology

who we are

- Ceph created at UC Santa Cruz (2004-2007)
- supported by DreamHost (2008-2011)
- Inktank (2012)
  - Los Angeles, Sunnyvale, San Francisco, remote
- growing user and developer community
  - Linux distros, users, cloud stacks, SIs, OEMs

http://ceph.com/

thanks

sage weil

sage@inktank.com

@liewegas

http://github.com/ceph

http://ceph.com/

inktank

ceph

why we like btrfs

- pervasive checksumming
- snapshots, copy-on-write
- efficient metadata (xattrs)
- inline data for small files
- transparent compression
- integrated volume management
  - software RAID, mirroring, error recovery
  - SSD-aware
- online fsck
- active development community

ceph