Ceph Administration (CP-ADM)

Keywords

SDS, Ceph, RADOS, RGW, RBD,
CephFS, Mon, OSD, CRUSH, PG, Pool

References

- Ceph Documentation: http://docs.ceph.com
- Ceph Cookbook Karan Singh, 2016
- Ceph Cookbook 2nd Edition– Vikhyat Umrao, Michael Hackett, Karan Singh, 2017

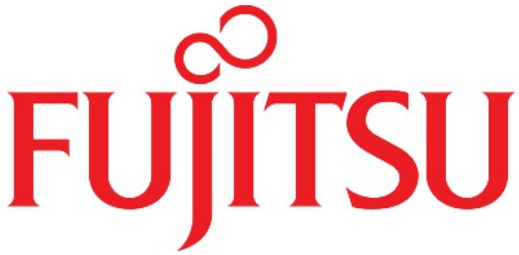
Software Defined Storage (SDS)

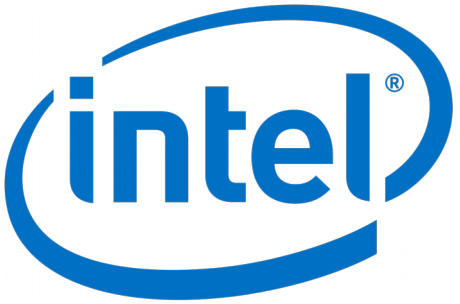
Unified object, block and file storage cluster system

Why Ceph?

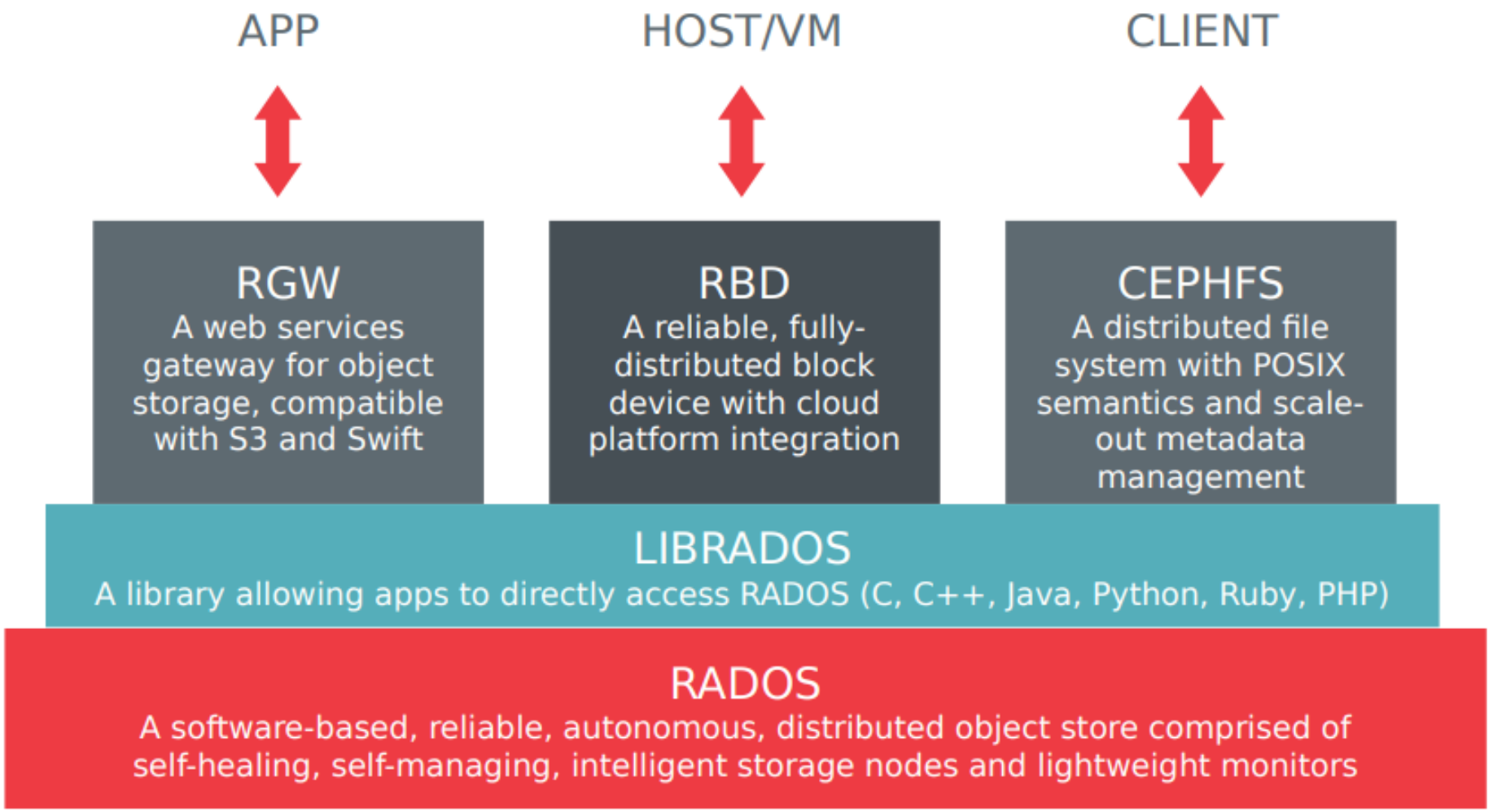
- Open Source Software Defined Storage
- Unified Storage Platform (Block , Object and File Storage)
- Runs on Commodity Hardware
- Self Managing, Self Healing
- Massively Scalable
- No Single Point of failure
- Designed for cloud infrastructure and emerging workloads
- Awesome release names

Contributors

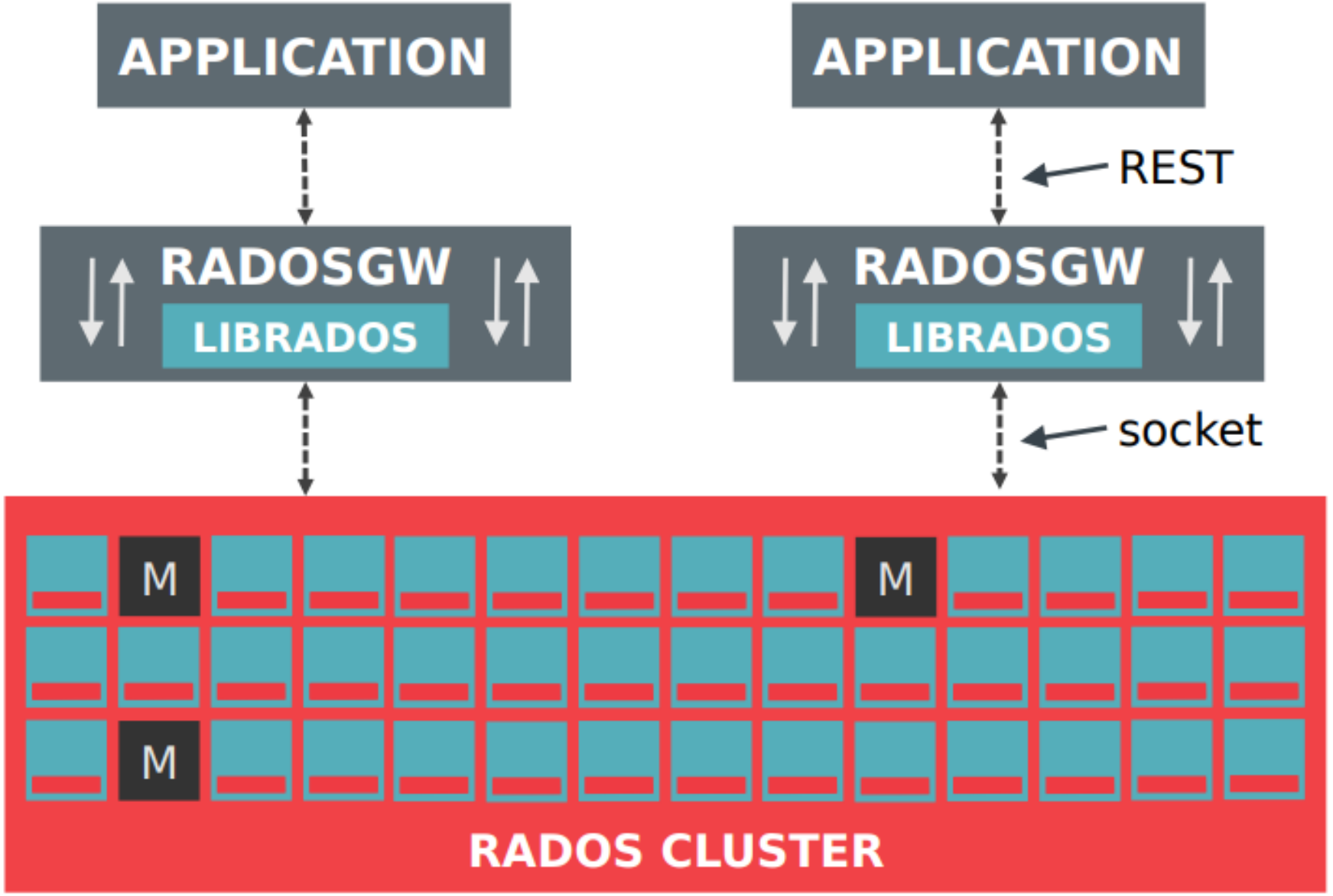




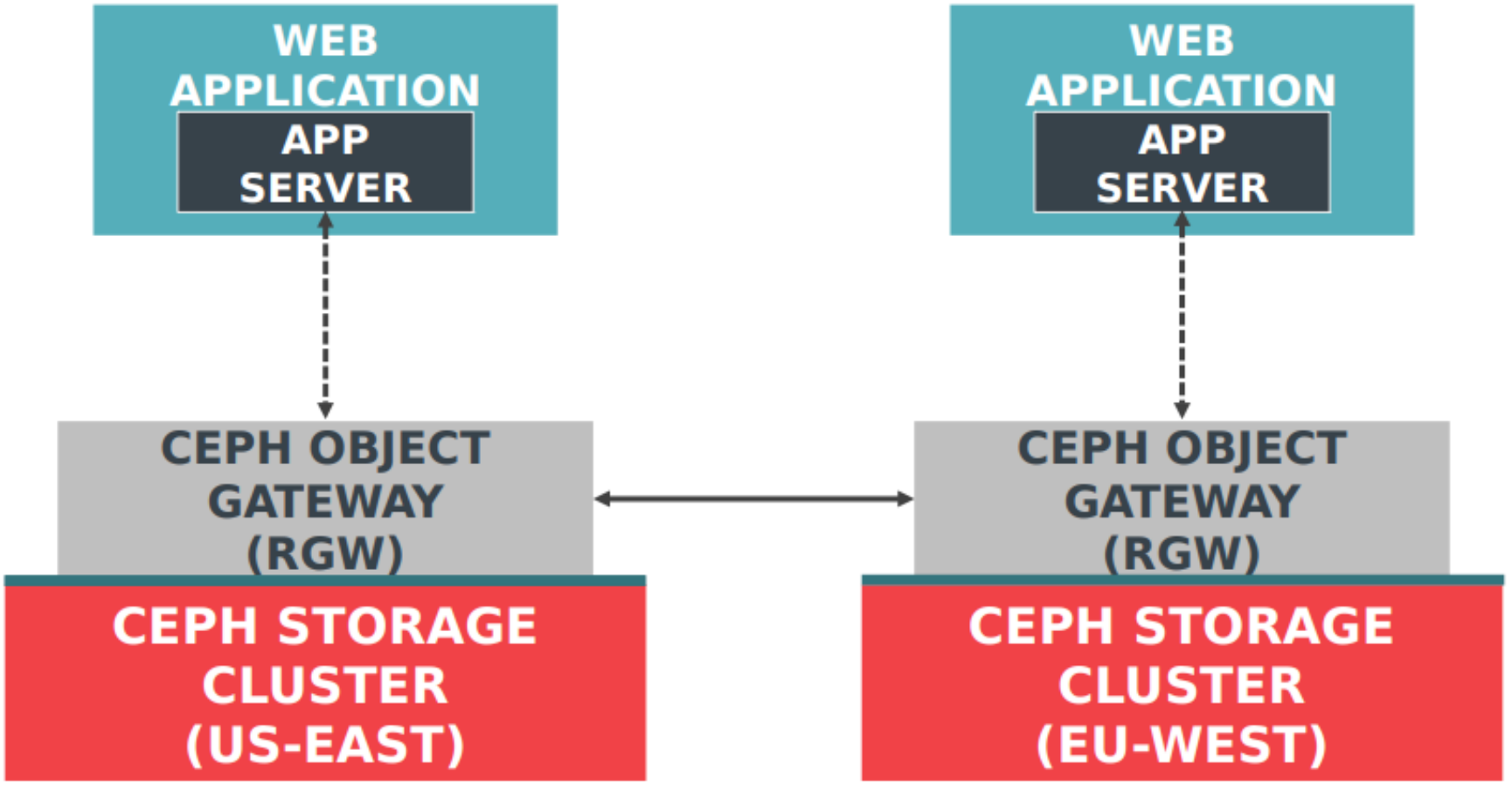
Ceph Component



Rados Gateway



Multi-Side Object Storage

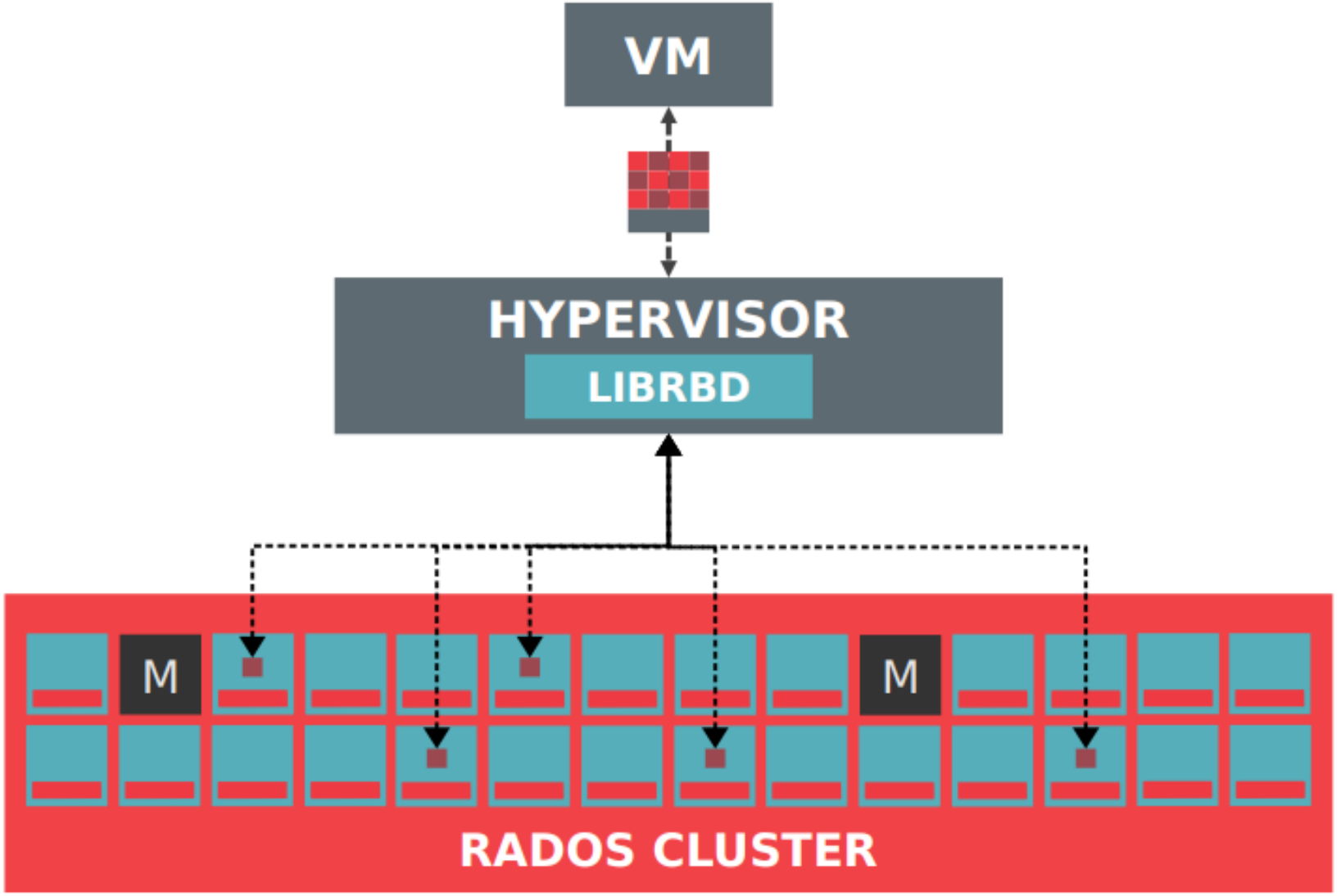


RADOSGW

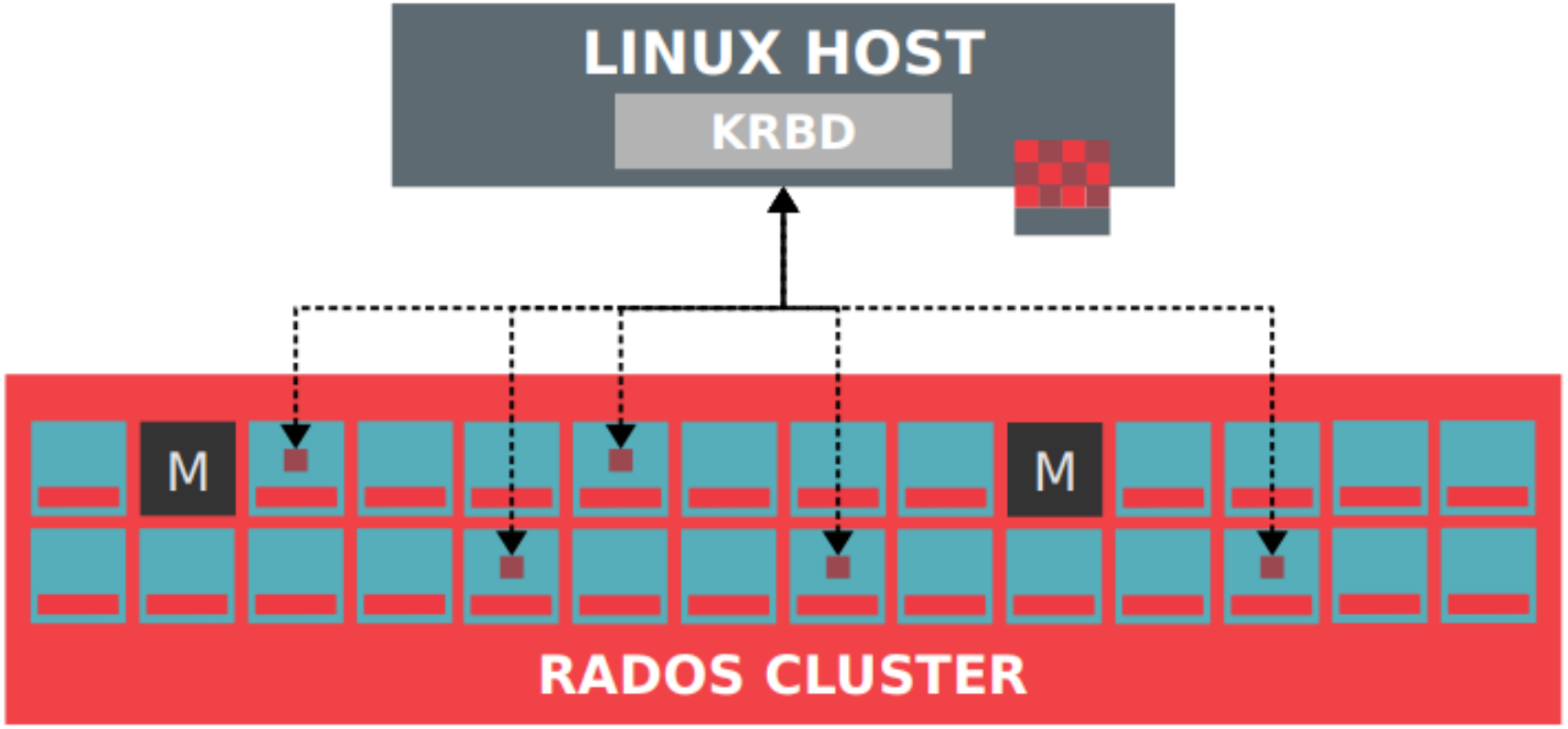
RADOSGW:

- REST-based object storage proxy
- Uses RADOS to store objects
  - Stripes large RESTful objects across many RADOS objects
- API supports buckets, accounts
- Usage accounting for billing
- Compatible with S3 and Swift applications

Rados Block Device



Kernel Module

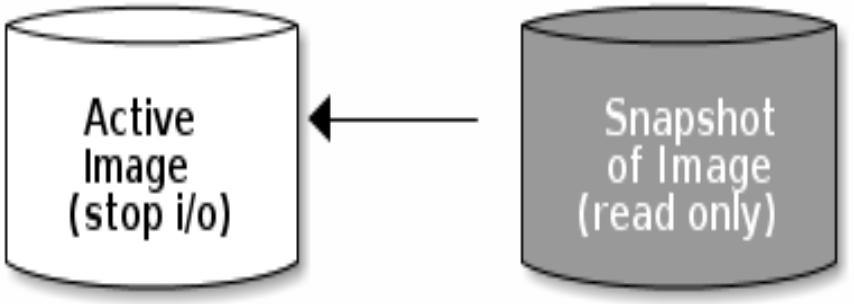


RBD

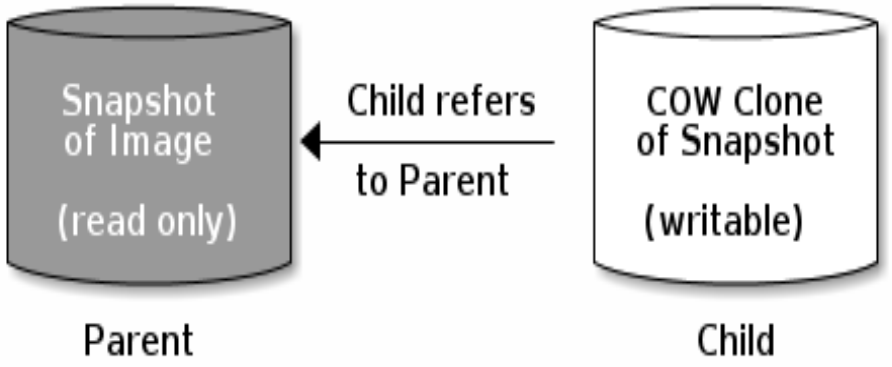
- Stripe images across entire cluster (pool)
- Read-only snapshots
- Copy-on-write clones
- Broad integration
  - Qemu
  - Linux kernel
  - iSCSI (STGT, LIO)
  - OpenStack, CloudStack, Nebula, Ganeti, Proxmox
- Incremental backup (relative to snapshots)

Snapshot & Layering (Cloning)

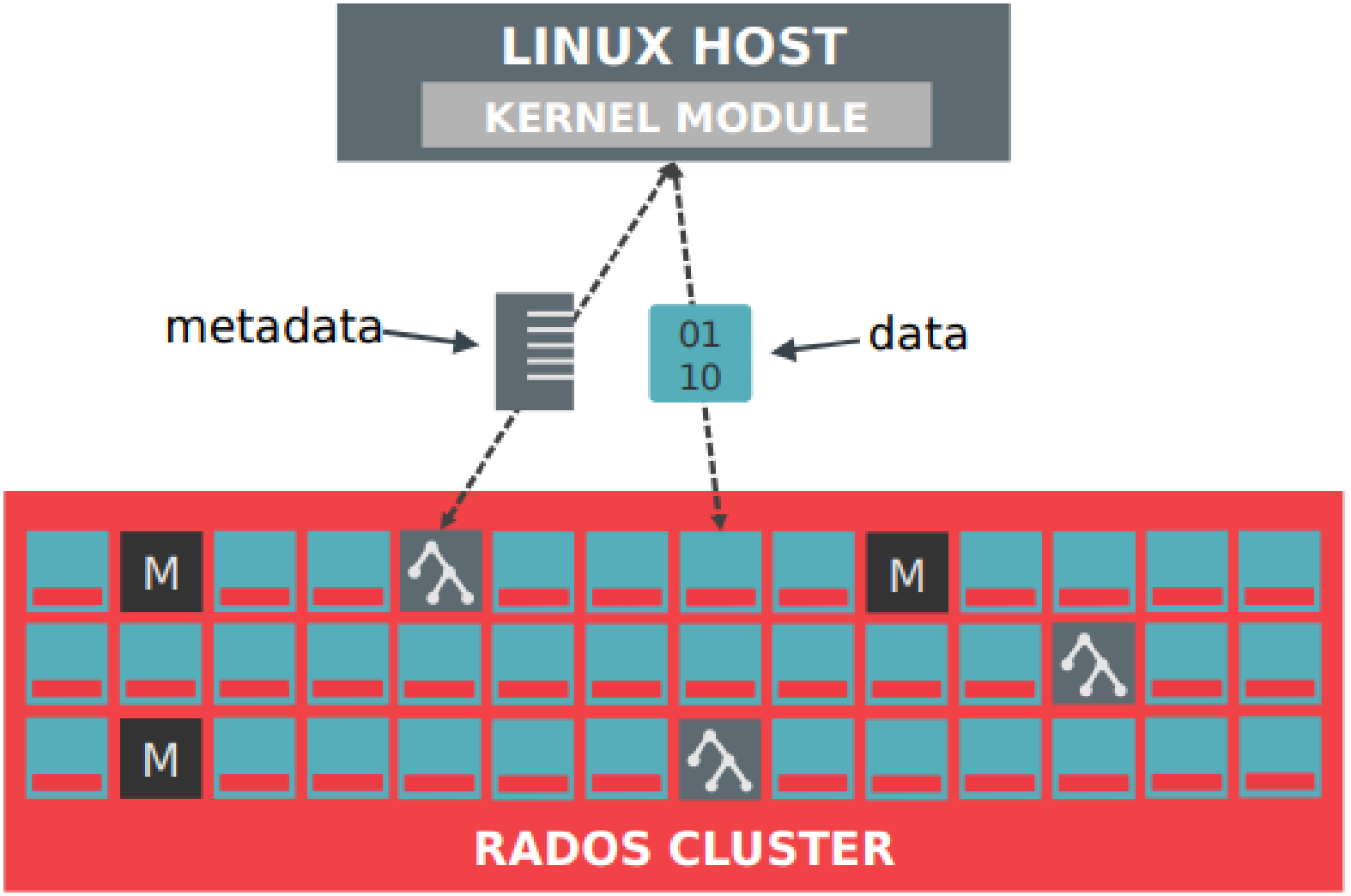
A snapshot is a read-only copy of the state of an image at a particular point in time.



Ceph supports the ability to create many copy-on-write (COW) clones of a block device shapshot.



Ceph File System



CephFS

METADATA SERVER

- Manages metadata for a POSIX-compliant shared filesystem
  - Directory hierarchy
  - File metadata (owner, timestamps, mode, etc.)
  - Snapshots on any directory
- Clients stripe file data in RADOS
  - MDS not in data path
- MDS stores metadata in RADOS
- **Dynamic MDS cluster** scales to 10s or 100s
- Only required for shared filesystem

Component Overview

- **RADOS Gateway Interface (RGW)**: RGW provides object storage service. It uses librgw (the Rados Gateway Library) and librados, allowing applications to establish connections with the Ceph object storage. The RGW provides RESTful APIs with interfaces that are compatible with Amazon S3 and OpenStack Swift.

- **RADOS Block Devices (RBDs)**: RBDs, which are now known as the Ceph block device, provide persistent block storage, which is thin-provisioned, resizable, and stores data striped over multiple OSDs. The RBD service has been built as a native interface on top of librados

Component Overview (2)

- **CephFS**: The Ceph filesystem provides a POSIX-compliant filesystem that uses the Ceph storage cluster to store user data on a filesystem. Like RBD and RGW, the CephFS service is also implemented as a native interface to librados.

Ceph Nodes

OSDs:

- 10s to 1000s in a cluster
- One per disk (or one per SSD, RAID group…)
- Serve stored objects to clients
- Intelligently peer for replication & recovery

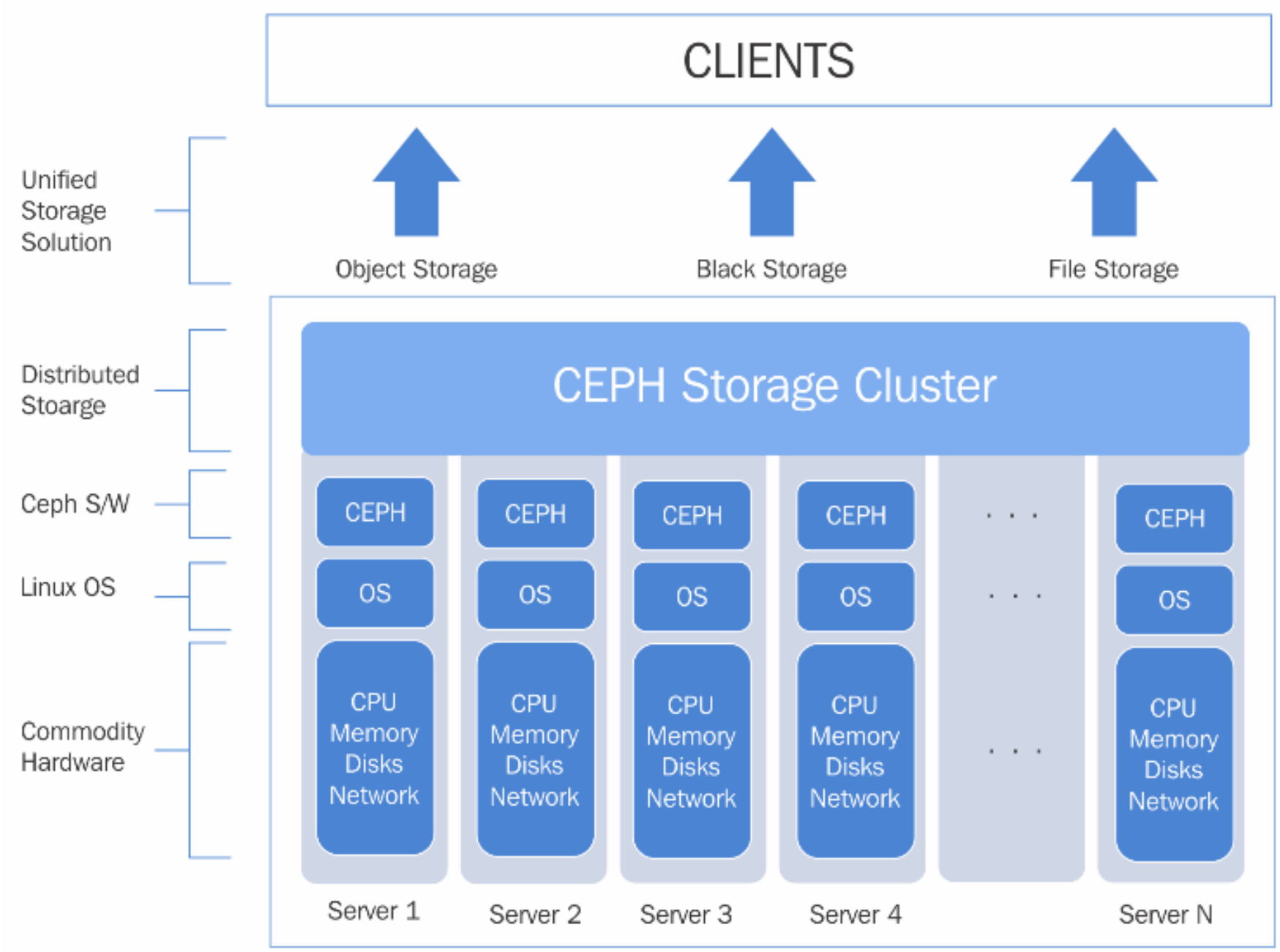
Monitors:

- Maintain cluster membership and state
- Provide consensus for distributed decision-making
- Small, odd number (e.g., 5)
- Not part of data path

Ceph Nodes (2)

- **Ceph object storage device (OSD)**: Stores data, replicate data, recovery, backfilling, rebalancing, check other OSD & info to Monitor. Make 3 copies of data by default.
- **Ceph monitors (MON)**: maintains maps of the cluster state (monitor map, OSD map, PG map, CRUSH map)
- **Ceph metadata server (MDS)**: store metadata on behalf of the CephFS. Make POSIX file system to execute basic commands like ls, find, etc.

Planning a Ceph deployment



Ceph Min H/W Requirements

Process	Criteria	Minimum Recommended
`ceph-osd`	Processor	• 1x 64-bit AMD-64 • 1x 32-bit ARM dual-core or better • 1x i386 dual-core
	RAM	~1GB for 1TB of storage per daemon
	Volume Storage	1x storage drive per daemon
	Journal	1x SSD partition per daemon (optional)
	Network	2x 1GB Ethernet NICs
`ceph-mon`	Processor	• 1x 64-bit AMD-64/i386 • 1x 32-bit ARM dual-core or better • 1x i386 dual-core
	RAM	1 GB per daemon
	Disk Space	10 GB per daemon
	Network	2x 1GB Ethernet NICs
`ceph-mds`	Processor	• 1x 64-bit AMD-64 quad-core • 1x 32-bit ARM quad-core • 1x i386 quad-core
	RAM	1 GB minimum per daemon
	Disk Space	1 MB per daemon
	Network	2x 1GB Ethernet NICs

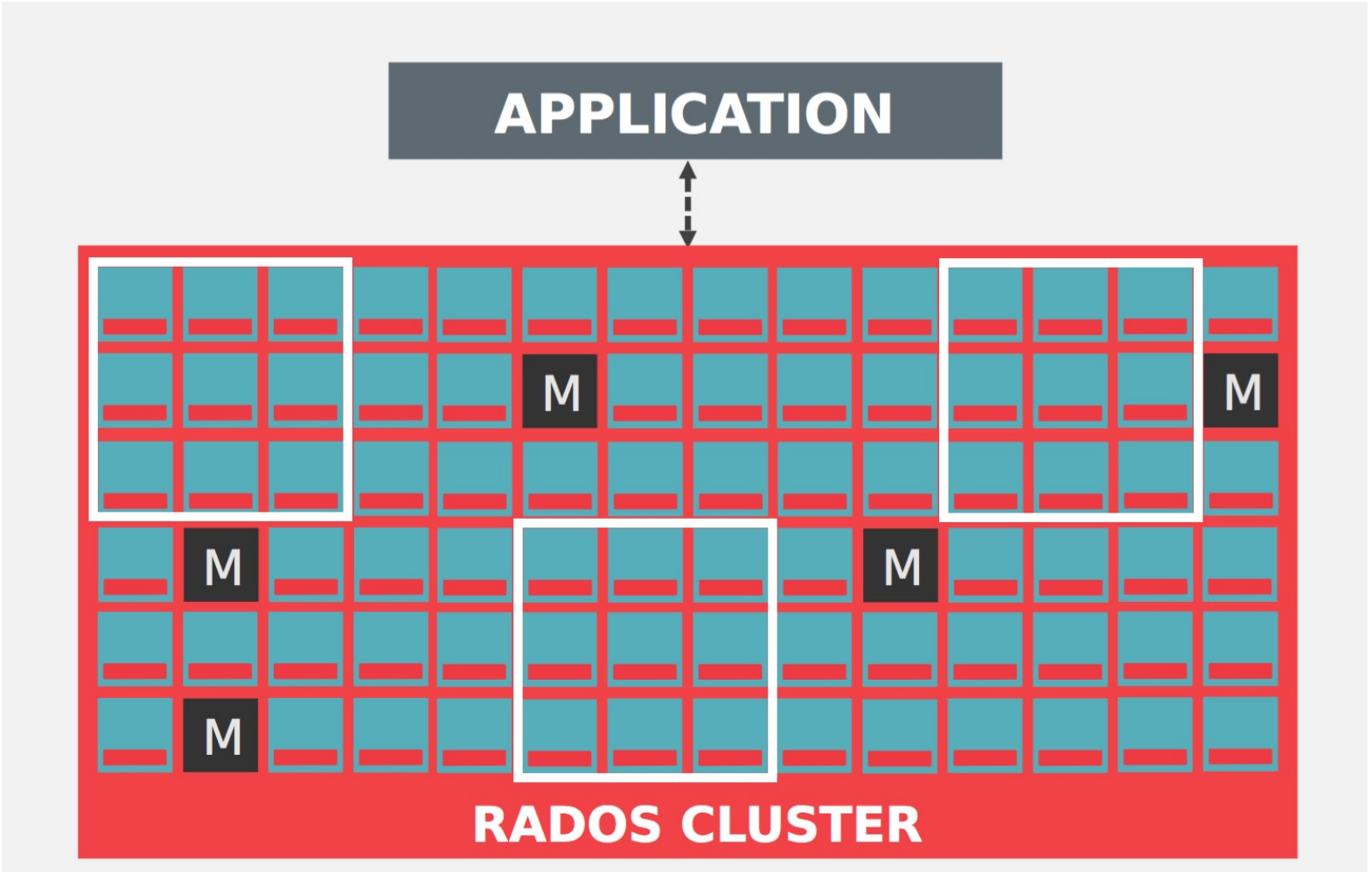
Ceph Releases

Ceph release name	Ceph release version	Released On
Argonaut	V0.48 (LTS)	July 3, 2012
Bobtail	V0.56 (LTS)	January 1, 2013
Cuttlefish	V0.61	May 7, 2013
Dumpling	V0.67 (LTS)	August 14, 2013
Emperor	V0.72	November 9, 2013
Firefly	V0.80 (LTS)	May 7, 2014
Giant	V0.87.1	Feb 26, 2015
Hammer	V0.94 (LTS)	April 7, 2015
Infernalis	V9.0.0	May 5, 2015
Jewel	V10.0.0 (LTS)	Nov, 2015
Kraken	V11.0.0	June 2016
Luminous	V12.0.0 (LTS)	Feb 2017

Here is a fact: Ceph release names follow an alphabetic order; the next one will be an *M* release. The term *Ceph* is a common nickname given to pet octopuses and is considered a short form of *Cephalopod*, which is a class of marine animals that belong to the mollusk phylum. Ceph has octopuses as its mascot, which represents Ceph's highly parallel behavior, similar to octopuses. http://docs.ceph.com/docs/master/releases/schedule/

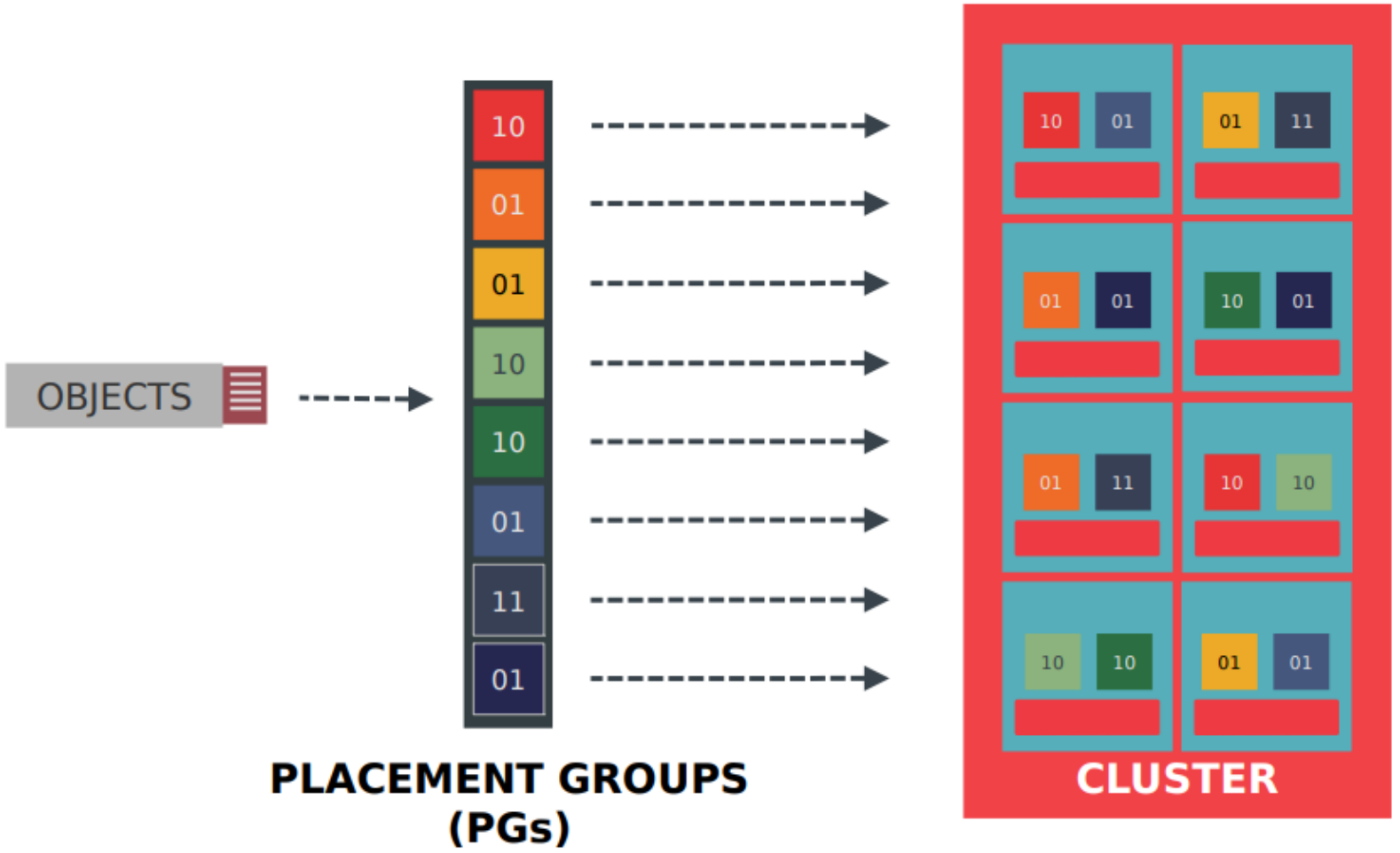
Pools

- Ceph stores data within pools, which are logical groups for storing objects. Pools manage the number of placement groups, the number of replicas, and the ruleset for the pool.



Placement Groups (PG)

- Placement groups (PGs) are shards or fragments of a logical object pool that place objects as a group into OSDs. Placement groups reduce the amount of per-object metadata when Ceph stores the data in OSDs.

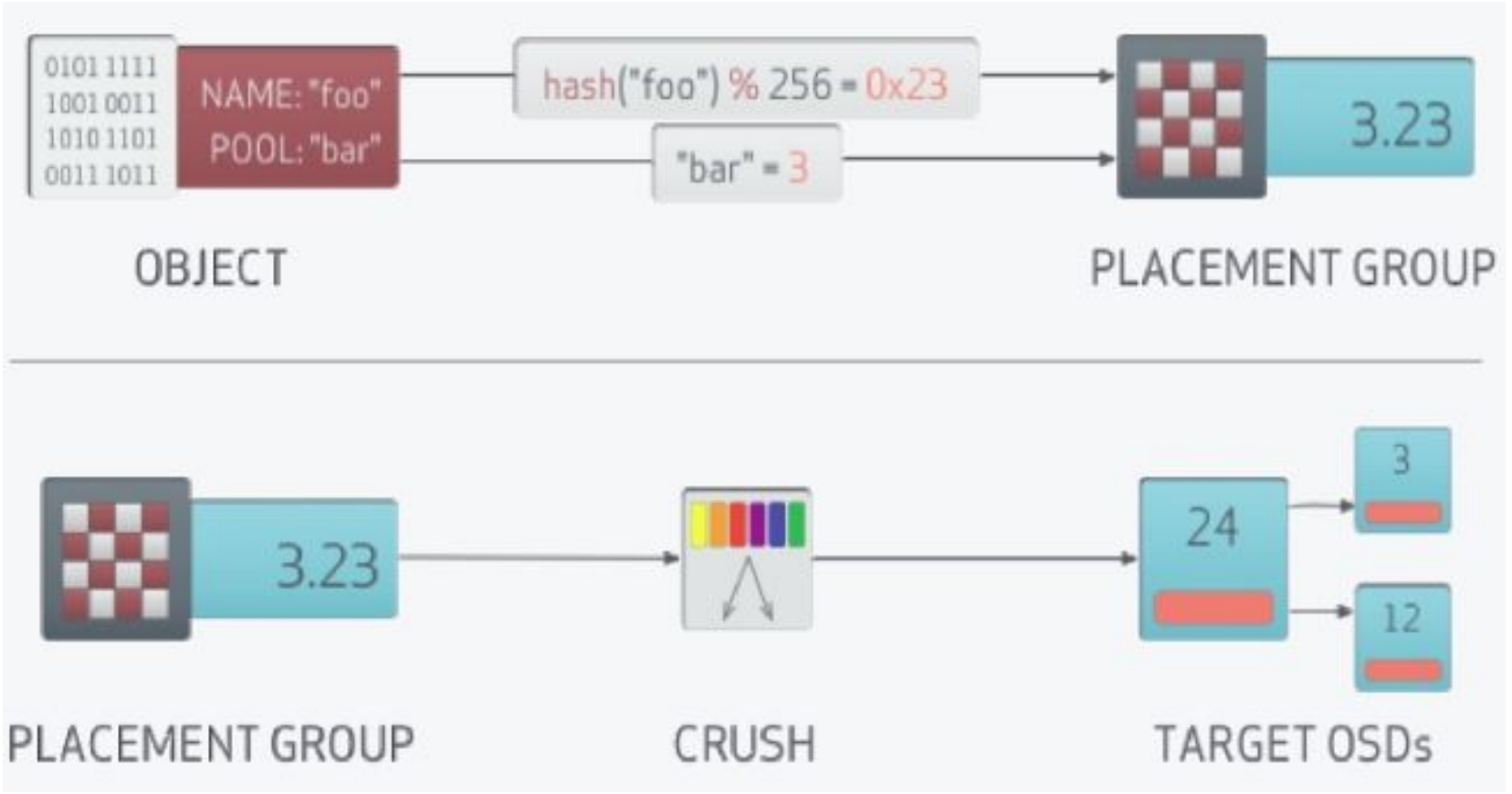


PG Number

- < 5 OSDs, pg_num = 128
- 5 – 10 OSDs, pg_num = 512
- 10 – 50 OSDs, pg_num = 4096
- PGCalc: http://ceph.com/pgcalc/

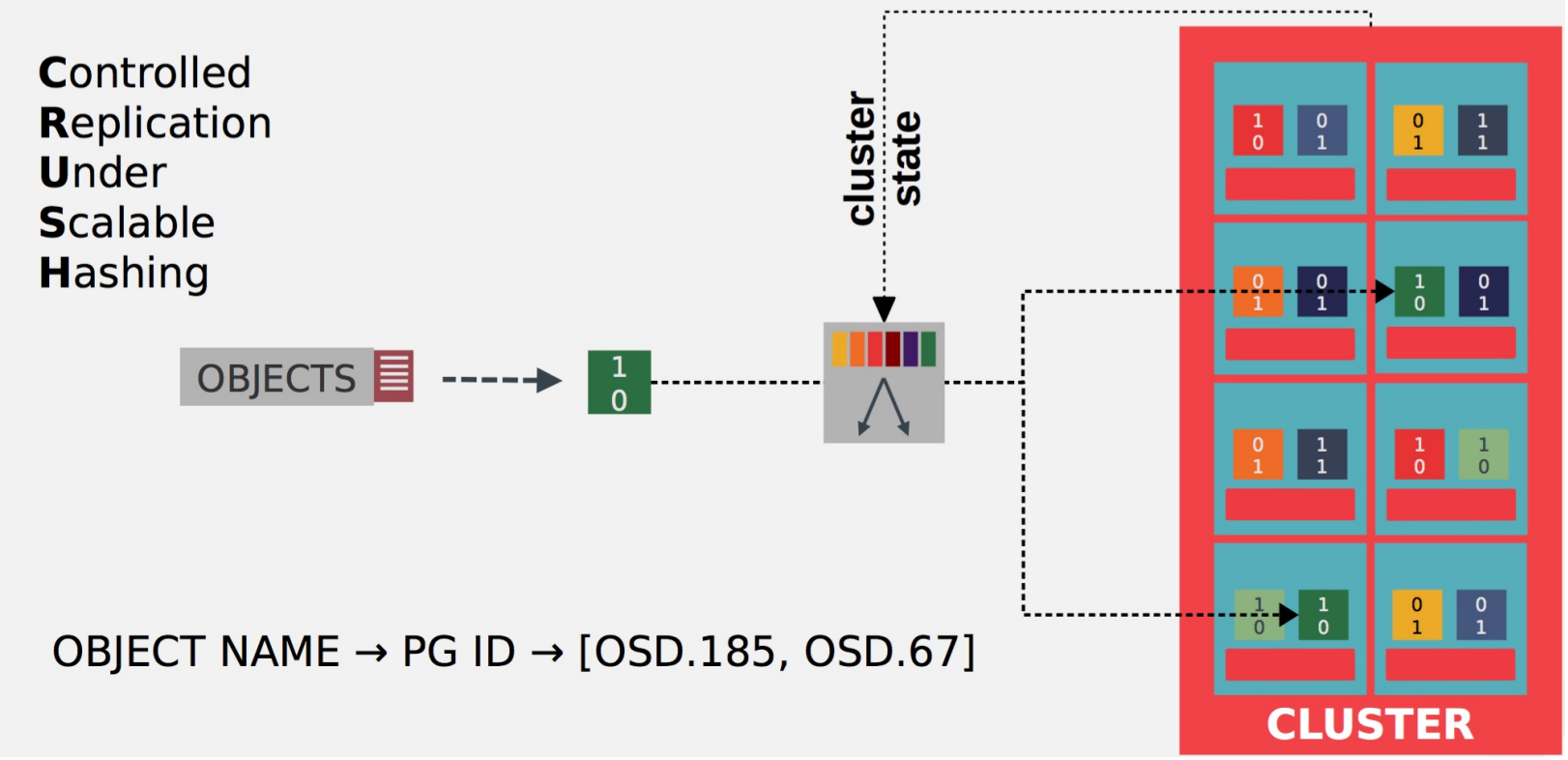
How it Work

- Ceph client first **hashes** the **object name** and splits it into a number of **Placement Groups(pg's)**. Placement groups are just split based on names.



CRUSH

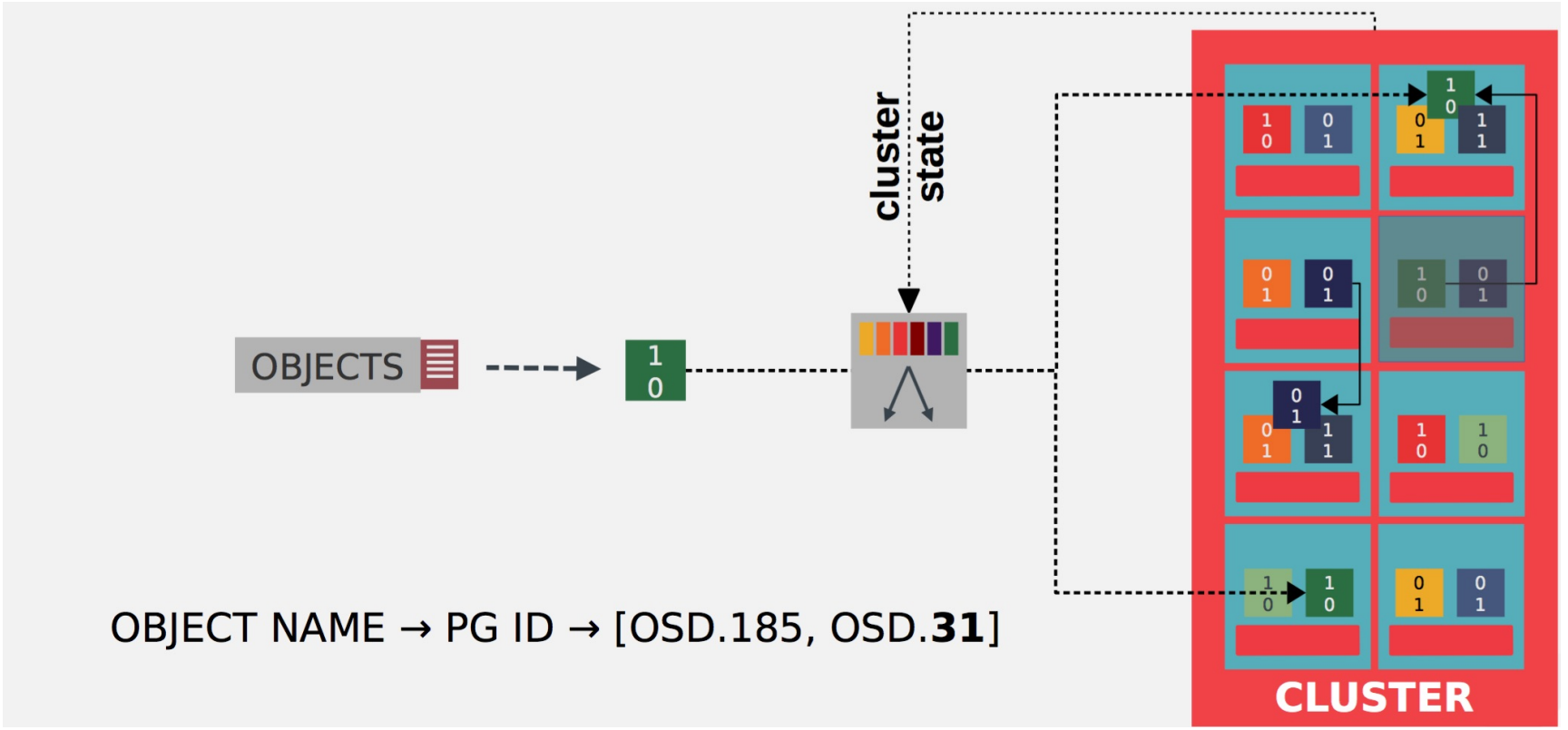
- CRUSH is a big part of what allows Ceph to scale without performance bottlenecks, without limitations to scalability, and without a single point of failure. CRUSH maps provide the physical topology of the cluster to the CRUSH algorithm to determine where the data for an object and its replicas should be stored, and how to do so across failure domains for added data safety among other things.



CRUSH Main Advantages

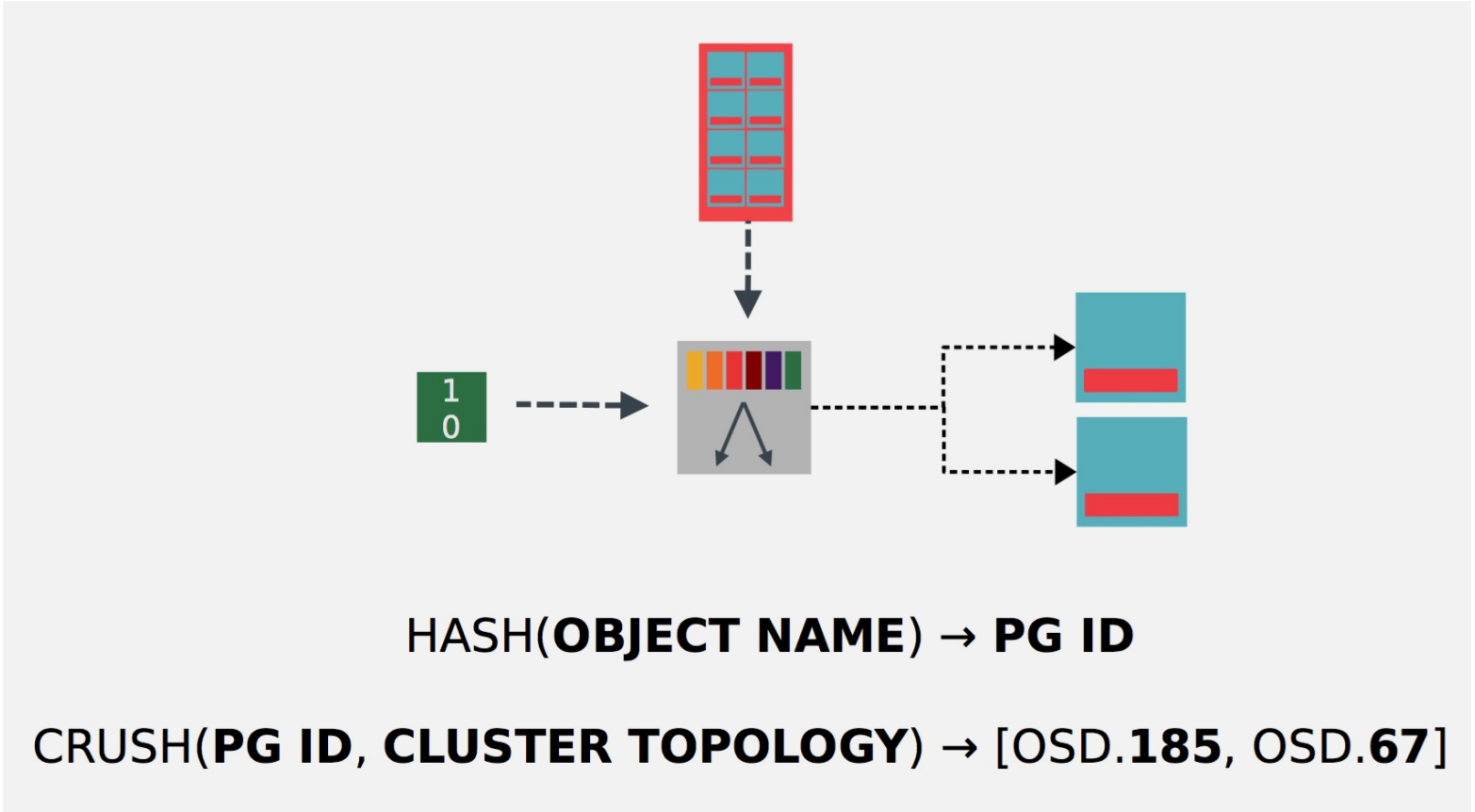
- **Avoid failed devices**: CRUSH take care of the data placement itself and can manage where to place object on the device.

  In this example, blue and green objects are relocate to others OSDs



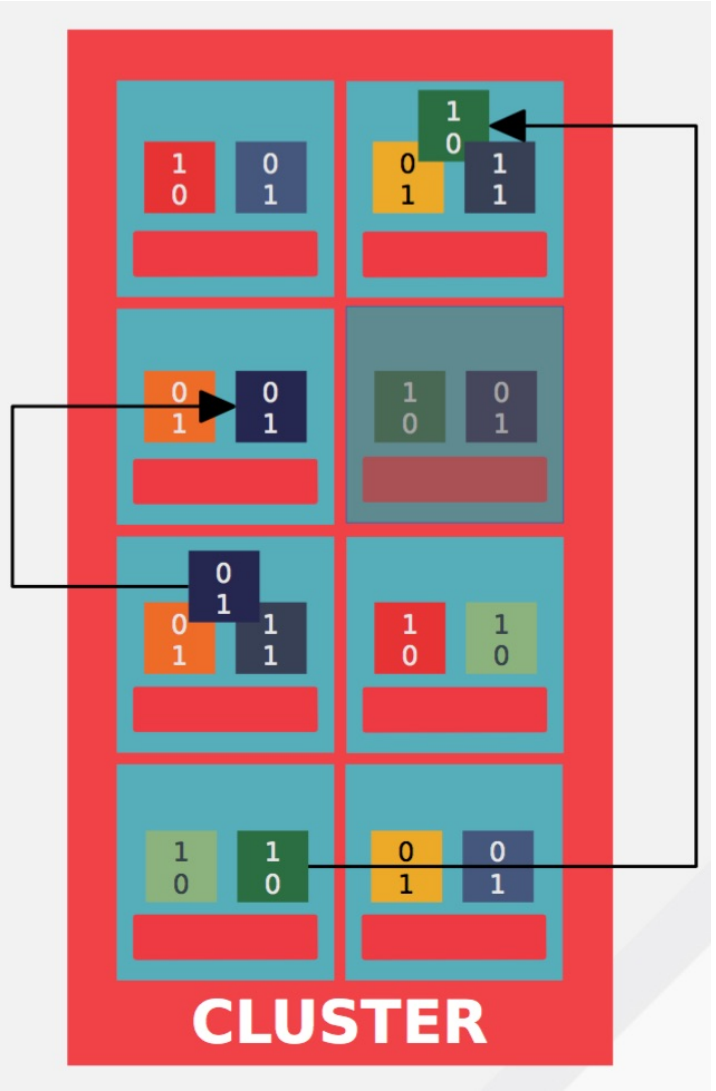
CRUSH Main Advantages (2)

- **Works as a function**: CRUSH allows clients to communicate directly with storage devices without the need for a central index server to manage data object locations, Ceph clusters can store and retrieve data very quickly and scale up or down quite easily.



CRUSH Main Advantages (3)

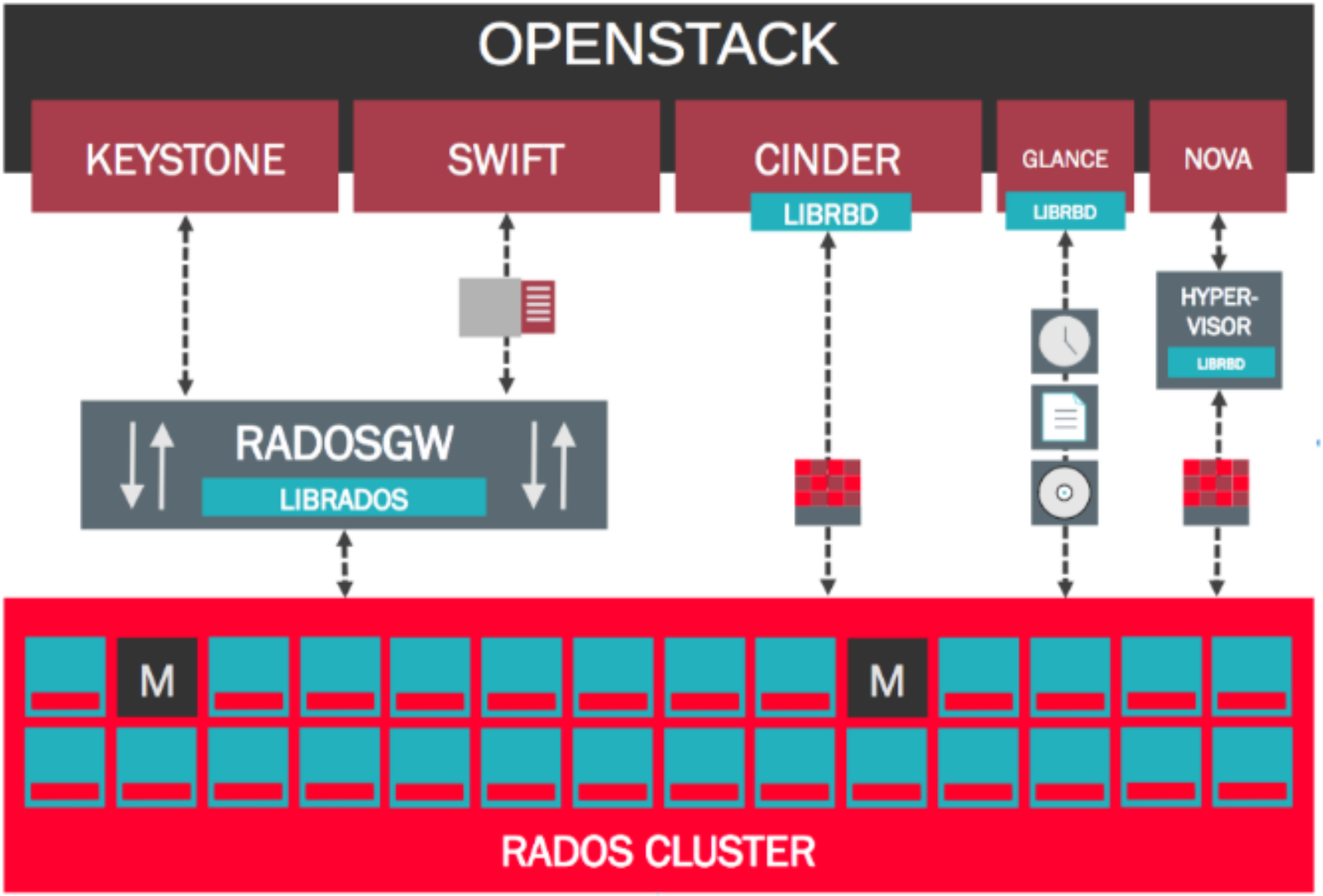
- **Pseudo-random placement**: CRUSH generates a declustered distribution of replicas in that the set of devices sharing replicas for one item also appears to be independent of all other items.



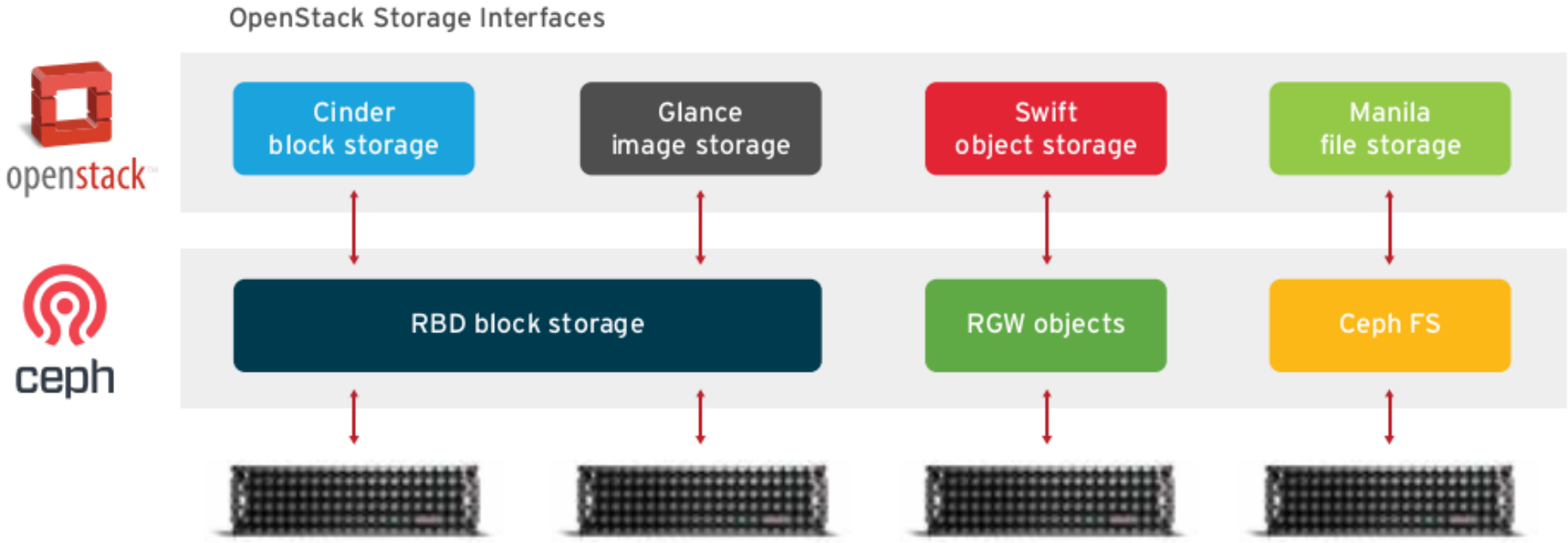
Ceph Cluster Maps

- **Monitor Map**: Contains the cluster fsid, the position, name address and port of each monitor.
- **OSD Map**: Contains the cluster fsid, when the map was created and last modified, a list of pools, replica sizes, PG numbers, a list of OSDs and their status (e.g., up, in).
- **PG Map**: Contains the PG version, its time stamp, the last OSD map epoch, the full ratios, and details on each placement group such as the PG ID, the Up Set, the Acting Set, the state of the PG (e.g., active + clean), and data usage statistics for each pool.
- **CRUSH Map**: Contains a list of storage devices, the failure domain hierarchy (e.g., device, host, rack, row, room, etc.), and rules for traversing the hierarchy when storing data.
- **MDS Map**: Contains the current MDS map epoch, when the map was created, and the last time it changed. It also contains the pool for storing metadata, a list of metadata servers, and which metadata servers are up and in.

Ceph and OpenStack



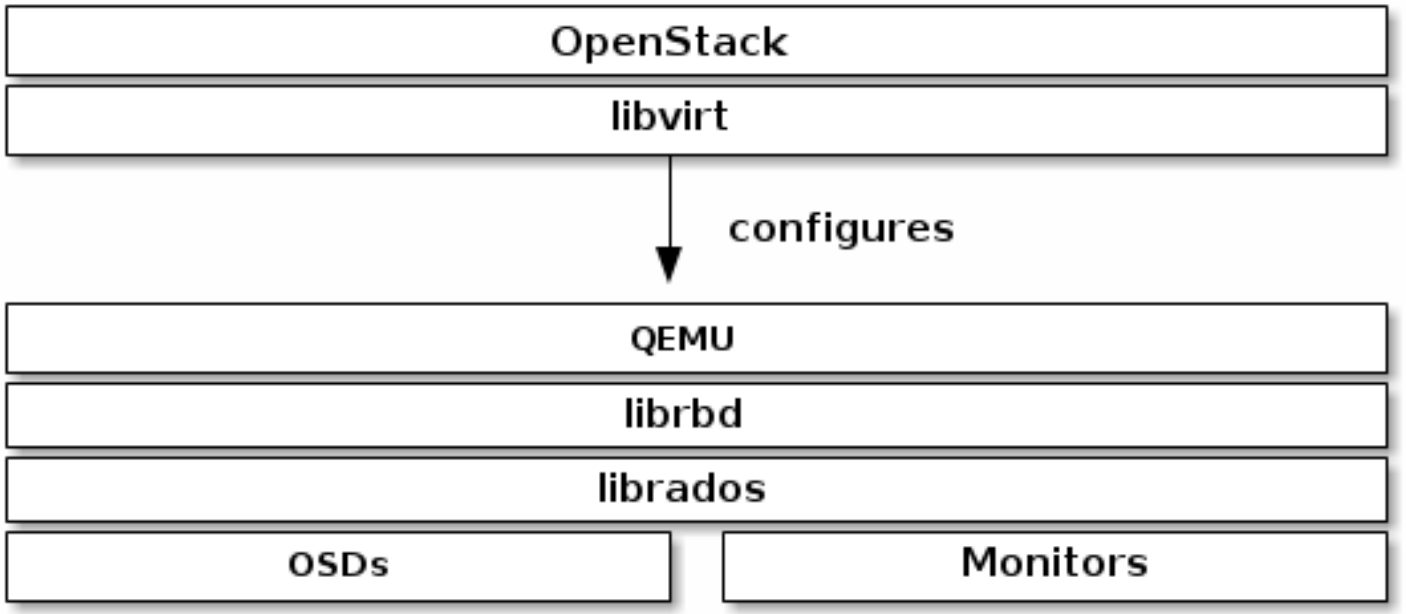
Ceph and OpenStack (2)



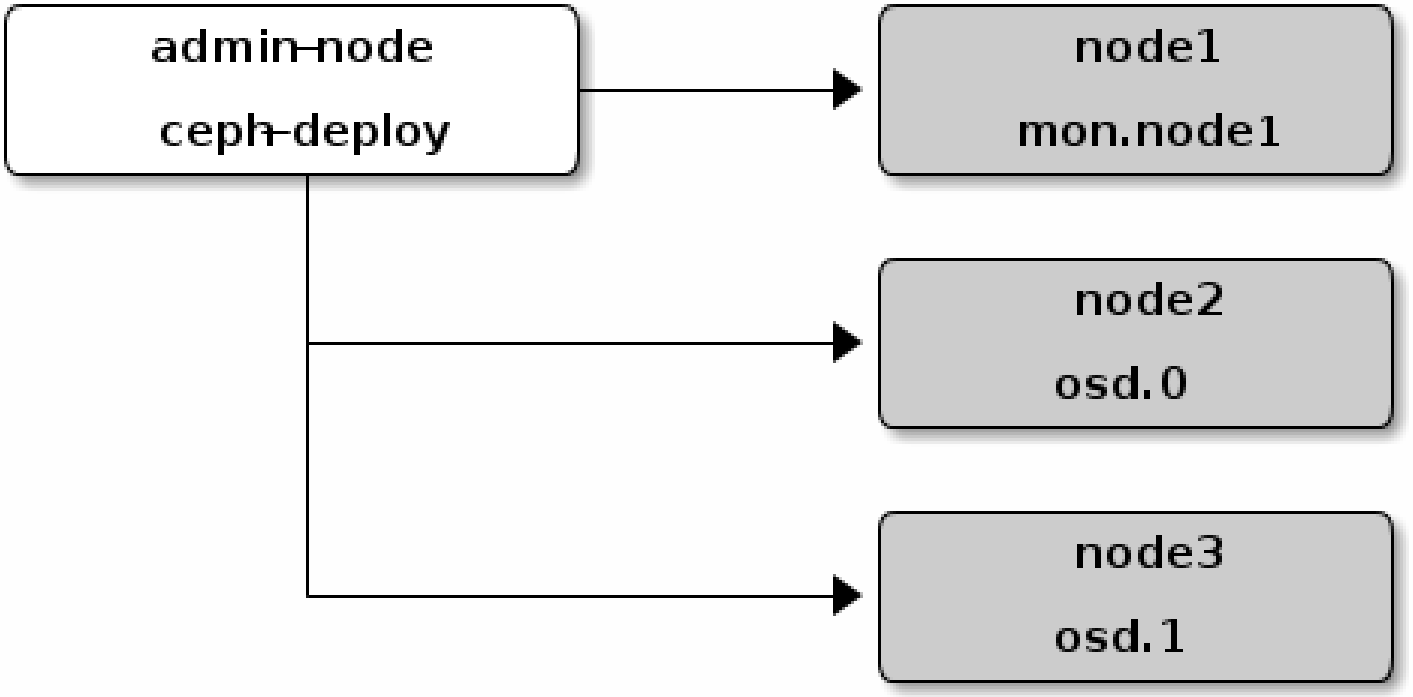
Ceph and OpenStack

Ceph Block Device (RBD):

- Glance images
- Cinder volumes
- Instance disks



Ceph-deploy Utility



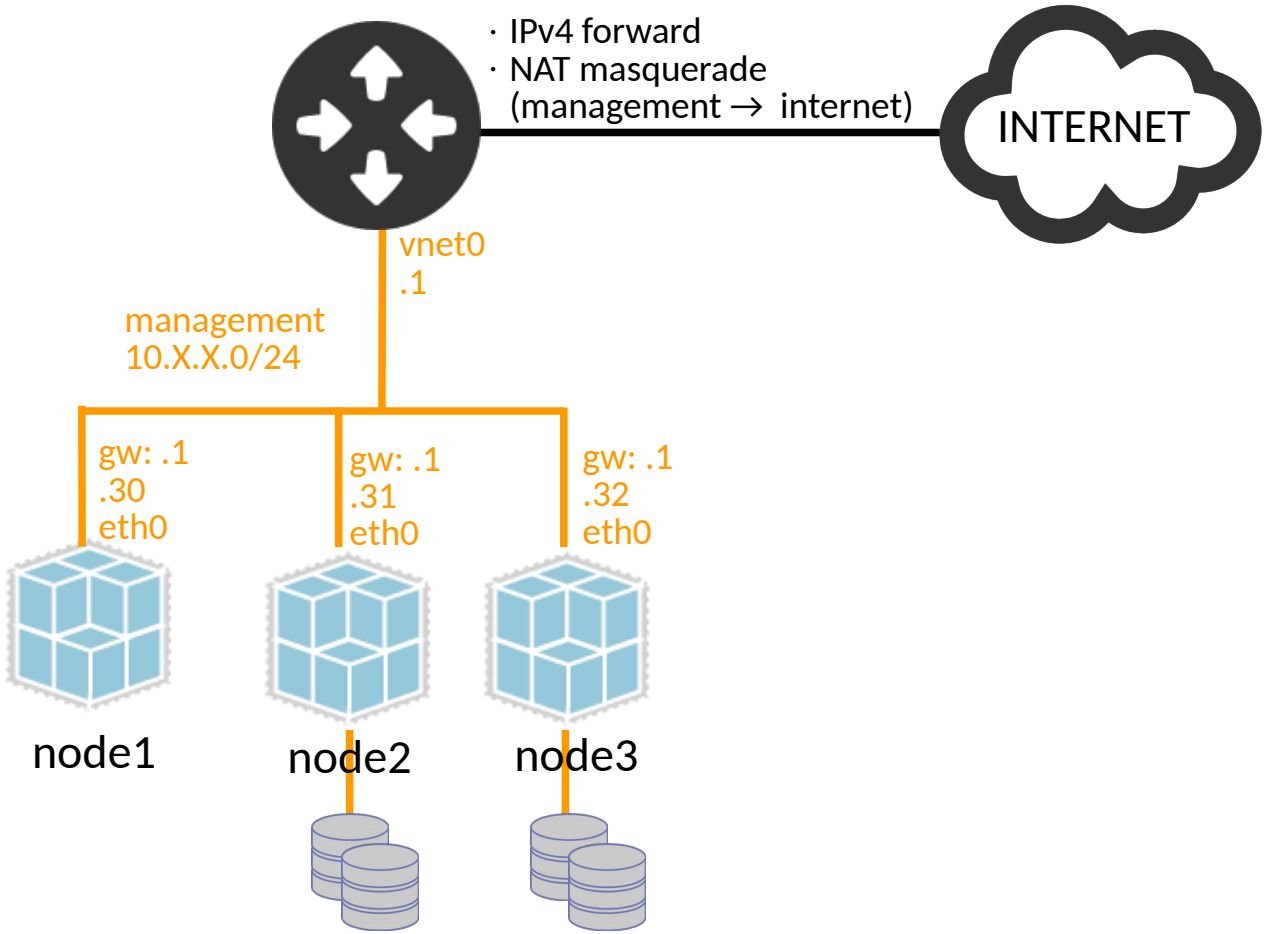
Ceph Preflight Checklist

- Repository: https://download.ceph.com. Mirrors is available
- NTP
- SSH Server with password-less access
- Ceph Deploy User, sudo without TTY
- Ceph Monitor port 6789/tcp.
- Ceph OSDs port 6800:7300/tcp
- SELinux Permissive

Lab

Ceph Cluster

Lab Topology



NolSatu.id