20 ΧΡΟΝΙΑ ΕΔΕΤ

Συμπόσιο Ψηφιακής Τεχνολογίας

Νέες Διαστάσεις
στην Έρευνα & Εκπαίδευση

grnet

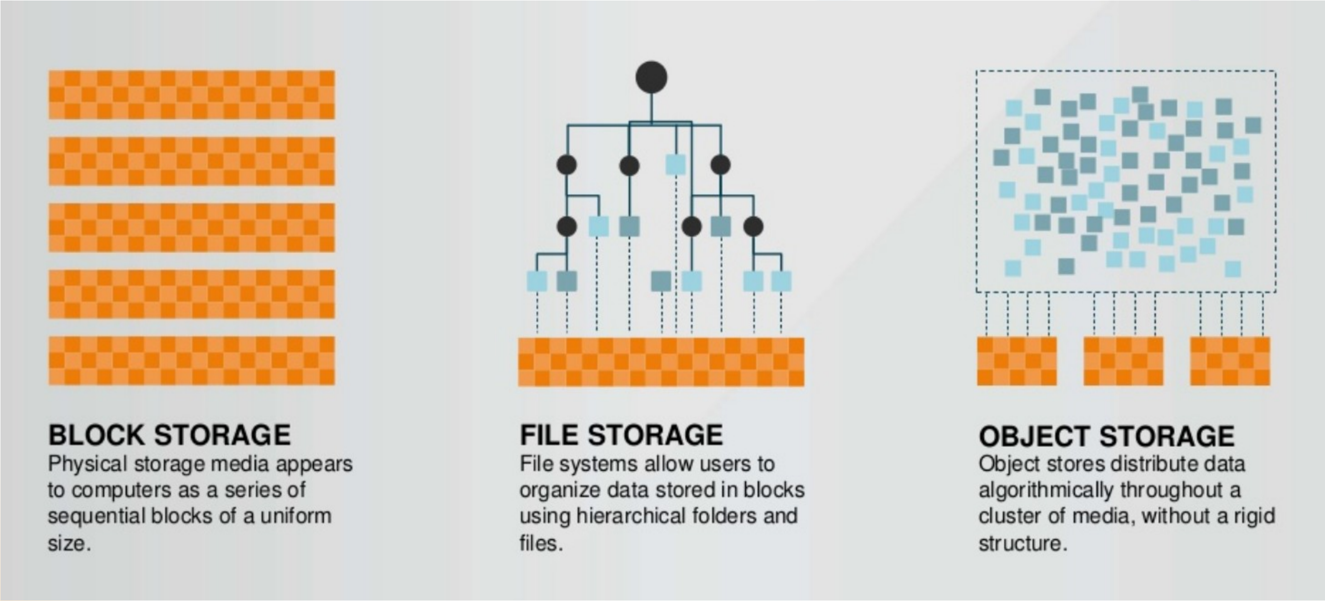
Open Source Storage at Scale: Ceph @ GRNET

Nikos Kormpakis – nkorb@noc.grnet.gr

Object Storage

Independent, unique entities including data, plus some metadata.

- Simple without hierarchy
- Useful for unstructured data
- Abstract lower layers (blocks, files, sectors)
- Let “users” create applications on top of it



BLOCK STORAGE
Physical storage media appears to computers as a series of sequential blocks of a uniform size.

FILE STORAGE
File systems allow users to organize data stored in blocks using hierarchical folders and files.

OBJECT STORAGE
Object stores distribute data algorithmically throughout a cluster of media, without a rigid structure.

Source: https://www.slideshare.net/alohamora/ceph-introduction-2017

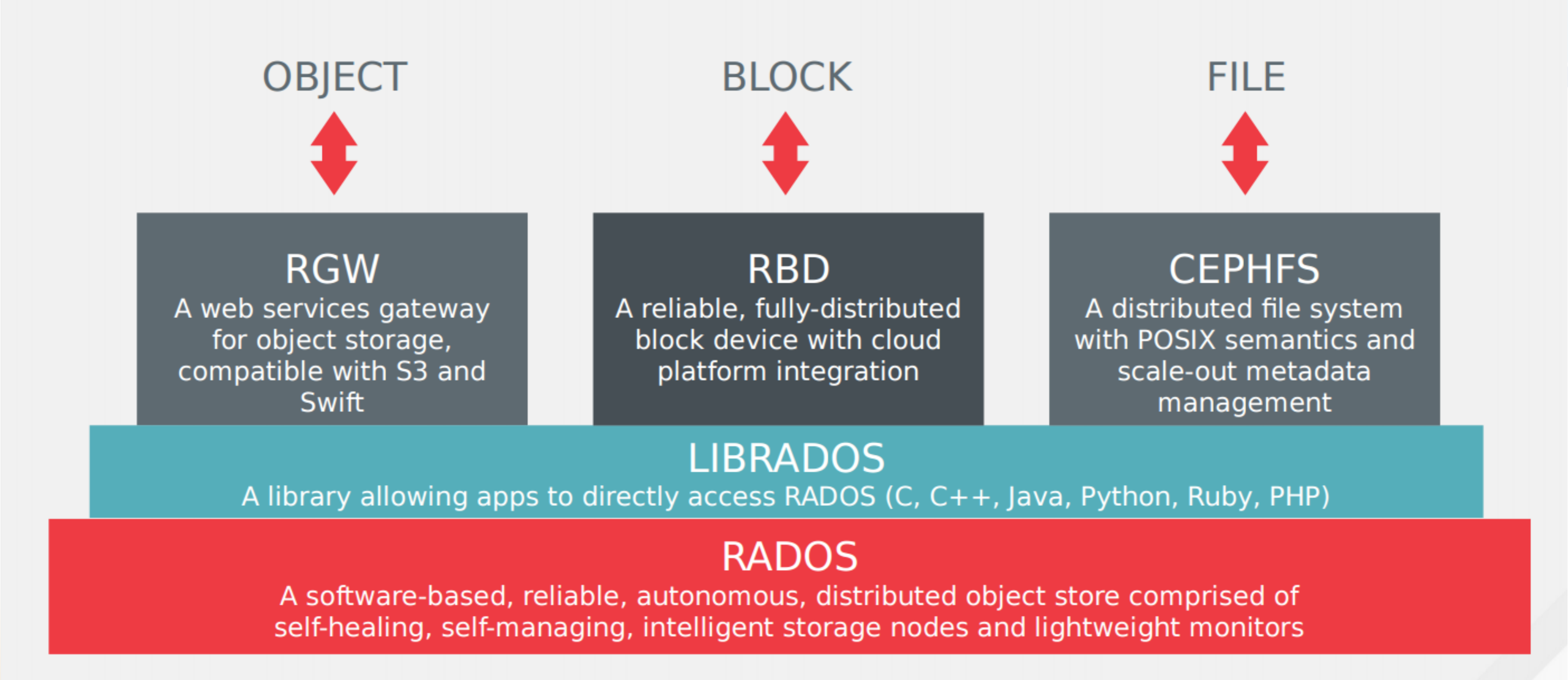
So, Ceph is…
“Object Storage”?

What is Ceph?

ceph.com states: *Ceph is a unified, distributed storage system designed for excellent performance, reliability and scalability.*

- Free and Open Source Software
- Started as a research project in UCSC, now a ~~Red Hat~~ IBM product
- Software Defined Storage (sic)
- Runs on commodity hardware
- Implements Object Storage internally, provides all types: **Block, Object, File**

Components



OBJECT

BLOCK

FILE

RGW
A web services gateway for object storage, compatible with S3 and Swift

RBD
A reliable, fully-distributed block device with cloud platform integration

CEPHFS
A distributed file system with POSIX semantics and scale-out metadata management

LIBRADOS
A library allowing apps to directly access RADOS (C, C++, Java, Python, Ruby, PHP)

RADOS
A software-based, reliable, autonomous, distributed object store comprised of self-healing, self-managing, intelligent storage nodes and lightweight monitors

Source: https://www.slideshare.net/sageweil1/a-crash-course-in-crush

RADOS

Reliable Autonomic Distributed Object Store

- Storage layer where all objects live
- Based on CRUSH (Controlled, Scalable, Decentralized Placement of Replicated Data)
- Maintains physical topology of cluster
- Handles placement of objects
- Monitors cluster health
- Two type of daemons: OSDs and MONs (plus some more)
- Instead of relying on a central directory, let each client calculate itself where to find or place objects

Daemons

OSDs

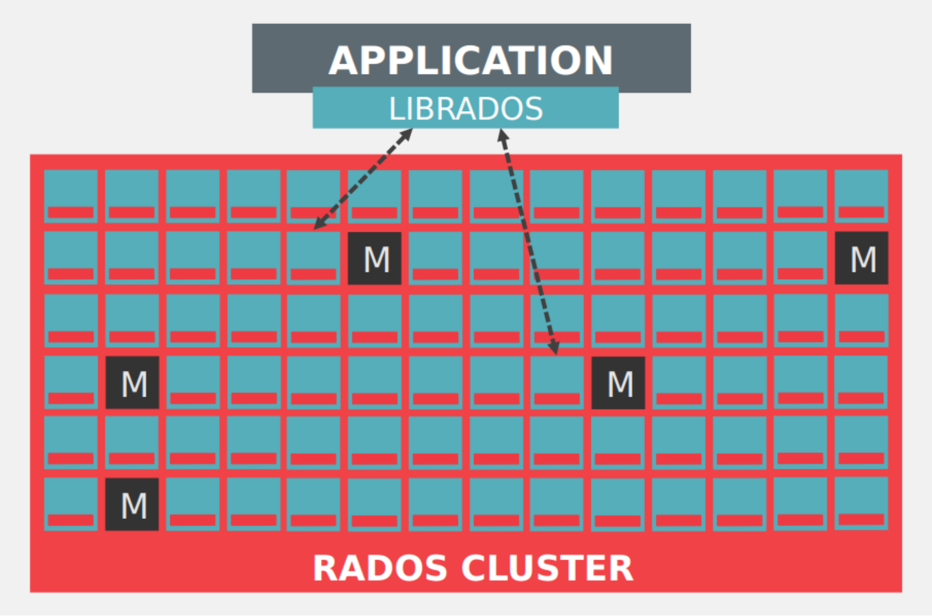
- 10-1000s per cluster
- 1 per disk
- Serve data to clients
- Peer for replication and recovery

MONs

- 3 or 5 per cluster
- Maintain cluster state
- Paxos for decisions
- Do not handle data

More daemons

- mgr, mds and more…



Source: https://www.slideshare.net/sageweil1/a-crash-course-in-crush

Ways of storing data

Pool Types

Replicated

- Each object has size (>=3) replicas
- Each object must have min_size replicas
- Faster than EC
- Larger space overhead than EC

Erasure Coding

- Each object gets divided in k chunks plus m additional
- An object can be recovered from any k chunks
- More CPU intensive

Objectstore

Filestore

- POSIX filesystem (XFS)
- Each object is a file + xattrs
- Has external journal
- LevelDB for metadata
- Deprecated

Bluestore

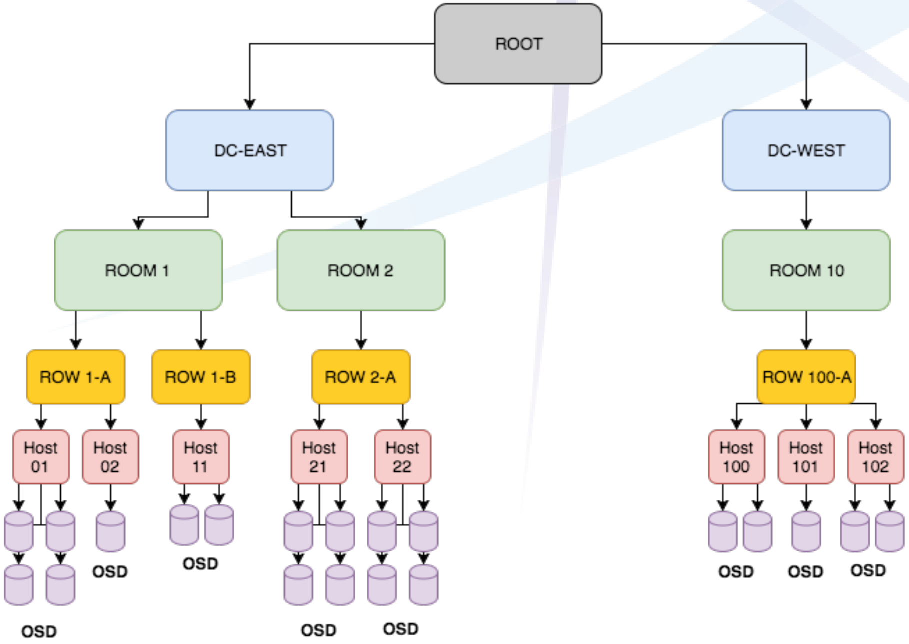
- Raw block device
- RocksDB for metadata
- No journals
- RocksDB can be moved to fast disks
- Faster for most workloads
- Checksumming
- Compression

CRUSH map

Physical Topology of Cluster

- Leaves: OSDs
- Nodes: Buckets: physical locations (rack, PDU, DC, chassis, etc)
- Custom placement policies (i.e, send secondary replica to other DC)
- Place data on different disks types
- Custom failure domain (bucket type)
- Replicas of same object are spread across different buckets of failure domain
- OSDs have weights, depending on their size (or not)

Example: With size=3 and failure domain = rack, you can even lose two racks without having data unavailable or lost



Source: http://yauuu.me/ride-around-ceph-crush-map.html

librados

- Low level interface for RADOS
- Simply read, write and manipulate objects
- Useful for custom apps (dovecot-ceph-plugin, Archipelago, etc)
- All other public interfaces use librados internally

RBD

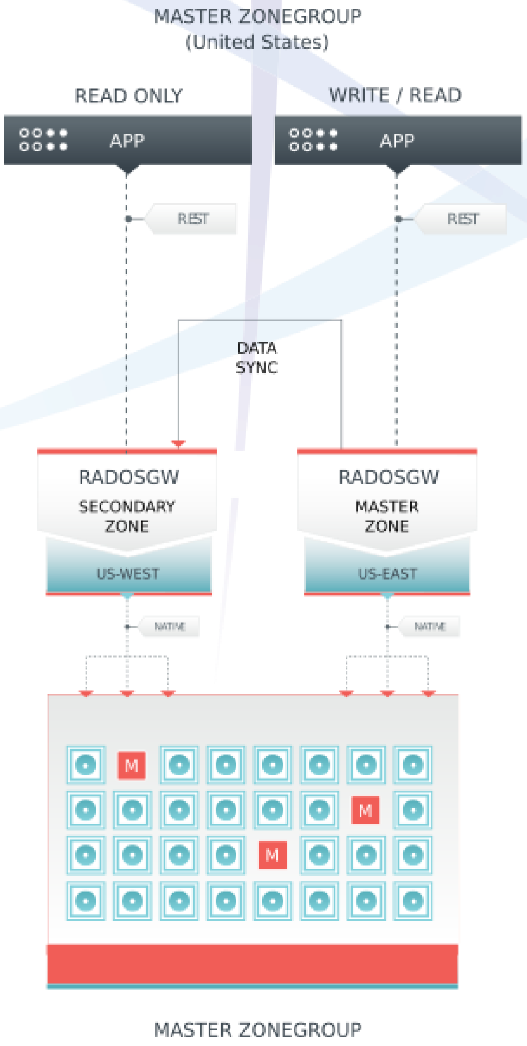
RADOS Block Device

- Provide block devices to clients
- Each block device (image) gets split into multiple RADOS objects (4MB by default)
- librbd (or kernel RBD) calculates on the fly offsets (and thus the target object)
- Has a lot of fancy features (object-maps, clones, snapshots, mirroring)
- 2 ways to access RBD images: librbd or kernel RBD (shipped with mainline kernel)
- RBD volumes can be mirrored to a different cluster

RADOS Gateway

Provide Ceph through S3 and Openstack Swift APIs

- A RESTful gateway to Ceph
- Maps internally each S3/Swift bucket/container/object to RADOS objects and metadata
- Runs a built-in HTTP server (civetweb)
- Can be multizoned
- Has mulitple auth backends (keystone, ldap, etc)

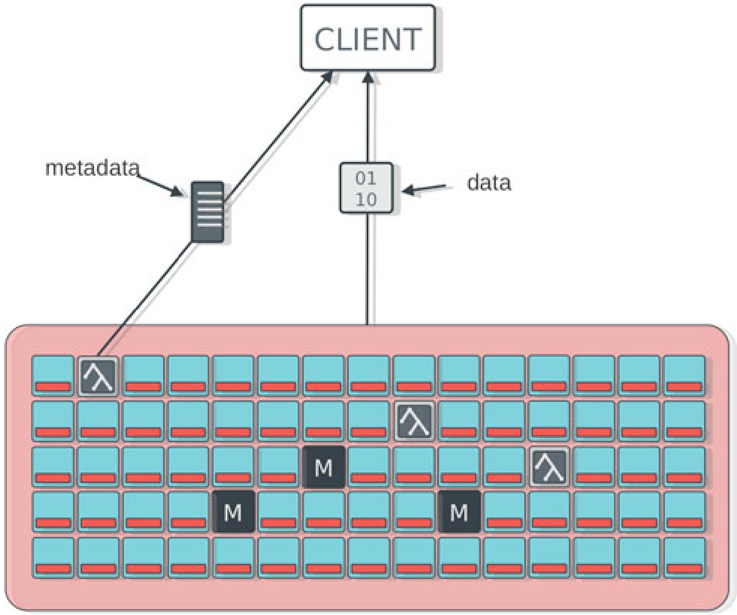


Source: http://docs.ceph.com/docs/mimic/radosgw/

CephFS

POSIX-compliant Shared Filesystem

- Exposes a shared filesystem (think NFS) to clients
- Can be mounted using Ceph clients (FUSE, kernel) and be exposed as a NFS filesystem (nfs-ganesha)
- Metadata (folders, filenames, etc) are stored in separate pools and managed by MDS



Source: https://www.sebastien-han.fr/blog/2012/12/03/ceph-and-mds/

Ceph @ GRNET

Ceph Infrastructure

Interesting numbers

- **6** clusters (4 prod, 2 staging)
- **900** OSDs
- **81** Hosts
- **2.5PB** total raw storage
- **100** Million objects
- **1500** RBD volumes
- **400,000** Swift objects
- **2** major outages
- **0 bytes corrupted or lost**

Facts

- librados, RBD, rgw (Swift)
- Variety of hardware
- Spine-leaf network topology
- Each cluster lives only in one DC
- Mix of Ceph versions and setups
- Filestore & Bluestore
- Failure domain host
- No mixed clusters
- 4/6 clusters are IPv6 only :)

Ceph Clusters

Name	rd0	rd1	rd2	rd3
Version	Jewel	Luminous	Luminous	Luminous
Location	YPEPTH	KNOSSOS	YPEPTH	KNOSSOS
Services	librados	librados, RBD	RBD	rgw (Swift)
Used by	~okeanos	~okeanos, ViMa	ViMa	ESA Copernicus
Pools	repl. size=2	repl. size=3	repl. Size=3	EC 6+3, size=3 on SSDs
Objectstore	Filestore	Filestore	Bluestore	Bluestore
Capacity	350TB	700TB	540TB	1PB
Usage	60%	25%	25%	22%
OSDs	186	192	192	350
Hosts	31	16	12	22

Open Source Tooling

- **FAI** for fully automated Bare-Metal Server provisioning
- **ceph-ansible** for provisioning plus some custom scripts
- **Puppet** for configuration management
- **Ansible** and **Python** tooling for maintenance, operations, upgrades
- **Icinga** for alerting/healthchecks
- **Prometheus** for Ceph and node metrics
- **ELK** for log aggregation

Also, started an effort to open-source our tooling (always GPL!) and provide it to the community.

`https://github.com/grnet/cephtools`

Ansible playbooks, helper scripts, health checks and more to come!

Outages

2 major outages

- **4 day outage caused by flapping hosts**
  - https://blog.noc.grnet.gr/2016/10/18/surviving-a-ceph-cluster-outage-the-hard-way/
- **Not exactly an “outage”: Huge performance degradation due to a single QSFP**
  - https://blog.noc.grnet.gr/2018/08/29/a-performance-story-how-a-faulty-qsfp-crippled-a-whole-ceph-cluster/

Future of Ceph @ GRNET

- Provide S3/Swift as a Service
- Use Ceph for Openstack Clouds
- Automate ourselves out of daily operations!
- Improve performance monitoring
- Contribute to Ceph with patches and docs
- Experiment with new features and tunings
- ....

Pros & Cons

Pros

- Free Software
- Cheaper than vendor solutions
- Can easily scale
- Fast and resilient
- Each release is getting more stable, faster and easier to manage
- Great community
- Provides a lot of services out of the box without much hassle
- Super easy upgrades & ops
- No central directory, no SPOFs
- Customizable
- No major problems so far

Cons

- Requires more technical insight than closed-source vendor solutions
- Ceph hates unstable networks!
- Benchmarking is not easy
- Increased latency in some scenarios
- Recovery is not always fast
- Wrong configuration can cause trouble
- Not FOSS tools for daily ops: you might have to implement your own

Questions?