CEPH WORKSHOP

GridKA School 2015

Diana Gudu
Uros Stevanovic

September 8, 2015

Karlsruhe Institute of Technology

INTRODUCTION ROUND

DIANA GUDU

- PhD researcher in Computer Science @KIT (SCC)
  - distributed multi-agent framework for trading cloud resources
- working in HBP @SCC on cloud storage
- MSc in Computational Science and Engineering @TU Munich
- BSc in Computer Science @Polytechnic University of Bucharest

UROS STEVANOVIC

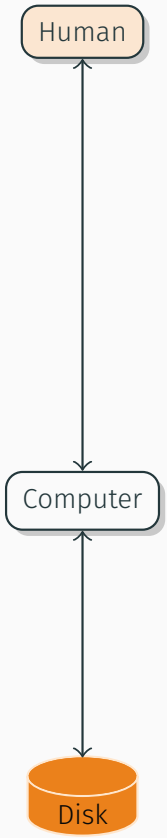
- working in AARC project @KIT (SCC)
- PhD @KIT (IPE): 2010-2015
  - building a custom smart camera framework
  - using FPGAs
  - implementing image processing algorithms
- studied Electrical Engineering @University of Belgrade

YOU

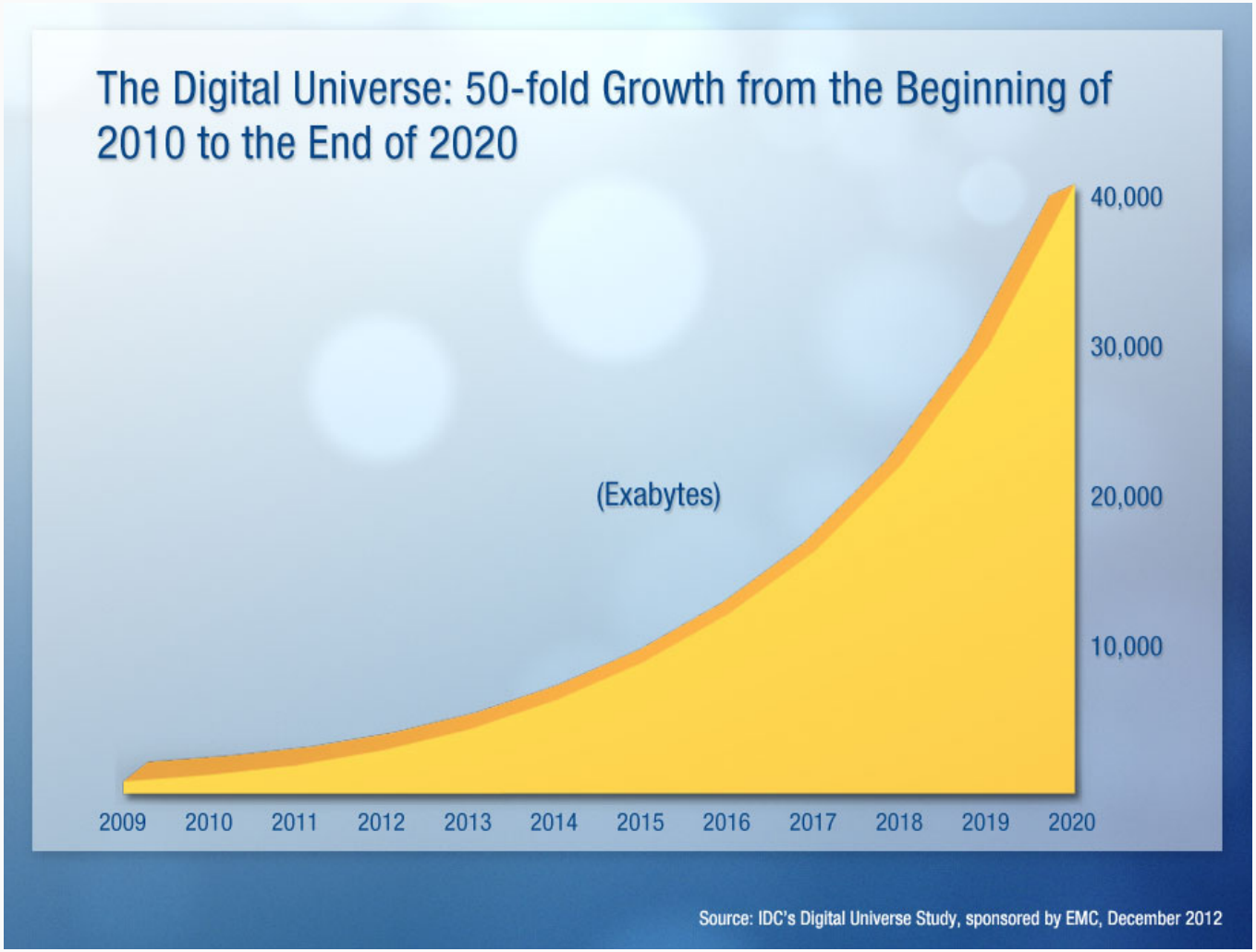
Your turn!

EVOLUTION OF STORAGE

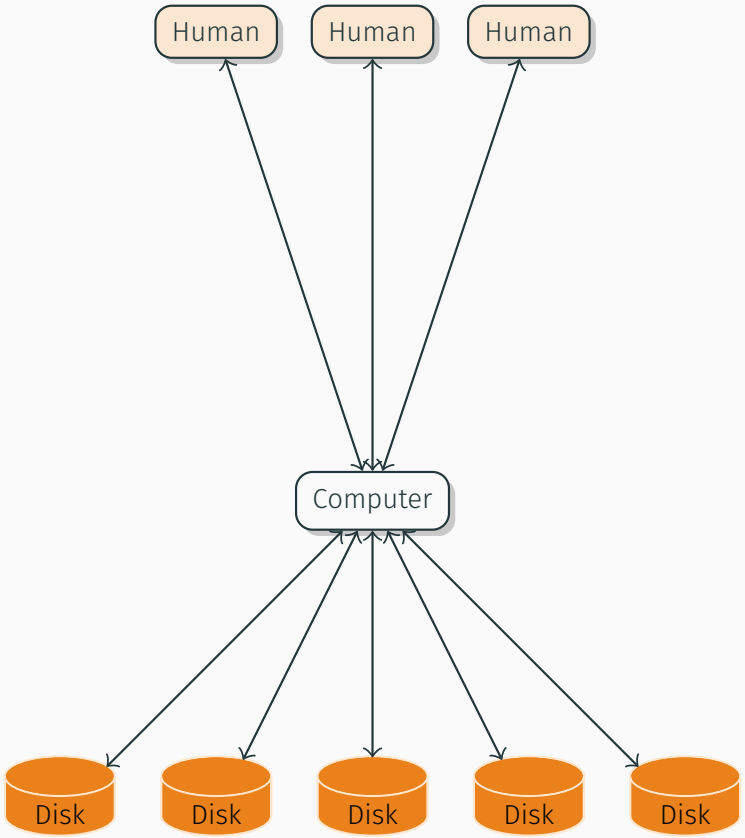
EVOLUTION OF STORAGE



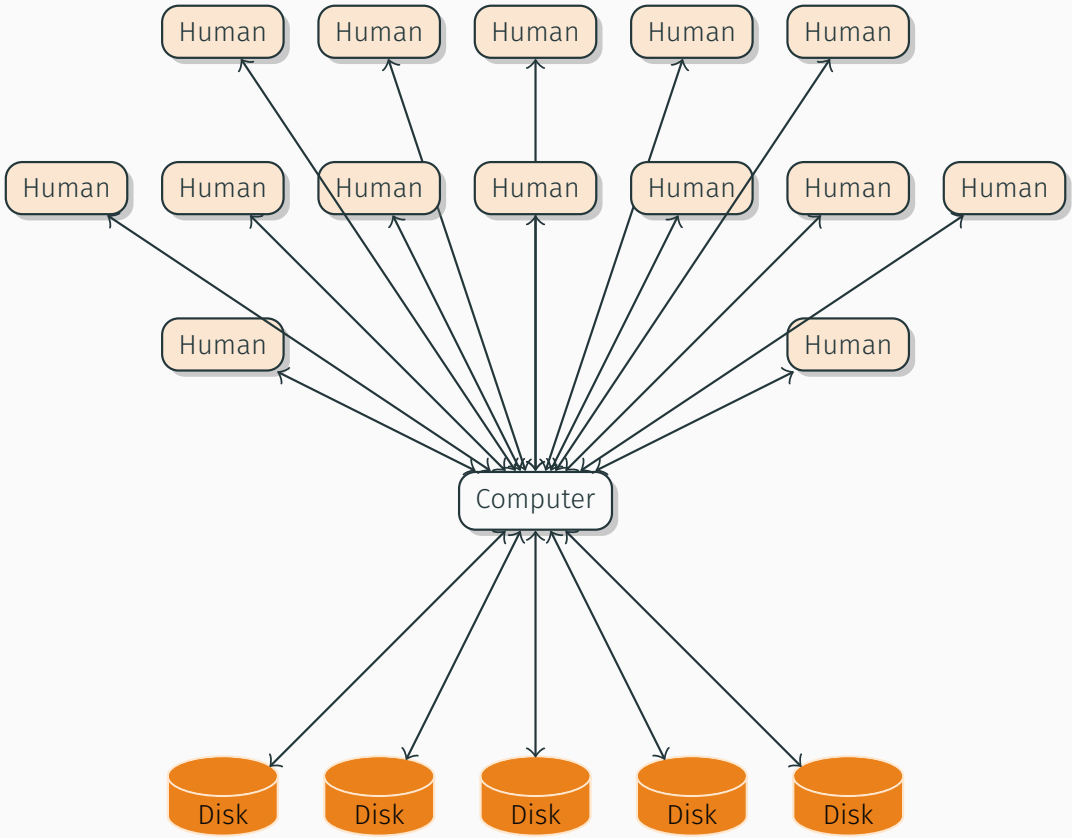
EVOLUTION OF STORAGE



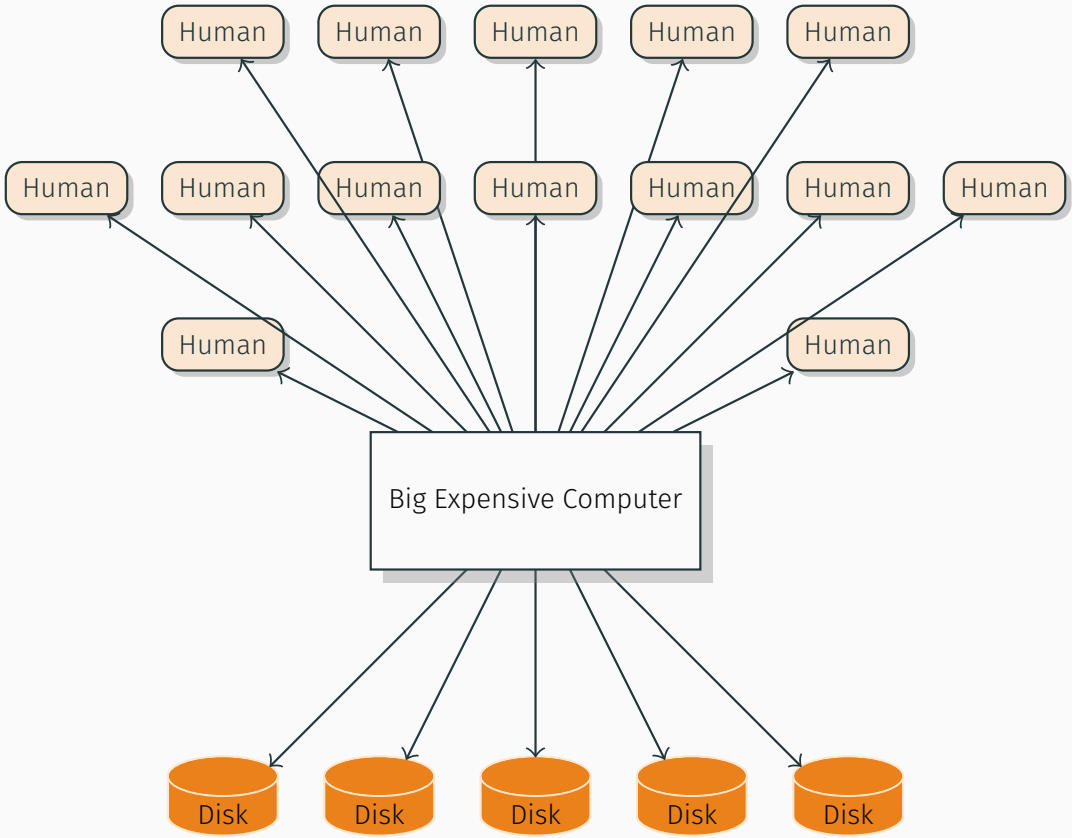
EVOLUTION OF STORAGE



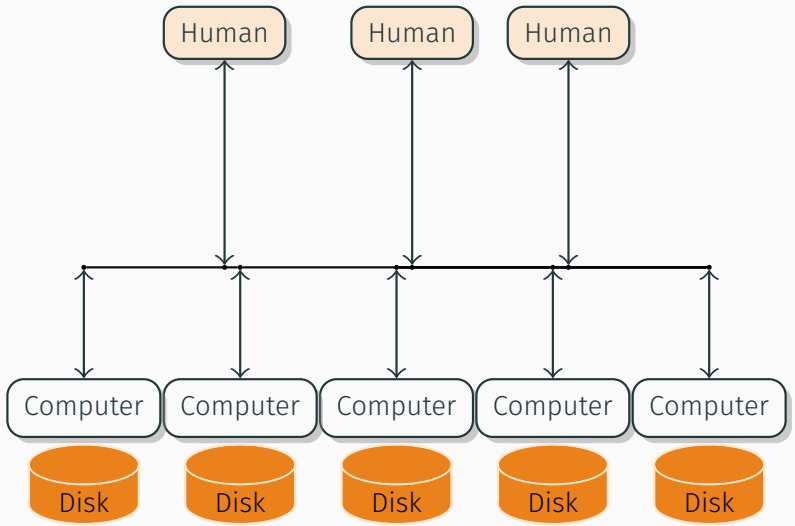
EVOLUTION OF STORAGE



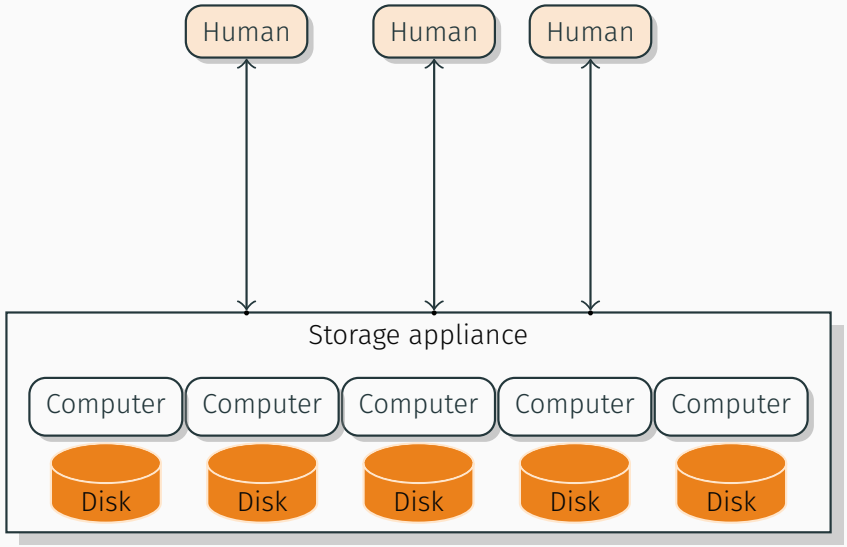
EVOLUTION OF STORAGE



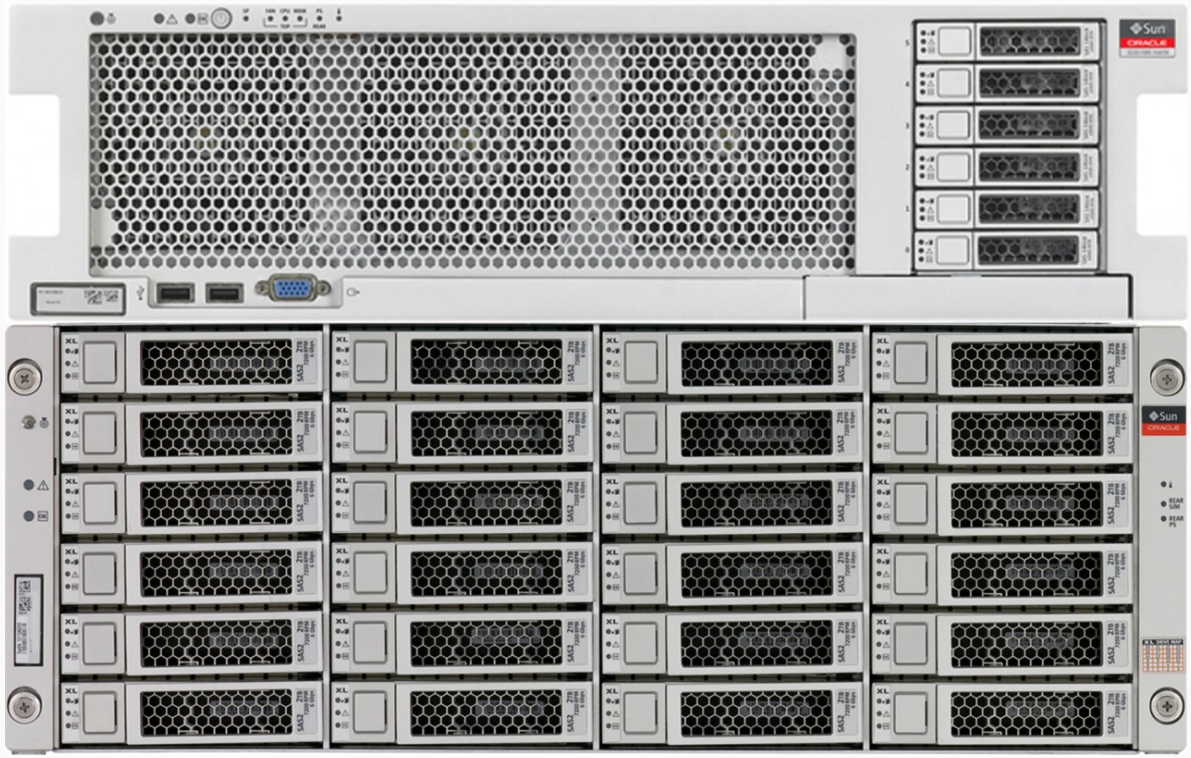
EVOLUTION OF STORAGE



EVOLUTION OF STORAGE

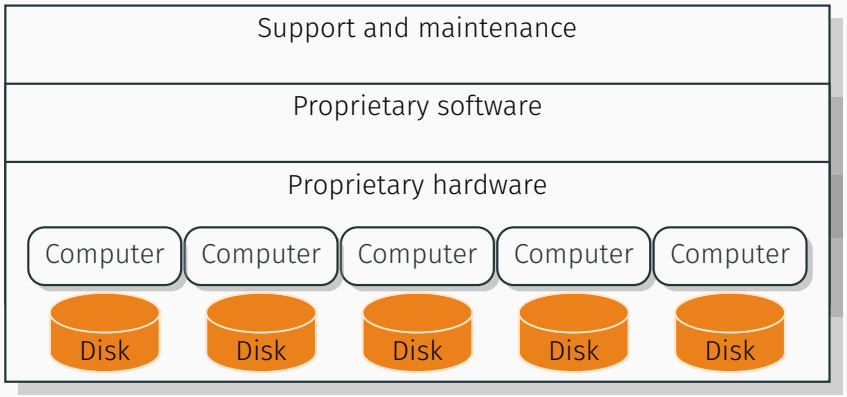


STORAGE APPLIANCE

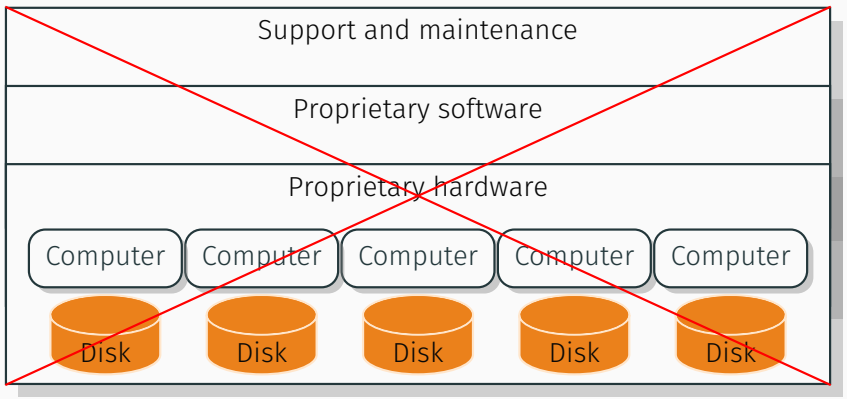


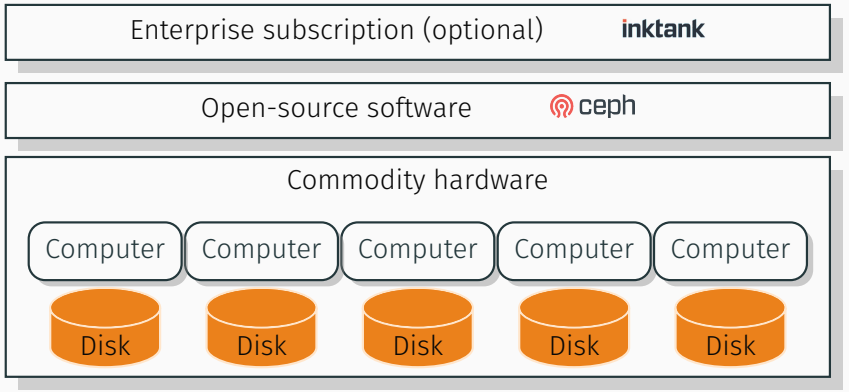
Oracle http://www.e-business.com/zfs-7420-storage-appliance

FUTURE OF STORAGE



FUTURE OF STORAGE





CEPH

Philosophy

- open-source

Philosophy

- open-source
- community focused

Philosophy

- open-source
- community focused
- software-defined

Philosophy

- open-source
- community focused
- software-defined
- scale-out hardware, no SPF

Philosophy

- open-source
- community focused
- software-defined
- scale-out hardware, no SPF
- self-managing

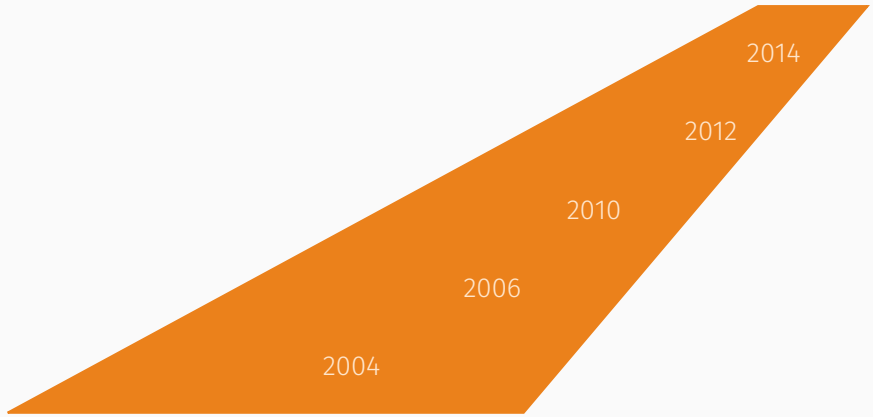
Philosophy

- open-source
- community focused
- software-defined
- scale-out hardware, no SPF
- self-managing
- failure is normal

Philosophy

- open-source
- community focused
- software-defined
- scale-out hardware, no SPF
- self-managing
- failure is normal

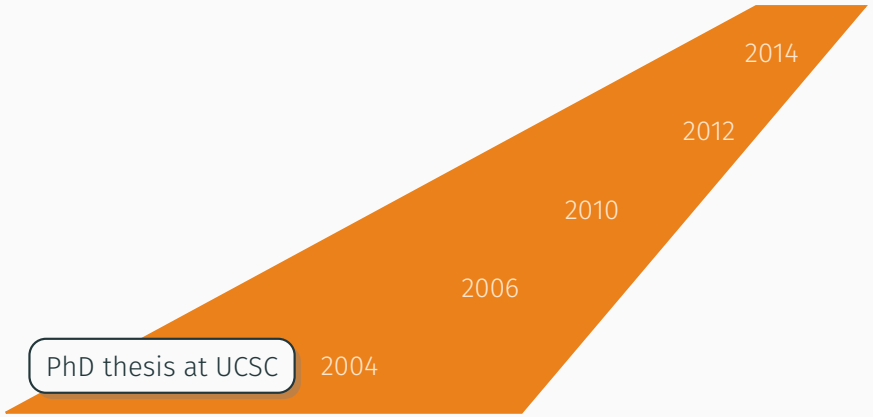
History



Philosophy

- open-source
- community focused
- software-defined
- scale-out hardware, no SPF
- self-managing
- failure is normal

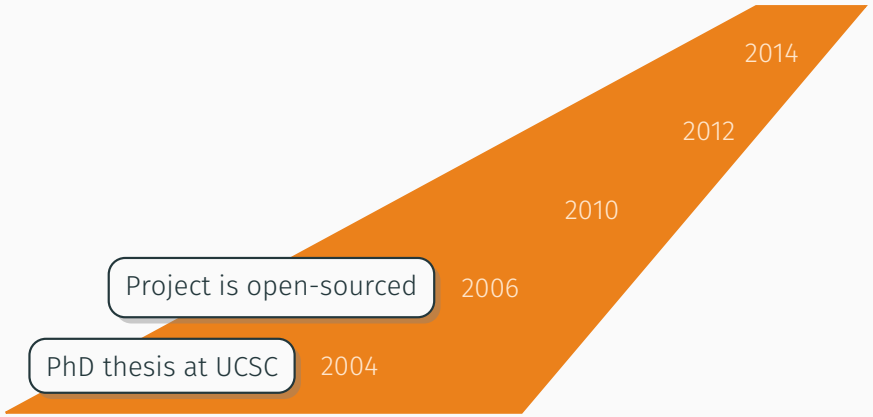
History



Philosophy

- open-source
- community focused
- software-defined
- scale-out hardware, no SPF
- self-managing
- failure is normal

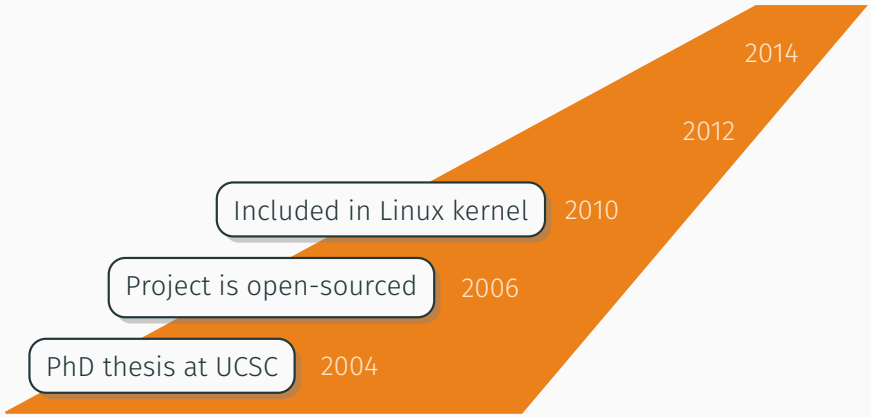
History



Philosophy

- open-source
- community focused
- software-defined
- scale-out hardware, no SPF
- self-managing
- failure is normal

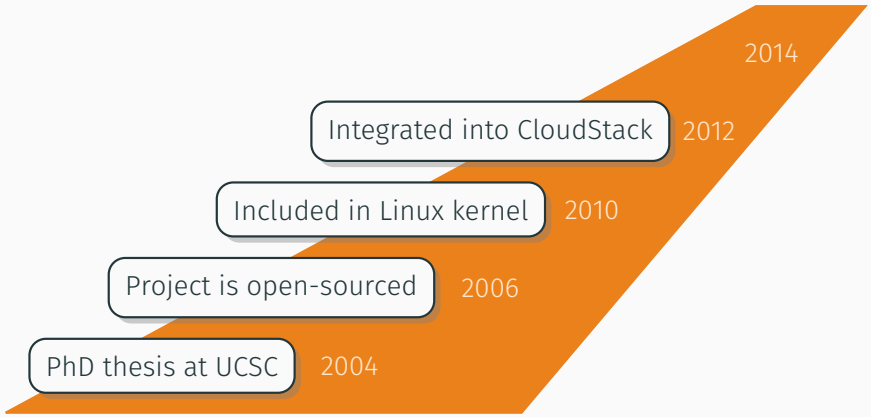
History



Philosophy

- open-source
- community focused
- software-defined
- scale-out hardware, no SPF
- self-managing
- failure is normal

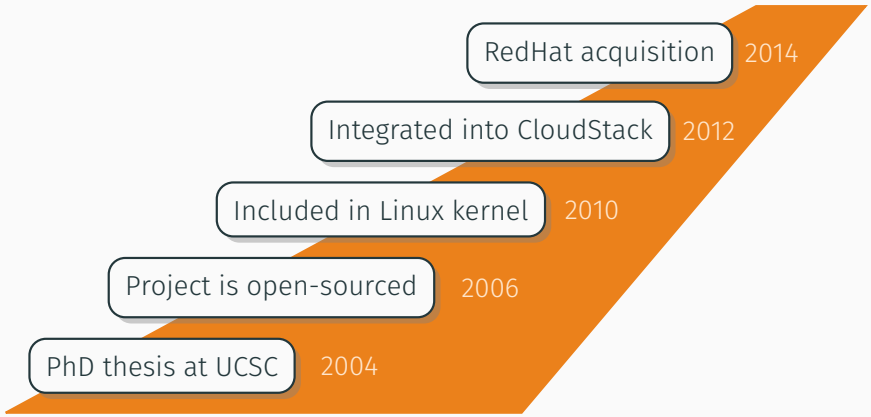
History



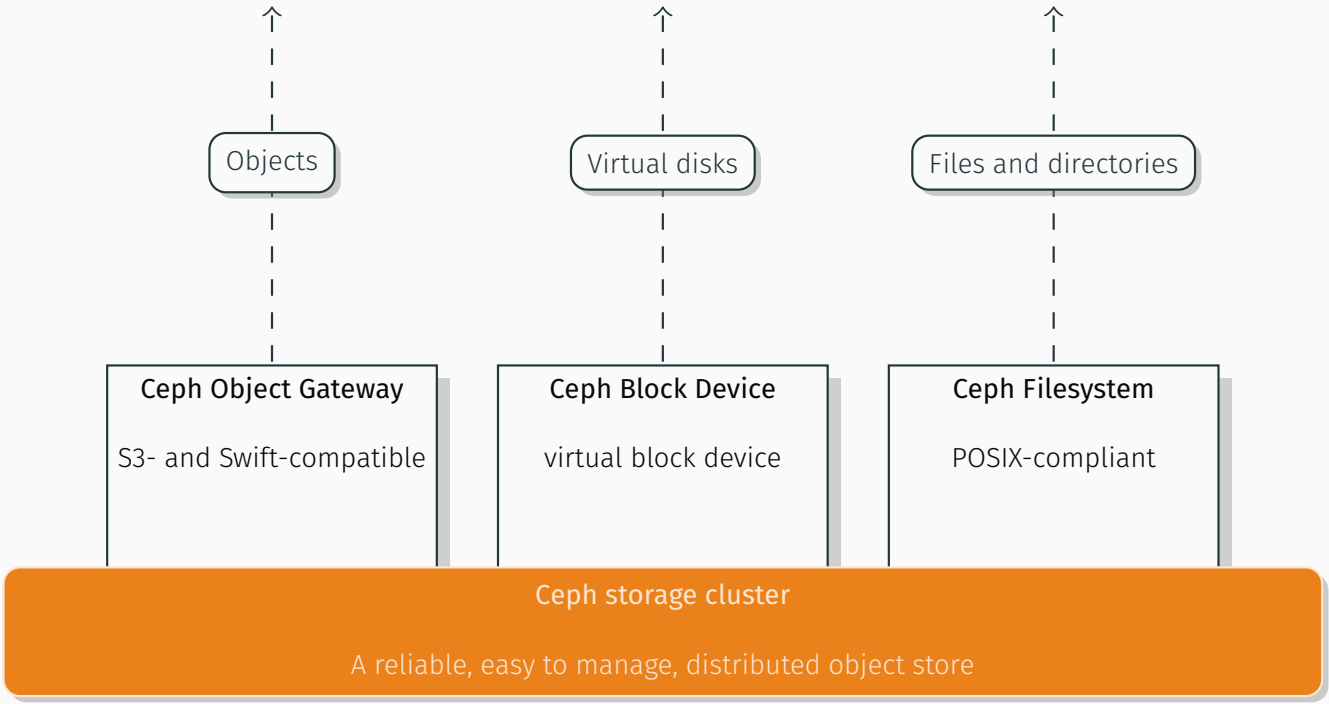
Philosophy

- open-source
- community focused
- software-defined
- scale-out hardware, no SPF
- self-managing
- failure is normal

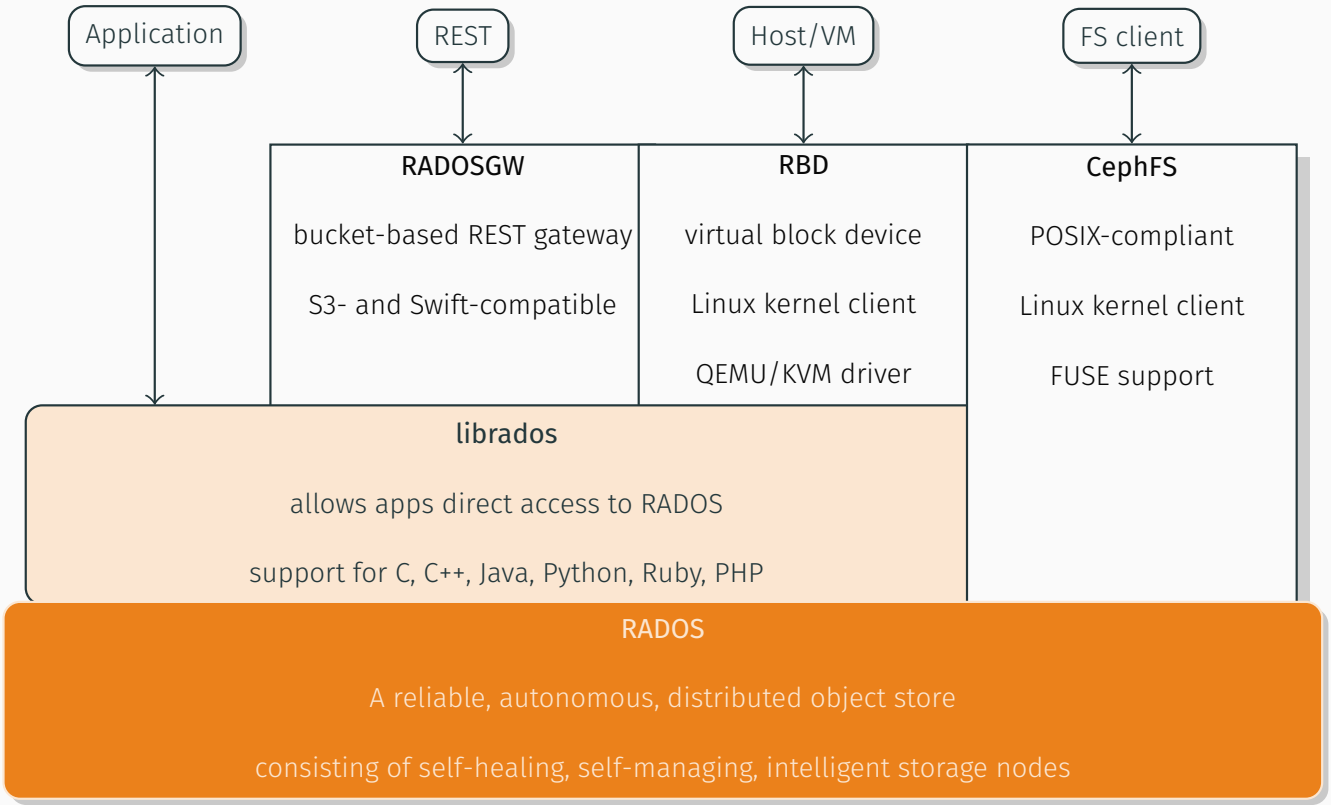
History



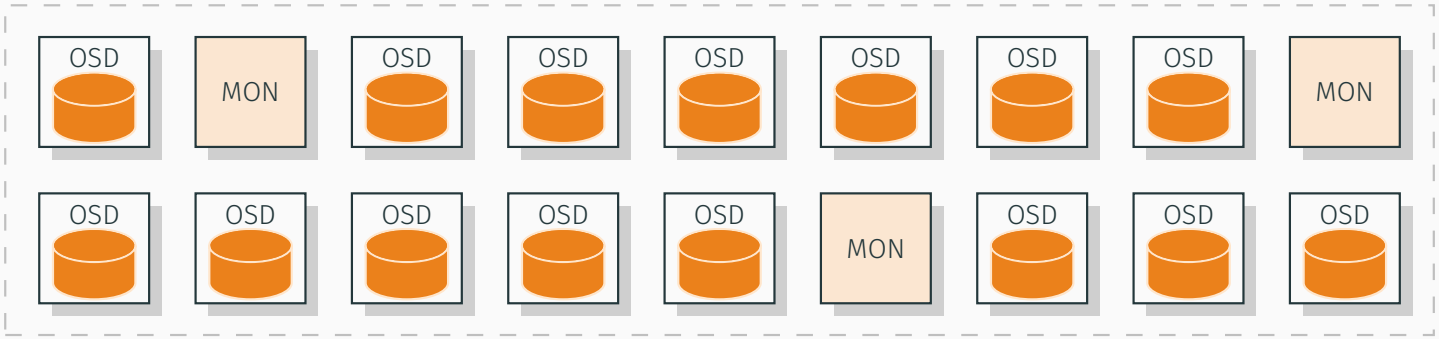
CEPH ARCHITECTURE



CEPH ARCHITECTURE



RADOS


OSD
MON
OSD
OSD
OSD
OSD
OSD
OSD
MON
OSD
OSD
OSD
OSD
OSD
MON
OSD
OSD
OSD

CEPH DAEMONS

OSD

- serve objects to clients
- one per disk
- backend: btrfs, xfs, ext4
- peer-to-peer replication and recovery
- write-ahead journal

MON

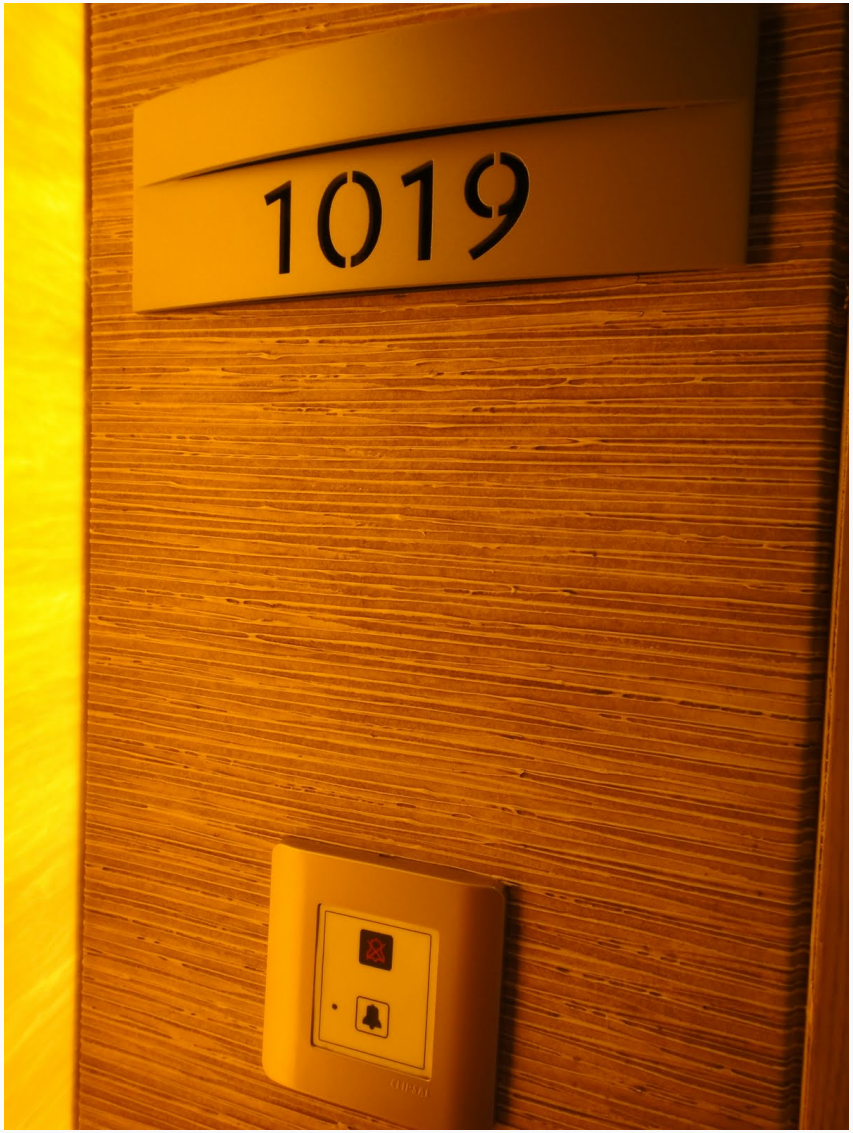
- maintain cluster state and membership
- vote for distributed decision-making
- small, odd number

DATA PLACEMENT

HOTELS

http://free-stock-illustration.com/hotel+key+card

HOTELS



http://2.bp.blogspot.com/-o-rlIrv094E/TXxj8D-B2LI/AAAAAAAAGh8/VEbrbHpxVxo/s1600/DSC02213.JPG

HOTELS

- What if the hotel had 1 billion rooms? Or ∞?

#13,565,983

What if the hotel changed constantly?

http://waltonian.com/news/eastern-library-renovations-continue/

Scale-up everything?

http://www.millenniumhotels.com/content/dam/global/en/the-heritage-hotel-manila/images/cons-photographics-lobby-reception-desk%2003062011_34-basicB-preview-2048.jpg

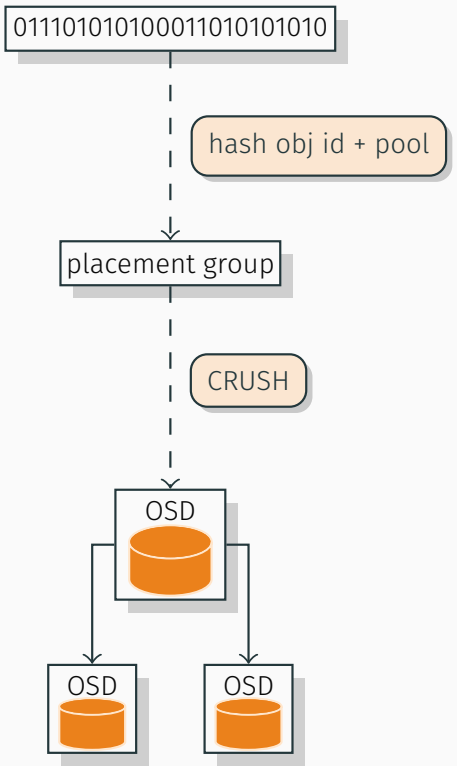
HOTELS

- The hotel itself must assign people to rooms instead of a centralized place
- The hotel should grow itself organically

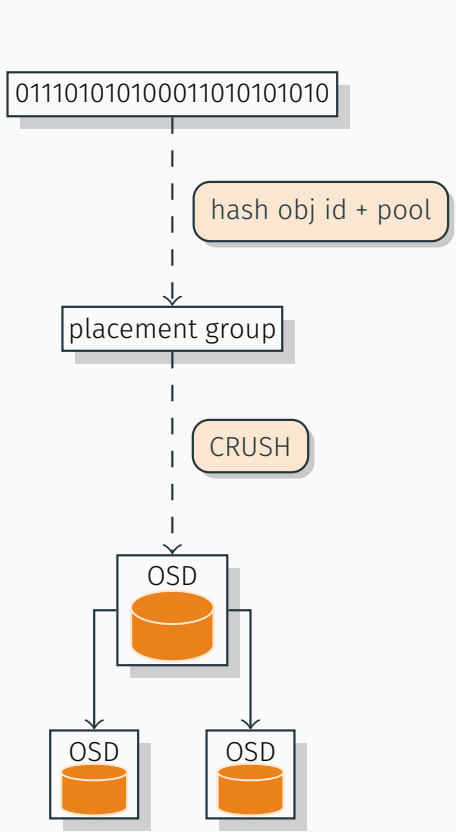
HOTELS

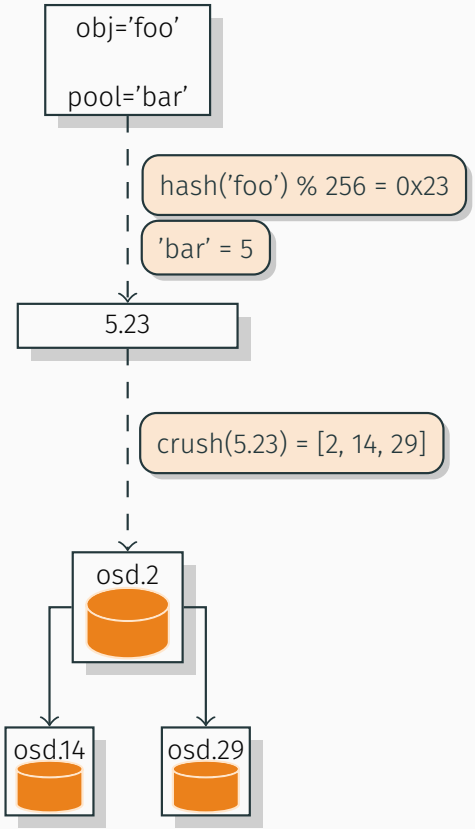
- The hotel itself must assign people to rooms instead of a centralized place
- The hotel should grow itself organically
- Deterministic placement algorithm
- Intelligent nodes

CRUSH


01110101010000110101010
hash obj id + pool
placement group
CRUSH
OSD
OSD
OSD

CRUSH


01110101010000110101010101010
hash obj id + pool
placement group
CRUSH
OSD
OSD
OSD

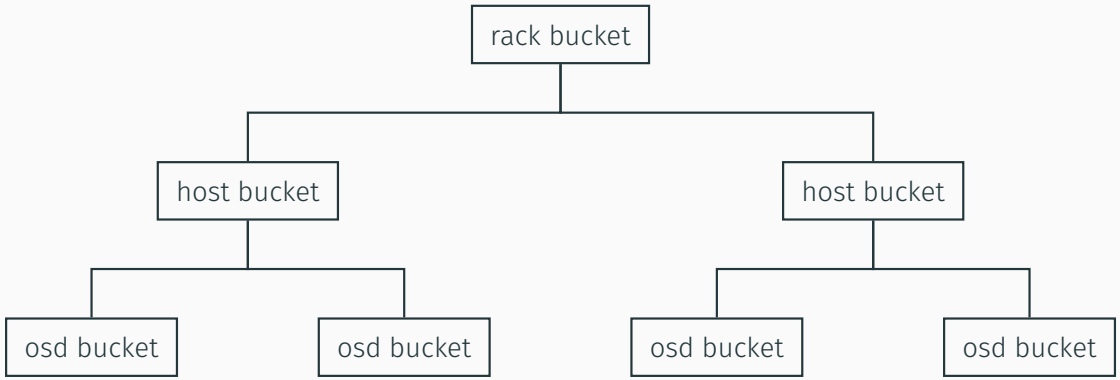

obj='foo'
pool='bar'
hash('foo') % 256 = 0x23
'bar' = 5
5.23
crush(5.23) = [2, 14, 29]
osd.2
osd.14
osd.29

Controlled Replication Under Scalable Hashing

- Pseudo-random placement algorithm
- Repeatable, deterministic
- Statistically uniform distribution
- Stable mapping: minimal data migration
- Rule-based configuration, topology aware

Controlled Replication Under Scalable Hashing

- Pseudo-random placement algorithm
- Repeatable, deterministic
- Statistically uniform distribution
- Stable mapping: minimal data migration
- Rule-based configuration, topology aware



CEPH CLIENTS

LIBRADOS

- direct access to RADOS for applications
- C, C++, Python, Java, Erlang, PHP
- native socket access, no HTTP overhead

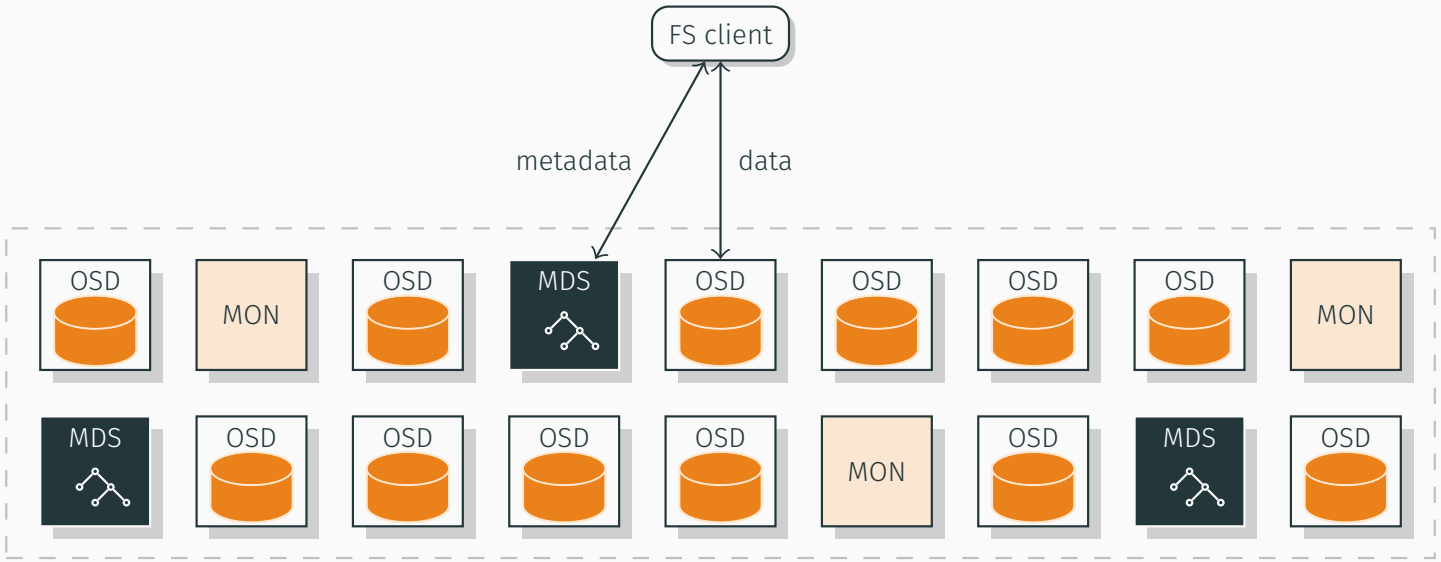
RADOSGW

- RESTful API
- unified object namespace
- S3 and Swift compatible
- user database and access control
- usage accounting, billing

RBD

- Storage of disk images in RADOS
- Images are striped across the cluster
- Decoupling of VMs from host
- Thin provisioning
  - physical storage only used once you begin writing
- Snaphots, copy-on-write clones
- Support in Qemu, KVM

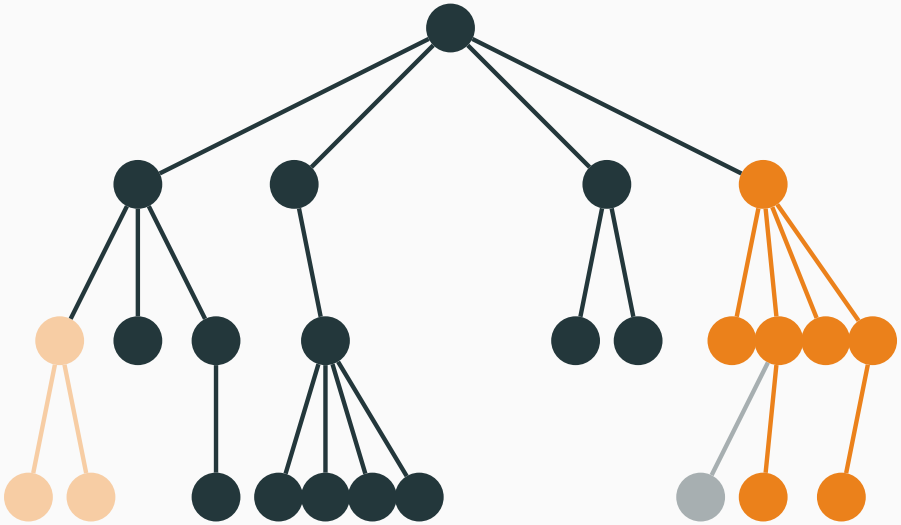
CephFS


FS client
metadata
data
OSD
MON
OSD
MDS
OSD
OSD
OSD
OSD
MON
MDS
OSD
OSD
OSD
OSD
MON
OSD
MDS
OSD

Metadata Server

- Manages metadata for POSIX-compliant filesystem
  - directory hierarchy
  - file metadata: owner, timestamps, mode etc
- Stores metadata in RADOS
- Multiple MDS for HA and load balancing

DYNAMIC SUBTREE PARTITIONING

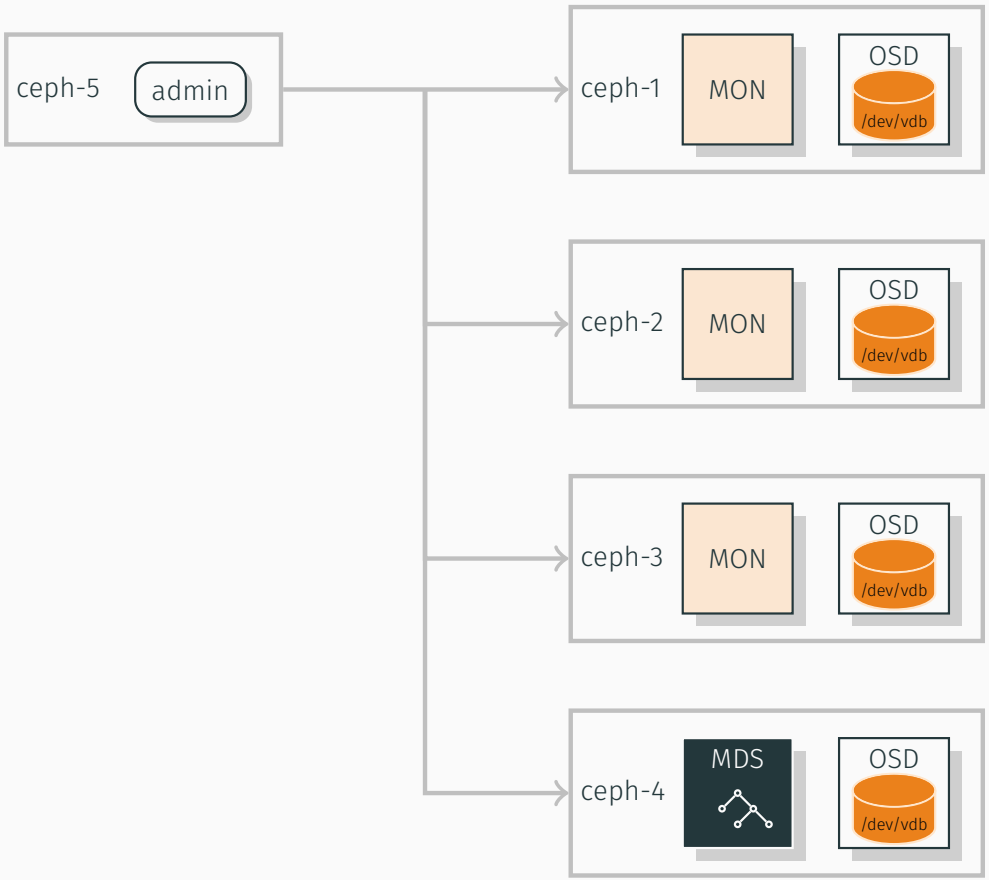


TUTORIAL

OVERVIEW

- Deploy a Ceph cluster
- Basic operations with the storage cluster
- Data placement: CRUSH
- Ceph Filesystem
- Block storage: RBD
- Advanced topics: erasure coding
- Troubleshooting challenge

CLUSTER SET-UP



Questions?