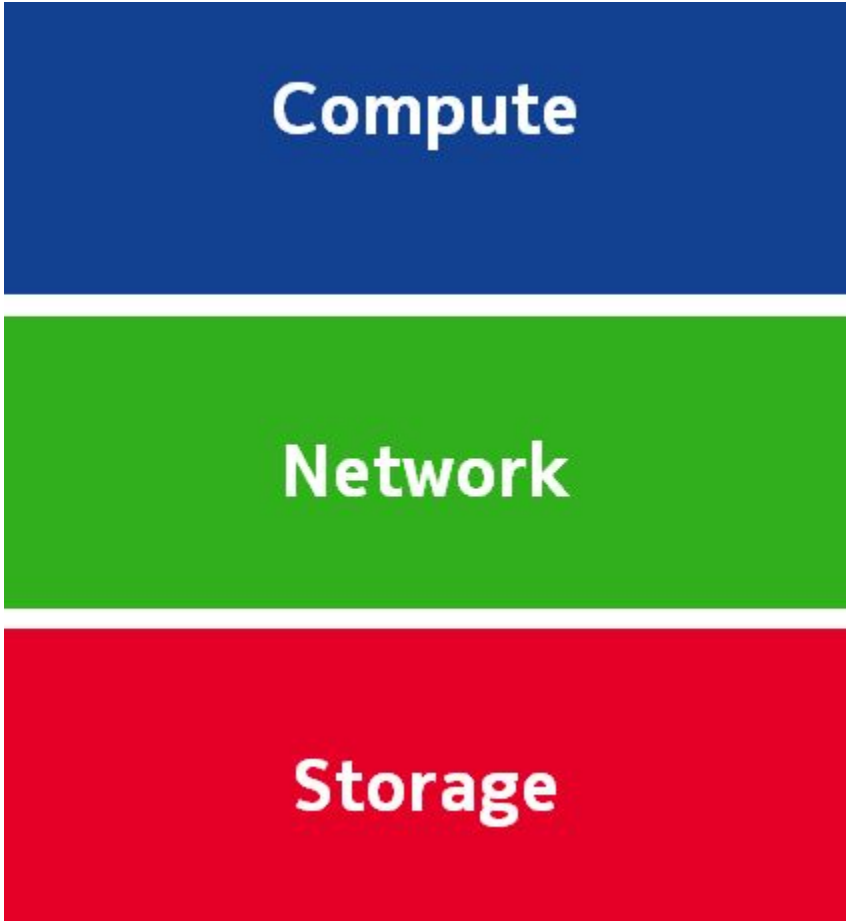
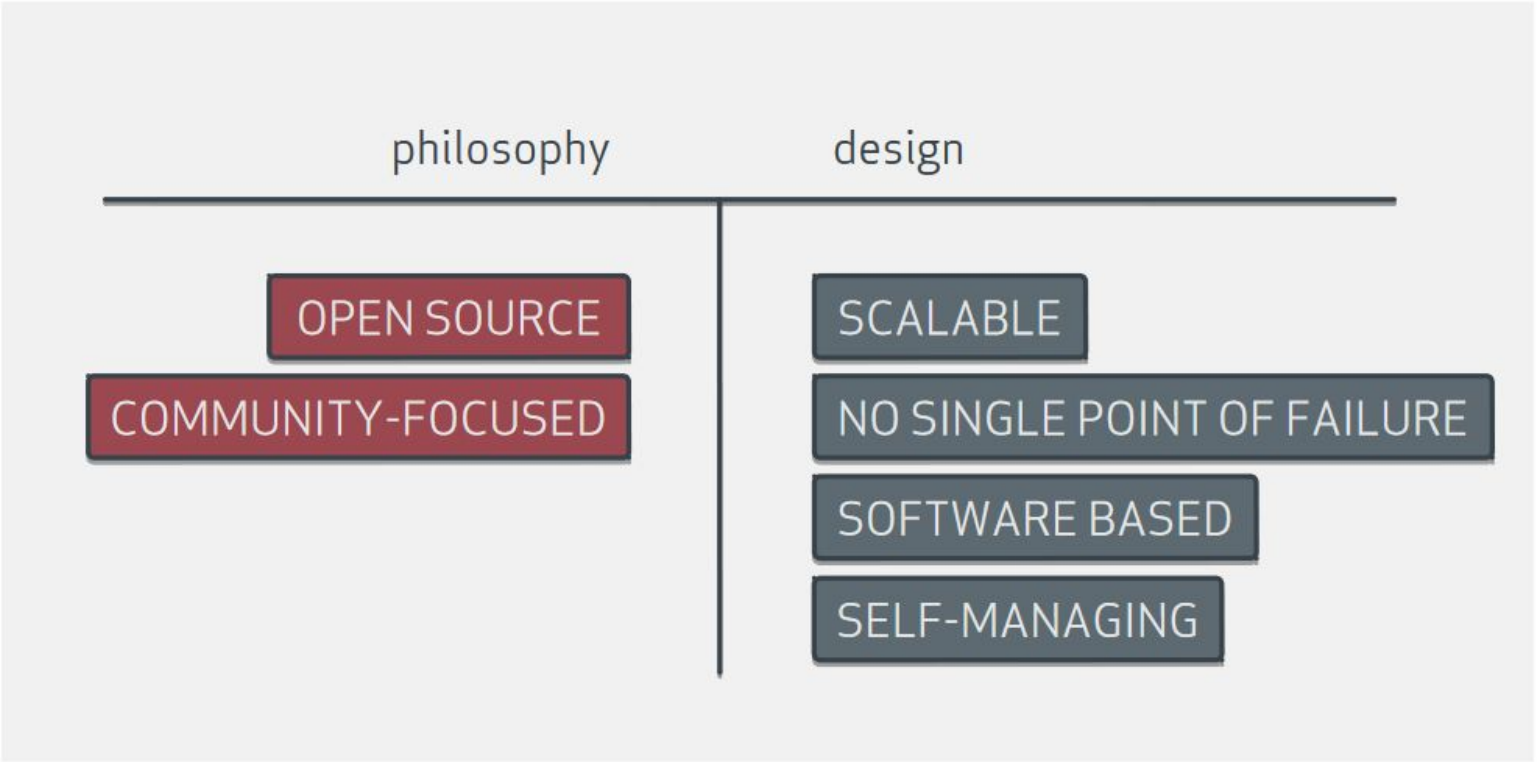
Software Defined Storage

08-Jan-2020


Compute
Network
Storage

Philosophy and Design



https://people.redhat.com/~flucifre/talks/OpenStackGDL%20-%20Ceph%20Intro%20&%20Architectural%20Overview.pdf

Users

https://ceph.io/users/

Enterprise Ceph providers

https://www.redhat.com/en/technologies/storage/ceph?intcmp=701f2000001OEGrAAO

https://ubuntu.com/ceph

https://www.suse.com/solutions/software-defined-storage/ceph/

Agenda

1. History
2. Proprietary Storage Appliances vs Opensource Storage
3. Architecture
4. Deployments

Agenda

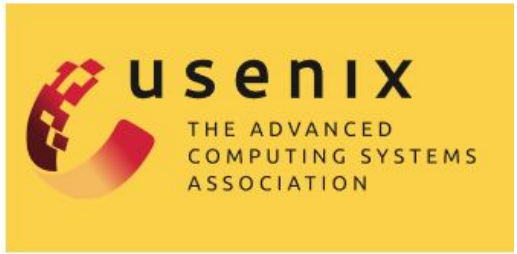
1. History
2. Proprietary Storage Appliances vs Opensource Storage
3. Architecture
4. Deployments

History

Ceph made is debut in 2006 in a whitepaper presented in the USENIX OSDI 2006 conference by:

1. Sage Weil (founder and chief architect of Ceph)
2. Scott Brandt (Professor University of California, Santa Cruz)
3. Ethan L. Miller (Professor University of California, Santa Cruz)
4. Darrell D. E. Long (Distinguished Professor and Director of SSRC)
5. Carlos Maltzahn (founder and director of the UC Santa Cruz Center for Research in Open Source Software - CROSS)

https://www.ssrc.ucsc.edu/Papers/weil-osdi06.pdf



History

1. Ceph was initially a community driven project supported mainly by Inktank Storage [created in 2012]
2. Initial contributors where Inktank, Dreamhost, Canonical,
3. Red Hat purchased Inktank in April 2014 for $175 Million.
4. In October 2015, Ceph Community Advisory Board was formed.

https://ceph.com/community/red-hat-to-acquire-inktank/

History

1. Community members include individuals from Canonical, CERN, Cisco, Fujitsu, Intel, Red Hat, SanDisk, and SUSE
2. Current stable release is *Nautilus* (v14.2.5) release date: Dec 19, 2019
3. Current Red Hat version is RHCS 3.2 (Bluestore) that supports containerized ceph daemons

Agenda

1. History
2. Proprietary Storage Appliances vs Opensource Storage
3. Architecture
4. Deployments

Proprietary Storage Appliances vs Opensource Storage (Ceph)

Proprietary Storage Appliance

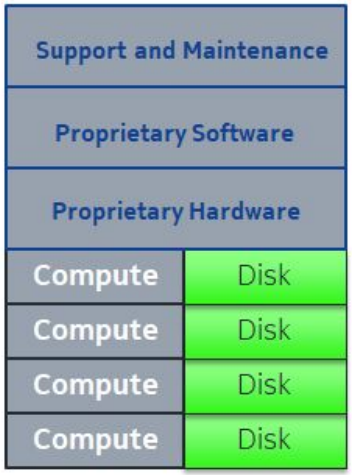
OpenSource Storage

- ❑ Cost ($/GB)
- ❑ Performance
- ❑ Configurable/Programmability
- ❑ Scalability
- ❑ Adaptability
- ❑ Support and Maintenance

https://people.redhat.com/~flucifre/talks/OpenStackGDL%20-%20Ceph%20Intro%20&%20Architectural%20Overview.pdf

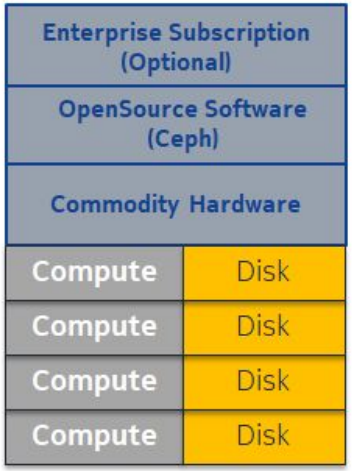
Proprietary Storage Appliances vs OpenSource Storage (Ceph)

Proprietary Storage Appliance



- $5.2 Billion in revenue for support and maintenance
- $1.1 Billion in R&D
- 1.6 million sq. ft. of manufacturing space

Opensource Storage (Ceph)



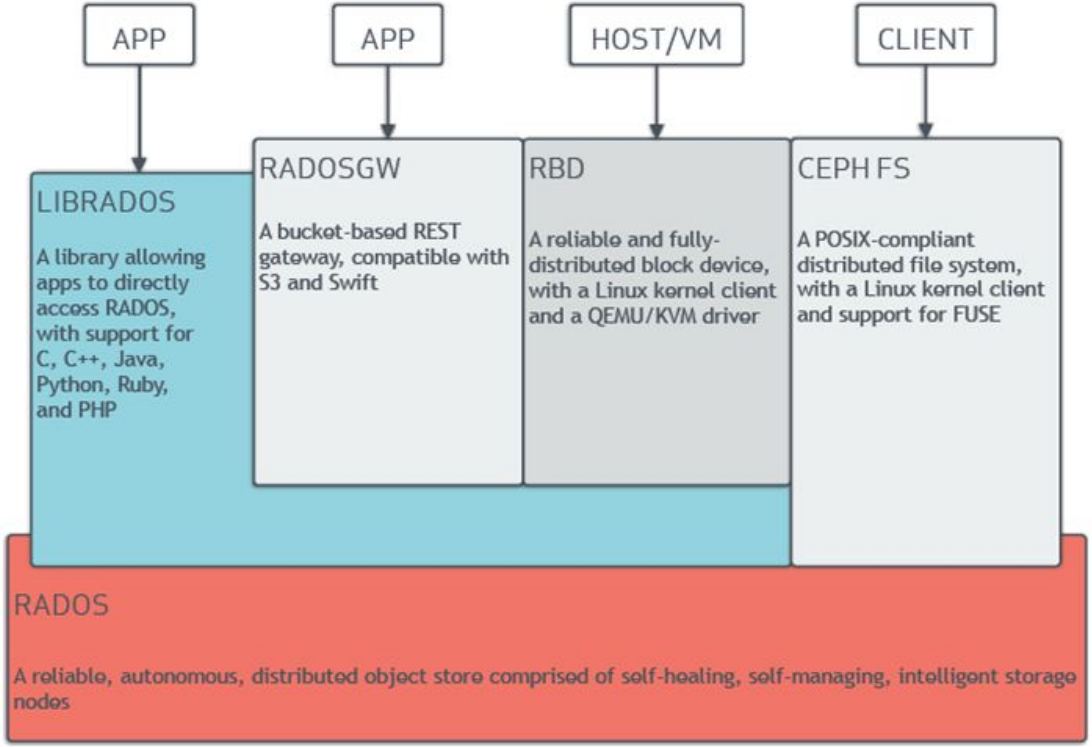
- Subscriptions provided by companies like Redhat, Suse, Canonical
- OpenSource software is free
- Can work with any commodity hardware

https://people.redhat.com/~flucifre/talks/OpenStackGDL%20-%20Ceph%20Intro%20&%20Architectural%20Overview.pdf

Agenda

1. History
2. Proprietary Storage Appliances vs Opensource Storage
3. Ceph Architecture
4. Deployments

Ceph Architecture



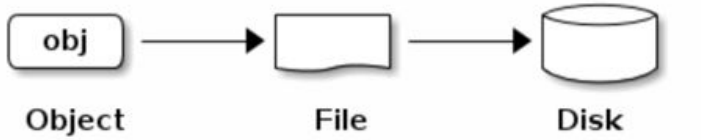
Ceph Architecture: Ceph Storage Cluster

A Ceph Storage Cluster consists of two types of daemons:

- **Ceph Monitor:**
  - Maintains a master copy of the cluster map
  - Works in high availability (at least 3 mon to form a quorum)
- **Ceph OSD Daemon:**
  - Ceph OSD Daemon checks its own state and the state of other OSDs and reports back to monitors.
  - Uses CRUSH algorithm to efficiently compute information about data location, instead of having to depend on a central lookup table.

Ceph Architecture: Storing Data

- A Ceph Storage Cluster receives data from Ceph clients and stores the data as objects.
- Each object corresponds to a file in a filesystem which is stored on an Object Storage Device.
- Ceph OSD Daemons handle the read/write operations on the storage disks.

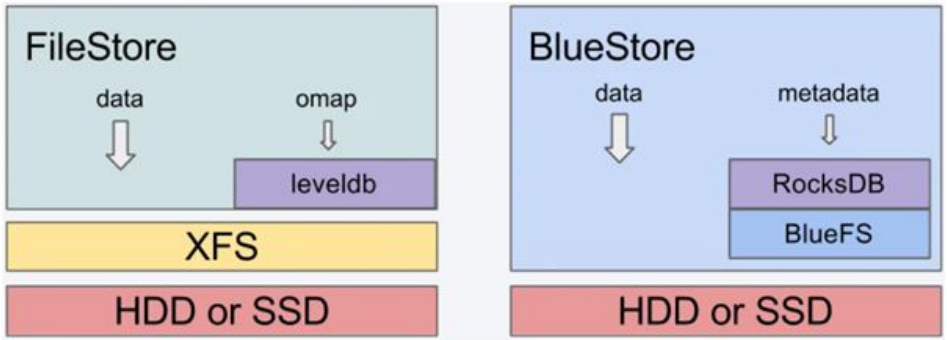


- An object has an identifier, binary data, and metadata consisting of a set of name/value pairs.

ID	Binary Data	Metadata
1234	01010101010101001101010010 01011000010101001101010010 01011000010101001101010010	name1 value1 name2 value2 nameN valueN

Ceph Architecture: Storage Backends – Filestore & Bluestore

- Bluestore has replaced Filestore as the default storage backend in newer versions of Ceph
- BlueStore is built on top of a raw underlying block device (or block devices).
- It embeds the RocksDB key/value database for managing its internal metadata.
- A small internal adapter component called BlueFS implements a file-system-like interface that provides just enough functionality to allow RocksDB to store its "files" and share the same raw device(s) with BlueStore.



https://ceph.io/community/new-luminous-bluestore/

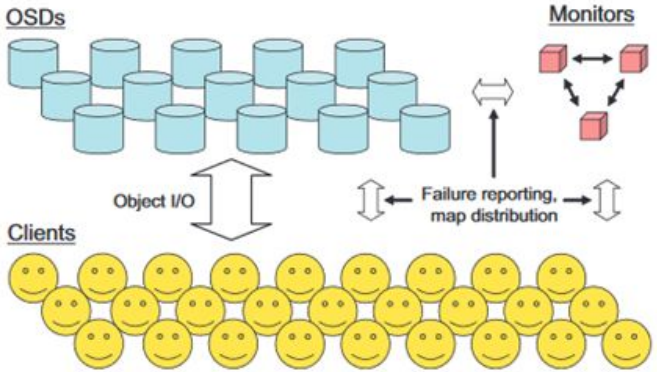
Ceph Architecture: Scalability and High Availability

- Ceph eliminates the centralized gateway to enable clients to interact with Ceph OSD Daemons directly.
- Ceph OSD Daemons create object replicas on other Ceph Nodes to ensure data safety and high availability.
- Ceph also uses a cluster of monitors to ensure high availability.
- To eliminate centralization, Ceph uses an algorithm called CRUSH.

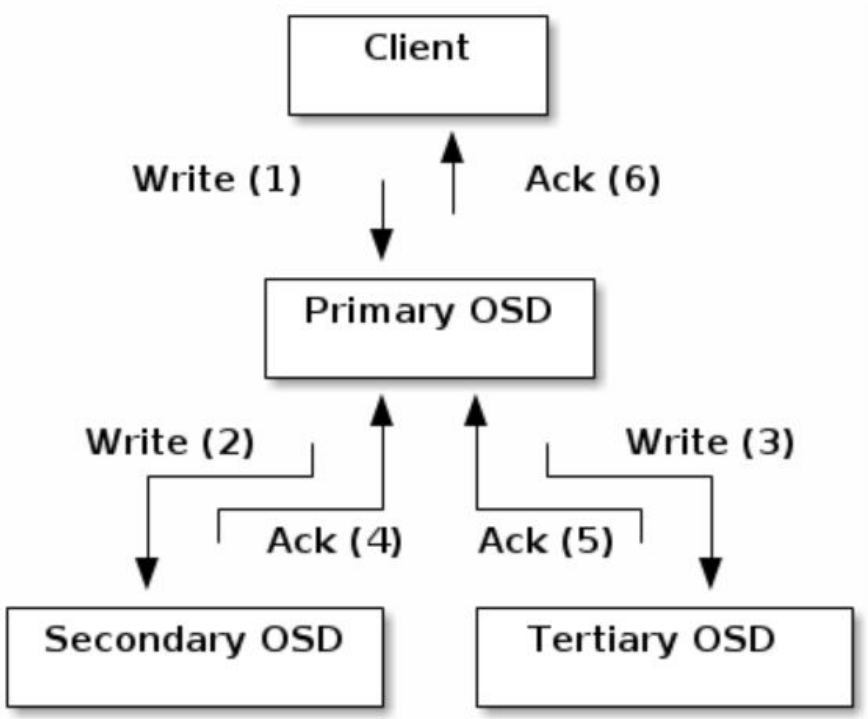
Ceph Architecture: Smart Daemons Enable Hyperscale

- Ceph's OSD Daemons AND Ceph Clients are cluster aware.
- Ceph OSD Daemon knows about other Ceph OSD Daemons in the cluster.
- Ceph OSD Daemon interact directly with other Ceph OSD Daemons and Ceph Monitors.
- Each Ceph daemon utilizes the CPU and RAM of the ceph node.

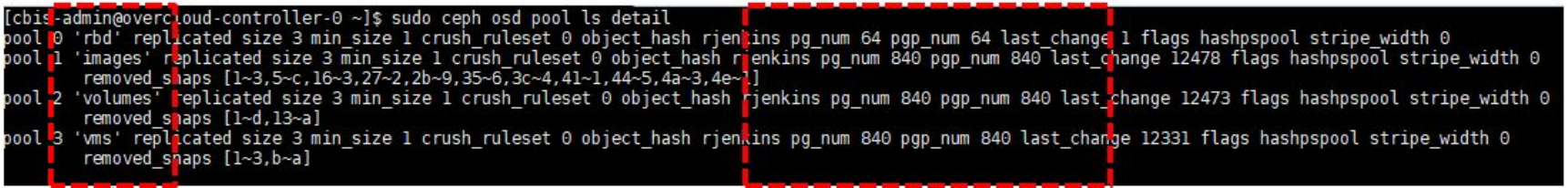
- **OSDs Service Clients Directly**
- **OSD Membership and Status**
- **Data Scrubbing**
- **Replication**



Ceph Architecture: Replication



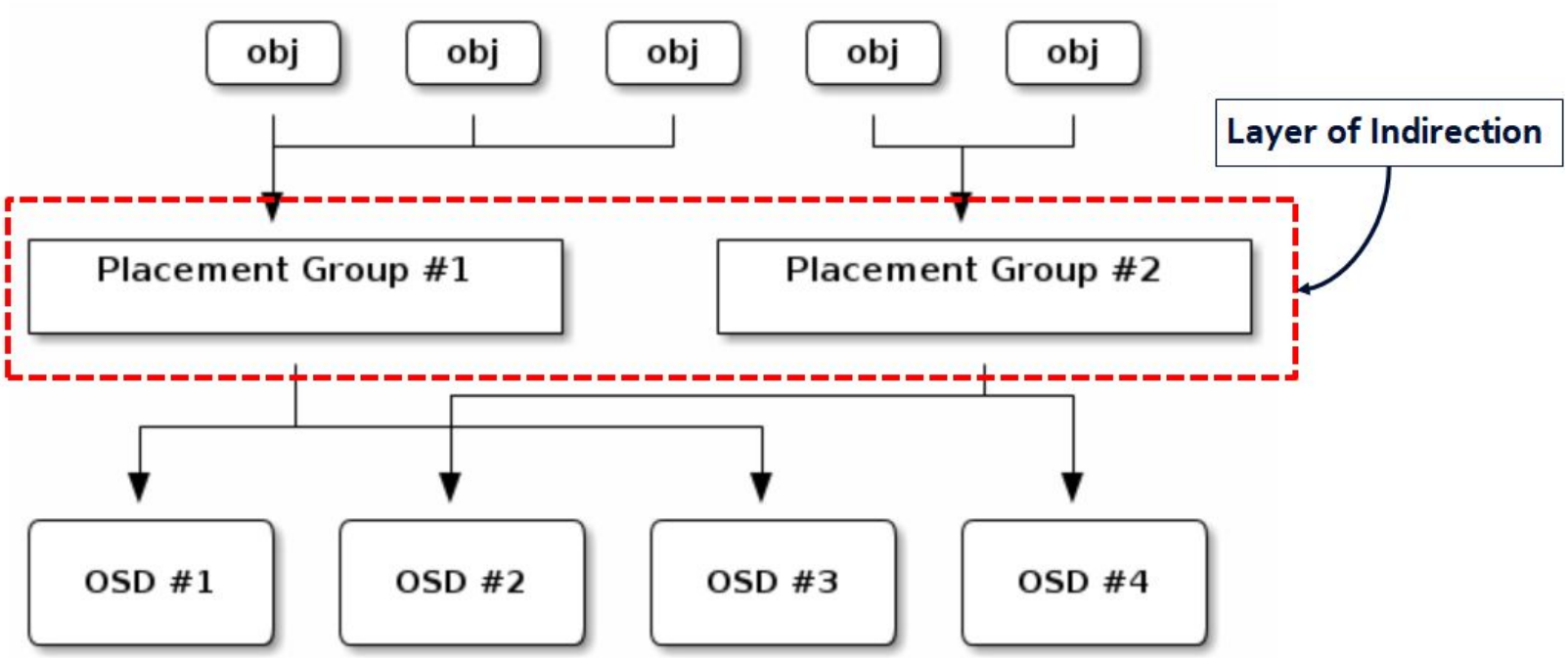
Ceph Architecture: Ceph Pools and Placement Groups



'Pools', which are logical partitions for storing objects

Each pool has a number of **placement groups**. CRUSH maps PGs to OSDs dynamically.

Ceph Architecture: Mapping PGs to OSDs



Ceph Architecture: Calculating PGs

$$\text{Total PGs} = \frac{(\text{OSDs} * 100)}{\text{pool size}}$$

The output value is then rounded to the **nearest power of 2**.
Tip: The nearest power of 2 provides a marginal improvement in efficiency of the CRUSH algorithm.

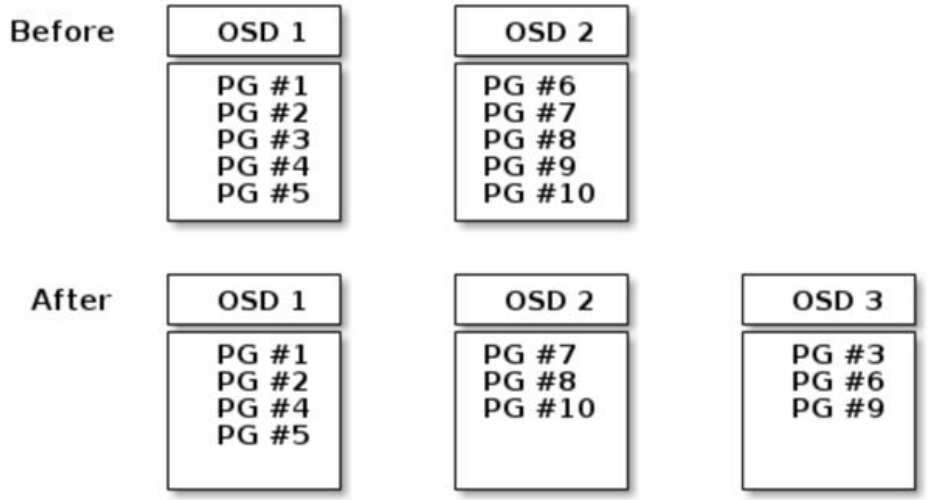
$$\frac{(200 * 100)}{3} = 6667.\ \text{Nearest power of 2: } 8192$$

It's also important to know that the PG count can be increased, but **NEVER** decreased without destroying / recreating the pool. However, increasing the PG Count of a pool is one of the most impactful events in a Ceph Cluster, and should be avoided for production clusters if possible.

https://docs.ceph.com/docs/jewel/rados/operations/placement-groups/
https://ceph.io/pgcalc/

Ceph Architecure: Rebalancing

- When you add a Ceph OSD Daemon to a Ceph Storage Cluster, the cluster map gets updated with the new OSD.
- **What can be said about the load spike when rebalancing is taking place?**



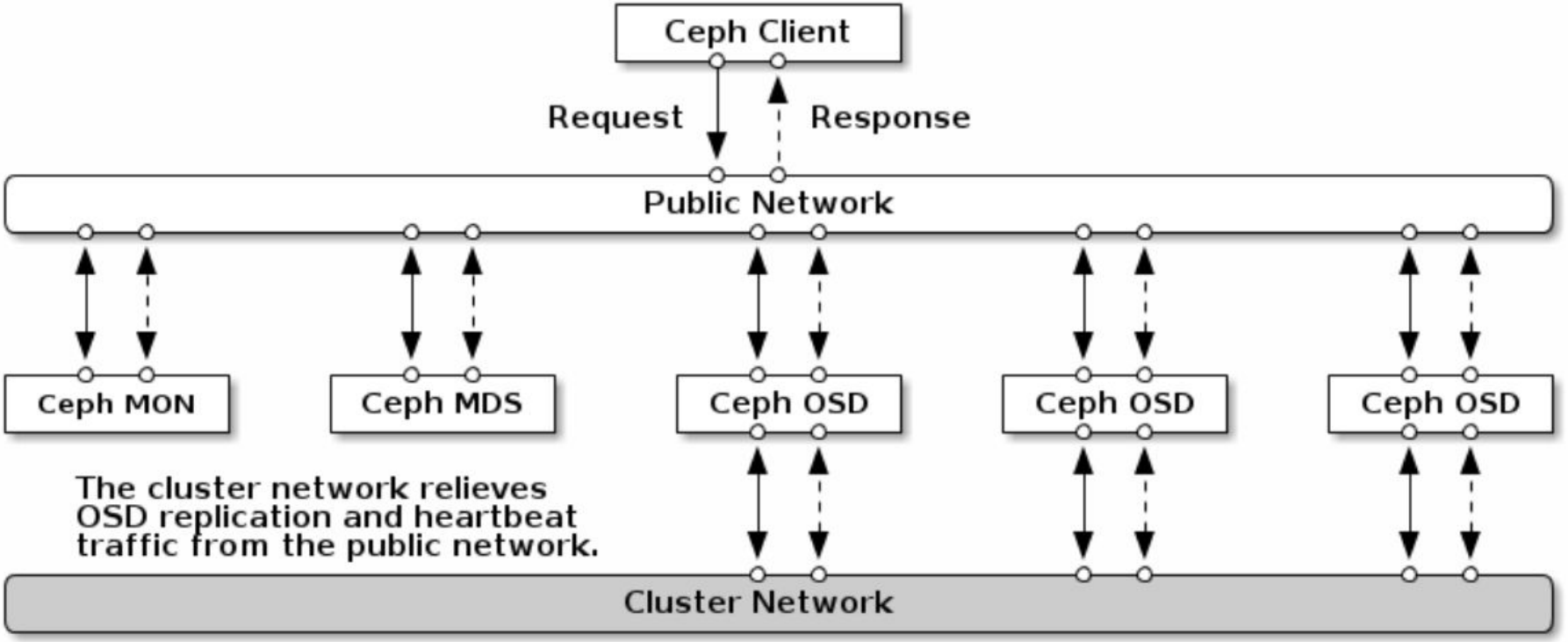
Ceph Architecure: Summary

- Ceph Storage Clusters are dynamic–like a living organism.
- Many storage appliances do not fully utilize the CPU and RAM of a typical commodity server, Ceph does.
- From heartbeats, to peering, to rebalancing the cluster or recovering from faults, Ceph offloads work from clients
- No centralized gateway, that enables it to scale easily

Agenda

1. History
2. Proprietary Storage Appliances vs Opensource Storage
3. Ceph Architecture
4. Deployments

Ceph Deployment: Reference Networking



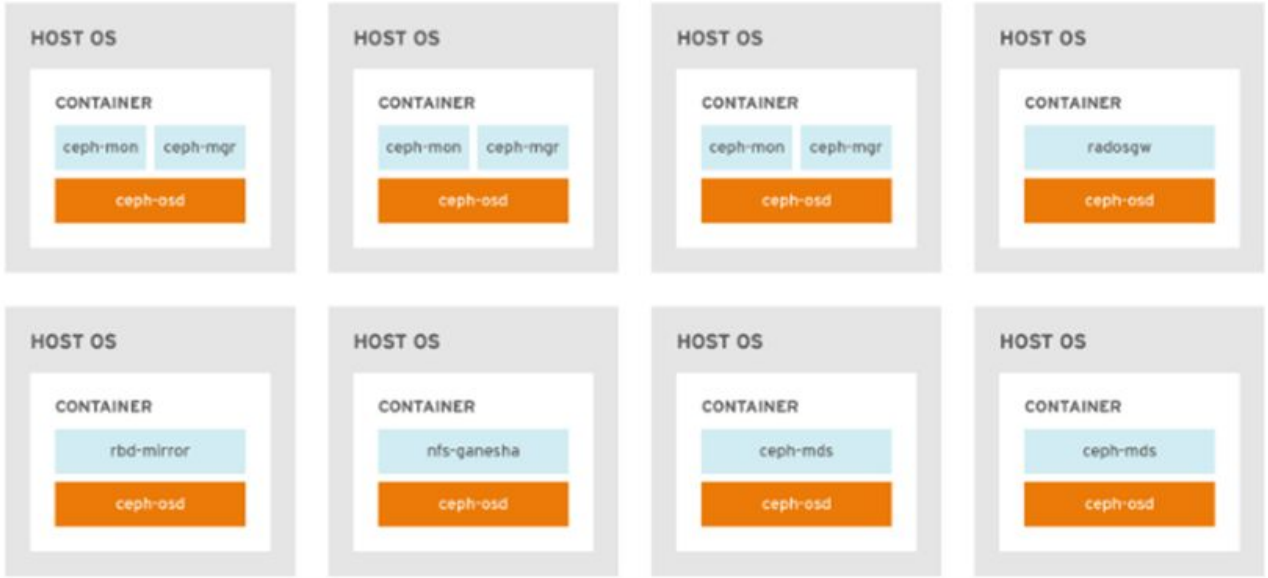
https://docs.ceph.com/docs/master/rados/configuration/network-config-ref/

Ceph Deployment: Minimum Hardware Requirements

Process	Criteria	Minimum Recommended
ceph-osd	Processor	• 1x 64-bit AMD-64 • 1x 32-bit ARM dual-core or better
	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 • 1x 32-bit ARM dual-core or better
	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
	RAM	1 GB minimum per daemon
	Disk Space	1 MB per daemon
	Network	2x 1GB Ethernet NICs

Ceph Deployments: Ceph Daemons placement

Figure 2.1. Colocated Daemons



Ceph Deployments: Ceph Daemons placement

Figure 2.2. Non-colocated Daemons

Ceph Deployment: Failure Domain

- A failure domain is any failure that prevents access to one or more OSDs.
- When planning out your hardware needs, you must balance the temptation to reduce costs by placing too many responsibilities into too few failure domains, and the added costs of isolating every potential failure domain.
- **Where can the knowledge of Failure Domain be helpful in cloud operations?**

Ceph Deployment: OSD Scenarios

- **collocated**: journals are stored alongside data. The cluster will lose performance.
- **non-collocated**: journals are stored on dedicated_devices (in this example both sda and sdb journals will be stored on sdc).

https://docs.ceph.com/ceph-ansible/master/osds/scenarios.html

Ceph: What to expect in new versions?

- Dashboard (ceph-dashboard)
- Containerized Ceph Daemons
- Better performance (IOPs and Latency)

Thank You

Q&A