CEPH come soluzione di storage consolidation

Marica Antonacci - Giacinto Donvito

INFN
Istituto Nazionale
di Fisica Nucleare
Sezione di Bari

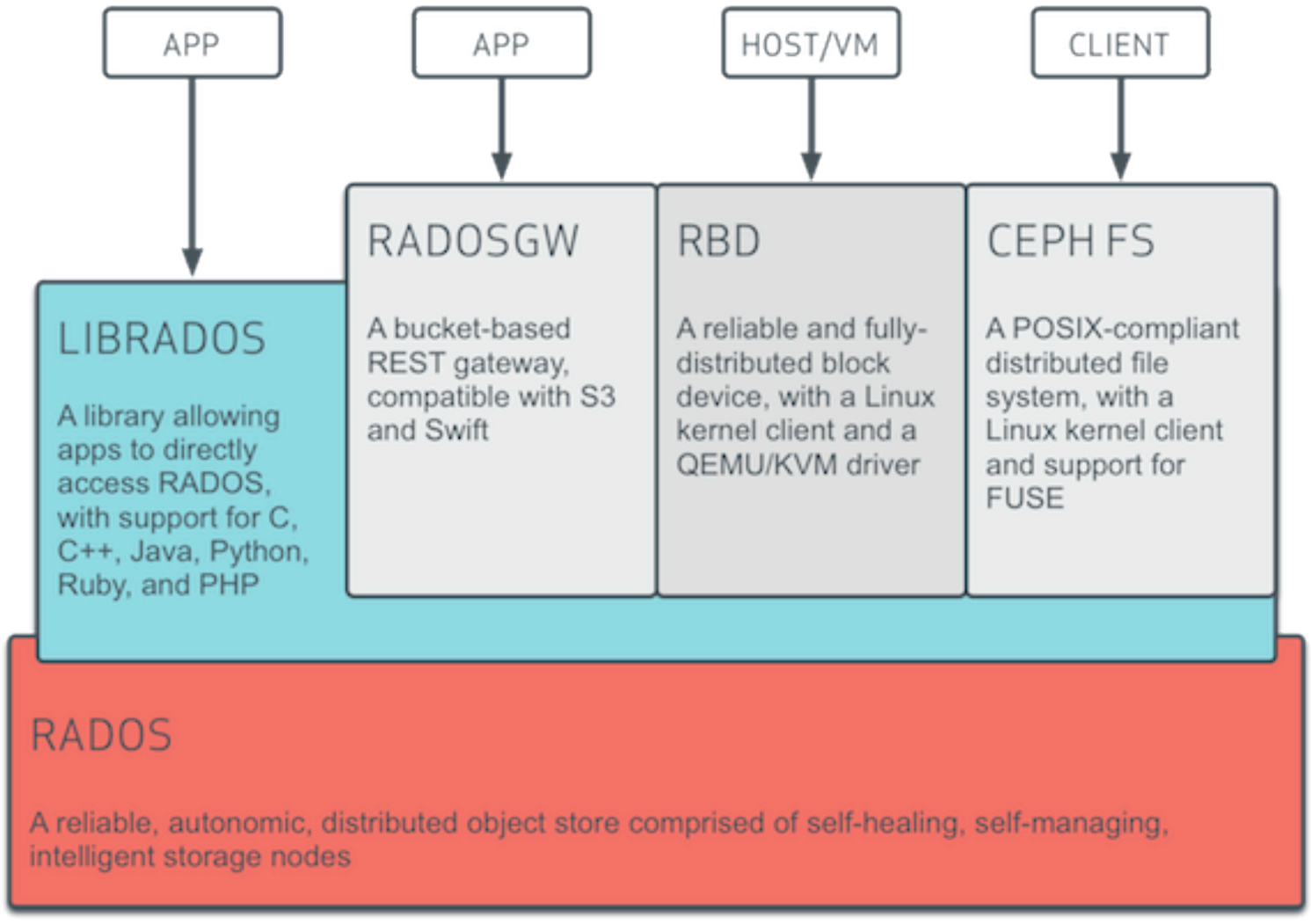
Workshop CCR - LNF
25-29 Maggio 2015

Outline

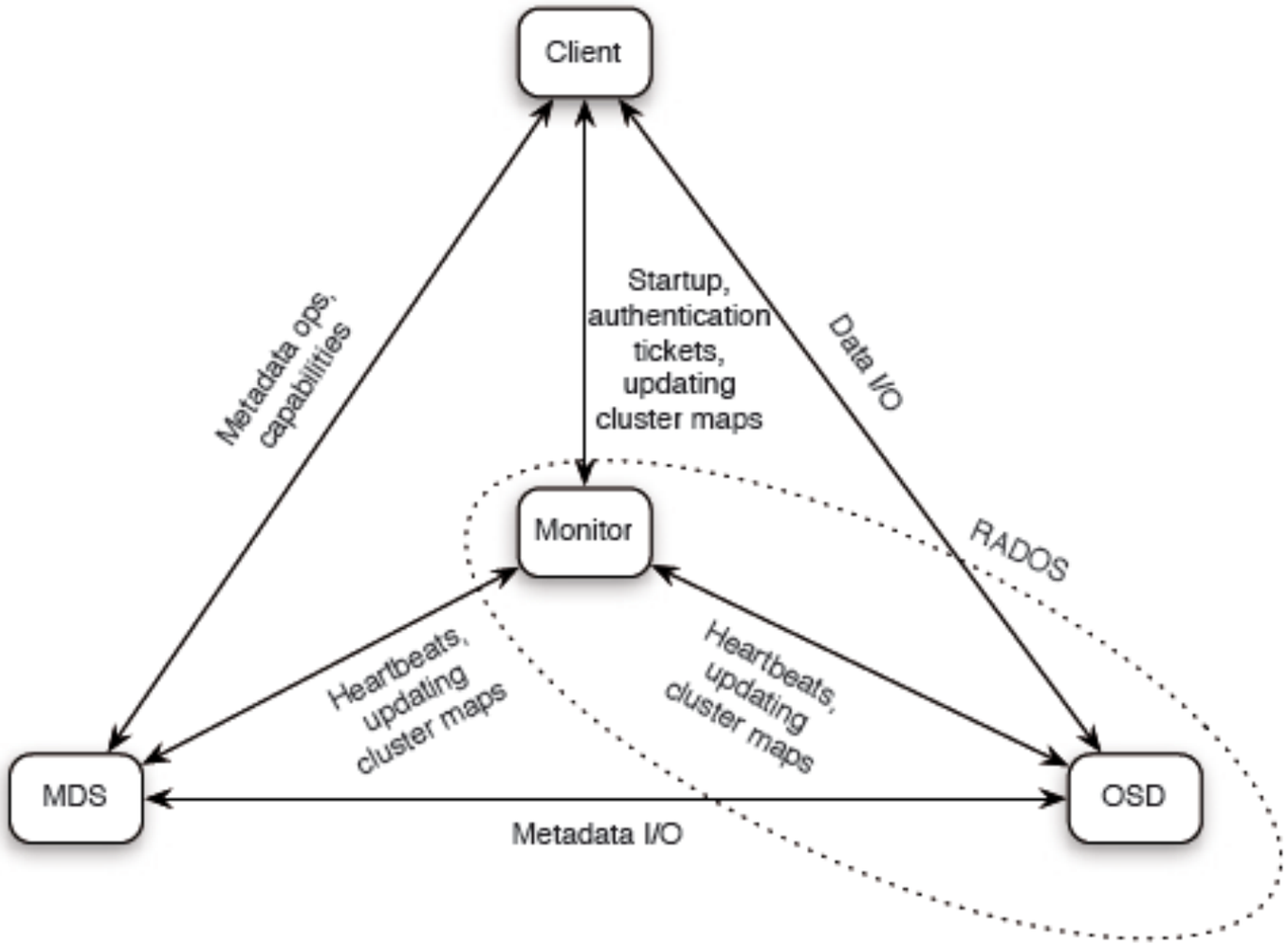
- Ceph - Intro
- Strategies for Performance improvements
- Ceph & Openstack
- Automatic Deployment

Ceph introduction

Ceph is an open-source, massively scalable, software-defined storage system which provides *object*, *block* and *file* system storage in a single platform.

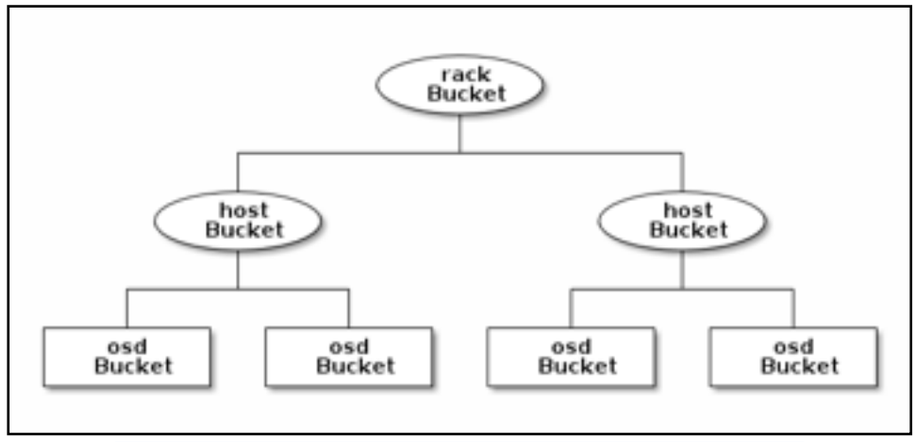


Ceph Architecture

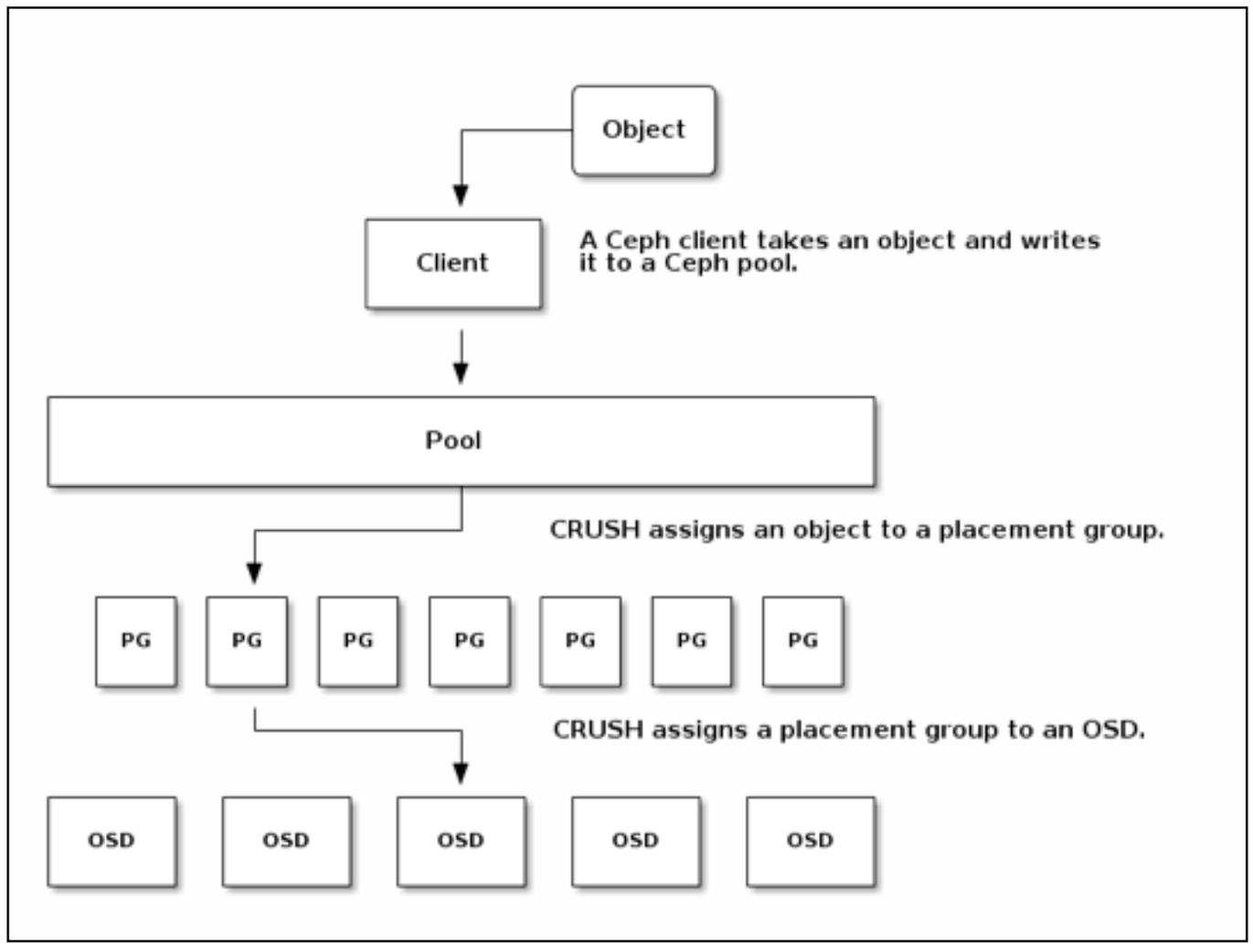


- Ceph **Clients** and Ceph **OSDs** both use the **CRUSH** map and the CRUSH algorithm.

Pools, PGs, CRUSH map

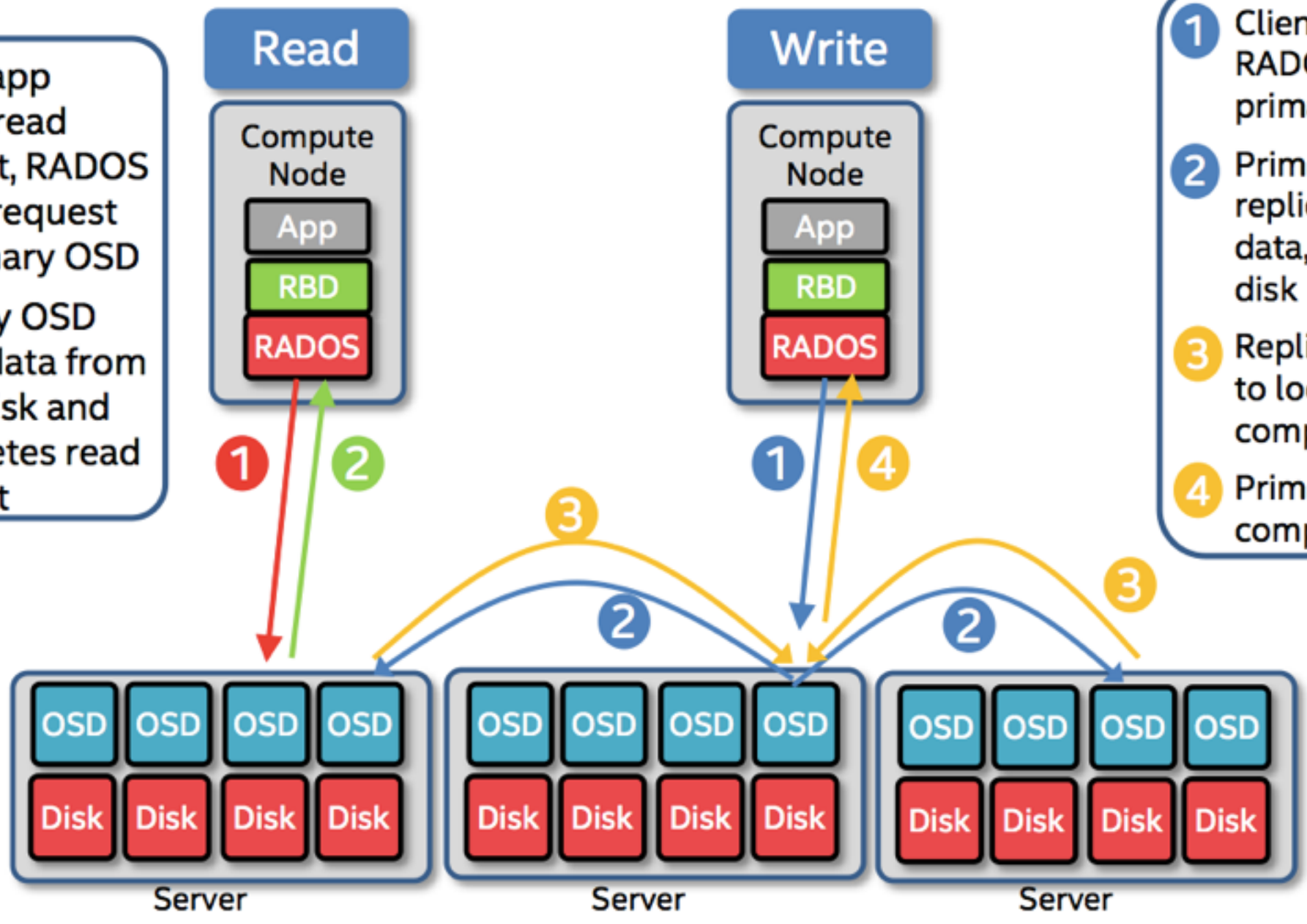



```
host node1 {
        id -1
        alg straw
        hash 0
        item osd.0 weight 1.00
        item osd.1 weight 1.00
}

host node2 {
        id -2
        alg straw
        hash 0
        item osd.2 weight 1.00
        item osd.3 weight 1.00
}

rack rack1 {
        id -3
        alg straw
        hash 0
        item node1 weight 2.00
        item node2 weight 2.00
}
```



Read/Write Flow

1. Client app issues read request, RADOS sends request to primary OSD
2. Primary OSD reads data from local disk and completes read request



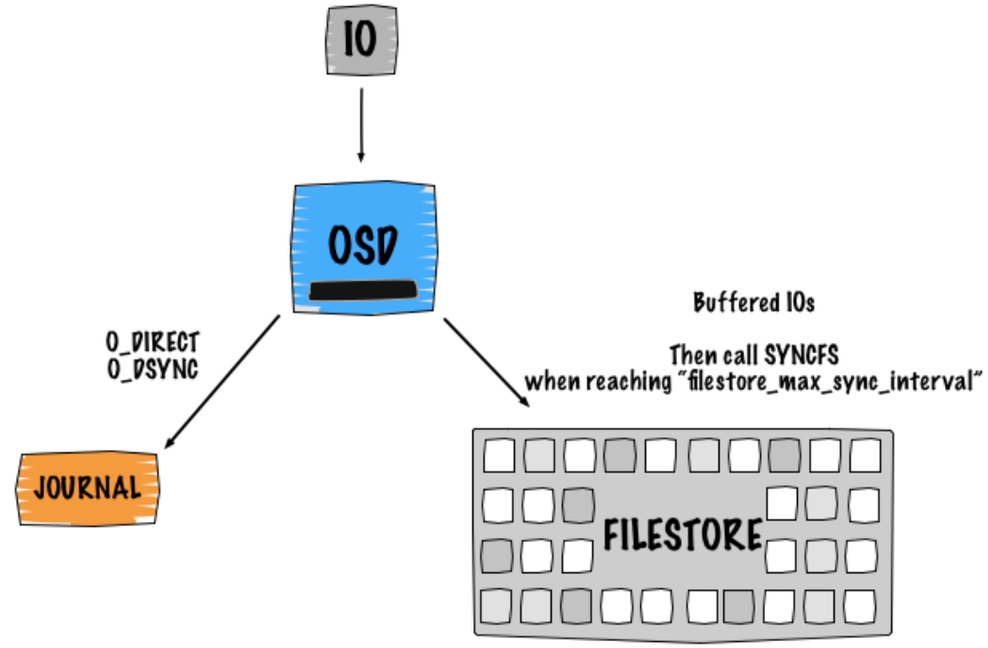
1. Client app writes data, RADOS sends data to primary OSD
2. Primary OSD identifies replica OSDs and sends data, writes data to local disk
3. Replica OSDs write data to local disk, signal completion to primary
4. Primary OSD signals completion to client app

Performance tuning

- We are testing strategies to improve performances:
- **Move OSD Journal to SSD**
  - Ceph guarantees data consistency using writeahead journalling
- **Add a Cache tier**
  - Cache tiering aims to improve the IO performance with the fast storage devices (e.g. SSD) acting as cache for an existing larger pool.

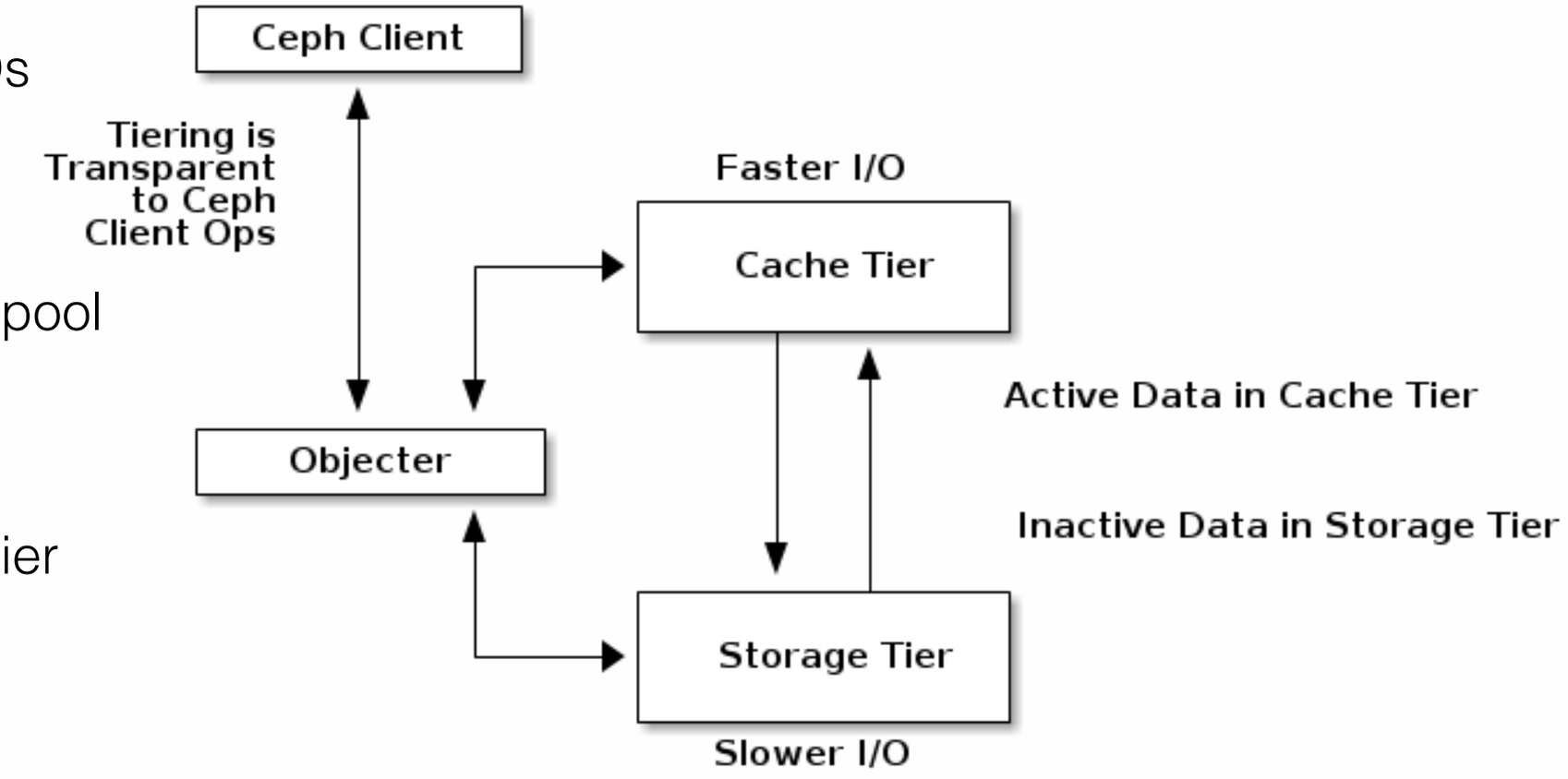
OSD journal

- Ceph writes synchronously to its OSD journal and asynchronously to the OSD filestore
- Everything is written twice
- Deployment question:
  - Shared vs. dedicated journal devices



Cache tiering set-up

- To enable cache tiering, you need to have two pools: the **base tier pool** and the **cache tier pool**.
- The cache tiering setup involves the following five steps:
  - ✓ Adding SSDs as OSDs
  - ✓ Edit the crush map
  - ✓ Create the cache tier pool
  - ✓ Create the cache tier
  - ✓ Configure the cache tier



Important note: Once we enabled the cache tiering, Ceph clients with kernel 3.13 stopped working with 'feature set mismatch' error —> kernel upgrade was needed.

RADOS Performance

Cluster set-up:
3 Monitors
3 storage nodes with 6x OSDs:
- 1x 250GB SSD Samsung 840 EVO
- 5x 550GB SAS HDD 10k rpm

We are testing Ceph Hammer version

```
root@cephsrv01:~# time rados put -p rbd filetest1 obj500M

real    0m6.358s
user    0m0.392s
sys     0m0.636s
root@cephsrv01:~#
root@cephsrv01:~#
root@cephsrv01:~# time rados put -p test filetest2 obj500M

real    0m12.045s
user    0m0.412s
sys     0m0.644s
```

cache tiering enabled for pool rbd

Where are my objects?

```
root@cephsrv01:~# ceph osd map rbd filetest1
osdmap e466 pool 'rbd' (0) object 'filetest1' -> pg 0.25f30571 (0.171) -> up ([0,15,6], p0) acting ([0,15,6], p0)
root@cephsrv01:~#
root@cephsrv01:~# ceph osd map test filetest2
osdmap e466 pool 'test' (6) object 'filetest2' -> pg 6.57f4c5aa (6.1aa) -> up ([9,4,15], p9) acting ([9,4,15], p9)
root@cephsrv01:~#
root@cephsrv01:~#
root@cephsrv01:~# find /var/lib/ceph/osd/ -name filetest* -exec ls -l {} \;
-rw-r--r-- 1 root root 524288000 May 25 22:14 /var/lib/ceph/osd/ceph-5/current/1.71_head/filetest1__head_25F30571__1
-rw-r--r-- 1 root root 524288000 May 25 22:15 /var/lib/ceph/osd/ceph-4/current/6.1aa_head/filetest2__head_57F4C5AA__6
root@cephsrv01:~#
root@cephsrv01:~# ls -l obj500M
-rw-r--r-- 1 root root 524288000 May 25 14:02 obj500M
```

osd.[5,11,17] —> cache tier

Test tools for RBD

- **FIO** (Flexible I/O Tester) with **librbd** support https://telekomcloud.github.io/ceph/2014/02/26/ceph-performance-analysis_fio_rbd.html

```
$ git clone git://git.kernel.dk/fio.git
$ cd fio
$ ./configure
[...]
Rados Block Device engine     yes
[...]
$ make
```

needs librbd-dev

```
[global]
ioengine=rbd
clientname=admin
pool=rbd
rbdname=fio_test
invalidate=0    # mandatory
rw=randwrite
bs=4k

[rbd_iodepth32]
iodepth=32
```

Preparatory step:

```
$ rbd -p rbd create fio_test —size 2048
```

Test tools for RBD (2)

- IOzone Filesystem Benchmark

```
$ iozone -r $BS -I -i 0 -i 1 -i 2 -t $THREADS -s $FILESIZE

-i 0=write/rewrite, 1=read/re-read, 2=random-read/write
```

- Preparatory steps:

$ rbd -p rbd create iozone_test —size 2048

$ rbd -p rbd map iozone_test

/dev/rbd0

$ mkfs.ext4 /dev/rbd0

$ mount /dev/rbd0 /ceph-test

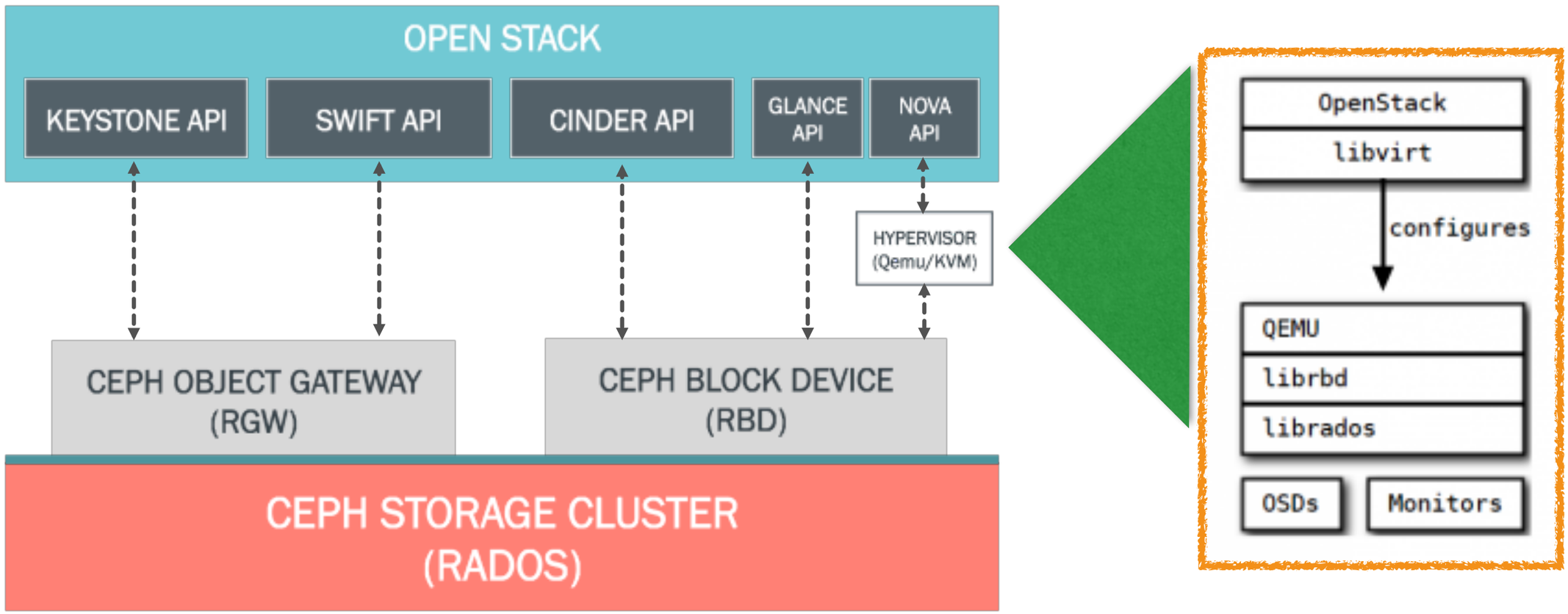
Cache flush & evict

- During tests we needed to manually clean the cache:

```
1. ceph osd tier cache-mode {cachepool} forward
2. rados -p {cachepool} cache-flush-evict-all
```

- Check the status with “rados df”
- When the cache is empty, restore the **`writeback`** cache mode

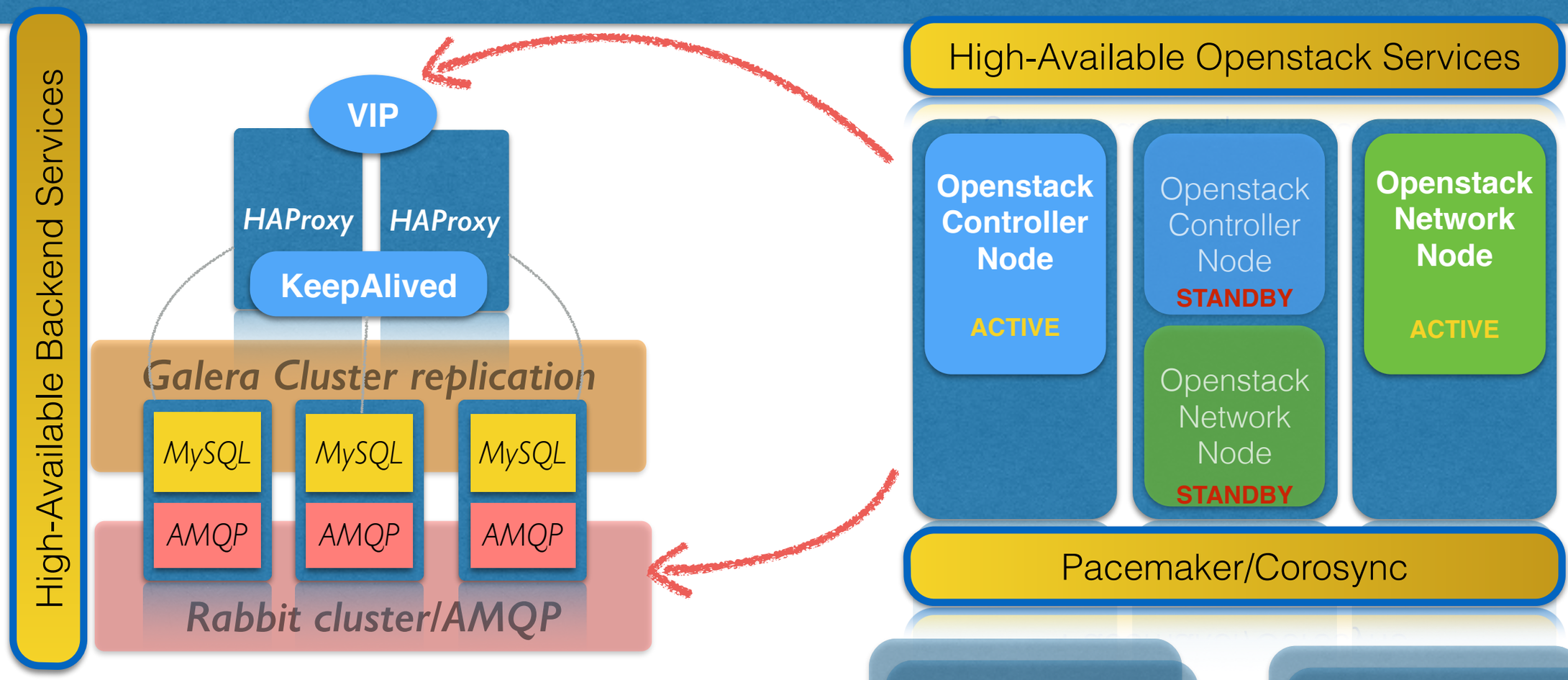
Ceph & Openstack



Ceph benefits:

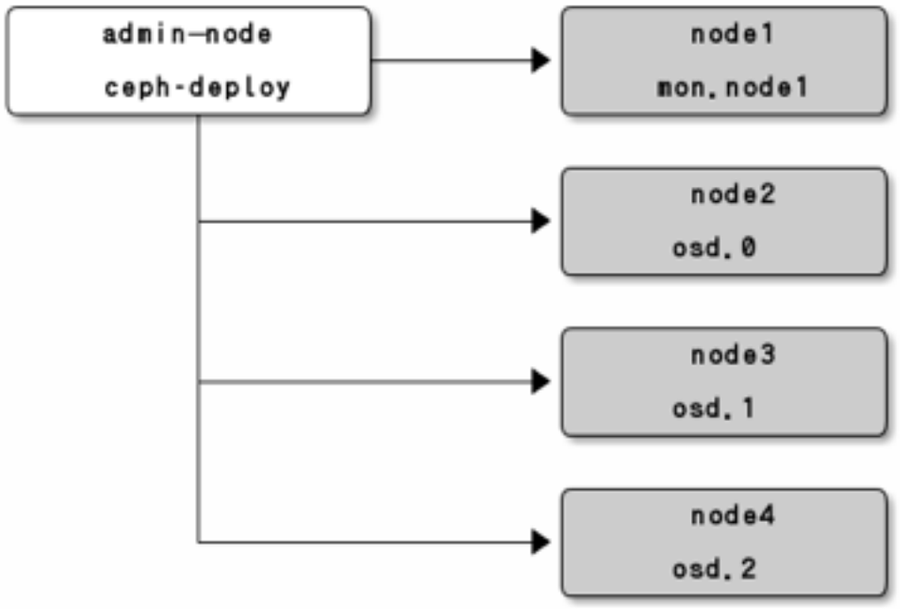
- Multi-node striping and redundancy for block storage (Cinder volumes and Nova ephemeral drives)
- Copy-on-write cloning of images to volumes and instances
- Unified storage pool for all types of storage (object, block, and POSIX)
- Live migration of Ceph-backed instances

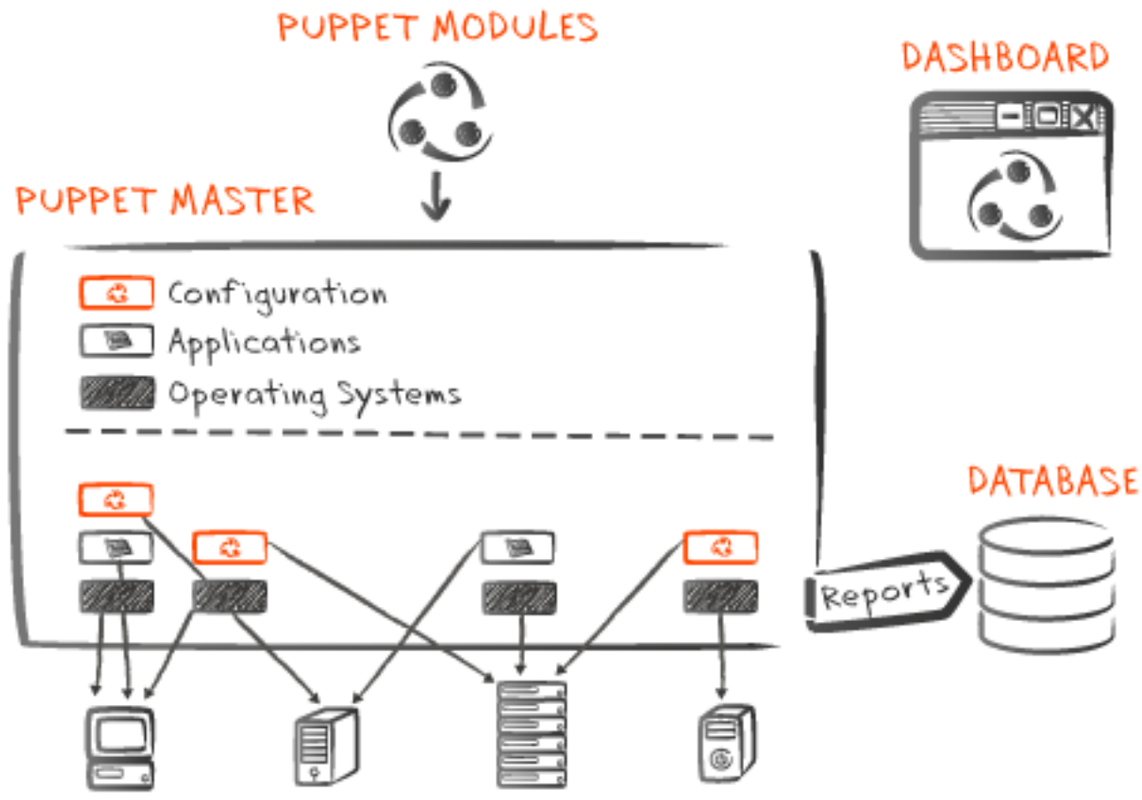
ReCaS/PRISMA Testbed



Ceph cluster deployment

From ceph-deploy to Puppet…





- Manage your cluster with a centralized and scalable architecture

Puppet modules

- We are testing the Puppet modules for Ceph:
  - https://github.com/stackforge/puppet-ceph.git
- We managed to deploy a cluster with basic configuration:
  - MONs, OSDs (MDS to be done, but we are not very interested in using it)
- We managed to configure Openstack components (glance, nova, cinder) to use Ceph as backend storage

Puppet module: classes & resources

- Ceph classes

```
---
stack-ceph:
    - ceph::profile::base
    - ceph::profile::client


ceph::fsid: 'f65809d3-7961-4cd7-b731-a9bc94bc6e9c'
ceph::mon_initial_members: 'ceph01,ceph02,ceph03'
ceph::mon_host: '172.20.0.74,172.20.0.75,172.20.0.76'
```

- Ceph resources

```
node 'ceph01.school.cloud.ba.infn.it' {

  hiera_include('stack-ceph')


  $mon = hiera('ceph_mon')
  create_resources('ceph::mon', $mon)
  $key = hiera('ceph_key')
  create_resources('ceph::key', $key)
  $osd = hiera('ceph_osd')
  create_resources('ceph::osd', $osd)

  $ceph_pool = hiera('ceph_pools')
  create_resources('ceph::pool', $ceph_pool)
```

Puppet module: configuration

```
ceph_mon:
  'ceph01':
    key: 'AQBxkvVU4F+VDBAArxUf+8s0LbxIxNrbyEC1kw=='
ceph_key:
  'client.admin':
    secret: 'AQDgL/hUSC2kLBAAnWJaSiqJG+YMk+XV9sapnw=='
    cap_mon: 'allow *'
    cap_osd: 'allow *'
    cap_mds: 'allow'
    inject: true
    inject_as_id: 'mon.'
    inject_keyring: '/var/lib/ceph/mon/ceph-ceph01/keyring'
  'client.bootstrap-osd':
    secret: 'AQDlL/hUUCpdFBAAZeo6mKj4yeKPmVKfUY5awA=='
    cap_mon: 'allow profile bootstrap-osd'
    keyring_path: '/var/lib/ceph/bootstrap-osd/ceph.keyring'
    inject: true
    inject_as_id: 'mon.'
    inject_keyring: '/var/lib/ceph/mon/ceph-ceph01/keyring'
  'client.cinder':
    secret: 'AQAvxQpVKJ03KxAADFv78tedrAWZx1SoRdsQUA=='
    cap_mon: 'allow r'
    cap_osd: 'allow class-read object_prefix rbd_children, allow rwx pool=cephPuppetDeployed, allow rwx pool=vms,
    mode: '644'
    inject: true
    inject_as_id: 'mon.'
    inject_keyring: '/var/lib/ceph/mon/ceph-ceph01/keyring'


ceph_pools:
  volumes:
    pg_num: 128
  images:
    pg_num: 128
  vms:
    pg_num: 128
```

- Configuration data for: Keyrings, MONs, OSDs, Pools

```
ceph_osd:
  '/dev/vdb':
    journal: '/data1'
```

RBD backup

- We are using a cron-job to implement the automatic backup of RBD volumes using the RBD import/export features

- The script scans all volumes in a SOURCE pool and replicates them in another pool (DEST pool)

- For Disaster Recovery: run two simultaneous Ceph clusters in different geographic locations and generate and transfer only the delta:

```
rbd export-diff --from-snap snap1 pool/image@snap2 - | ssh
user@second_cluster rbd import-diff - pool2/image
```

Conclusions

- Currently we are testing Ceph for
  - farm posix filesystem (cephFS)
  - backend storage for the cloud services provided by our Openstack-based IaaS (glance, cinder, nova)
- We are evaluating strategies for improving performances:
  - OSD journal on SSD
  - cache tiering
- We are starting to use Puppet for cluster installation, configuration and management

People involved

- Marica Antonacci (INFN-BA)
- Giacinto Donvito (INFN-BA)
- Alessandro Italiano (INFN-BA)
- Fabrizio Ventola (UNIBA)