Build an High-Performance and High-Durable Block Storage Service Based on Ceph

Rongze & Haomai
[rongze | haomai]@unitedstack.com

UnitedStack

CONTENTS

01

THE FIRST PART

Block Storage Service

Block Storage Service Highlight

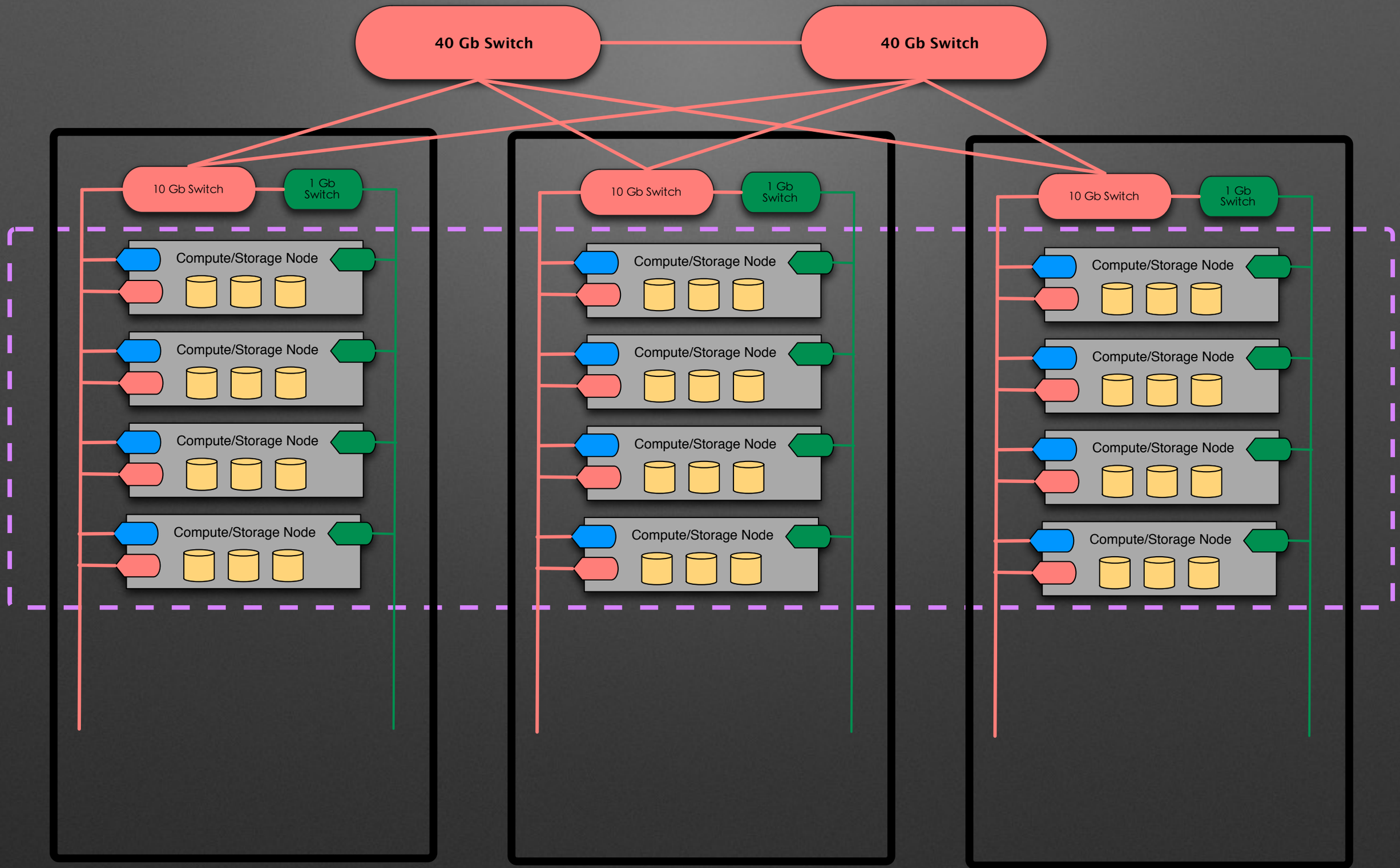
- 6000 IOPS 170 MB/s 95% < 2ms SLA
- 3 copys, strong consistency, 99.99999999% durability
- All management ops in seconds
- Real-time snapshot
- Performance volume type and capacity volume type

Software used

				Now	
OpenStack	Essex	Folsom	Havana	Icehouse/ Juno	Juno
Ceph	0.42	0.67.2	base on 0.67.5	base on 0.67.5	base on 0.80.7
CentOS		6.4	6.5	6.5	6.6
Qemu	0.12	0.12	base on 1.2	base on 1.2	2.0
Kernel		2.6.32	3.12.21	3.12.21	?

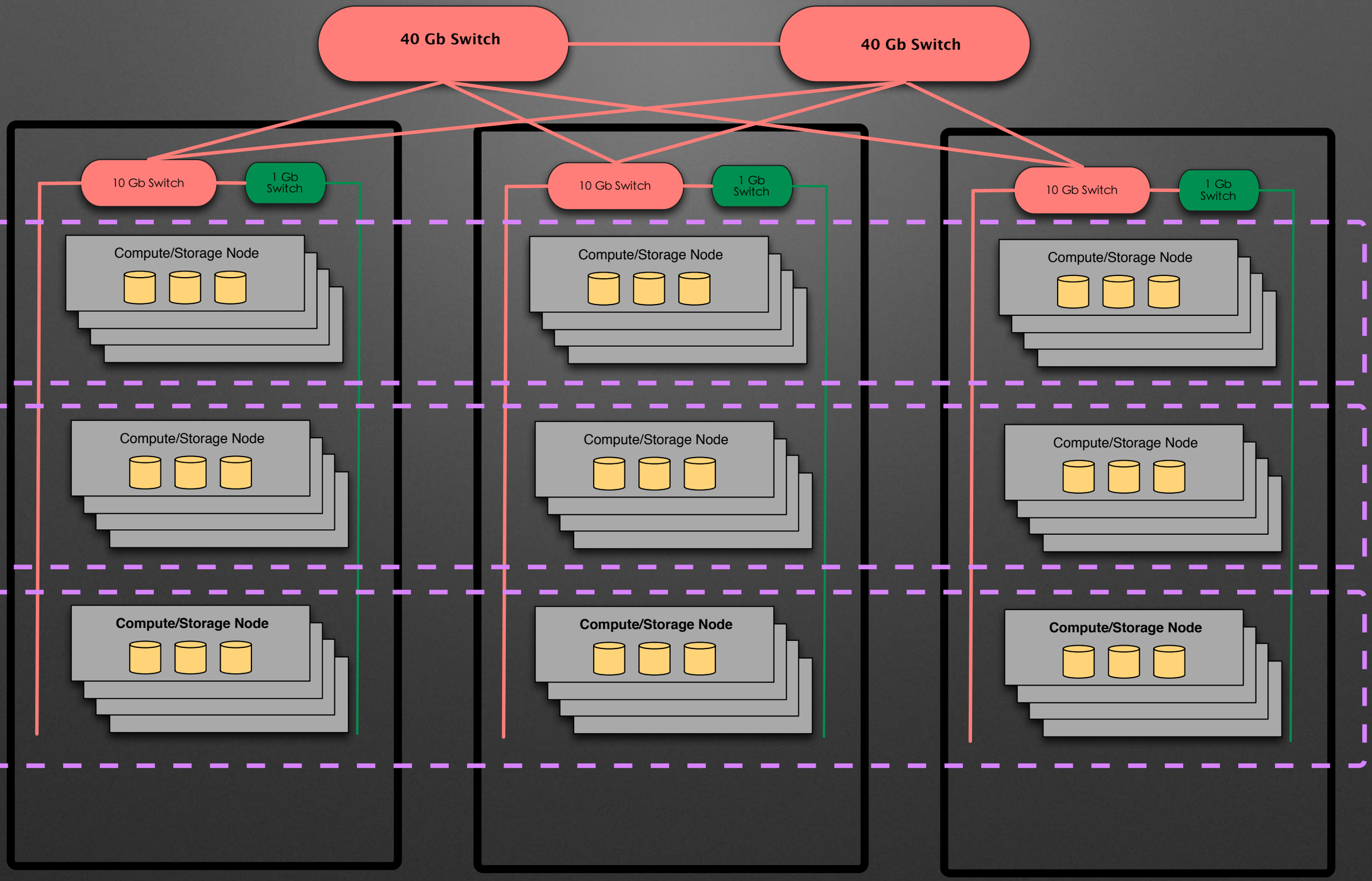
Deployment Architecture

minimum deployment
12 osd nodes and 3 monitor nodes



Scale-out

the minimum scale deployment
12 osd nodes

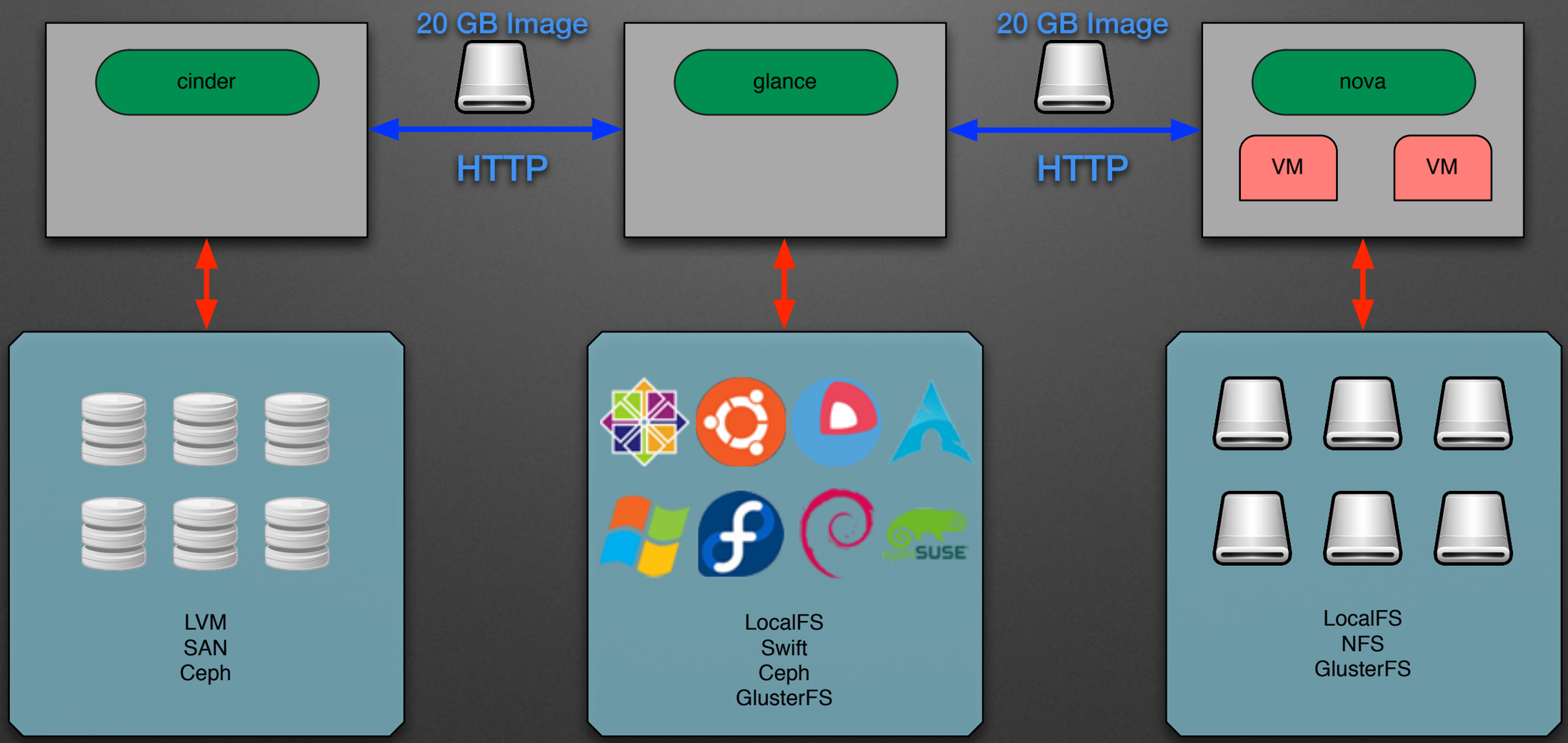


144 osd nodes

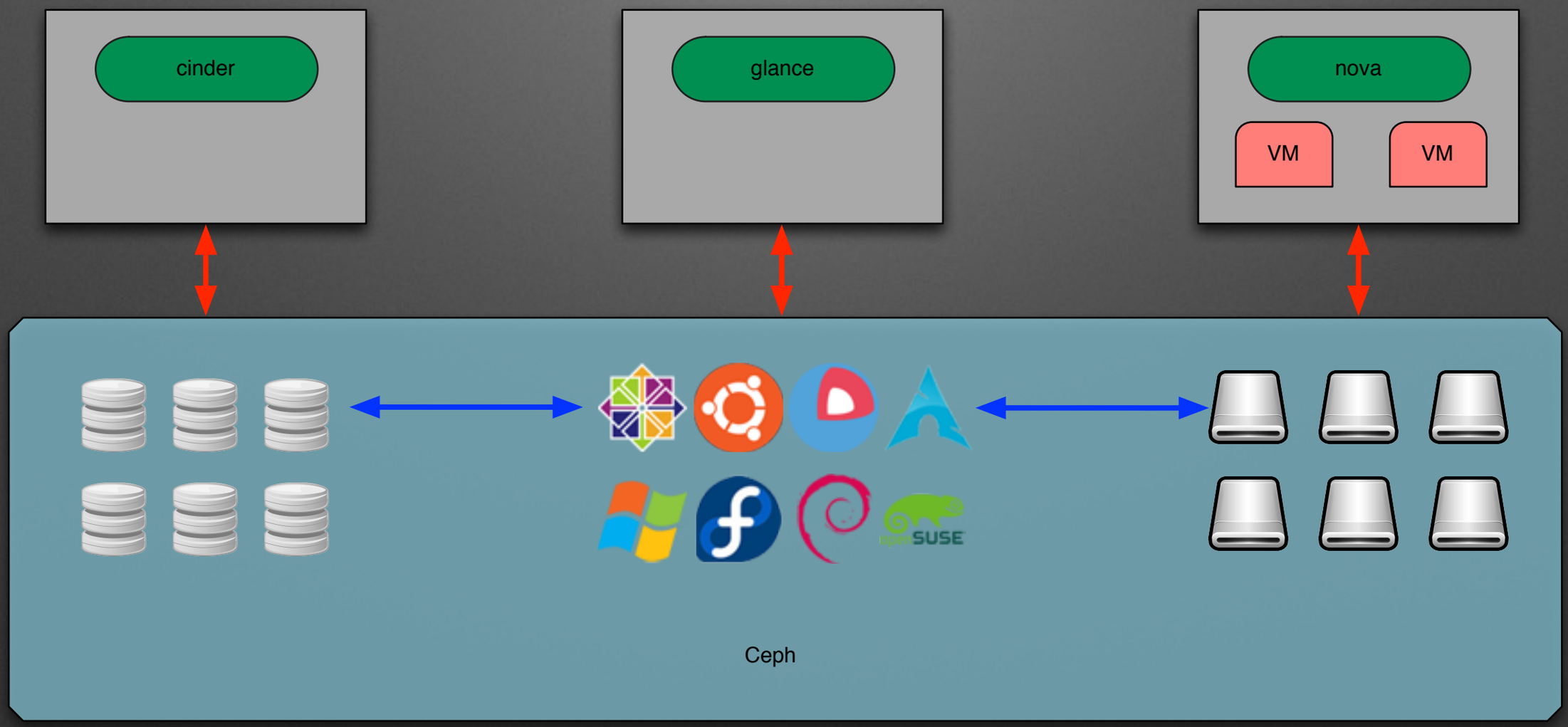
OpenStack

1 Gb Network: 20 GB / 100 MB = 200 s = 3 mins
10 Gb Network: 20 GB / 1000 MB = 20 s

Boot Storm



Nova, Glance, Cinder use the same ceph pool
All action in seconds
No boot storm



QoS

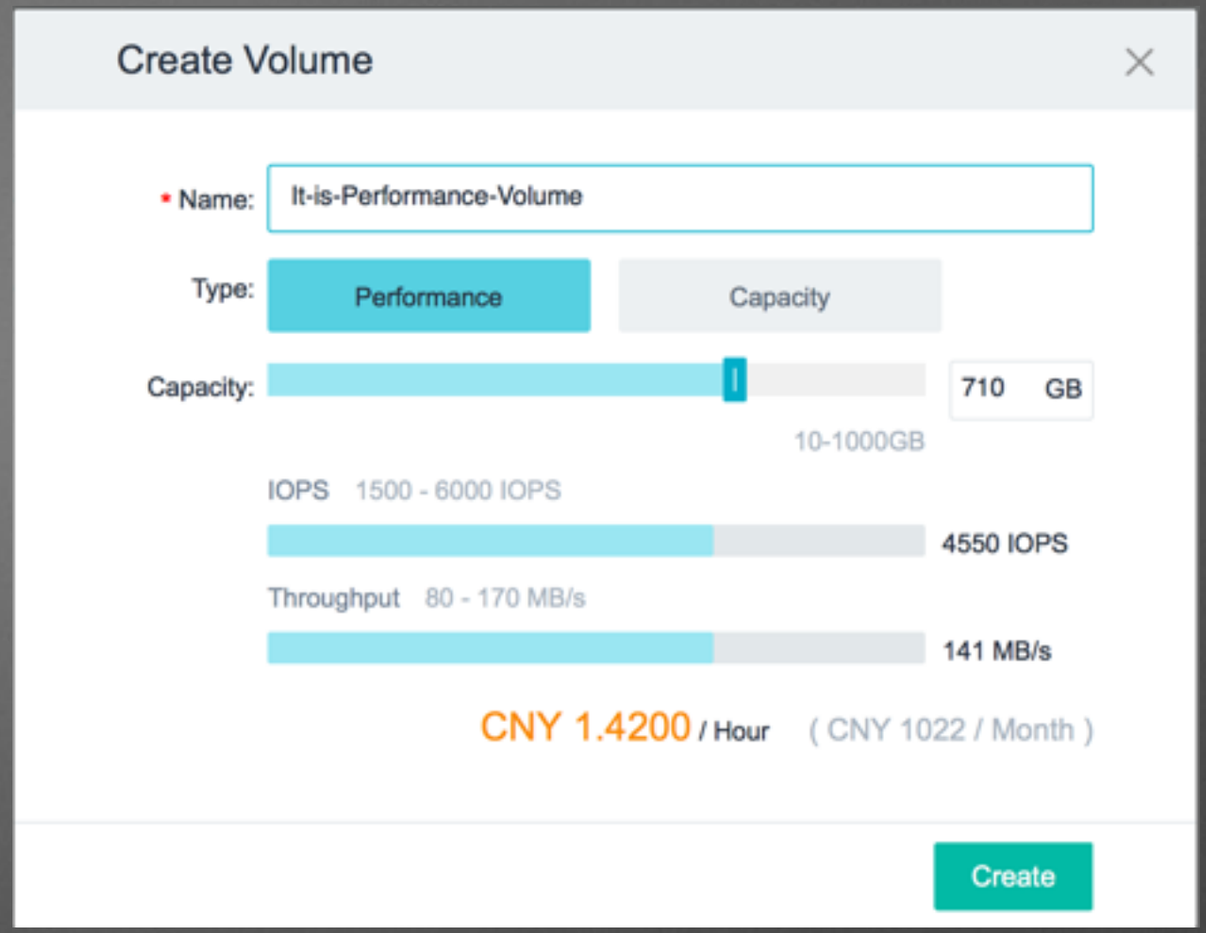
- Nova
- Libvirt
- Qemu(throttle)

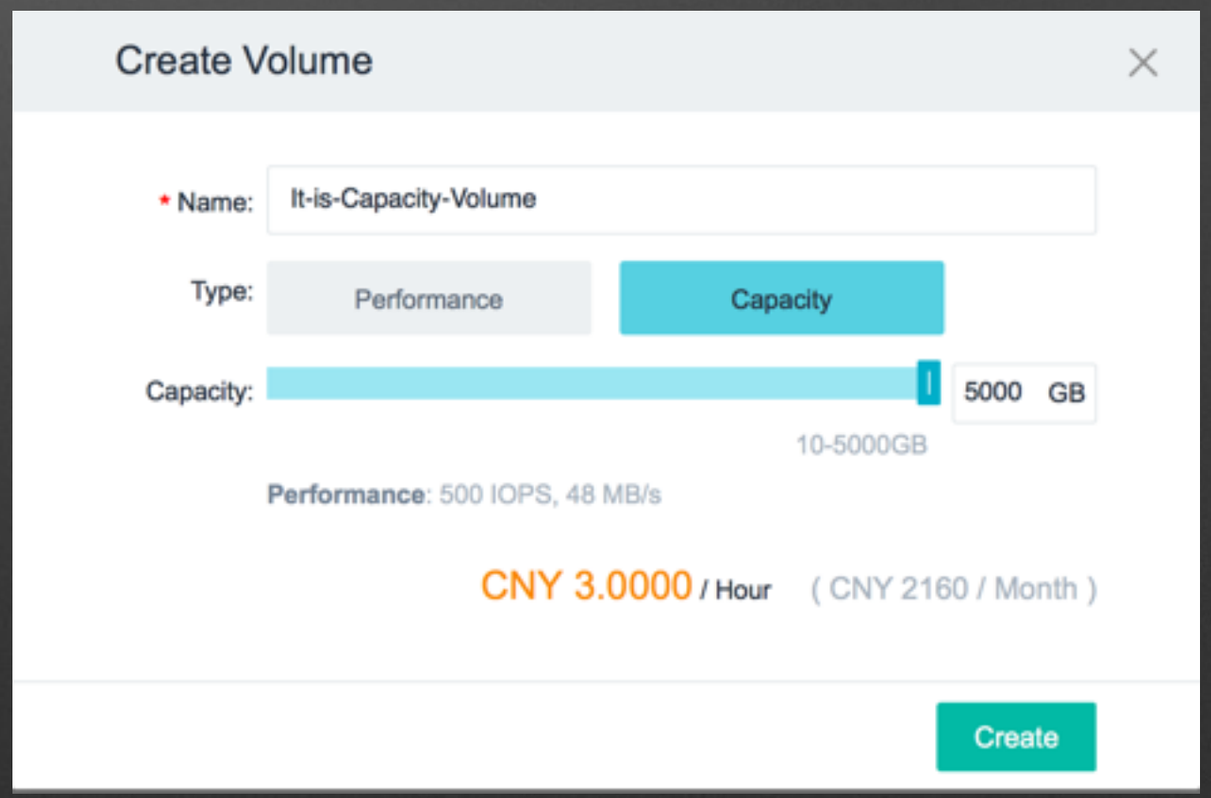
Two Volume Types

- Cinder multi-backend
  - Ceph SSD Pool
  - Ceph SATA Pool

Shared Volume

- Read Only
- Multi-attach





02

THE SECOND PART

High Performance

OS configure

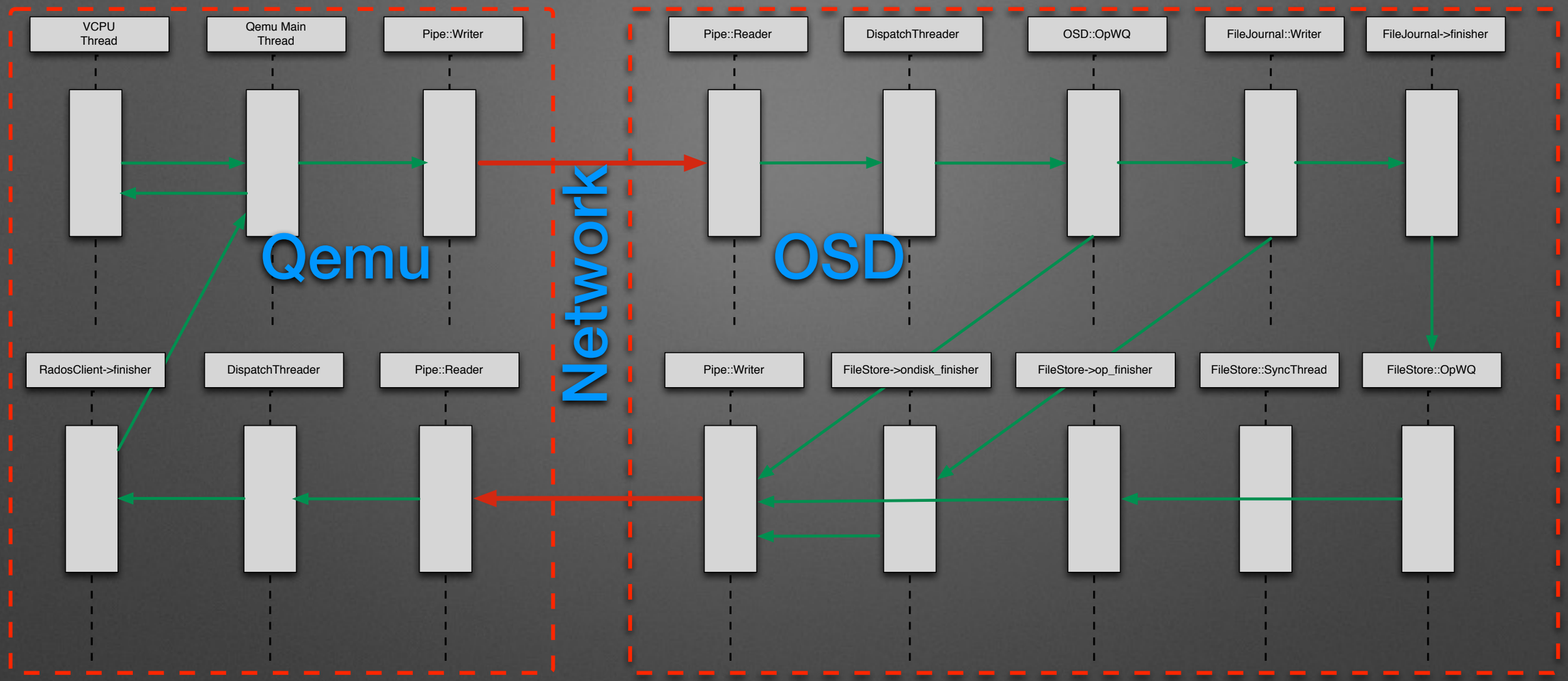
- CPU:
  - Get out of CPU out of power save mode:
    - echo performance | tee /sys/devices/system/cpu/cpu*/cpufreq/scaling_governor >/dev/null
  - Cgroup:
    - Bind Ceph-OSD processes to fixed cores(1-2 cores per OSD)
- Memory:
  - Turn off NUMA if support NUMA in /etc/grub.conf
  - Set vm.swappiness = 0
- Block:
  - echo deadline > /sys/block/sd[x]/queue/scheduler
- FileSystem
  - Mount with “noatime nobarrier"

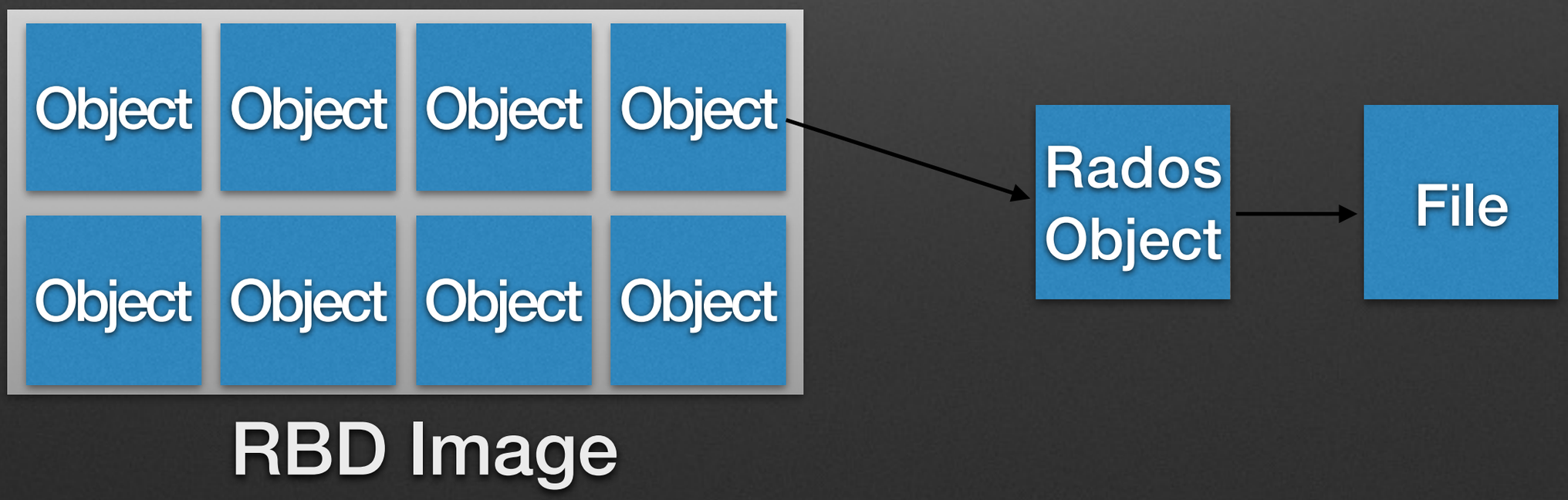
Qemu

- Throttle: Smooth IO limit algorithm(backport)
- RBD enhance: Discard and flush enhance(backport)
- Burst:
- Virt-scsi: Multi-queue support

Ceph IO Stack

data flow


VCPU Thread
Qemu Main Thread
Pipe::Writer
Pipe::Reader
DispatchThreader
OSD::OpWQ
FileJournal::Writer
FileJournal->finisher
Qemu
Network
OSD
RadosClient->finisher
DispatchThreader
Pipe::Reader
Pipe::Writer
FileStore->ondisk_finisher
FileStore->op_finisher
FileStore::SyncThread
FileStore::OpWQ


Object
Object
Object
Object
Object
Object
Object
Object
Rados Object
File
RBD Image

Ceph Optimization

Rule 1: Keep FD

- Facts:
  - FileStore Bottleneck: Remarkable performance degraded when FD cache missed
  - SSD = 480GB = 122880 Objects(4MB) = 30720 objects(16MB) in theory
- Action:
  - Increase FDCache/OMapHeader to very large to hold all objects
  - Increase object size to 16MB instead of 4MB(default)
  - Improve default OS fd limits
- Configuration:
  - “filestore_omap_header_cache_size”
  - “filestore_fd_cache_size”
  - “rbd_chunk_size”(OpenStack Cinder)

Rule 2: Sparse Read/Write

- Facts:
  - Only few KB exists in Object for RBD usage
  - Clone/Recovery will copy full object, harmful to performance and capacity
- Action:
  - Use sparse read/write
- Problem:
  - XFS or other local filesystems exists existing bugs for fiemap
- Configuration:
  - “filestore_fiemap=true”

Rule 3: Drop default limits

- Facts:
  - Default configuration value is suitable for HDD backend
- Action:
  - Change all throttle-related configuration value
- Configuration:
  - “filestore_wbthrottle_*”
  - “filestore_queue_*”
  - “journal_queue_*”
  - “…” More related configs(recovery, scrub)

Rule 4: Use RBD cache

- Facts:
  - RBD cache has remarkable performance improvement for seq read/write
- Action:
  - Enable RBD cache
- Configuration:
  - “rbd_cache = true”

Rule 5: Speed Cache

- Facts:
  - Default cache container implementation isn’t suitable for large cache capacity
- Temporary Action:
  - Change cache container to “RandomCache” (Out of Master Branch)
  - FDCache, OMap header cache, ObjectCacher
- Next:
  - RandomCache isn’t suitable for generic situations
  - Implementation Effective ARC replacing RandomCache

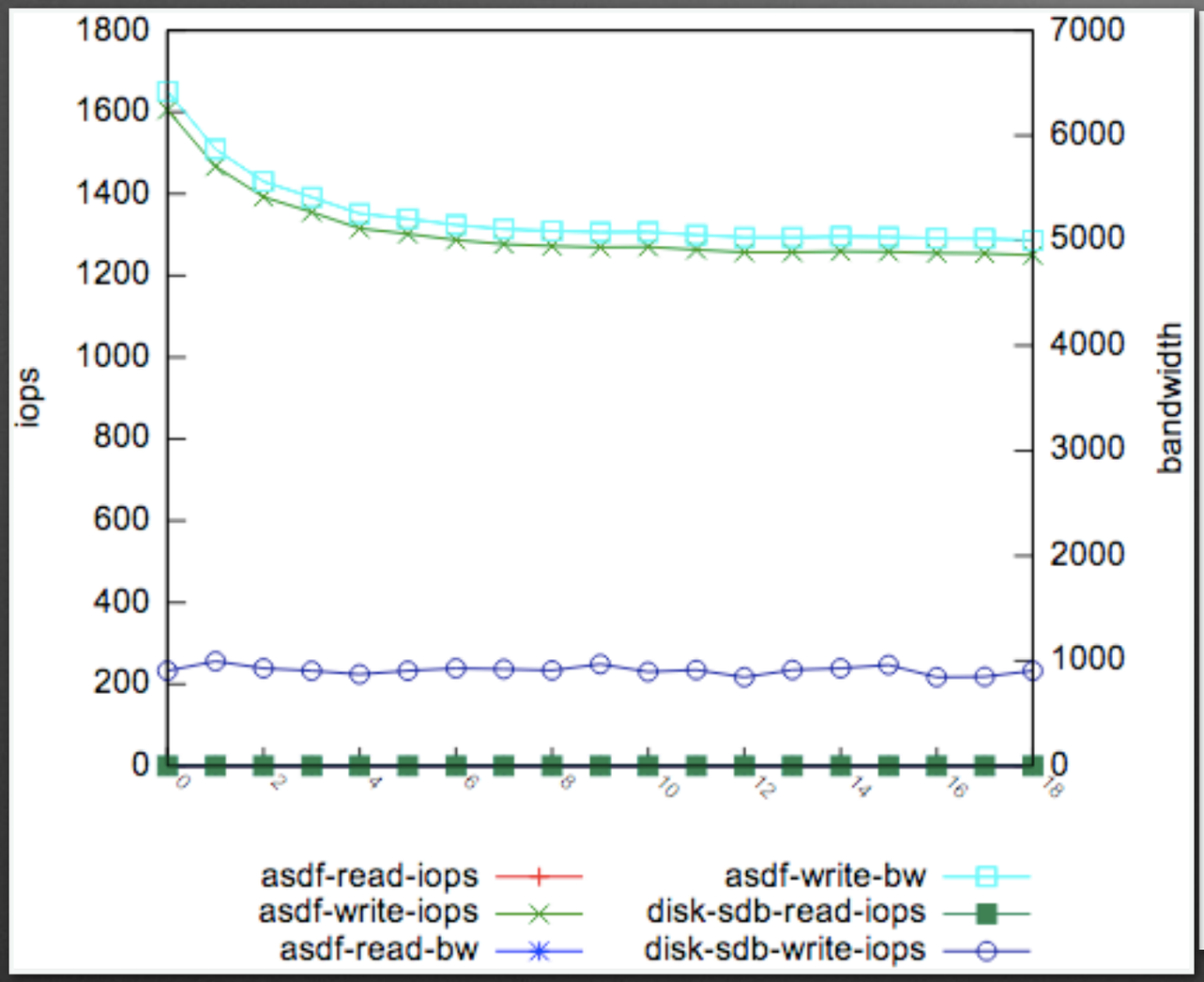
Rule 6: Keep Thread Running

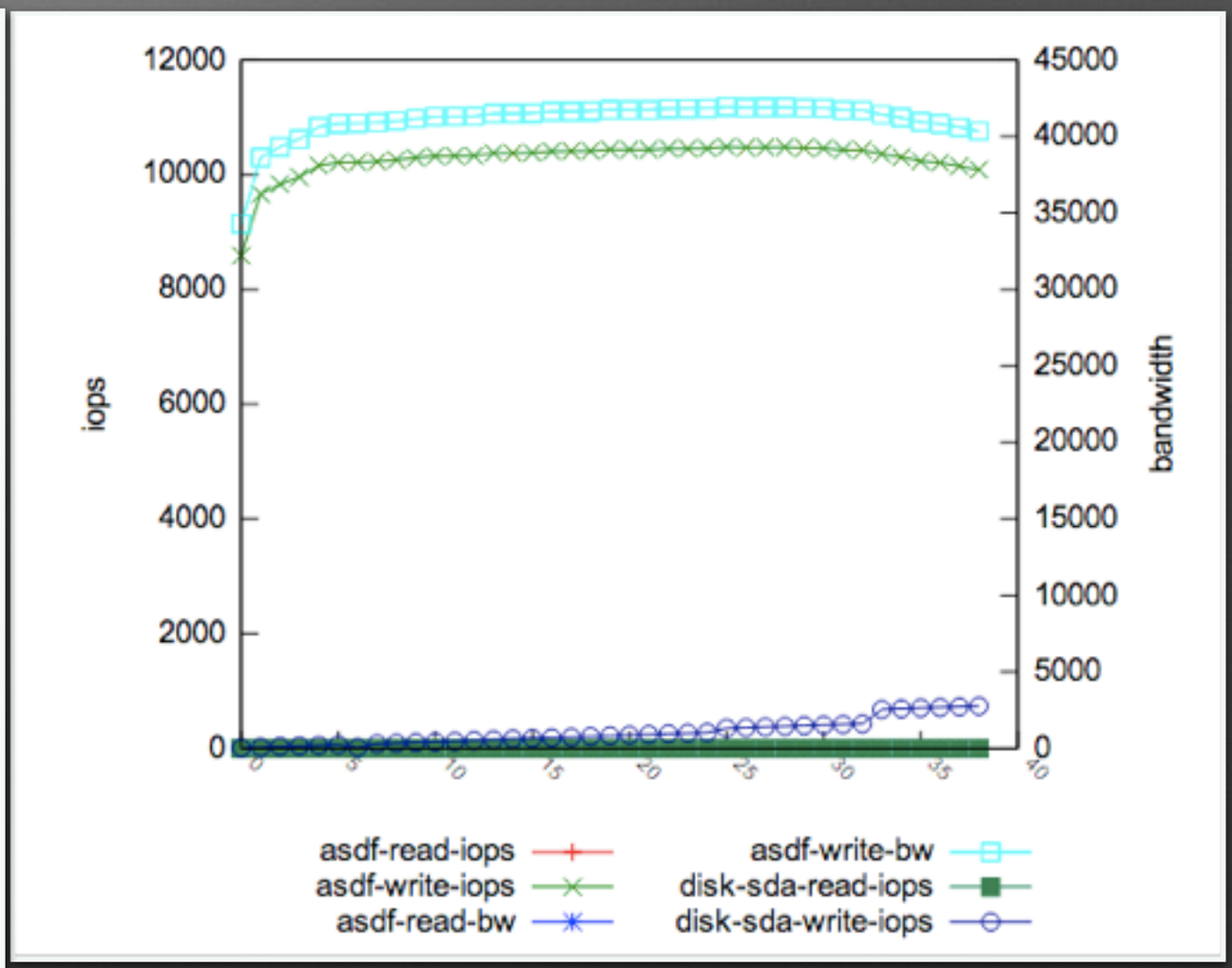
- Facts:
  - Ineffective thread wakeup
- Action:
  - Make OSD queue running
- Configuration:
  - Still in Pull Request(https://github.com/ceph/ceph/pull/2727)
  - “osd_op_worker_wake_time”

Rule 7: Async Messenger(experiment)

- Facts:
  - Each client need two threads on OSD side
  - Painful context switch latency
- Action:
  - Use Async Messenger
- Configuration:
  - “ms_type = async”
  - “ms_async_op_threads = 5”

Result: IOPS





Based on Ceph 0.67.5 Single OSD

Result: Latency

- 4K random write for 1TB rbd image: 1.05 ms per IO
- 4K random read for 1TB rbd image: 0.5 ms per IO
- 1.5x latency performance improvement
- Outstanding large dataset performance

03

THE THIRD PART

High Durability

Ceph Reliability Model

- https://wiki.ceph.com/Development/Reliability_model
- 《CRUSH: Controlled, Scalable, Decentralized Placement of Replicated Data》
- 《Copysets: Reducing the Frequency of Data Loss in Cloud Storage》
- Ceph CRUSH code

Durability Formula

- P = func(N, R, S, AFR)
- P = Pr * M / C(R, N)

Where need to optimize ?

- DataPlacement decides Durability
- CRUSH-MAP decides DataPlacement
- CRUSH-MAP decides Durability

What need to optimize ?

- Durability depend on OSD recovery time
- Durability depend on the number of Copy set in ceph pool

the possible PG's OSD set is Copy set,

the data loss in ceph is loss of any PG, actually is loss of any Copy set.

If the replication number is 3 and we lost 3 osds in ceph, the probability of data loss depend on the number of copy set, because the 3 osds may be not the Copy set.

- The shorter recovery time, the higher Durability
- The less the number of Copy set, the higher Durability

How to optimize it?

- The CRUSH-MAP setting decides osd recovery time
- The CRUSH-MAP setting decides the number of Copy set

Default CRUSH-MAP setting

3 racks
24 nodes
72 osds

if R = 3, M = 24 * 24 * 24 = 13824

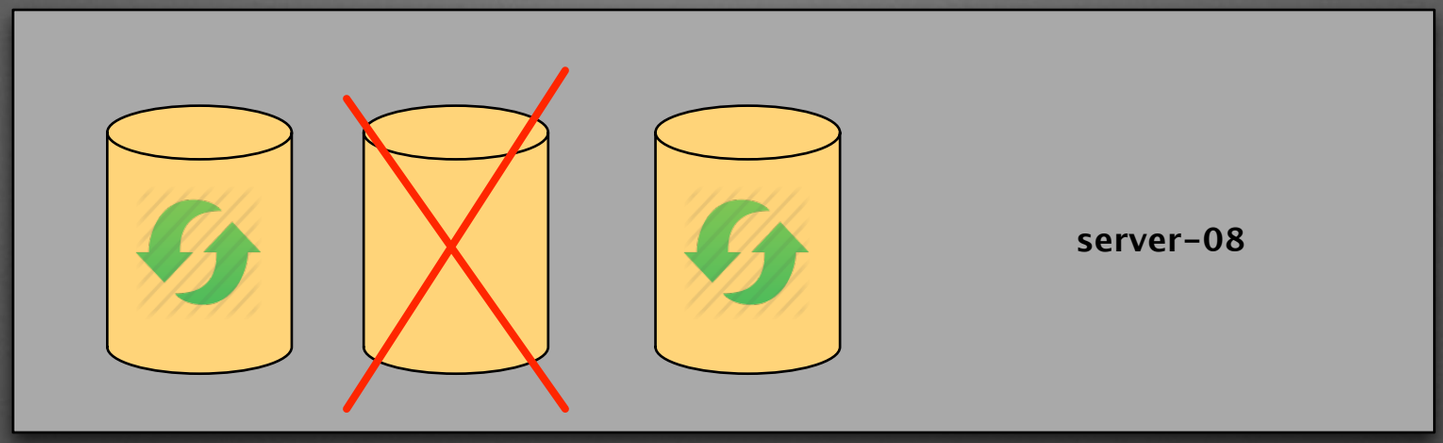
default crush setting

```
root default
    rack rack-01
        host server-01
            osd.0   up      1
            osd.1   up      1
            osd.2   up      1
        host server-02
            ......
    host server-03
    ......
    ......
    host server-08
        osd.21  up      1
        osd.22  up      1
        osd.23  up      1
rack rack-02
    ......
    ......
rack rack-03
    ......
    ......
```

N = 72 S = 3	R = 1	R = 2	R = 3
C(R, N)	72	2556	119280
M	72	1728	13824
Pr	0.99	2.1*10E-4	4.6*10E-8
P	0.99	1.4*10E-4	5.4*10E-9
Nines		3	8

Reduce recovery time

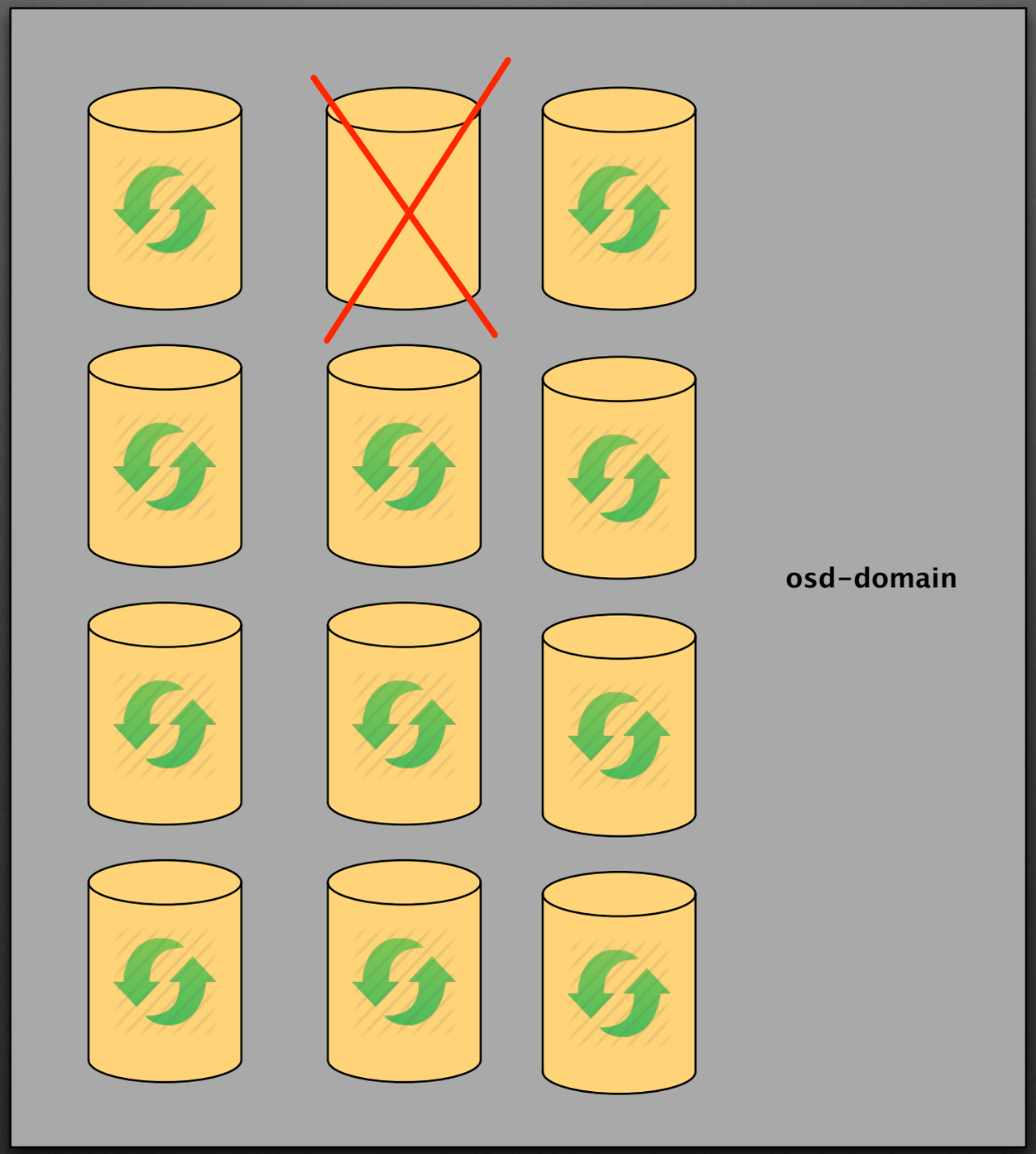
default CRUSH-MAP setting



if one OSD out, only two OSDs can do data recovery
so
recovery time is too longer

we
need more OSD to do data recovery to
reduce recovery time

use osd-domain instead of host bucket
reduce recovery time



if R = 3, M = 24 * 24 * 24 = 13824

new crush map

```
root default
    rack rack-01
        osd-domain osdm-01
            osd.0   up     1
            osd.1   up     1
            ......
            ......
            osd.11  up     1
        osd-domain osdm-02
            osd.12  up     1
            osd.13  up     1
            ......
            ......
            osd.23  up     1
    rack rack-02
        osd-domain osdm-03
            ......
        osd-domain osdm-04
            ......
    rack rack-03
        osd-domain osdm-05
            ......
        osd-domain osdm-06
            ......
```

N = 72 S = 12	R = 1	R = 2	R = 3
C(R, N)	72	2556	119280
M	72	1728	13824
Pr	0.99	7.8*10E-5	6.7*10E-9
P	0.99	5.4*10E-5	7.7*10E-10
Nines		4	9

Reduce the number of copy set

use replica-domain instead of rack bucket

if R = 3, M = (12*12*12) * 2 = 3456

new crush map

```
failure-domain apple
    replica-domain replica-01
        osd-domain osdm-01
            osd.0   up      1
            ......
            ......
            osd.11  up      1
        osd-domain osdm-03
            osd.24  up      1
            ......
            ......
            osd.35  up      1
    osd-domain osdm-05
            osd.48  up      1
            ......
            ......
            osd.59  up      1
replica-domain replica-02
    osd-domain osdm-02
            ......
    osd-domain osdm-04
            ......
    osd-domain osdm-06
            ......
```

```
rule sym-apple {
        ruleset 6
        type replicated
        min_size 1
        max_size 10
        step take apple
        step choose  firstn 1 type replica-domain
        step chooseleaf firstn 0 type osd-domain
        step emit
}
```

N = 72 S = 12	R = 1	R = 2	R = 3
C(R, N)	72	2556	119280
M	72	864	3456
Pr	0.99	7.8*10E-5	6.7*10E-9
P	0.99	2.7*10E-5	1.9*10E-10
Nines	0	4	≈ 10

04

THE FOURTH PART

Operation Expericence

deploy

- eNovance: puppet-ceph
- Stackforge: puppet-ceph
- UnitedStack: puppet-ceph

- shorter deploy time
- support all ceph options
- support multi disk type
- wwn-id instead of disk label
- hieradata

Operation goal: Availability

- reduce unnecessary data migration
- reduce slow requests

upgrade ceph

1. noout: ceph osd set noout
2. mark down: ceph osd down x
3. restart: service ceph restart osd.x

reboot host

1. migrate vm
2. mark down osd
3. reboot host

expand ceph capacity

1. setting crushmap
2. setting recovery options
3. trigger data migration
4. observe data recovery rate
5. observe slow requests

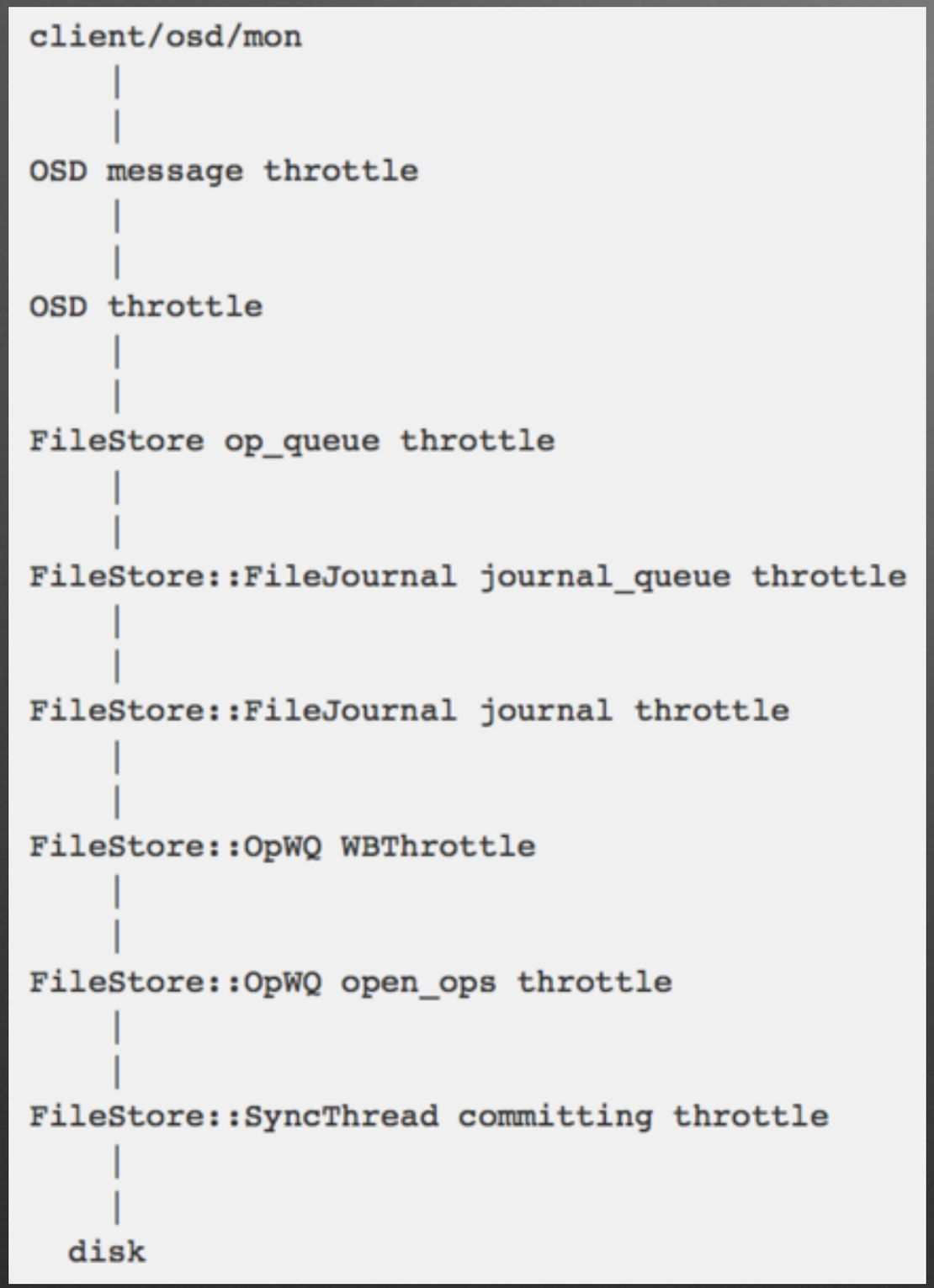
replace disk

- be careful
- ensure replica-domain’s weight unchanged，otherwise data(pg) migrate to another replica-domain

monitoring

- diamond: add new collector, ceph perf dump, ceph status
- graphite: store data
- grafana: display
- alert: zabbix && ceph health command

throttle model



add new collector in diamond redefine metric name in graphite

[process].[what].[component].[attr]

process
osd
mon

what
stat
throttle

component
osd_client_messenger
osd_dispatch_client
osd_dispatch_cluster
osd_pg
osd_pg_client_w
osd_pg_client_r
osd_pg_client_rw
osd_pg_cluster_w
filestore_op_queue
filestore_journal_queue
filestore_journal
filestore_wb
filestore_leveldb
filestore_commit

attr
max_bytes
max_ops
ops
bytes
op/s
in_b/s
out_b/s
lat

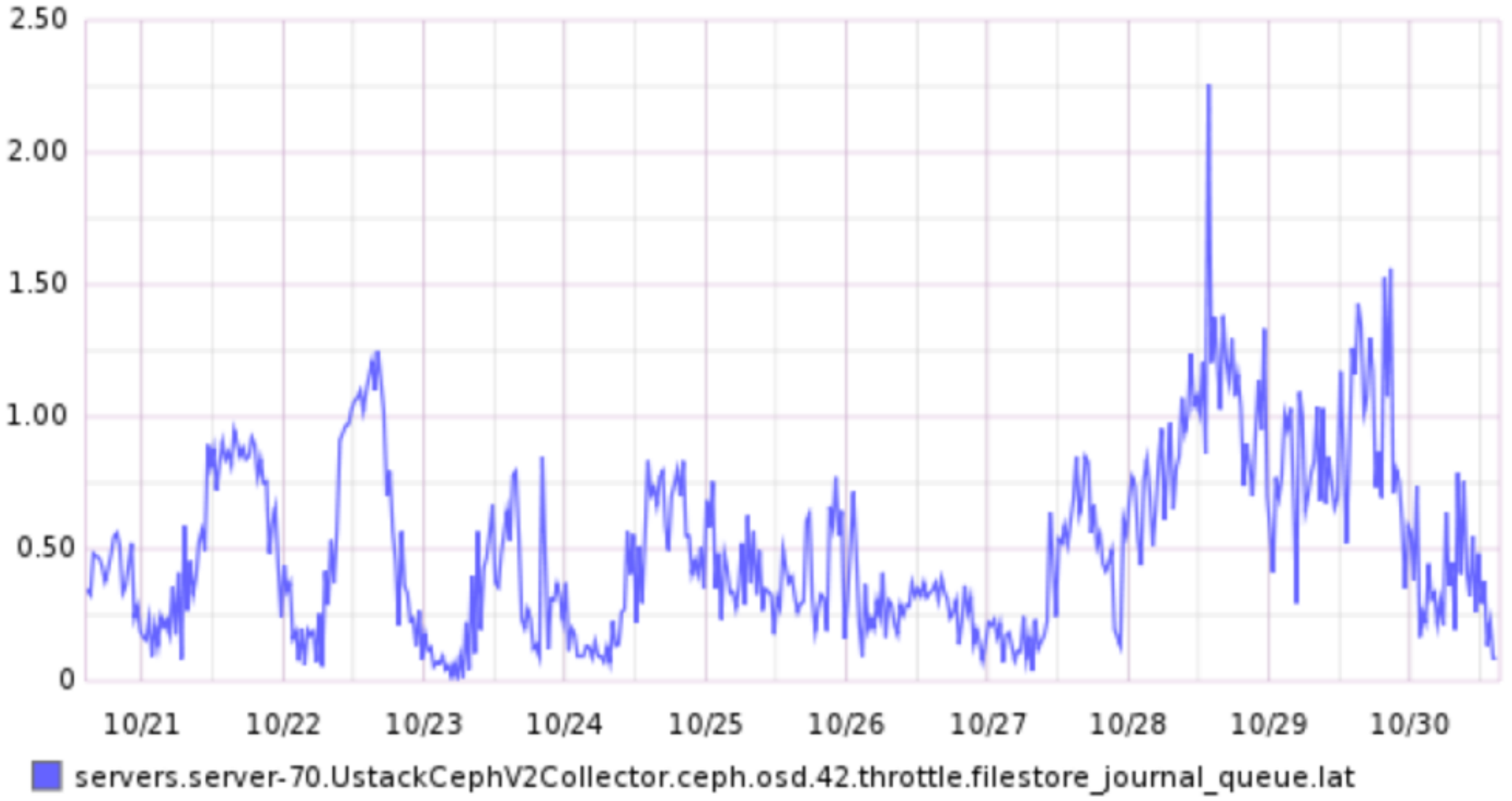
- df
  - total_avail
  - total_space
  - total_used
- pool
  - 3
    - num_bytes
    - num_bytes_recovered
    - num_deep_scrub_errors
    - num_keys_recovered
    - num_object_clones
    - num_object_copies
    - num_objects
    - num_objects_degraded
    - num_objects_missing_on_primary
    - num_objects_recovered
    - num_objects_unfound
    - num_read
    - num_read_kb
    - num_scrub_errors
    - num_shallow_scrub_errors
    - num_write
    - num_write_kb
  - 4
  - 5
  - all

- stat
  - osd_pg
    - numpg
    - numpg_primary
    - numpg_replica
    - numpg_stray
- throttle
  - filestore_commit
  - filestore_journal
  - filestore_journal_queue
    - bytes
    - in_b_p_s
    - lat
    - max_bytes
    - max_ops
    - op_p_s
    - ops
  - filestore_leveldb_get
  - filestore_leveldb_put
  - filestore_op_queue
  - filestore_wb
  - osd_client_messenger
  - osd_dispatch_client
  - osd_dispatch_cluster
  - osd_pg_client
  - osd_pg_client_r
  - osd_pg_client_rw
  - osd_pg_client_w
    - in_b_p_s
    - lat

- stat
  - cluster
- throttle
  - mon_client
  - mon_daemon
  - mon_dispatch
    - bytes
    - max_bytes
  - mon_leveldb_get
  - mon_leveldb_put

Graphite Composer

Now showing the past 10 days



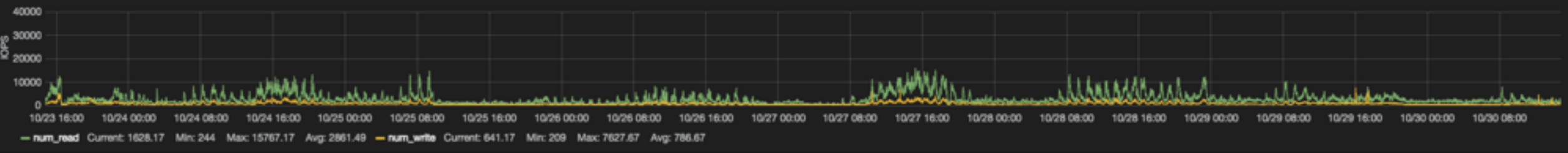
Graph Options ▾ | Graph Data | Auto-Refresh

Ceph-Status

Zoom Out | 7 days ago to a minute ago

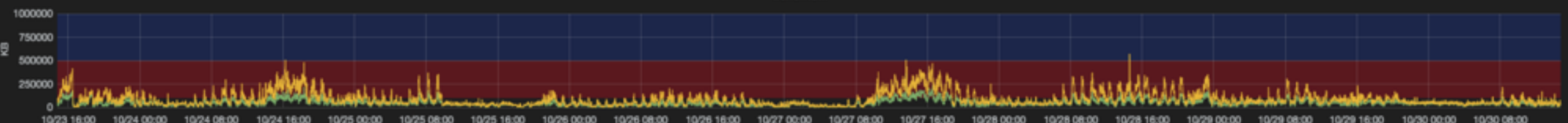
FILTERING:

SSD IOPS



num_read Current: 1628.17 Min: 244 Max: 15767.17 Avg: 2861.49 num_write Current: 641.17 Min: 209 Max: 7627.67 Avg: 786.67

Bandwidth

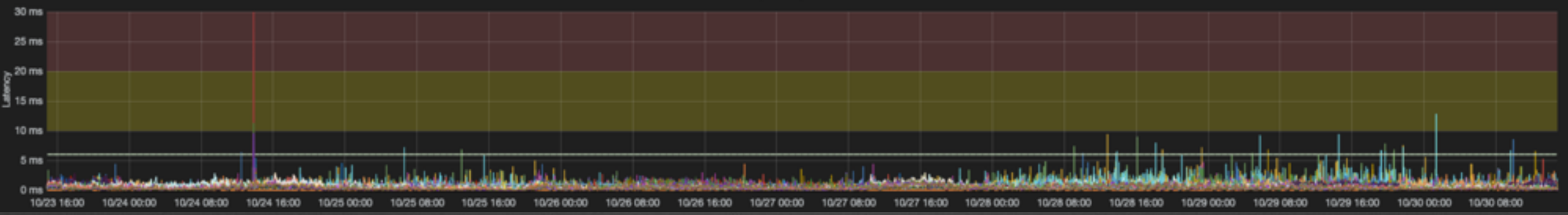


num_read_kb Current: 27166.67 Min: 47.50 Max: 358002 Avg: 43538.44 num_write_kb Current: 17246.33 Min: 2802.50 Max: 341247.50 Avg: 42925.70

All-OSD-Journal-Queue-Lat

Zoom Out | 7 days ago to a minute ago

ceph-osd-journal-lat

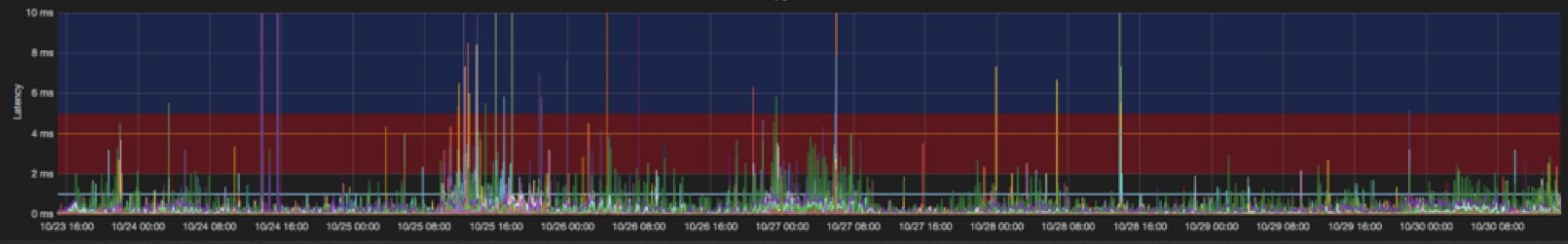


All-OSD-Pg-Client-Read-Lat

Back to dashboard | Zoom Out | 7 days ago to a minute ago

FILTERING: osd : 0 | compoenent : filestore_op_queue

osd pg client read lat



Accidents

- SSD GC
- network failure
- Ceph bug
- XFS bug
- SSD corruption
- PG inconsistent
- recovery data filled network bandwidth

@ UnitedStack

THANK YOU FOR WATCHING

2014/11/05

UnitedStack

https://www.ustack.com/jobs/

About UnitedStack

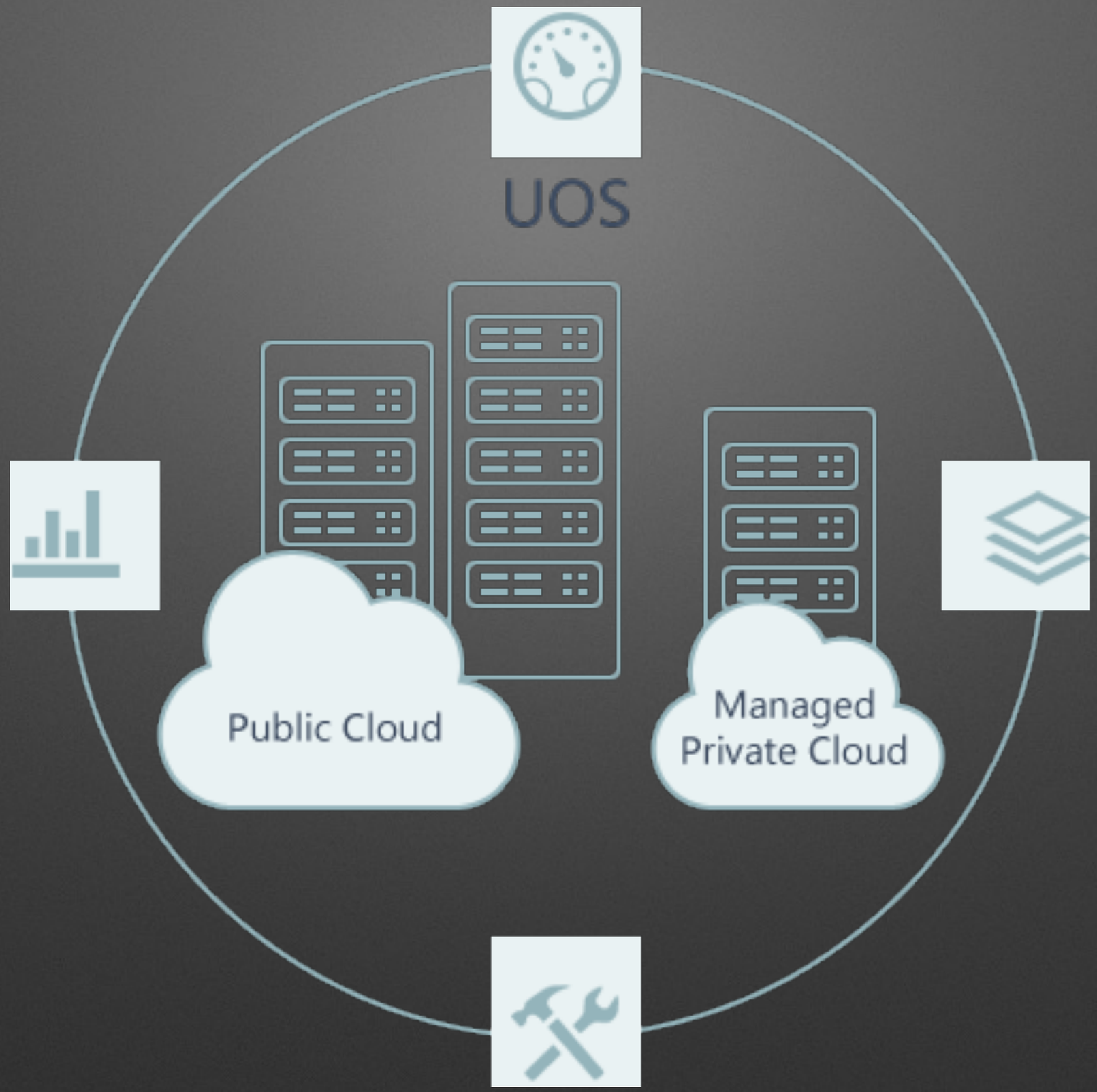
UnitedStack - The Leading OpenStack Cloud Service Solution Provider in China

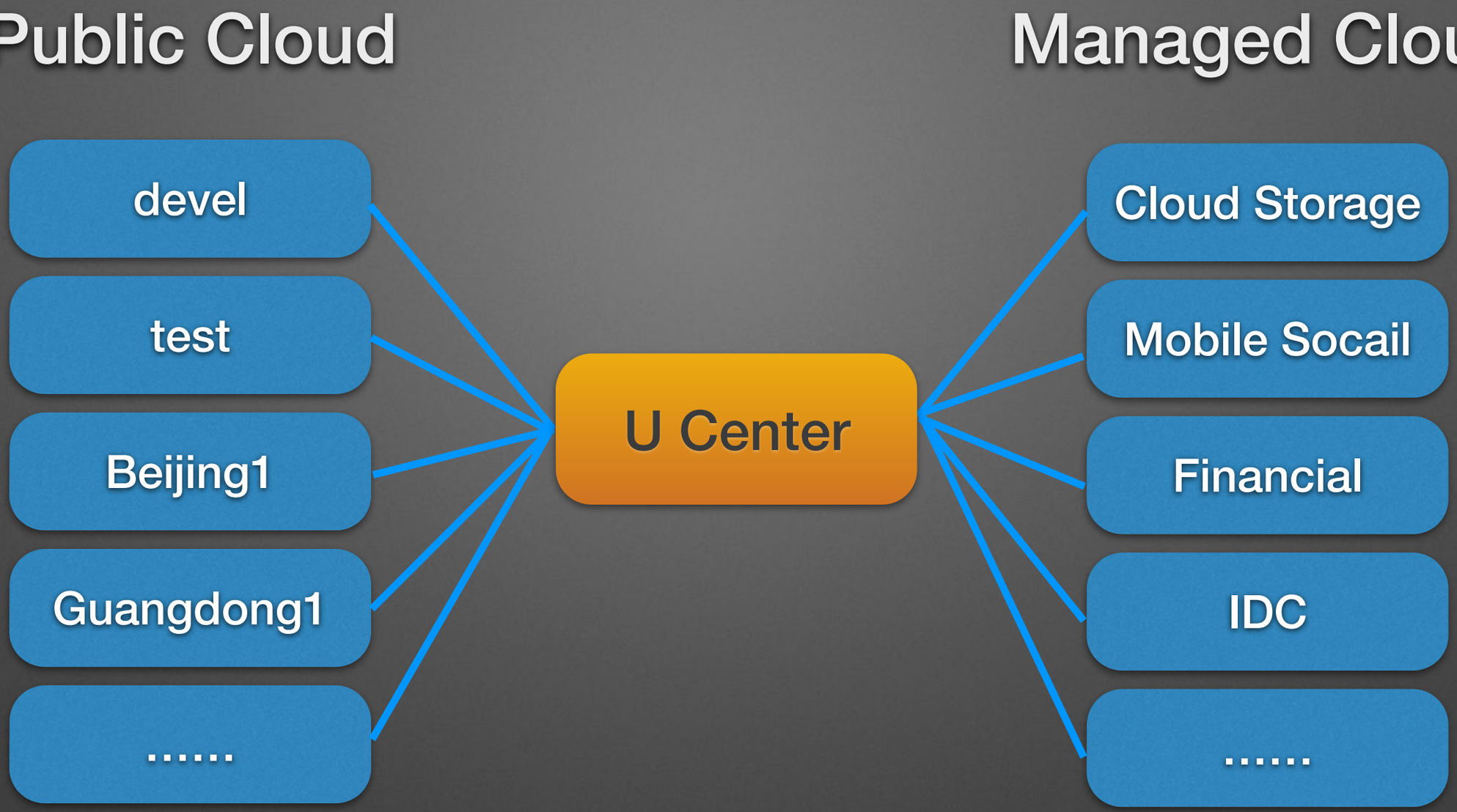
VM deployment in seconds

WYSIWYG network topology

High performance cloud storage

Billing by seconds


UOS
Public Cloud
Managed
Private Cloud



Unified Cloud Service Platform

Unified Ops

Unified SLA

The detail of durability formula

- DataPlacement decides Durability
- CRUSH-MAP decides DataPlacement
- CRUSH-MAP decides Durability

Default crush setting

3 racks
24 nodes
72 osds

How to compute Durability?

Ceph Reliability Model

- https://wiki.ceph.com/Development/Reliability_model
- 《CRUSH: Controlled, Scalable, Decentralized Placement of Replicated Data》
- 《Copysets: Reducing the Frequency of Data Loss in Cloud Storage》
- Ceph CRUSH code

Durability Formula

P = func(N, R, S, AFR)

- P: the probability of losing all copy
- N: the number of OSD in ceph pool
- R: the number of copy
- S: the number of OSD in bucket(it decide recovery time)
- AFR: disk annualized failure rate

Failure events are considered to be Poisson

- Failure rates are characterized in units of failures per billion hours(FITs), and so I have tried to represent all periodicities in FITs and all times in hours:
  fit = failures in time = 1/MTTF ~= 1/MTBF = AFR/(24*365)

- Event Probabilities, λ is the failure rate, the probability of n failure events during time t:
  Pn(λ,t) = (λt)n e-λt / n!

The probability of data loss

- OSD set: copy set, any PG reside in
- data loss: any OSD set loss
- ignore Non-Recoverable Errrors, NRE’s never happen which might be true on scrubbed osd

Non-Recoverable Errors
NREs are read errors that cannot be corrected by retries or ECC.

- media noise
- high-fly
- off-track writes

The probability of R OSDs loss

1. The probability of an initial OSD loss incident.
2. Having sufferred this loss, the probability of losing R-1 OSDs is based on the recovery time.
3. Multiplied by the probability of the above. The result is Pr。

The probability of Copy sets loss

1. M = Copy Sets Number in Ceph Pool
2. any R OSDs is C(R, N)
3. the probability of copy sets loss is Pr * M / C(R, N)

P = Pr * M / C(R, N)

If R = 3, One PG -> (osd.x, osd.y, osd.z)

(osd.x, osd.y, osd.z) is a Copy Set

All Copy Sets are in line with the rule of CRUSH MAP

M = The numer of Copy Sets in Ceph Pool

Pr = The probability of R OSDs loss

C(R, N) = any R OSDs in N

P = The probability of any Copy Sets loss(data loss)

default crush setting

if R = 3, M = 24 * 24 * 24 = 13824

default crush setting

```
root default
    rack rack-01
        host server-01
            osd.0   up      1
            osd.1   up      1
            osd.2   up      1
        host server-02
            ......
    host server-03
    ......
    ......
    host server-08
        osd.21  up      1
        osd.22  up      1
        osd.23  up      1
rack rack-02
    ......
    ......
rack rack-03
    ......
    ......
```

AFR = 0.04

One Disk Recovery Rate = 100 MB/s

Mark Out Time = 10 mins

N = 72 S = 3	R = 1	R = 2	R = 3
C(R, N)	72	2556	119280
M	72	1728	13824
Pr	0.99	2.1*10E-4	4.6*10E-8
P	0.99	1.4*10E-4	5.4*10E-9
Nines		3	8

Trade-off

trade off between durability and availability

new crush	N	R	S	Nines	R
Ceph	72	3	3	11	31 mins
Ceph	72	3	6	10	13 mins
Ceph	72	3	12	10	6 mins
Ceph	72	3	24	9	3 mins

Shorter recovery time

Minimize the impact of SLA

Final crush map

old map:

root

rack

host

osd

new map:

failure-domain

replica-domain

osd-domain

osd