netmarble

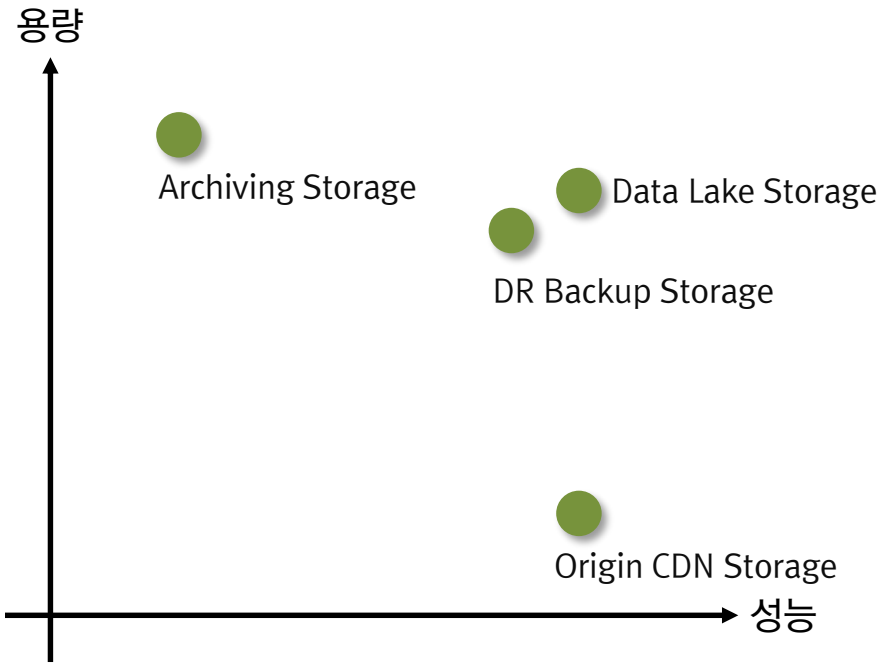
| 보안등급 | 배포범위 | 작성부서 | 작성일자 | 보존기한 |
| --- | --- | --- | --- | --- |
| 대외반출 | | | | |
| 반출사유 | 반출범위 | 반출부서 | 반출일자 | 비고 |
| | | | | |

# (CEPH 운영자를 위한)
# 오브젝트 스토리지 성능 튜닝


본 문서는 넷마블(주)의 자산으로 지정된 수신자 외 타인 열람 및 유출 시 산업 스파이로 간주될 수 있으며 민.형사상의 책임을 질 수 있습니다.

# Table of Contents

# Object Storage at Netmarble




## Archiving Storage

- Erasure Coded Pool
- Big Capacity (8TB Disk)

## Origin CDN Storage

- Product Ready
- Custom Archtecture
- GET requeste Optimized

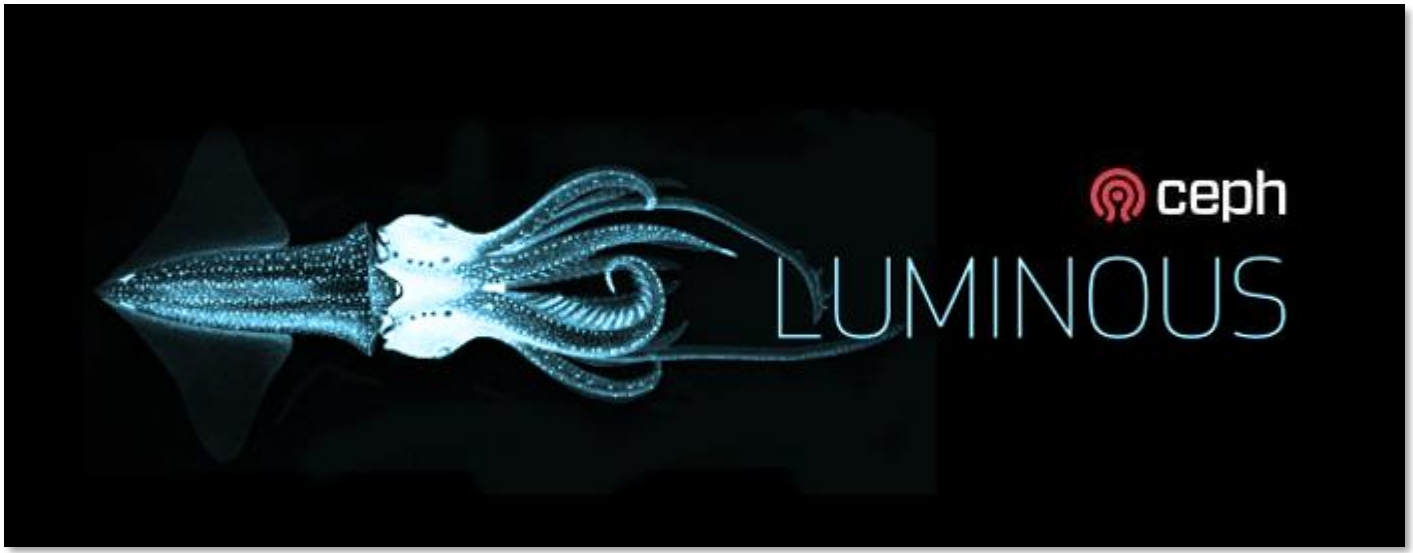
## DR Backup Storage

- 3 replica
- SSD Index Pool

## Data Lake Storage

- BMT with HDFS

# Luminous



# Luminous – Remove OSDs

## Before Luminous

```
$ ceph osd out 1
$ systemctl stop ceph-osd@1.service
$ ceph osd crush remove osd.1
$ ceph auth del osd.1
$ ceph osd rm 1
```

## Luminous

```
$ ceph osd purge 1 –yes-i-really-mean-it
```

# Luminous – ceph-volume

- ceph-disk
  - mimic 버전부터 disabled
  - 문제점
    - udev 기반의 설계로 여러 조건에서 버그 발생
      - reboot했는데 OSD가 안올라와요..등
    - 디버깅 하기 힘듦
    - OSD 추가 시 많은 시간 소요
- ceph-volume
  - 여러 device type을 modular 방식으로 지원
    - gpt type → simple
    - lvm type
    - NVMe witk SPDK (will be added)
  - 각 장치에 대한 메타데이터 정보 활용 (cluster uuid, db/wal device 정보, secret key 정보)
  - 여러 device mapper와 호환 가능

# Luminous – ceph-volume

## fdisk –l output of ceph-disk

```
Disk /dev/sdx: 1.7 TiB, 1800326569984 bytes, 3516262832 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 262144 bytes / 262144 bytes
Disklabel type: gpt
Disk identifier: 62BD69DD-3402-40C5-BA10-E62CE7D19C9B
```

## fdisk –l output of ceph-volume

```
Disk /dev/mapper/ceph--6ba8c86a--d4f6--4458--89de--6e6b1010e2fe-osd--block--56a2f04d--944e--47c1--a808--6cc9d4302901: 1.7 TiB, 1800325300224 bytes, 3516260352 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 262144 bytes / 262144 bytes
```

# Luminous – ceph-volume

## ceph-disk list

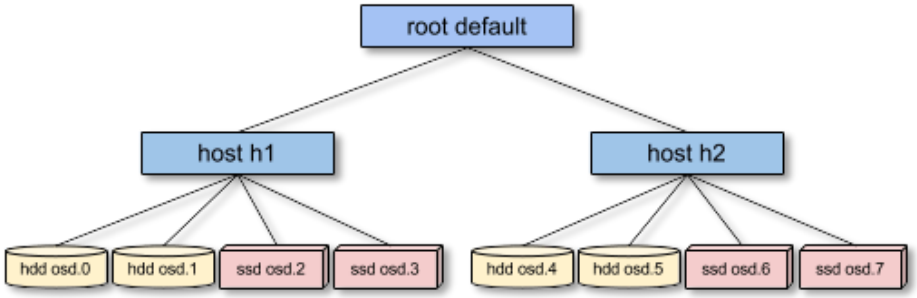
```
/dev/sdb7 ceph journal, for /dev/sdi1
 /dev/sdb8 ceph journal, for /dev/sdj1
 /dev/sdb9 ceph journal, for /dev/sdk1
/dev/sdc :
/dev/sdc1 ceph data, active, cluster ceph, osd.24, journal /dev/sdb1
/dev/sdd :
/dev/sdd1 ceph data, active, cluster ceph, osd.25, journal /dev/sdb2
/dev/sde :
/dev/sde1 ceph data, active, cluster ceph, osd.26, journal /dev/sdb3
```

## ceph-volume lvm list

```
====== osd.246 ======
[block] /dev/ceph-508cf292-0579-404f-9aa3-88a77f1cafb4/osd-block-eebabda3-cb13-4326-80e9-882846db64d8
type block
  osd id 246
  cluster fsid 7dddeb97-3807-4e3a-a727-e6dacb4d30b9
  cluster name ceph
  osd fsid eebabda3-cb13-4326-80e9-882846db64d8
  db device /dev/sdb3
  encrypted 0
  db uuid 0d536967-f376-4d35-8f78-42795d7ace7e
  cephx lockbox secret
  block uuid dPJ4YU-U3zX-fu8C-g1lF-We78-kYNb-2Yv6Mc
  block device /dev/ceph-508cf292-0579-404f-9aa3-88a77f1cafb4/osd-block-eebabda3-cb13-4326-80e9-882846db64d8
  crush device class None
[ db] /dev/sdb3
PARTUUID 0d536967-f376-4d35-8f78-42795d7ace7e
```

# Luminous – Device Class

- 여러 개의 device를 class 별로 분리
- device class 별로 rule을 정의함
- crush map을 직접 수정해야 하는 부담을 줄일 수 있음.



# Luminous – Device Class

## Before Luminous

```
$ ceph osd getcrushmap –o crush.map

$ crushtool –d crush.map –o crush.txt
$ vi crush.txt


$ crushtool –c crush.txt –o crush.map

$ ceph osd setcrushmap –I crush.map
```

```
### crushrule

host hdd-host01 {
}
host ssd-host01 {
}
rack ssd-rack {
}
rack hdd-rack {
}

# rules
rule ssd-rule {
        step take default
        step chooseleaf firstn 0 type ssd-rack
}
rule hdd-rule {
        step take default
        step chooseleaf firstn 0 type hdd-rack
}
```

# Luminous – Device Class

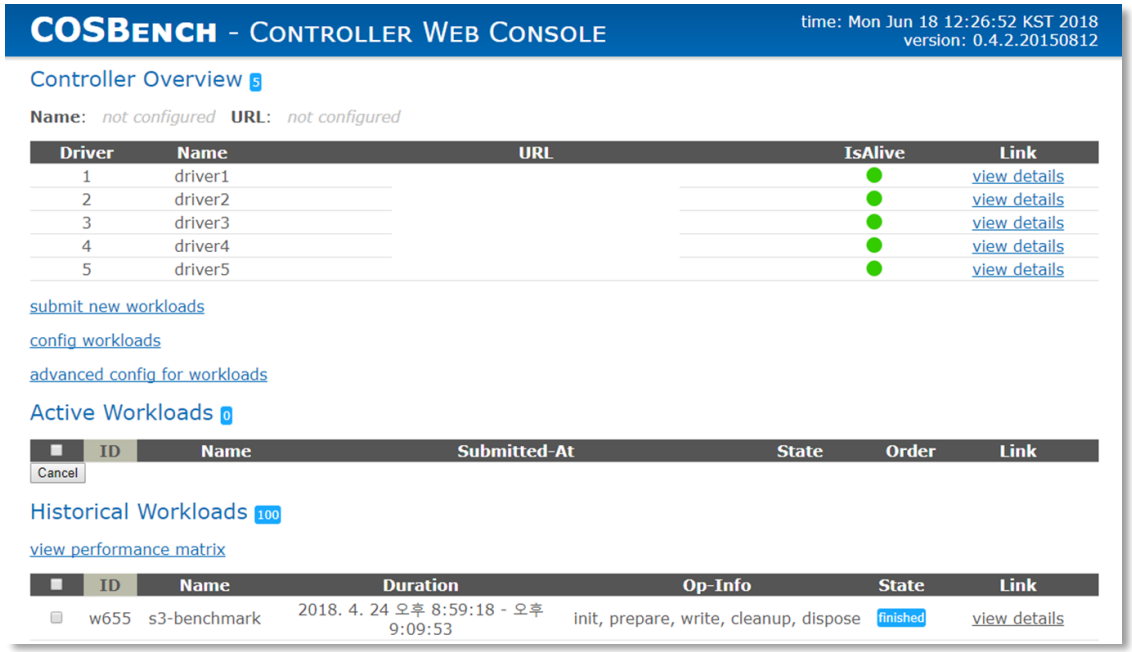
## Luminous

```
$ ceph osd set-device-class ssd osd.0

$ ceph osd crush create-replicated ₩
ssd-rule default host ssd

$ ceph osd crush tree --show-shadow
```

```
ID  CLASS WEIGHT     TYPE NAME
-45   ssd 122.24930 root default~ssd
-39   ssd   8.73245     host BBigpilakeslv100~ssd
292   ssd   1.74649         osd.292
293   ssd   1.74649         osd.293
294   ssd   1.74649         osd.294
295   ssd   1.74649         osd.295
296   ssd   1.74649         osd.296
.
.
 -2   hdd 412.60089 root default~hdd
-20   hdd  29.47302     host BBigpilakeslv100~hdd
144   hdd   1.63739         osd.144
145   hdd   1.63739         osd.145
146   hdd   1.63739         osd.146
147   hdd   1.63739         osd.147
148   hdd   1.63739         osd.148
149   hdd   1.63739         osd.149
150   hdd   1.63739         osd.150
.
```

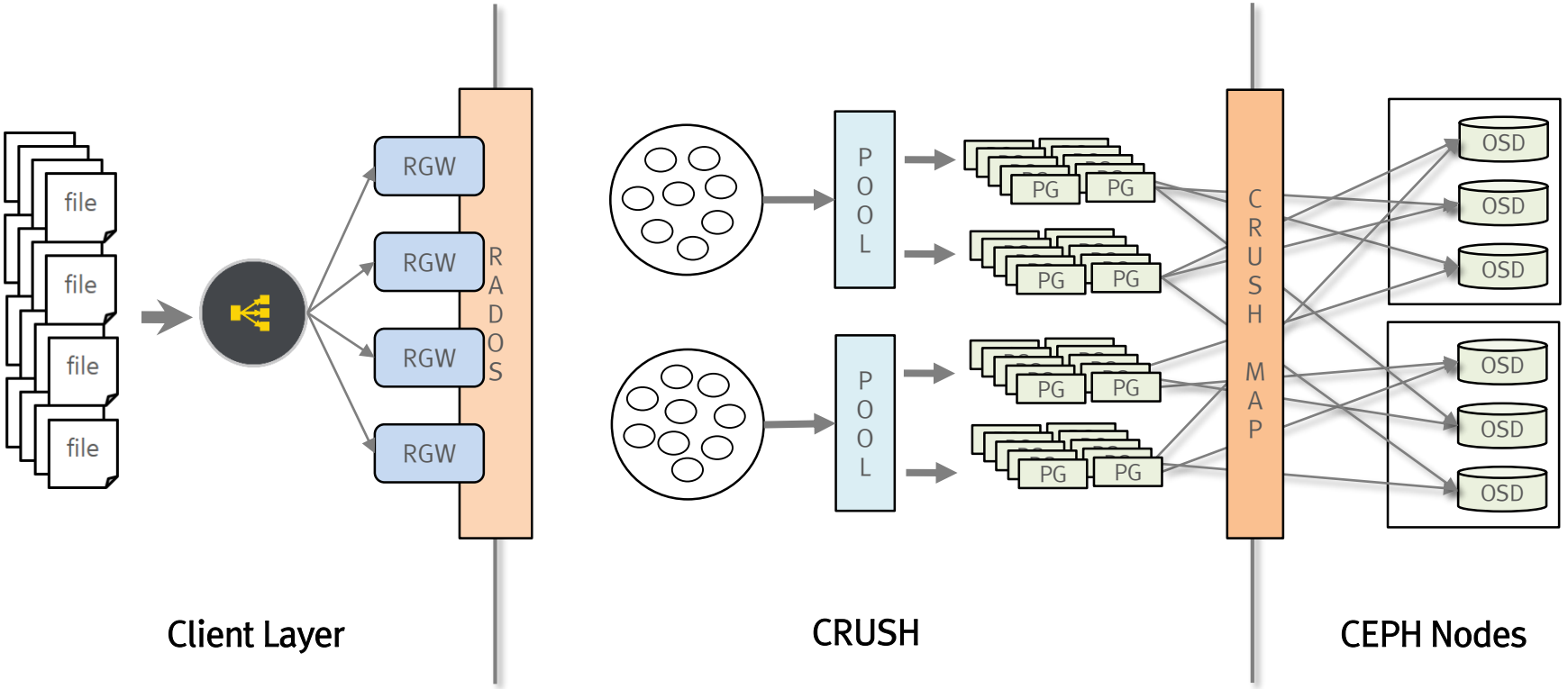
# Performance of Object Storage

# Performance of Object Storage

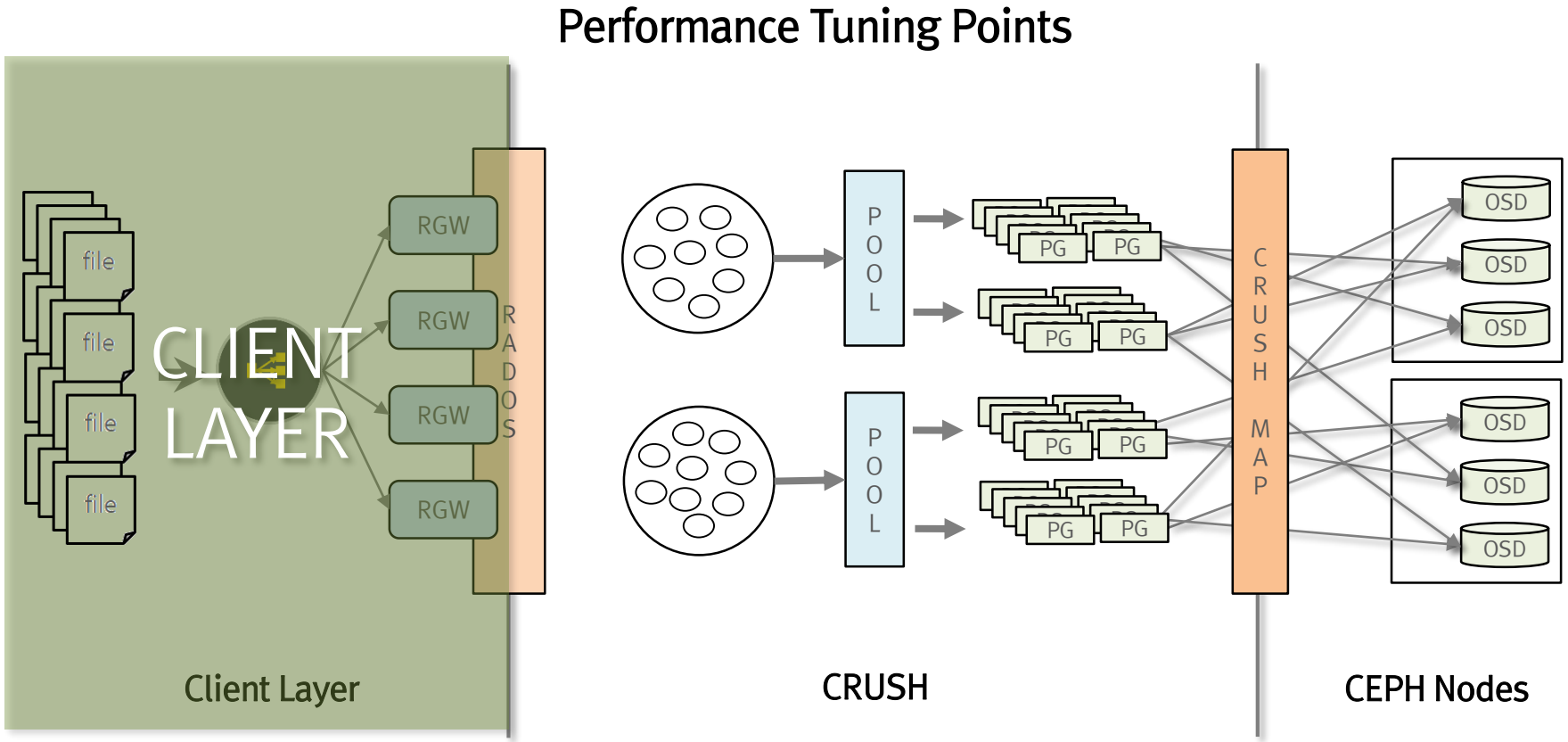



## Performance Benchmark Tool

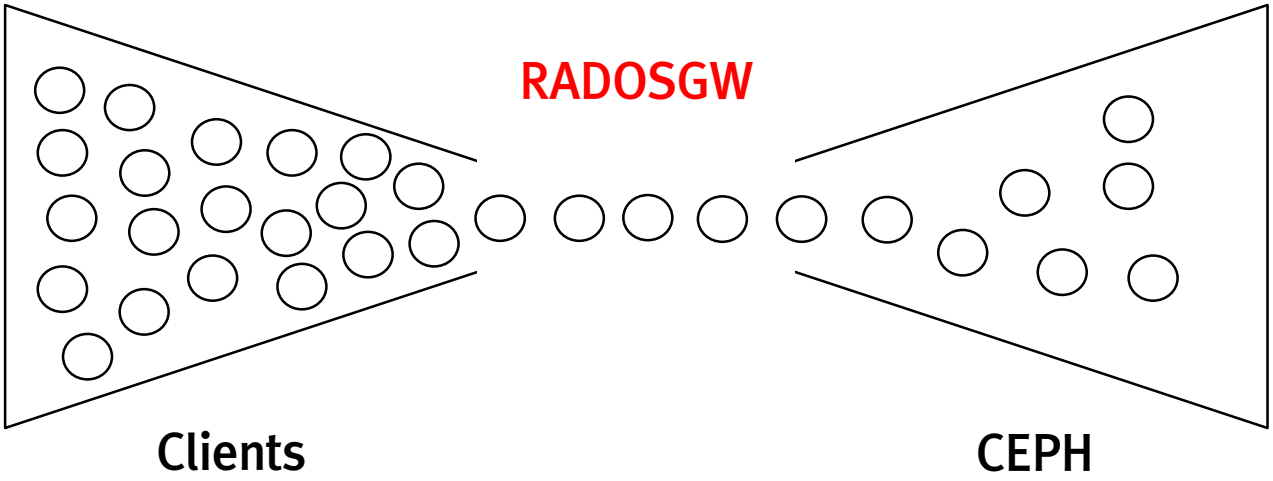
# Performance of Object Storage



# Performance of Object Storage



# Performance of RadosGW



# Performance of RadosGW

**RADOSGW**

RGW의 최대 성능 확인

Object Storage 환경에 얼마나 많은 RGW 노드를 구성해야 하는가?

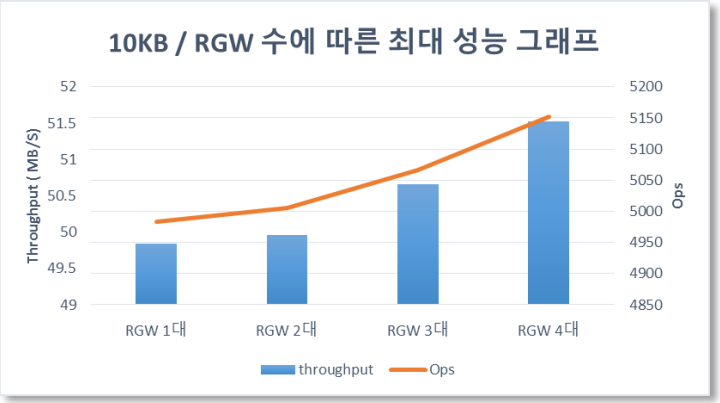
# Performance of RadosGW

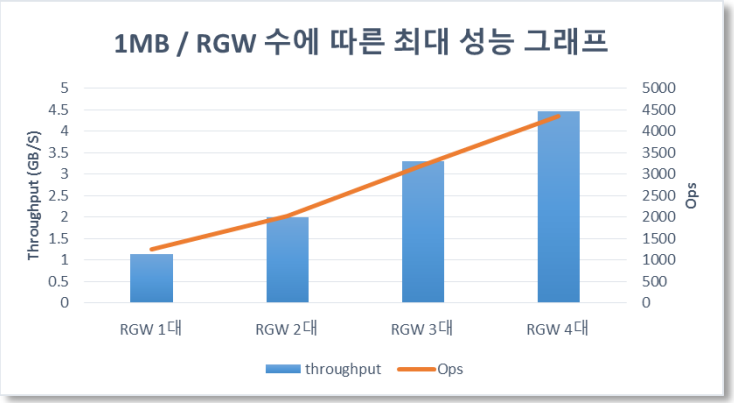
## Test Environments

| 구분 | Specification |
|---|---|
| ceph | 버전 : Luminous(12.2.2)<br>파일시스템 : bluestore<br>**Replication :** 3 replica<br>Mon 총 1개<br>OSD 총 464개 |
| OSD 노드 | 수량 : 3대(SSD) + 20대(HDD)<br>CPU : 10 Cores<br>Memory : 128GB<br>Journal : 800GB SAS WI<br>Disk : SAS 1.8TB (서버별 22대) |
| radosgw | 수량 : **1대 ~ 4대**<br>**OS : Ubuntu 16.04**<br>CPU : 10 Cores<br>Memory : 128GB<br>Network B/W : 10Gbps → 20Gbps<br>VIP : cephpilot.nmn.io |

# Performance of RadosGW

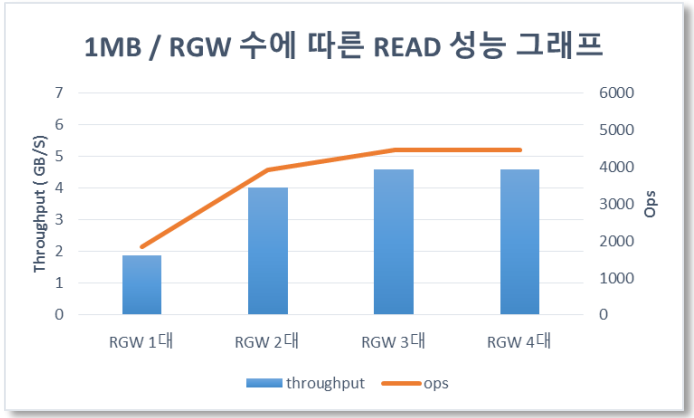
## Write Performance



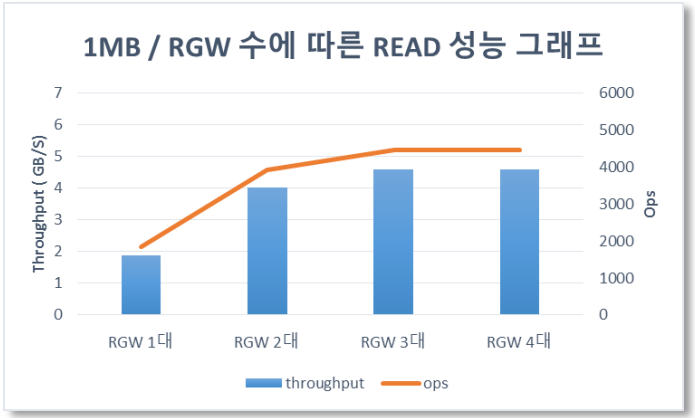
# Performance of RadosGW

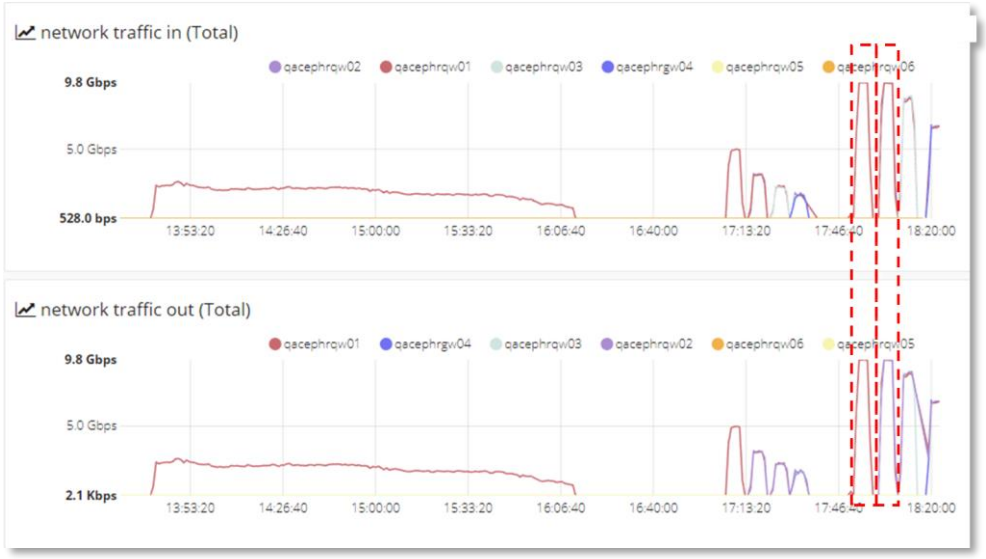
## Read Performance



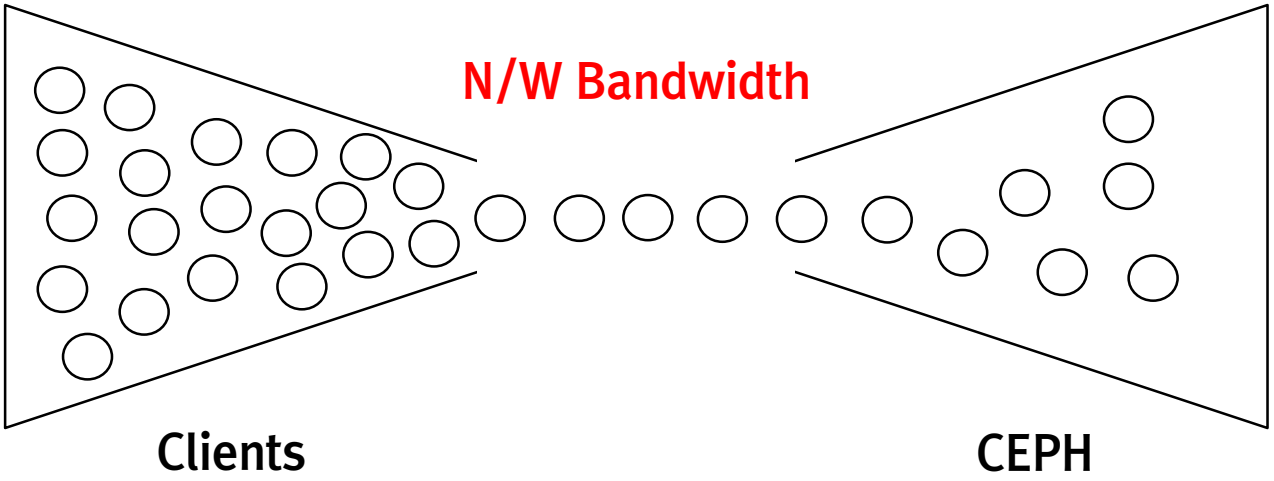
# Performance of RadosGW

## Read Performance



# Performance of RadosGW



# Performance of RadosGW

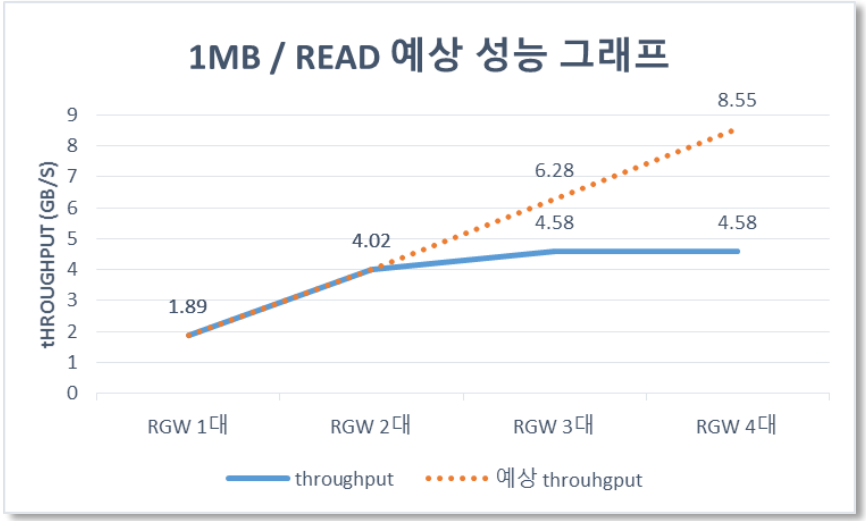
## N/W Bandwidth

20Gbps의 상의 네트워크 대역폭

Load Balancer의 네트워크 대역폭

# Performance of RadosGW

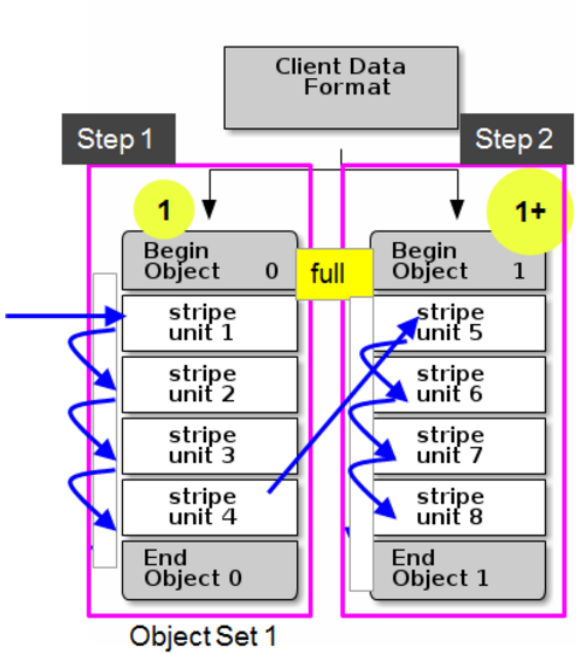
## Read Performance



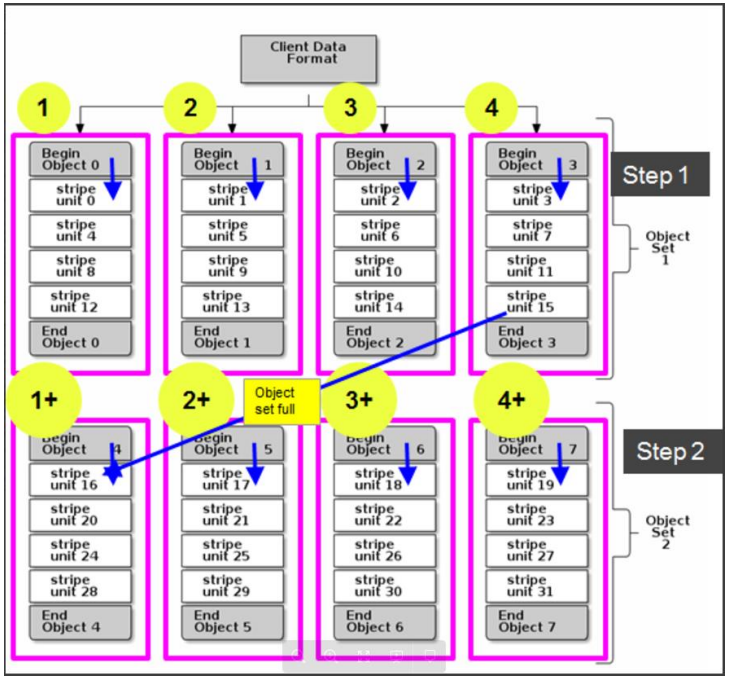
# Performance Tuning for Client Layer

- Data Striping
  - Storage Device
    - 처리 능력의 한계 → 여러 장비에 striping 저장 방식을 지원
    - 대표적인 예 → RAID 구성
  - Data Striping of Ceph
    - CEPH의 3가지 Client(RBD, MDS, RGW)에서 이 기능을 제공함.
    - rados object 들은 다른 placement group에 할당되어 있으므로 write 시에 다른 OSD에 동시 저장할 수 있다.
    - NOTE: ceph의 client 레벨에서 object에 data를 striping 하기 때문에, librados를 통해 직접 ceph cluster에 데이터를 저장하는 client의 경우 striping을 직접 구현해야 함.
  - 용어들
    - stripe count = object set을 의미
    - stripe width = object에 저장하기 위해 client에서 data를 나누는 단위(=striping unit)
    - object size = ceph의 rados object

# Performance Tuning for Client Layer

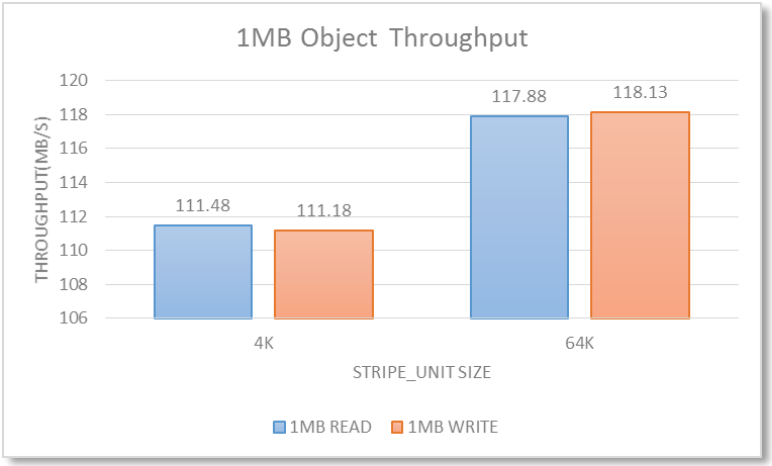


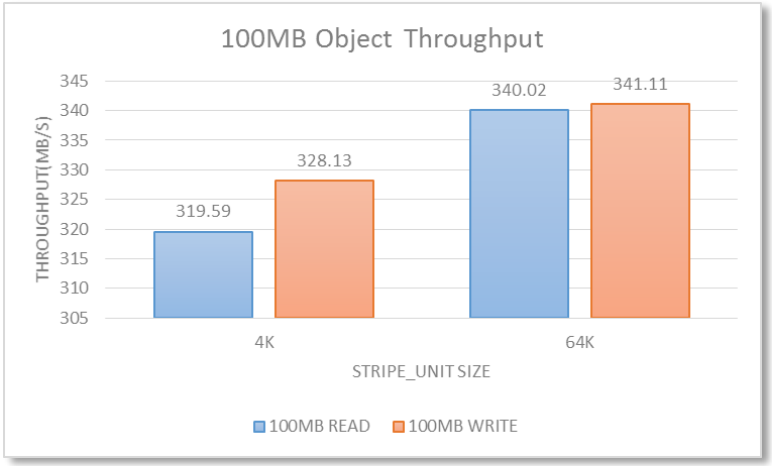

1 Stripe Count




4 Stripe Count

# Performance Tuning for Client Layer



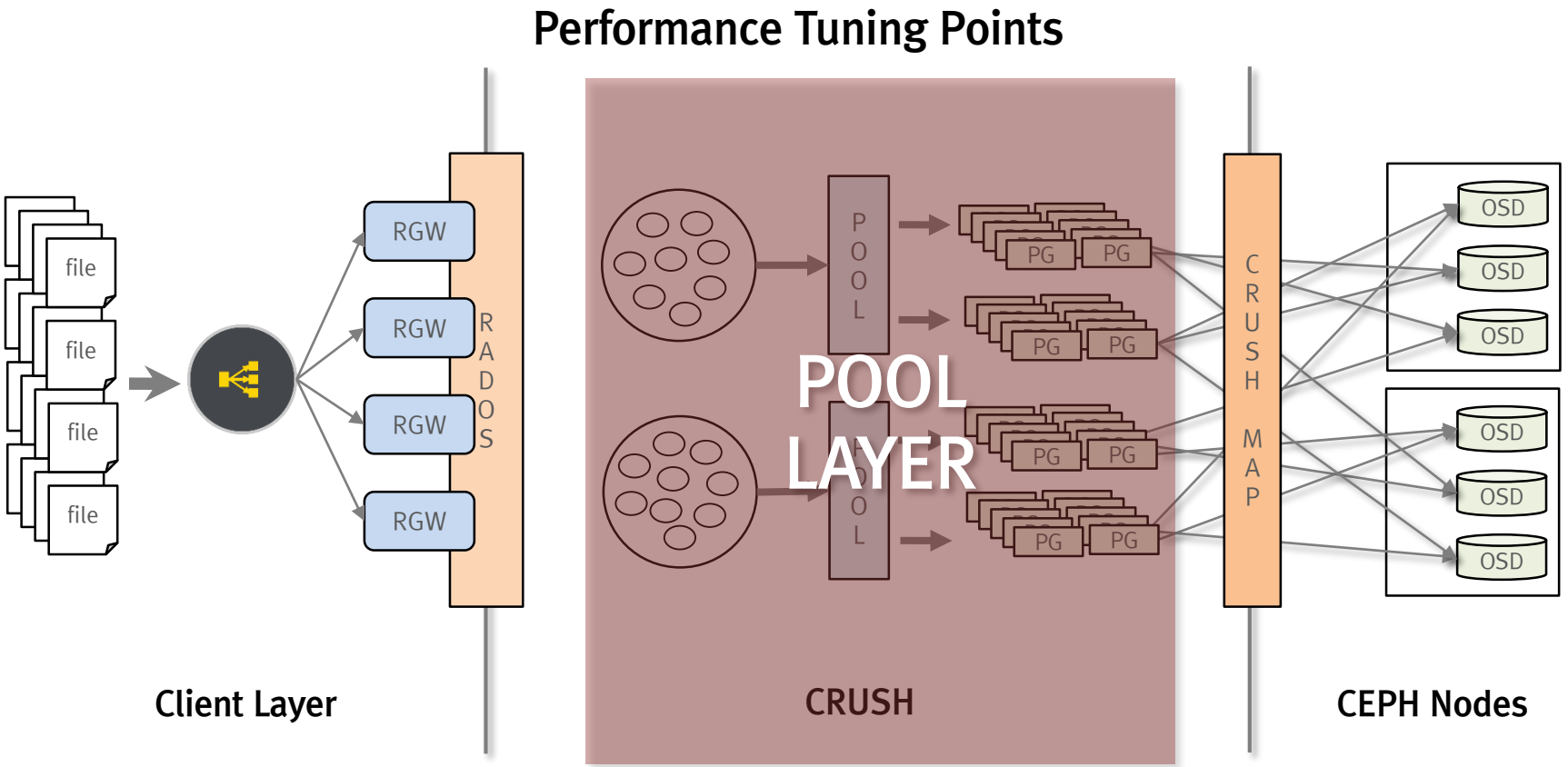




**Striping Unit의 크기에 따른 성능 차이**

# Performance Tuning for Client Layer

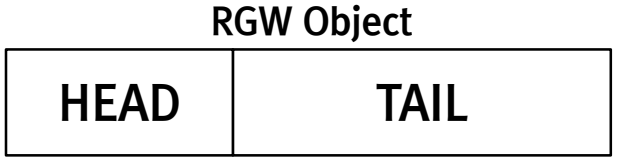
- 그 외에도
  - usage/acces log disable → 소폭 향상
  - rgw_num_rados_handles → 차이 미비
  - civetweb_threads → 차이 미비
  - rgw_thread_pool_size
    - 성능 차이는 많이 없음.
    - 안정적인 RGW 동작을 위해서는 size를 지정해 주는 것이 좋음.

# Performance of Object Storage

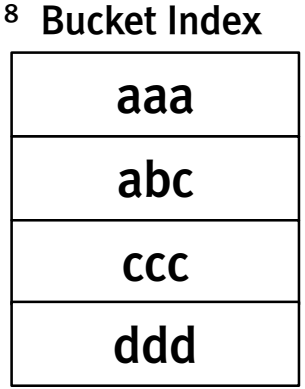


# Performance Tuning for for Rados Pool

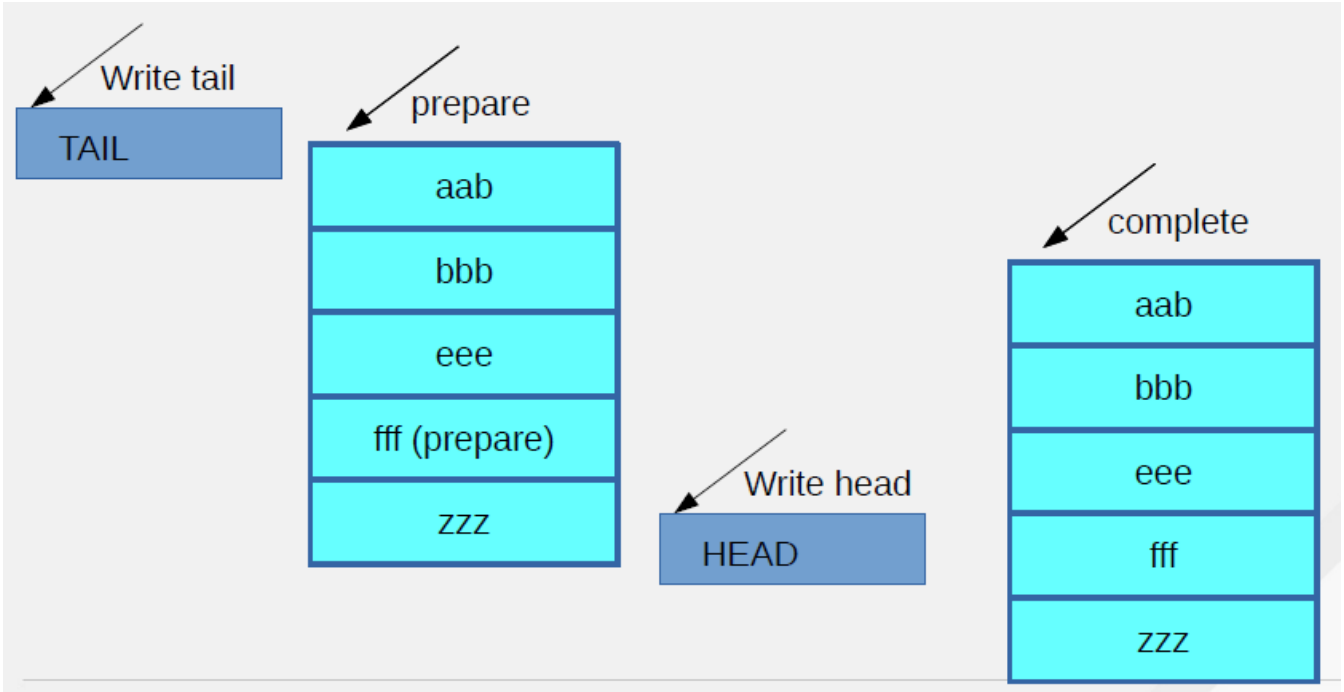
- RGW Object
  - HEAD
    - Single rados object
    - object metadata(acl, user attrs, manifest)
    - Optional start of data
  - TAIL
    - Striped data
    - 0 or more rados object
- RGW Bucket Index
  - Bucket에 포함된 오브젝트 정보
  - 하나의 Rados object로 이루어짐.
  - 많은 RGW Object → sharding





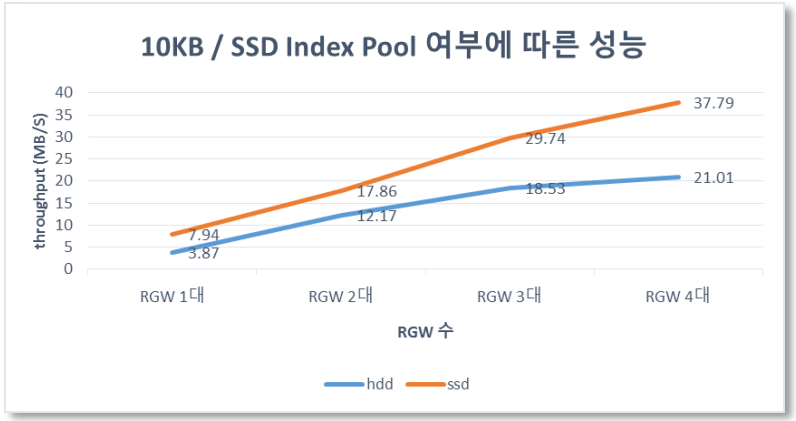
# Performance Tuning for for Rados Pool



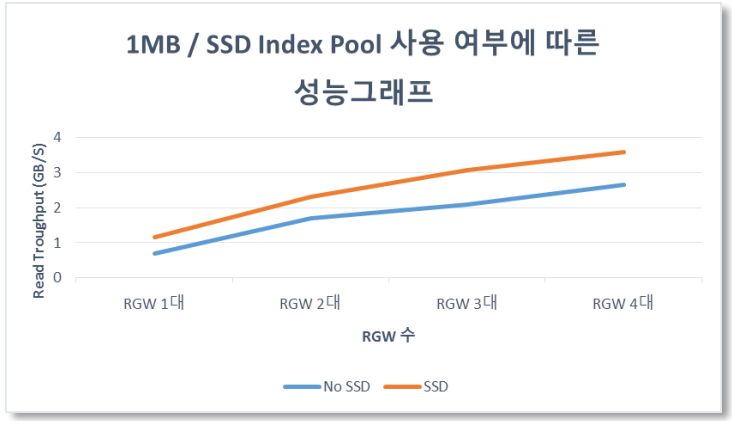

참조 : Fosdem_object_storage_ceph from Redhat

# Performance Tuning for for Rados Pool

- **SSD Index Pool**

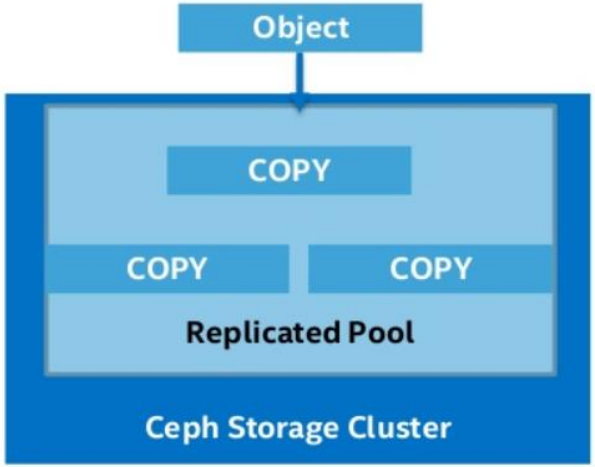



**Write**



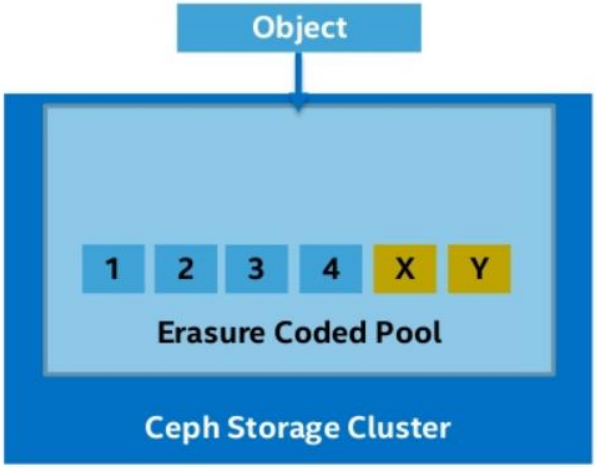

**Read**

# Performance Tuning for for Rados Pool




Full Copies of stored objects

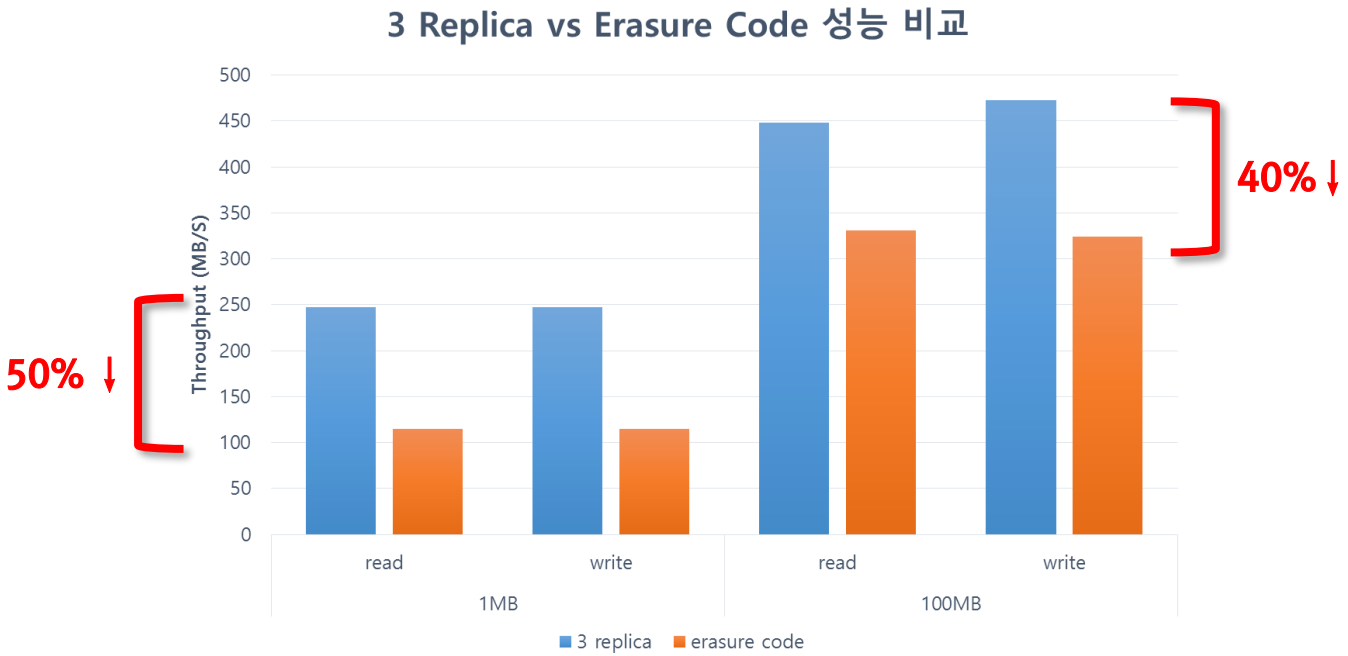
- Very high durability
- 3x (200% overhead)
- Quicker recovery



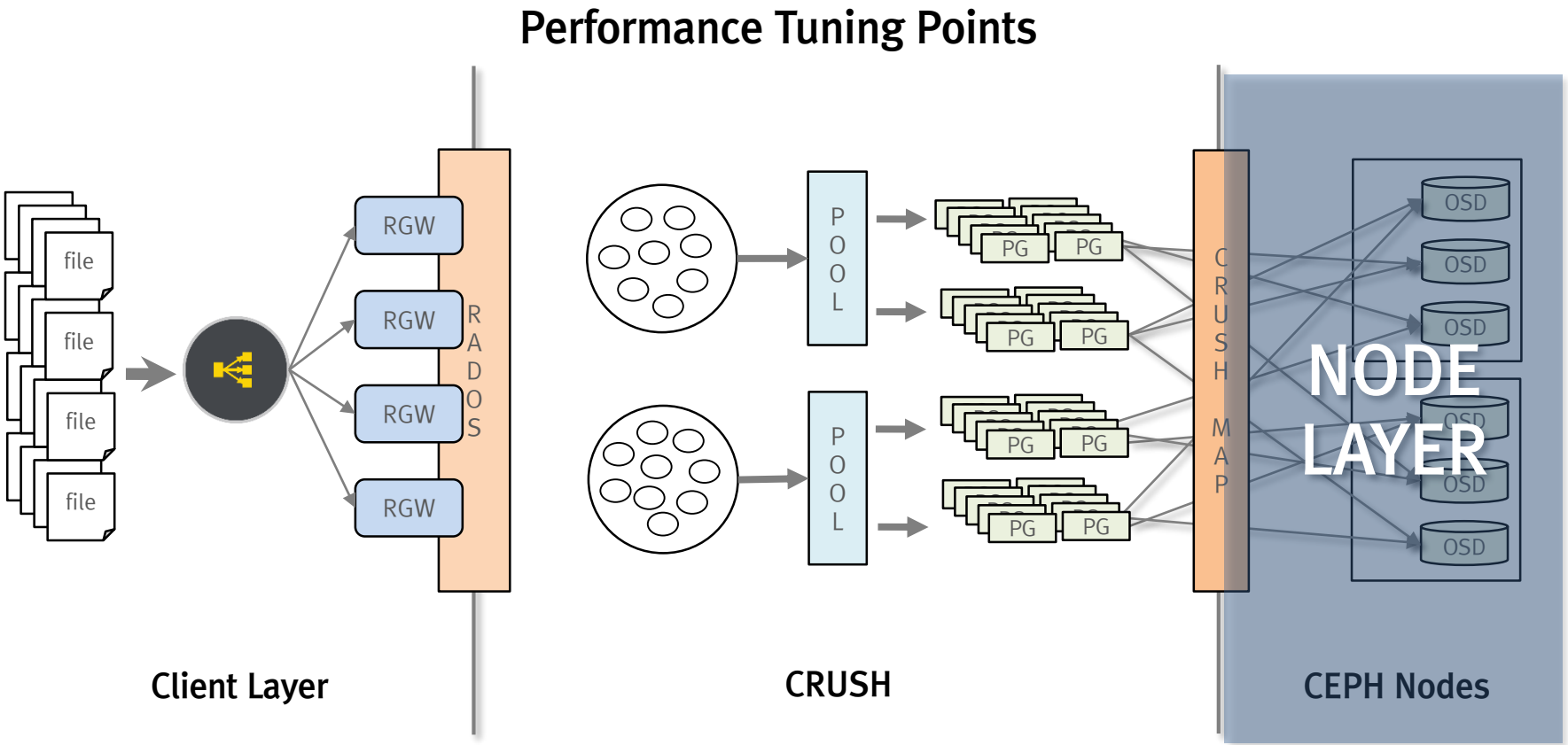

One Copy plus parity

- Cost-effective durability
- 1.5X (50% store overhead)
- Expensive recovery

# Performance Tuning for for Rados Pool



# Performance of Object Storage

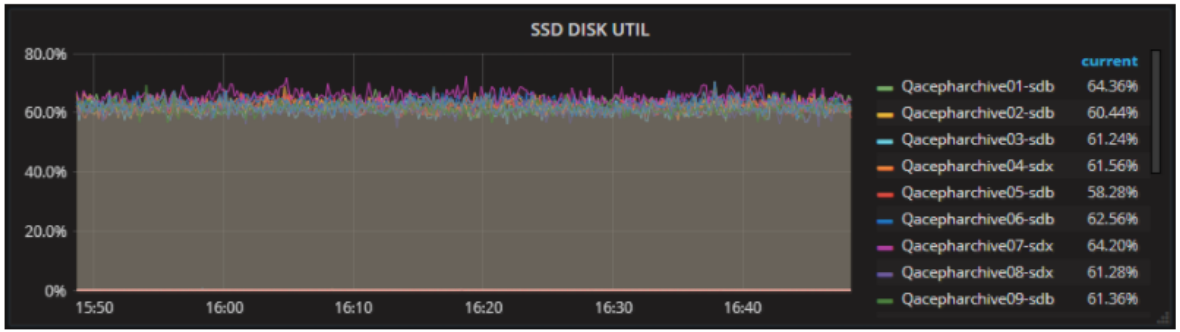


# Tips for Bluestore Design

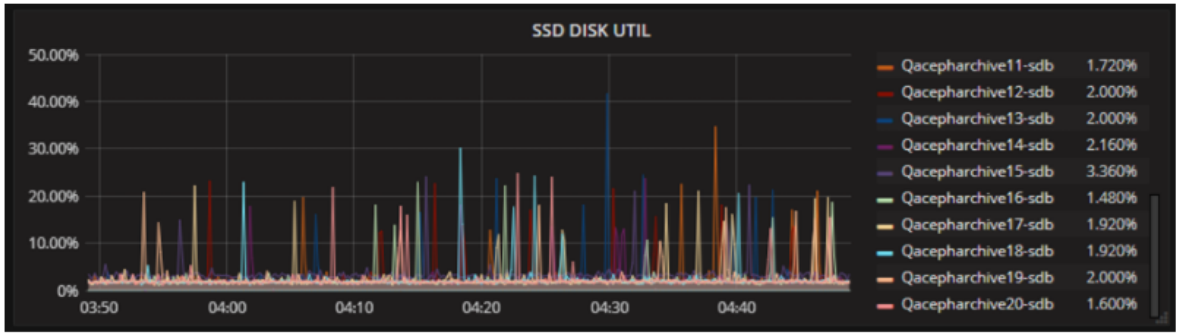
- Multi Device 구성
  - 3 종류의 Device가 존재할 경우
    - WAL : 가장 빠른 DEVICE / DB : 다음으로 빠른 DEVICE / DATA : HDD 영역
  - DB,WAL이 동일한 디바이스를 사용할 경우 → block.db만 지정
  - 주의사항
    - WAL의 경우는 할당된 용량의 파티션만 사용.
    - DB의 경우 할당된 용량의 파티션을 다 쓰게 되면 Data 영역에 데이터를 저장 → 성능 저하 발생
  - **Sizing Guide**
    - 보통 WAL → 512 MB ~ 1GB
    - DB
      - 특별한 가이드가 없음
      - OSD 하나 기준으로 object 하나당 6KB 정도의 DB를 사용함.
      - 백만 개의 object를 하나의 OSD에 저장하려면 6GB가 필요
      - 1TB OSD당 10GB DB를 사용

# Tips for Bluestore Design

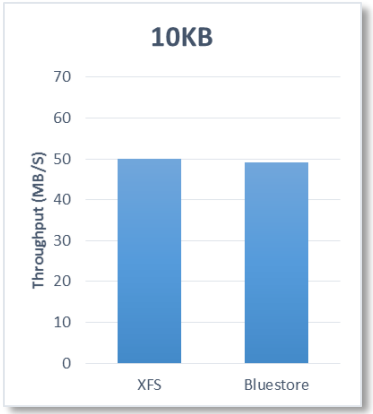
XFS




Bluestore



# Tips for Bluestore Design





Disk 부하가 없을 경우


Disk 부하가 있을 경우

# Performance Tuning for H/W

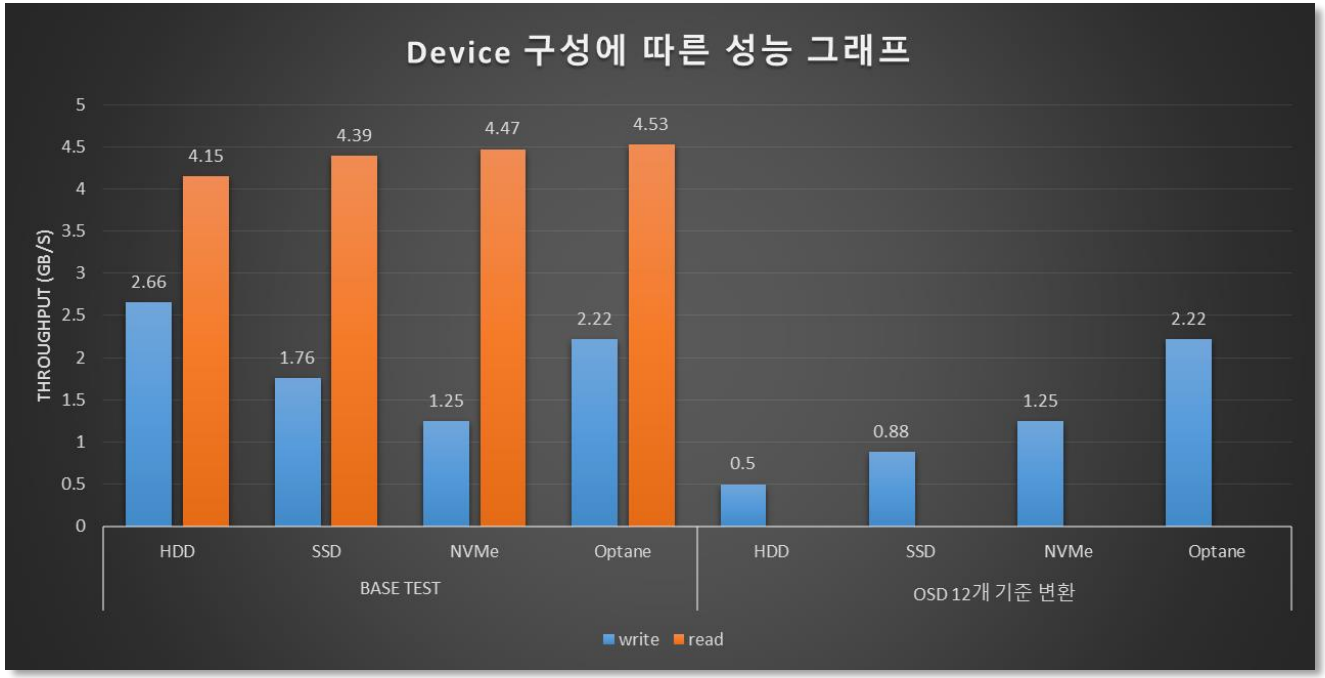

SATA SSD

SAS SSD

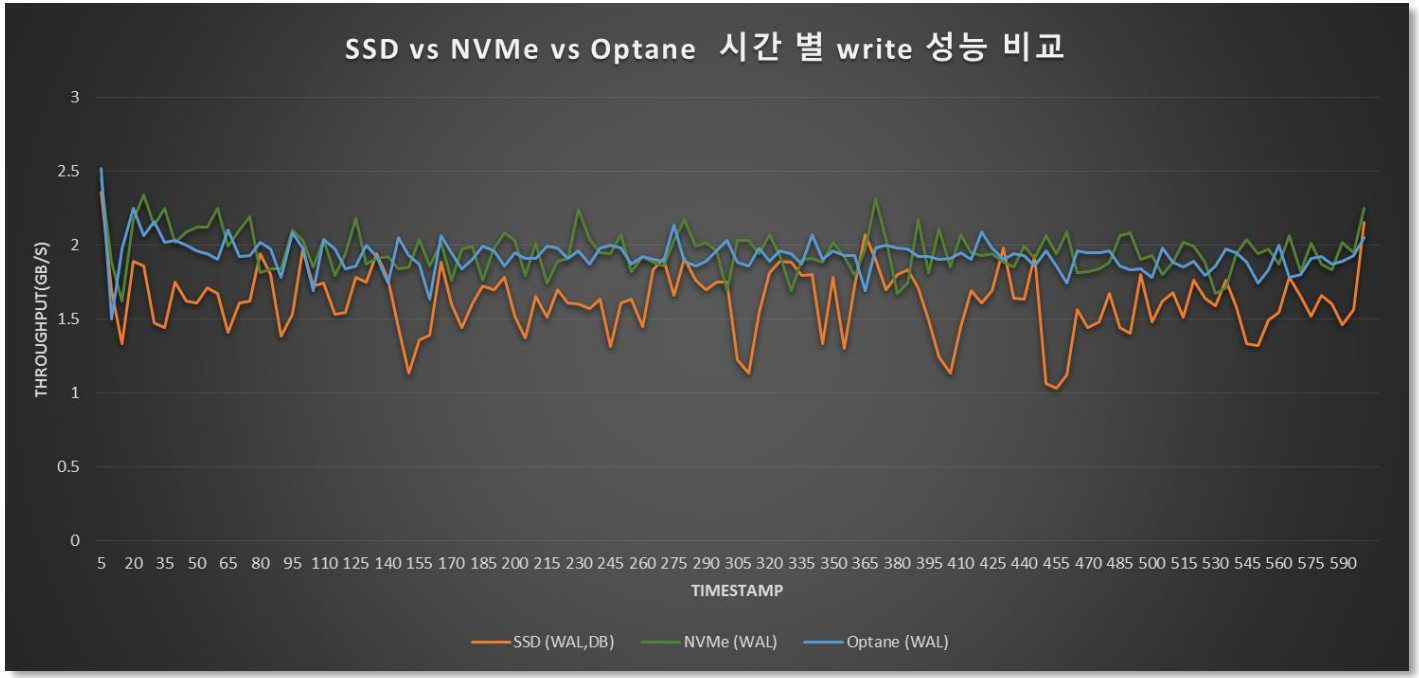
NVMe SSD

Optane

# Performance Tuning for H/W



# Performance Tuning for H/W



# NEXT..

- Multi Sites
- Object Storage as a Data Lake
- Origin CDN Storage

1 x 9 + 2 = 11
12 x 9 + 3 = 111
123 x 9 + 4 = 1111
1234 x 9 + 5 = 11111
12345 x 9 + 6 = 111111
123456 x 9 + 7 = 1111111
1234567 x 9 + 8 = 11111111
12345678 x 9 + 9 = 111111111
123456789 x 9 +10= 1111111111

There is no MAGIC NUMBER

# Q & A

netmarble
Games

# 감사합니다