SK telecom

Advanced Features for Ceph: Deduplication and QoS

Myoungwon Oh(omwmw@sk.com), Byungsu Park(bspark8@sk.com)

SW-Defined Storage Lab.

Network-IT Convergence R&D Center

SK텔레콤 NIC 기술원

- COSMOS(Composable, Open, Scalable, Mobile-Oriented System)
  - Telco 인프라 혁신
  - 개방형 All-IT 인프라 구축

Open Software

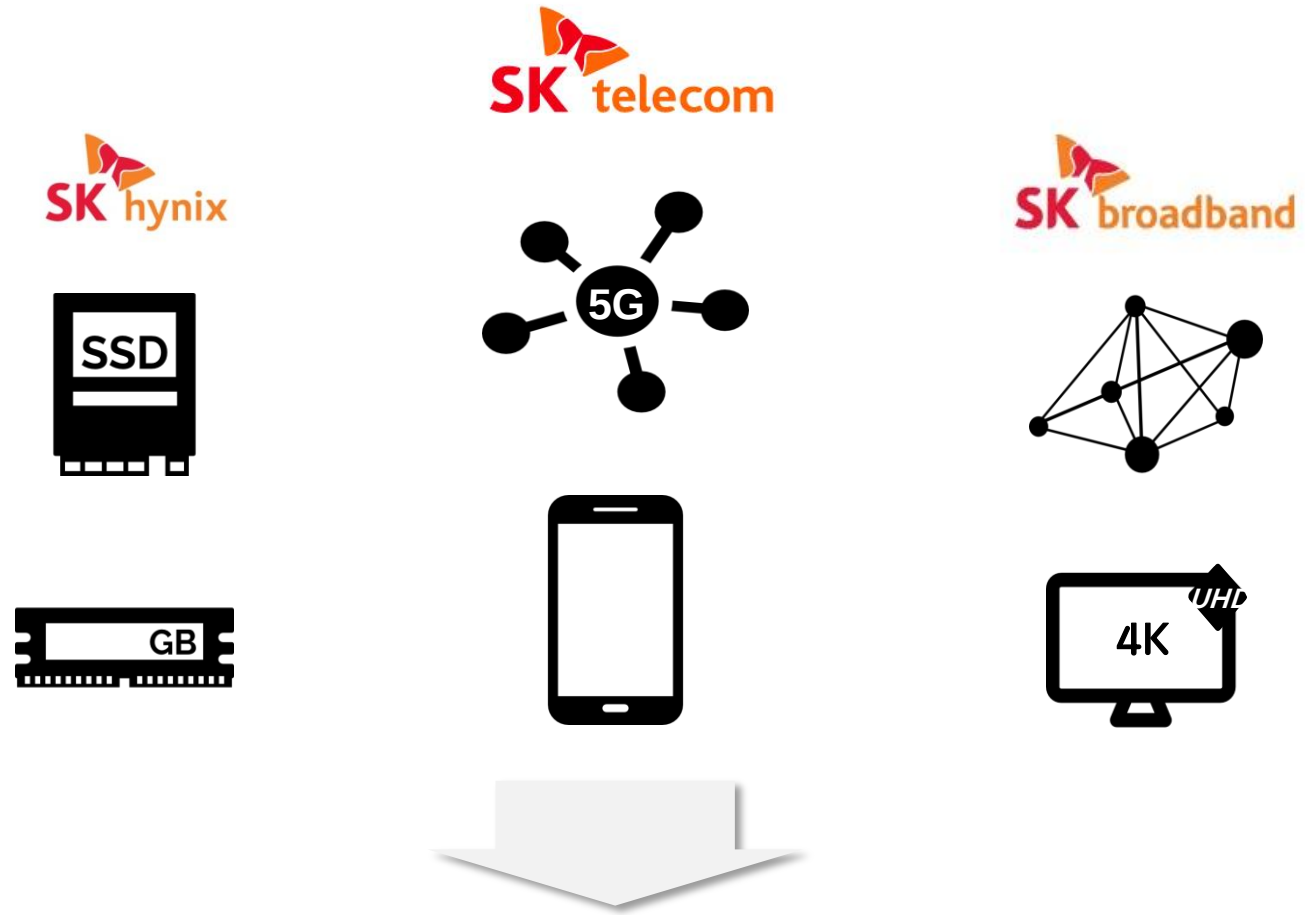
OpenStack, ONOS, Ceph, Cloud Foundry, Hadoop …

Software-Defined Technologies

Open Hardware

Open Compute Project (OCP), Telecom Infra Project (TIP)
All-flash Storage, Server Switch, Telco Specific H/W…

왜 SKT는 All-Flash Storage에 관심을?



Flash device

High Performance, Low Latency, SLA

Software-Defined Storage

	Traditional Storage	Emerging SDS
Architecture	**Proprietary** H/W, Proprietary S/W Only	**Commodity** H/W, **Open Source** S/W Available
Benefit	Turnkey Solution	Low Cost, **Flexible** Mgmt., High Scalability
Considerations	High Cost, **Inflexible** Mgmt., Limited Scalability, **Vendor Lock-in**	Tuning & **Development Efforts** Needed

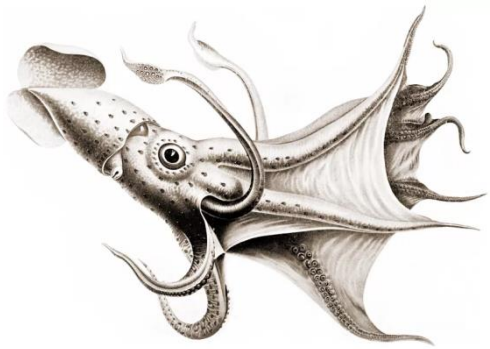
Target Workload

- IOPS-Sensitive Workload

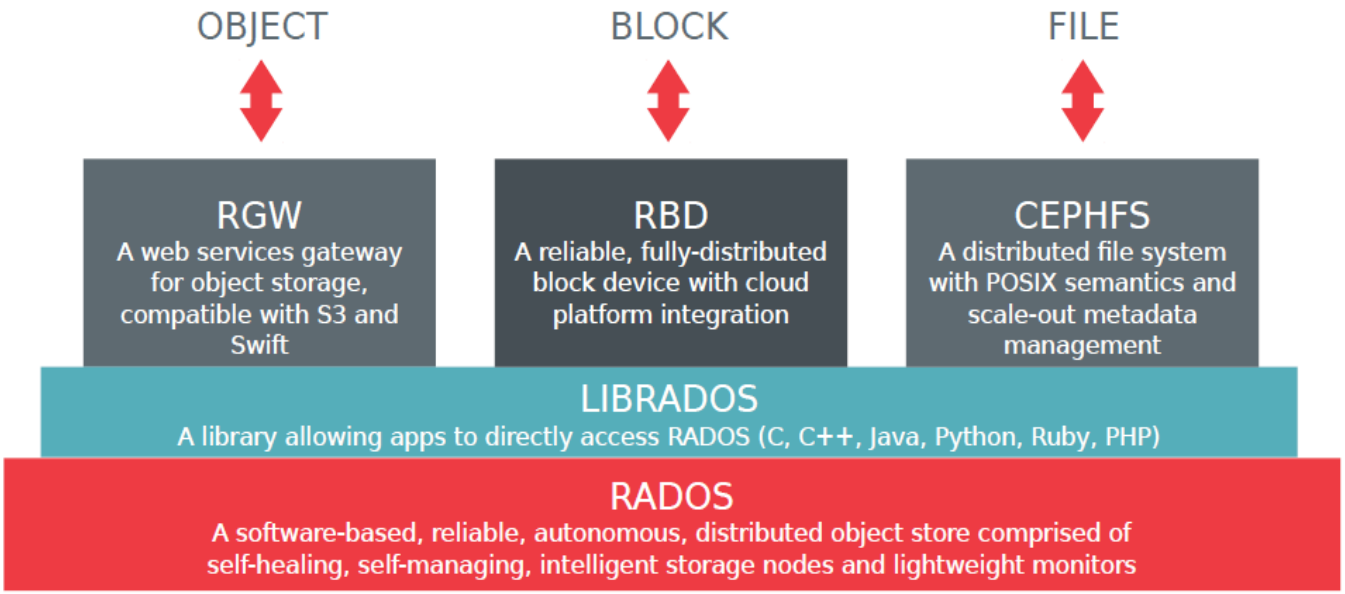
IOPS Sensitive	Throughput Sensitive	Capacity Sensitive
• Block-based(iSCSI, RBD) • SSD • Block Device, DB • **Random IO**	• File-based(NFS, CIFS) • SSD, HDD • Contents Sharing • **Sequential IO**	• Object-based (S3) • HDD • Archiving, Backup • **Sequential IO**

Ceph

- Ceph is a unified, distributed, massively scalable open source storage solution
  - Object, Block and File storage
- mostly LGPL open source project



- ✓ Failure is normal
- ✓ Self managing
- ✓ Scale out on commodity HW
- ✓ Everything runs in software



Community Contribution

- From Sage Weil's 'Ceph Project Update' in OpenStack Summit 2017 Boston

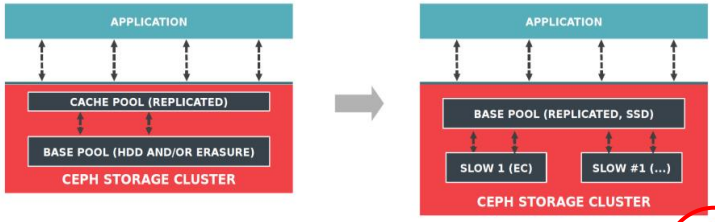
QUALITY OF SERVICE

- Ongoing background development
  - dmclock distributed QoS queuing
  - minimum reservations and priority weighting
- Range of policies
  - IO type (background, client)
  - pool-based
  - client-based
- Theory is complex
- Prototype is promising, despite simplicity
- Missing management framework

25

TIERING

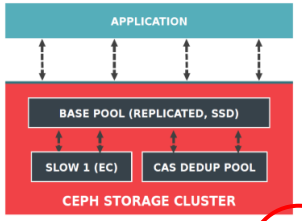
- new RADOS 'redirect' primitive
  - basically a symlink, transparent to librados
  - replace "sparse" cache tier with base pool "index"



27

DEDUPLICATION WIP

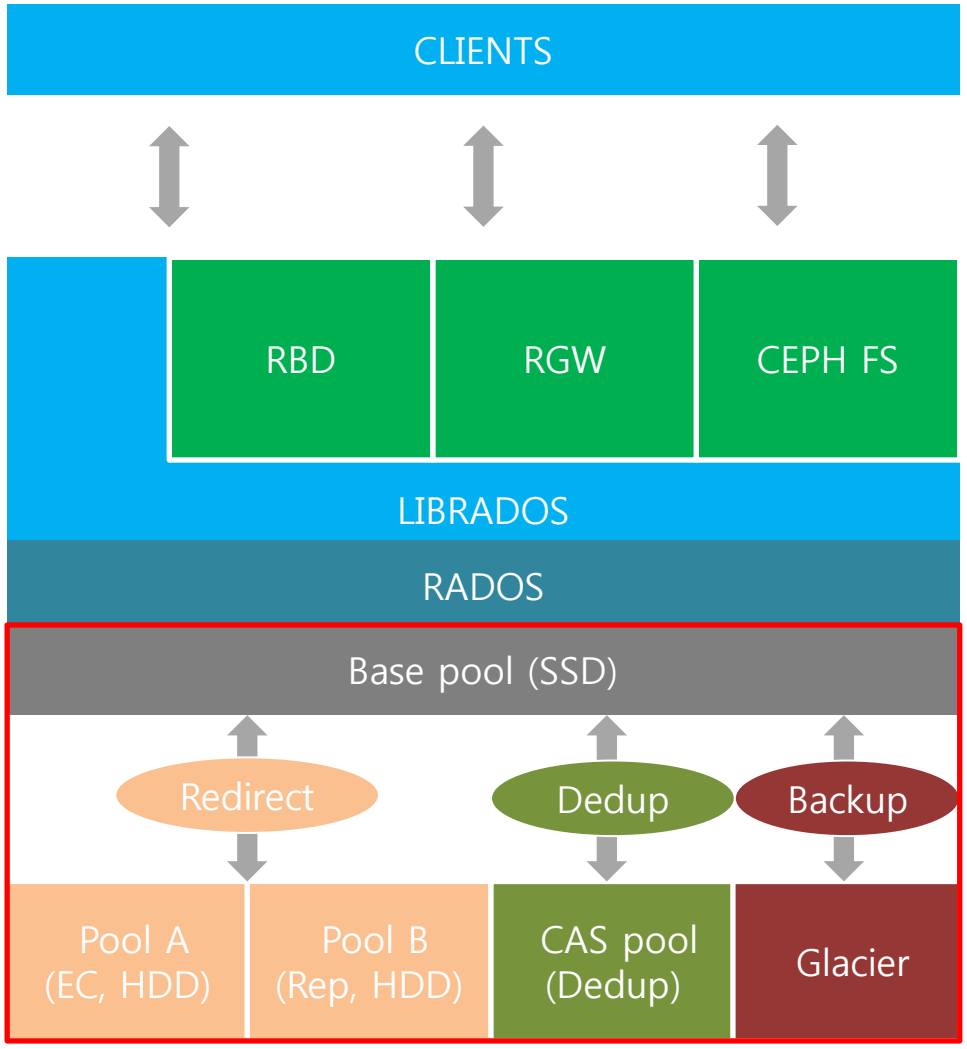
- Generalize redirect to a "manifest"
  - map of offsets to object "fragments" (vs a full object copy)
- Break objects into chunks
  - fixed size, or content fingerprint
- Store chunks in content-addressable pool
  - name object by sha256(content)
  - reference count chunks
- TBD
  - inline or post?
  - policy (when to promote inline, etc.)
  - agent home (embed in OSD, ...)



28

EXTENSIBLE TIER AND DEDUPLICATION

Ceph Extensible Tier

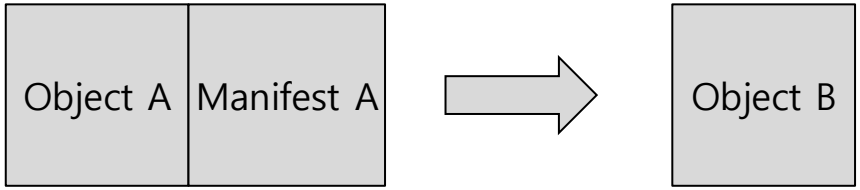


Object_manifest

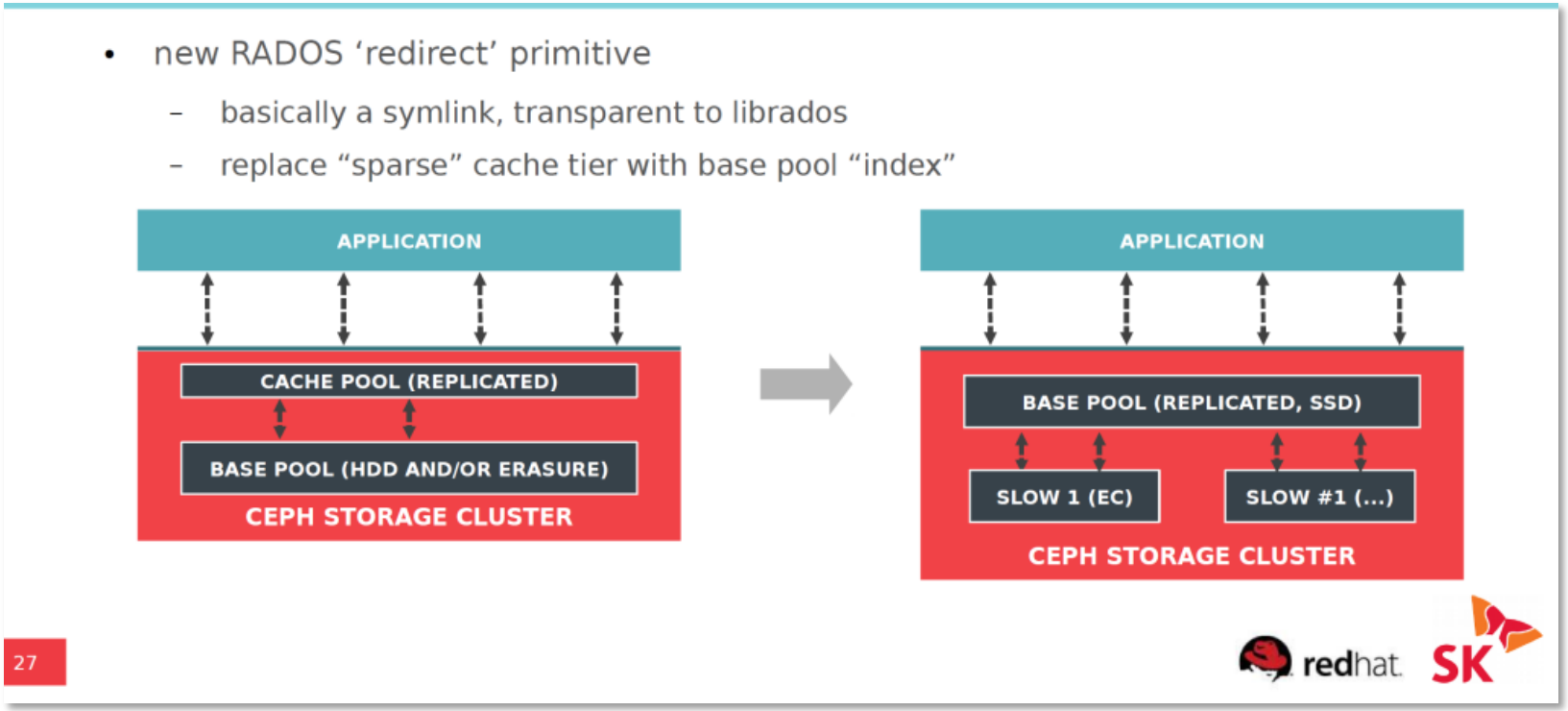
- **The key structure for extensible tier**

```
struct object_manifest_t {
        enum {
            TYPE_NONE = 0,
            TYPE_REDIRECT = 1,
            TYPE_CHUNKED = 2,
            TYPE_DEDUP = 3,
         };
         uint8_t type;  // redirect, chunked, ...
         ghobject_t redirect_target;
         map <uint64_t(offset), chunk_info_t> chunk_map;
};
```

- **Operations**
  - Proxy read, write
  - Flush, promote



Redirection

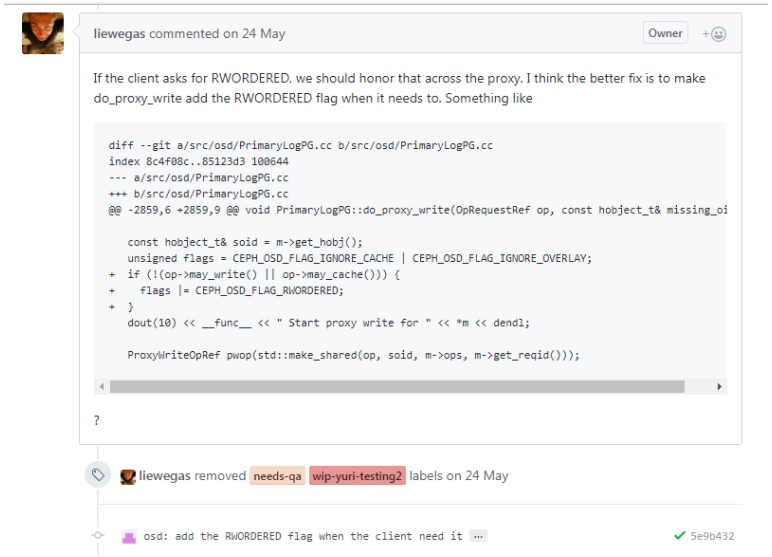


- Disadvantages of current tiering
  - Cache pool and base pool should be created at the same time if user wants to use "tier approach"
  - Existing cache tier only can move the data to a target pool (not different pools)

- Redirection
  - Pools (for backup, archiving) can be attached to the base pool
  - An object can be located in any pools (not a target pool)

Redirection

- Ceph upsteam process
  - Propose a new design
  - Create a pull request
  - Review and discussion
  - QA (if approved)
  - Merged (if QA is passed)



- Status
  - https://github.com/ceph/ceph/pull/14894 (merged)
  - https://github.com/ceph/ceph/pull/15325

Deduplication

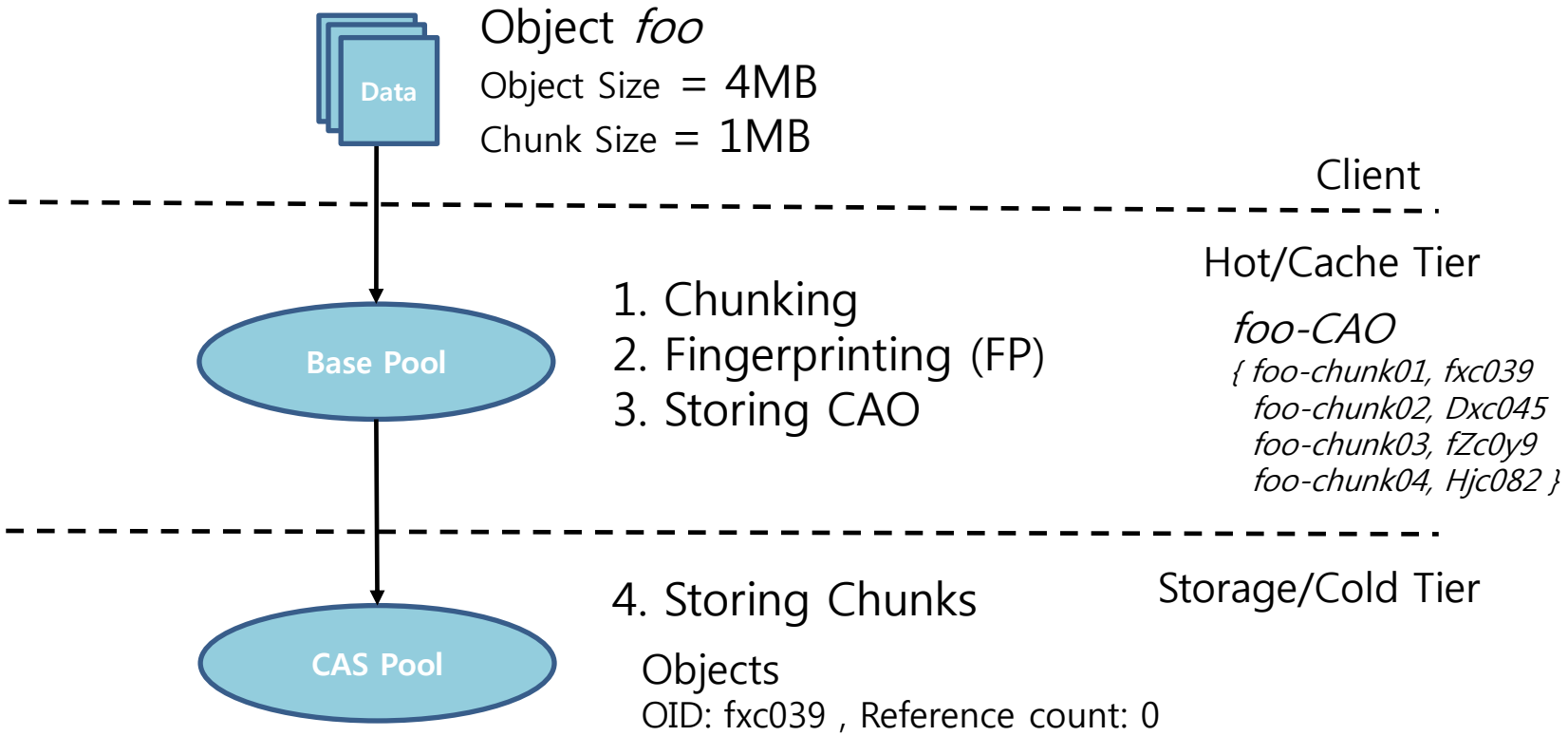
- Motivation (global deduplication)
  - Local dedup (in a block device level) can not cover whole data reduction in Cluster-wide environment.
  - Central dedup-metadata server does not fit in with shared-nothing distributed filesystem.

- Design goals
  - Efficient Metadata Management
  - Transparency to the application
  - Applicable to existing source
  - Minimizing performance degradation

  → Extensible tier + 2nd CRUSH

Deduplication

- Design

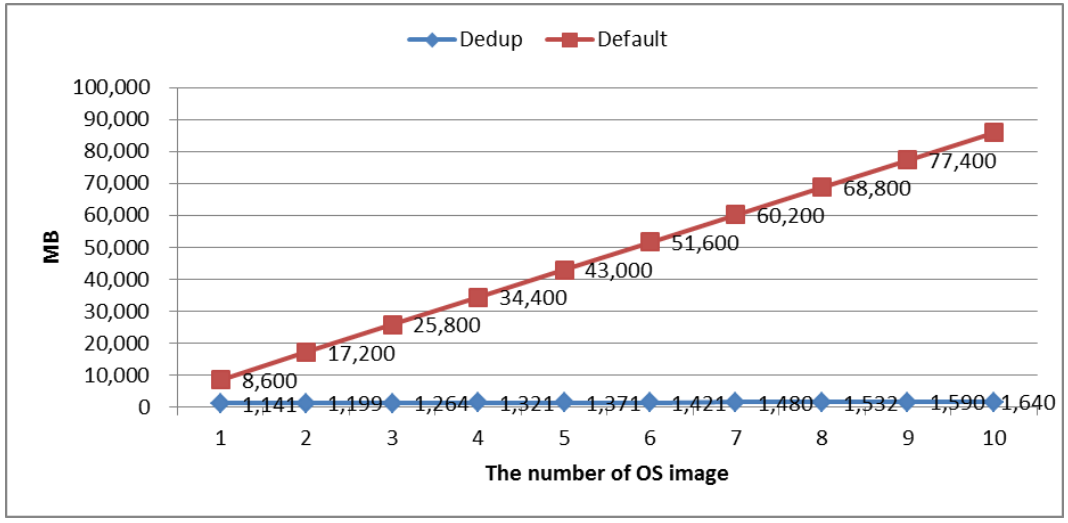


CAO→ Content Addressable Object

Deduplication

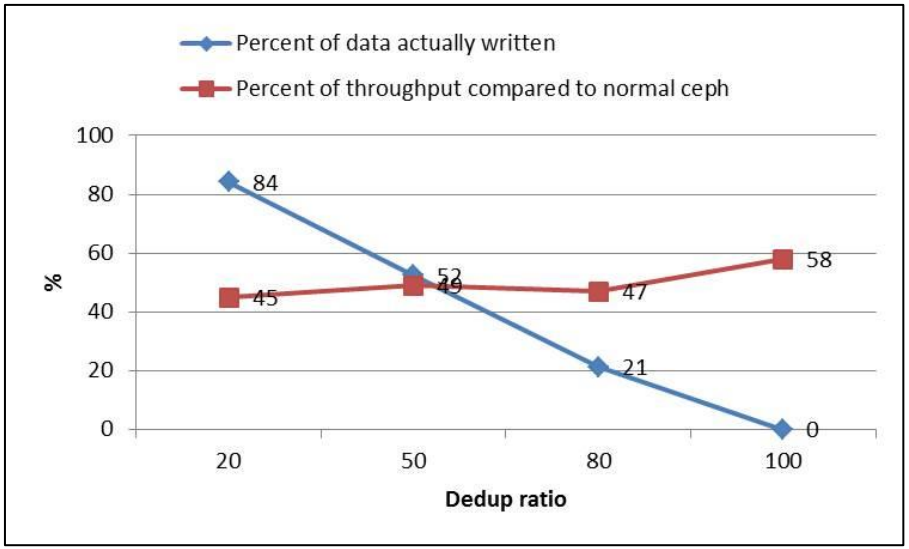
- The advantages
  - No change in current structure
  - No need to develop a separated deduplication metadata server
  - EC and Replication can be selected and used.
  - Don't need to consider data placement, load balancing and recovery.

- Evaluation



Deduplication

- Evaluation



- Problem (Low performance)
  - 512K chunk, RBD, Seq. write, FIO, single thread

dedup_ratio (write)	0	20	40	60	80	100
PROXY (KB)	45586	47804	51120	52844	56167	55302
WRITEBACK (KB)	13151	11078	9531	13010	9518	8319
ORIGINAL (KB)	121209	124786	122140	121195	122540	132363

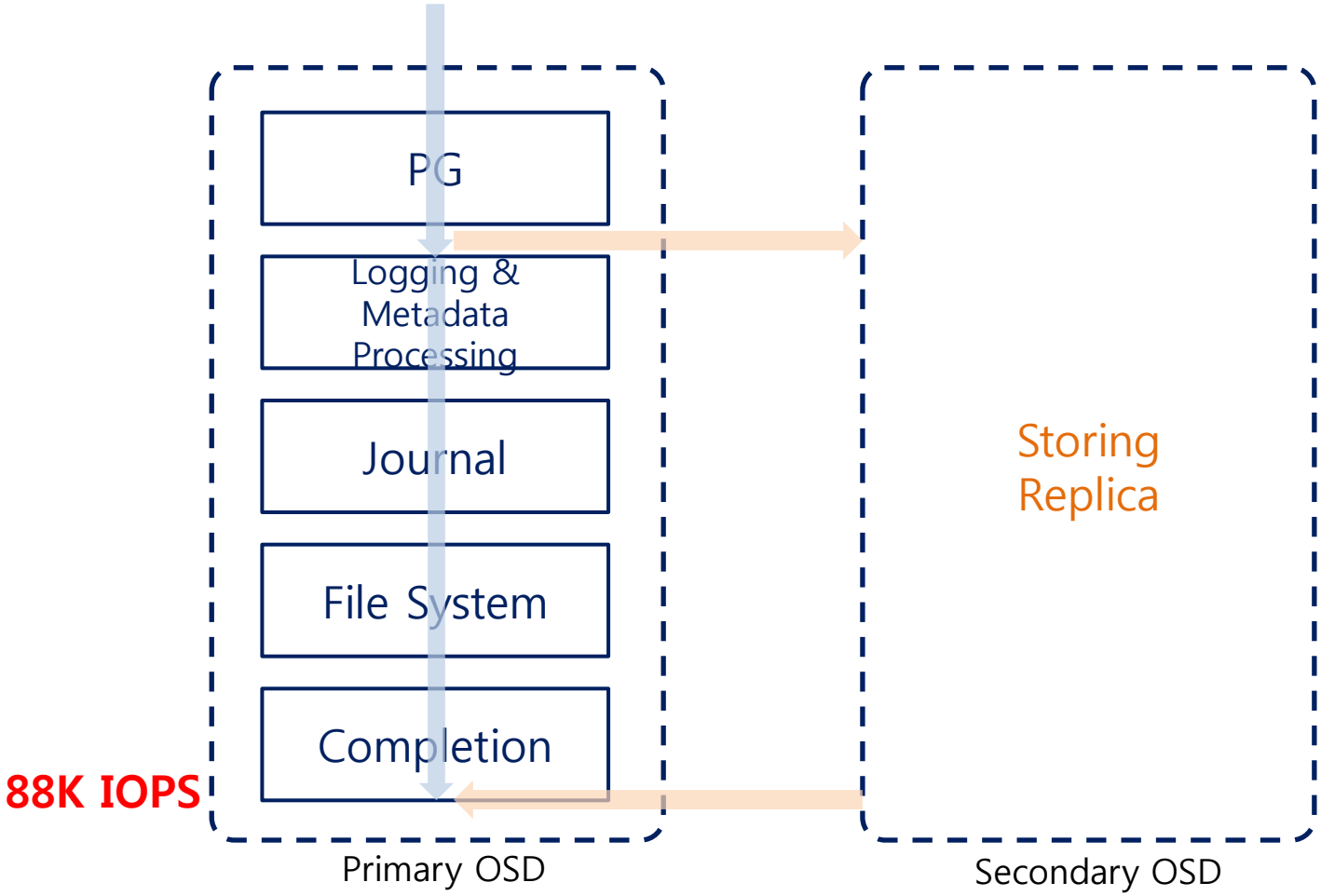
Deduplication

- Proposal
  - http://marc.info/?l=ceph-devel&m=148172886923985&w=2
- Design
  - http://marc.info/?l=ceph-devel&m=148646542200947&w=2
  - Pad document (with Sage Weil)
    - http://pad.ceph.com/p/deduplication_how_dedup_manifists
    - http://pad.ceph.com/p/deduplication_how_do_we_store_chunk
    - http://pad.ceph.com/p/deduplication_how_do_we_chunk
    - http://pad.ceph.com/p/deduplication_how_to_drive_dedup_process

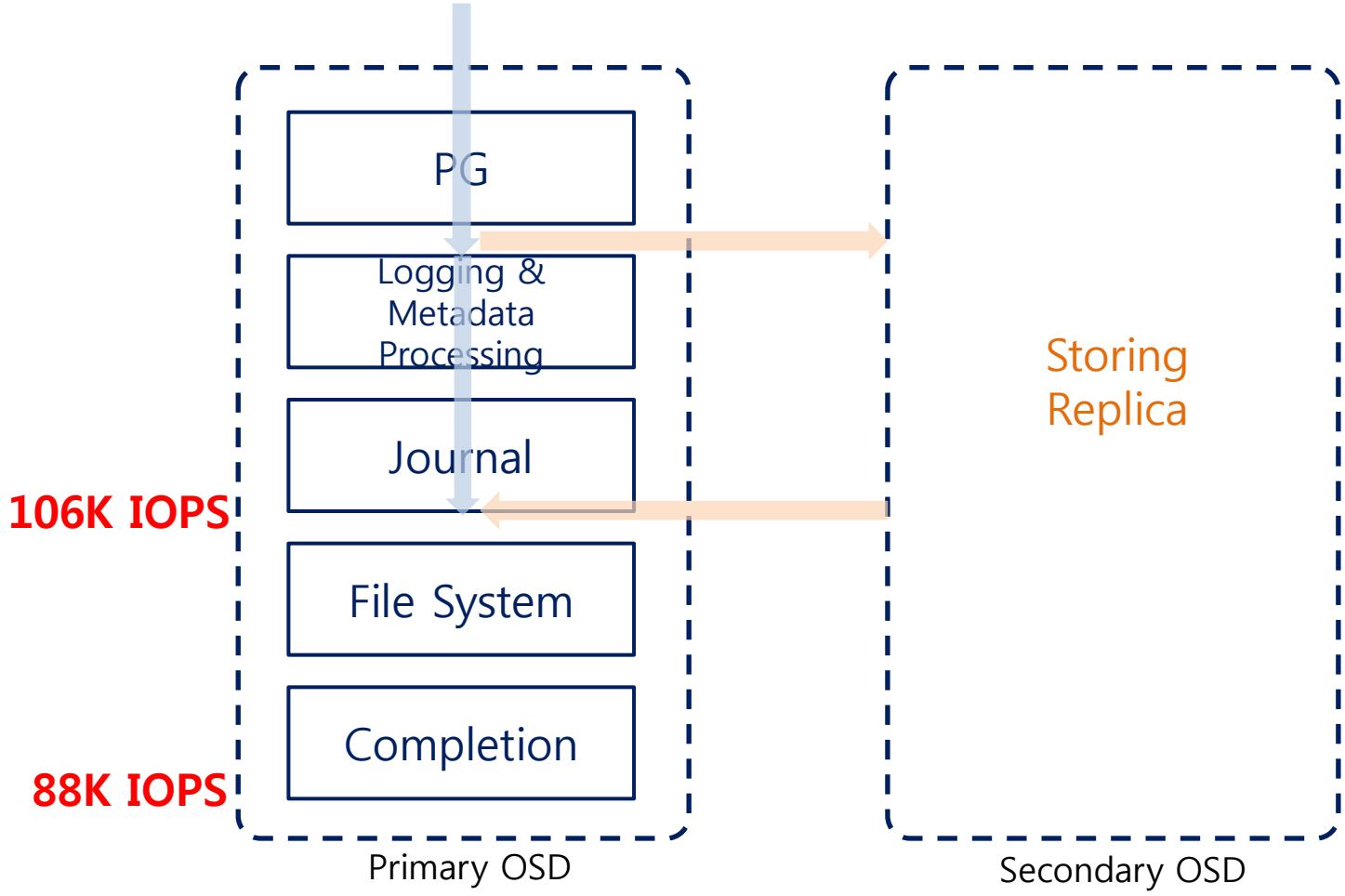
- Plan
  - Code in tier_agent or a rados op to safely/atomically install redirect manifests. (done)
    - Add new set-manifest rados operation, with encoding
    - Make osd proxy reads and writes when encountering a manifest
    - Add functional tests to test/librados/tier.cc
    - Make RadosModel randomly install redirects
  - Implement chunked manifest
    - Changes the read and write code to proxied reads/writes to many chunks work
    - https://github.com/ceph/ceph/pull/15482 (in progress)
  - Offline fingerprinting and then storing of chunked manifest (whole object or parts of it)

LOCK

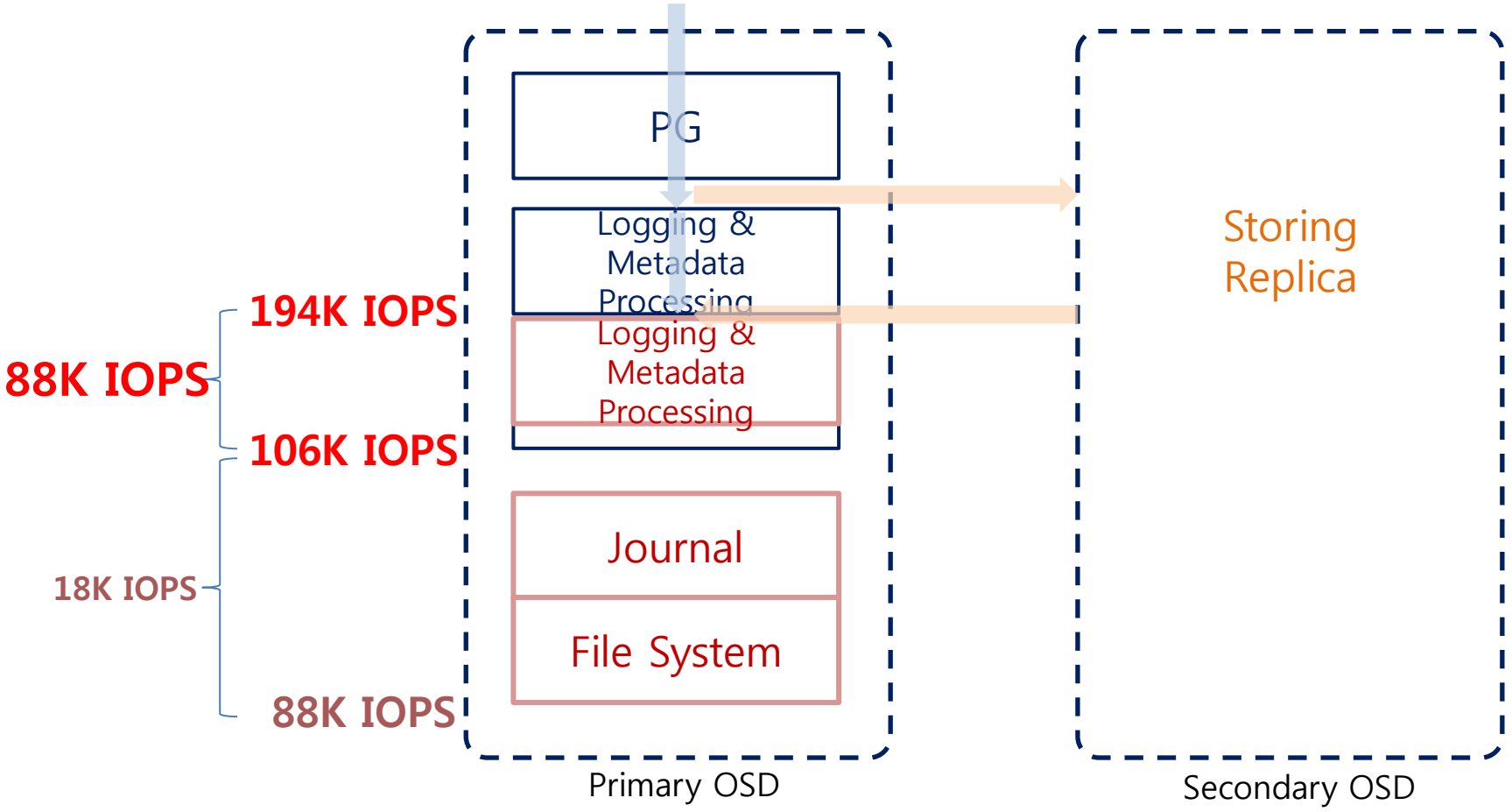
Overhead/Bottleneck Analysis



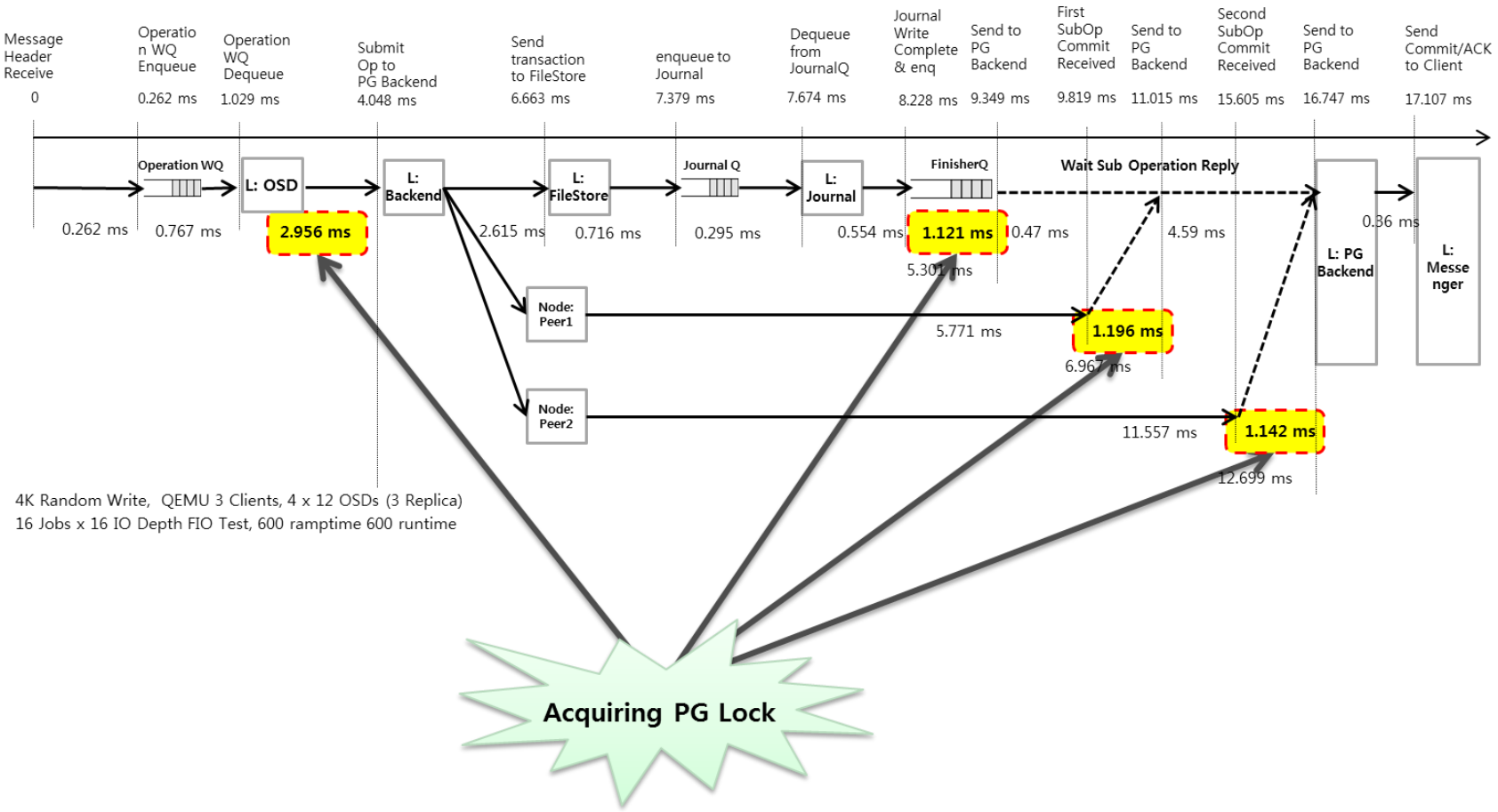
Overhead/Bottleneck Analysis



Overhead/Bottleneck Analysis



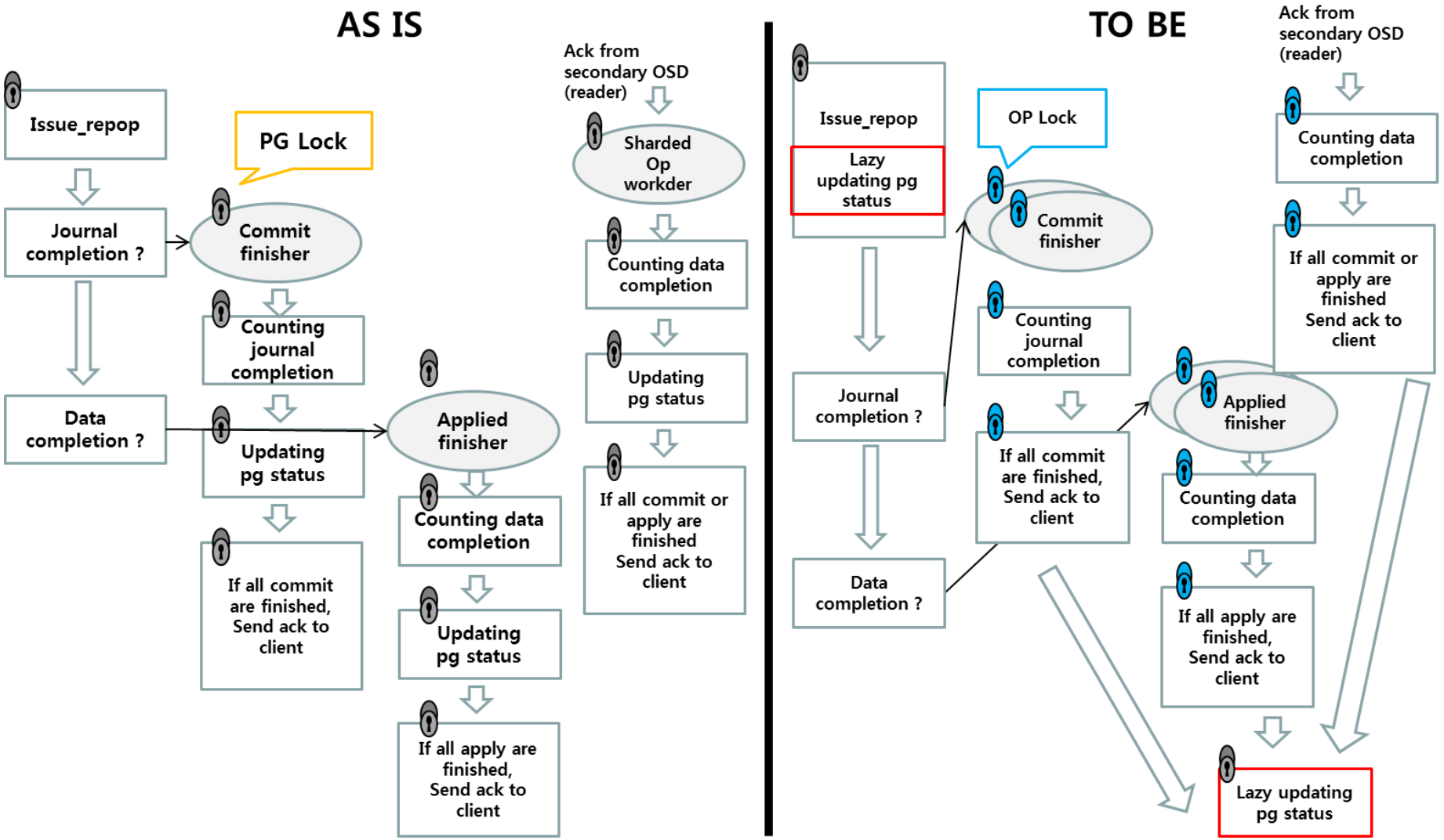
OSD Write Operation Latency Breakdown



6.3ms delay / Total 17ms

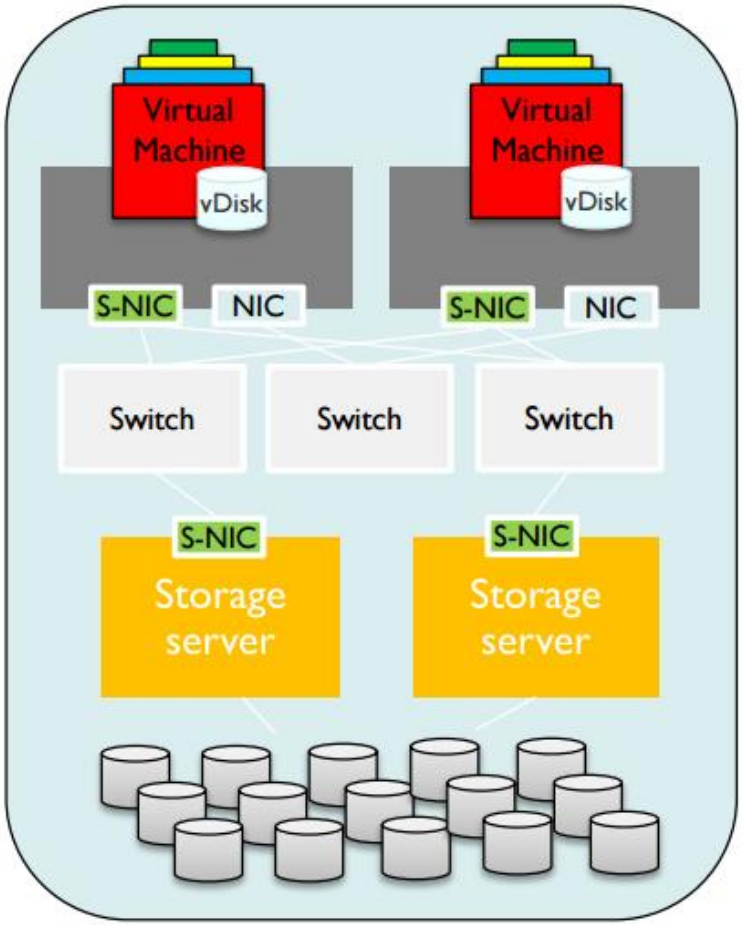
OP Lock Completion

- https://github.com/ceph/ceph/pull/11188
- https://github.com/ceph/ceph/pull/16030



QOS

Background : Enterprise Data Centers



- General purpose applications
- Application runs on several VMs
  - E.g. a multi-tier service
- Separate networks for VM-to-VM traffic and VM-to-Storage traffic
- Storage is virtualized
- Resources are shared
  - Filesharing protocols deployed

Source : Storage QoS for Enterprise Workloads, Tom Talpey & Eno Thereska, SNIA 2014
http://www.snia.org/sites/default/files/TomTalpey_Storage_Quality_Service.pdf

Service Level Agreement (SLA) Examples

- Minimum guaranteed (Reservation)
  - Reliability & Availability
    - MTBF (Mean Time Between Failure)
  - Performance
    - Storage Throughput
    - Storage IOPS
- Max allowed (Limit)
- Distribution of Remaining Available Resources (Proportional Share)
  - Proportional share based on client weight
  - Resource allocation to maximize service provider profitability

Storage SLA at Ceph

- Storage SLA at Ceph
  - Reliability & Availability
    - Supports various levels using replication or erasure code
    - Supports multiple levels with multiple OSD nodes and monitor node configuration based on Paxos protocol
    - c.f., Redhat All-flash storage system for Ceph's MTBF: 1.5 million hours *
  - Performance
    - Configured to effectively support performance
    - But doesn't supports different performance needs for each client

- Storage SLA Issues (challenges)
  - Various operations
    - Read/Write, Random/Sequential, Various IO sizes
  - Multiple clients
    - Shared resource competition among clients
  - Various device types
    - Various performance according to device type
  - Storage throughput change

*https://www.sandisk.com/content/dam/sandisk-main/en_us/assets/resources/enterprise/solutionbriefs/redhat-ceph-openstack-infiniflash-technology-brief.pdf

mClock

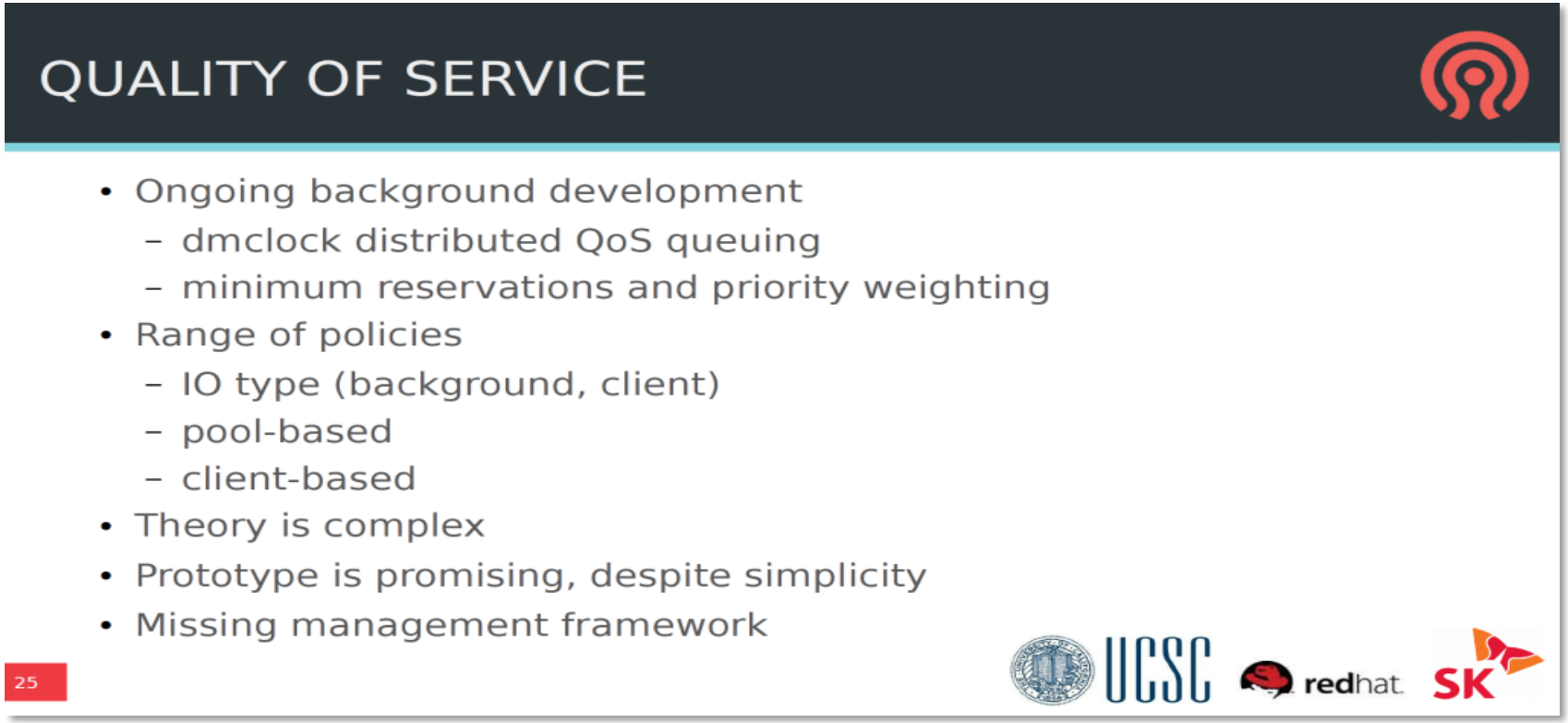
- Motivation
  - Lack of existing research to support QoS (reservation + proportional share) for storage
    - Support only simple proportional share
    - Support for other hardware devices (CPU, memory)

- Key Idea
  - Controls the number of I/O requests serviced by clients based on tags
  - Algorithm

```
if ( smallest reservation tag < current time) // constraint-based
    Schedule smallest eligible reservation tag
else // weight-based, reservations are met
    Schedule smallest eligible shares tag
    Subtract 1/r_k from reservation tags of VM k.
    A VM is eligible if (limit tag < current time)
```

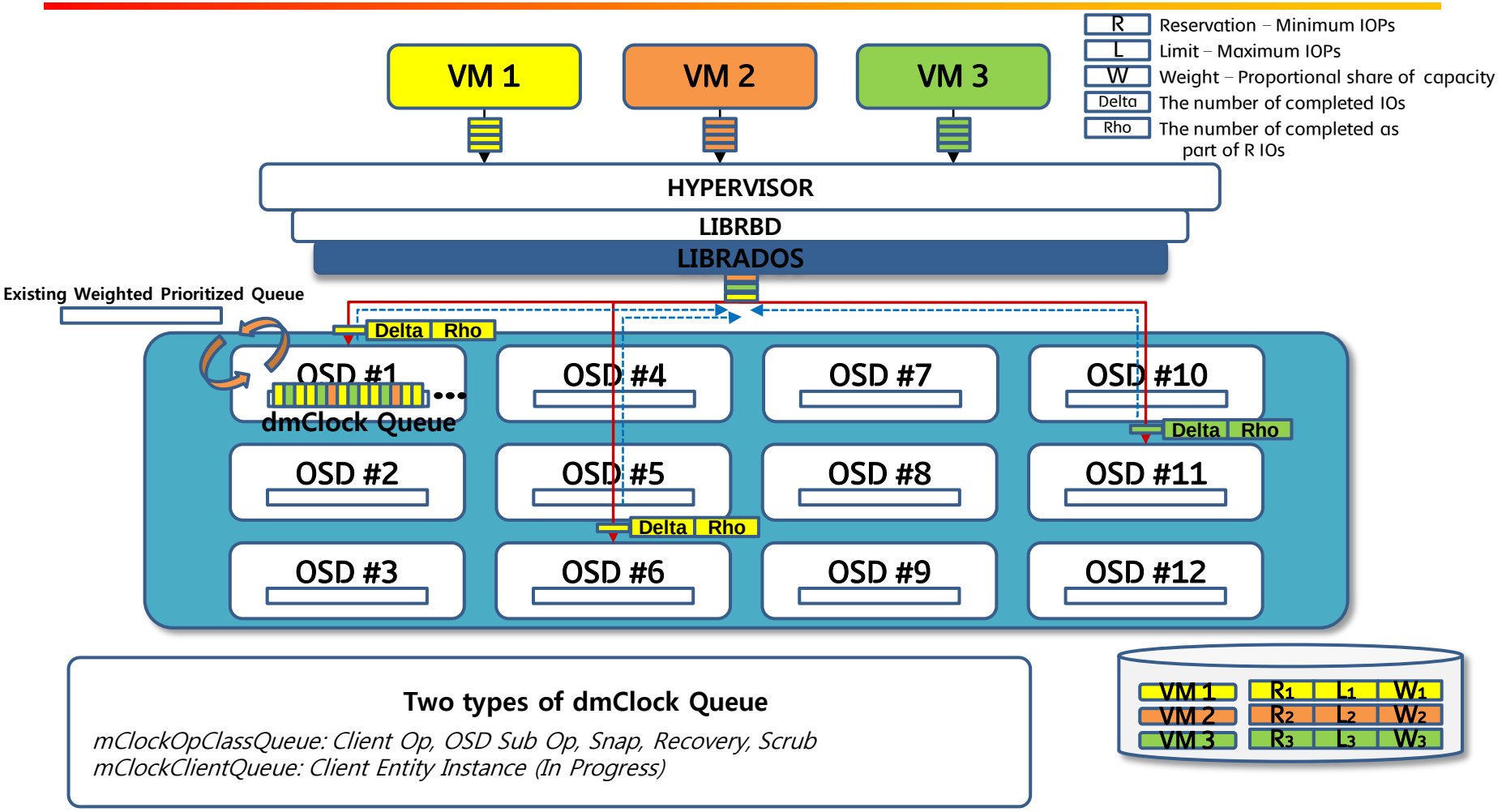
- Limitation
  - No consideration for multi-metric support
  - Proportional share may not be supported in some cases

Current QoS Progress in Community #1



- Source: Sage Weil's 'Ceph Project Update' in OpenStack Summit 2017 Boston

Current QoS Progress in Community #2



- Two PRs related to QoS in Ceph
  - Initial commit of dmclock QoS library (https://github.com/ceph/ceph/pull/14330)
  - Two new mClock implementations of the PG sharded operator queue (https://github.com/ceph/ceph/pull/14997)

dmClock based QoS Issue

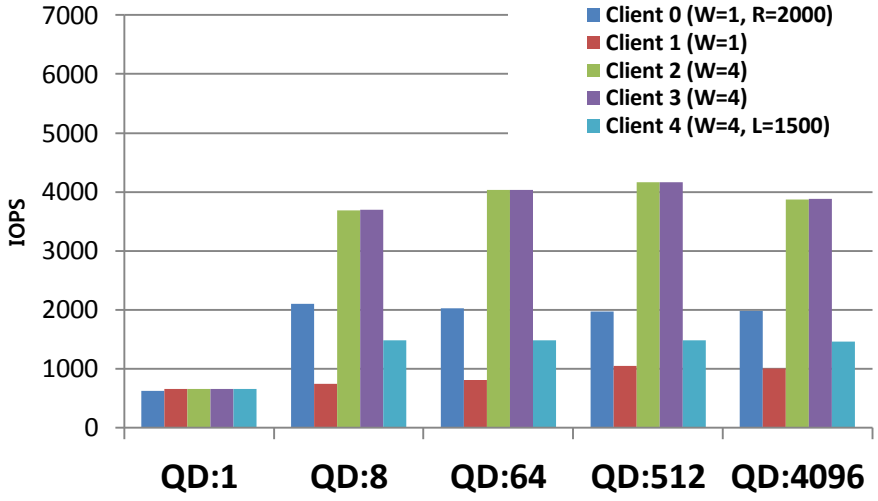


Fig.1. RW, BS:4K, TH:1, Shard:1
Shard per Thread:1

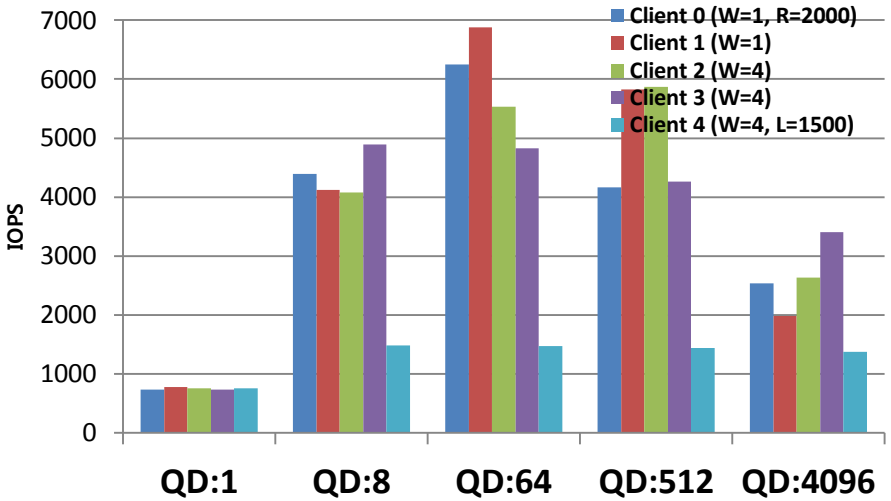


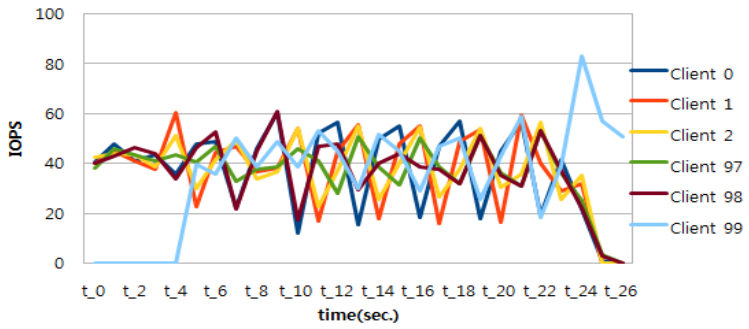
Fig.2. RW, BS:4K, TH:1, Shard:1
Shard per Thread:10

- dmClock Algorithm Failure Condition
  - Not enough workload has occurred at dmClock queue
  - Multiple dequeue threads try to pull OPs from dmClock queue at the same time

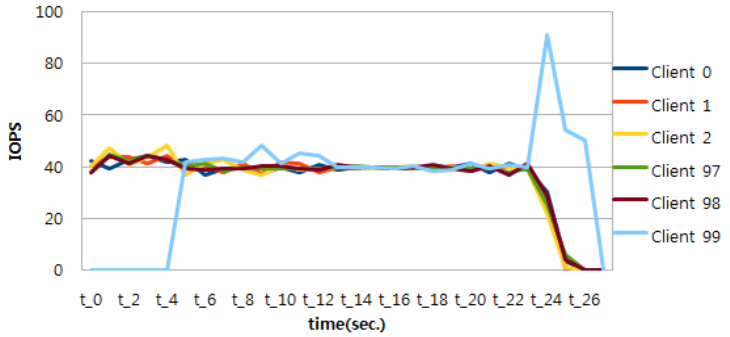
※ In both cases, no appropriate OPs rearrangement occurs

QoS on SKT: Contributions #1

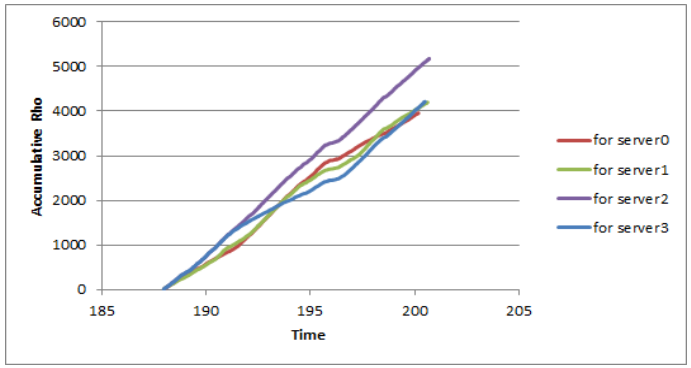
- Development and stabilization of QoS algorithm (https://github.com/ceph/dmclock/pull/)
  - Improved QoS instability in high load situations

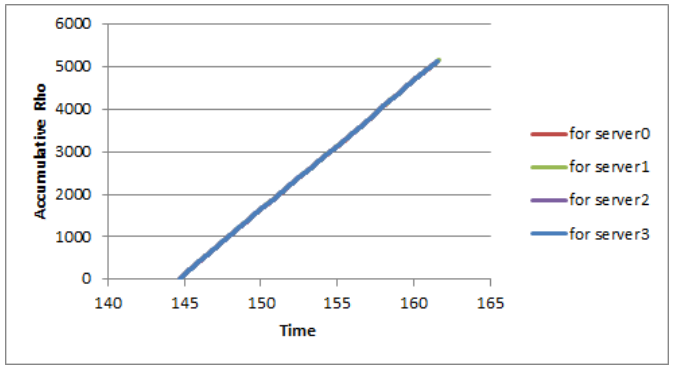


5.6x Improving



  - Improved QoS error due to heap management bug
  - Fix Tag adjustment algorithm calibrating proportional share tags against real time
  - Enable changing QoS parameters in run time (In Progress)
  - Improved Client QoS service status monitoring and reporting algorithm (In Progress)





QoS on SKT: Contributions #2

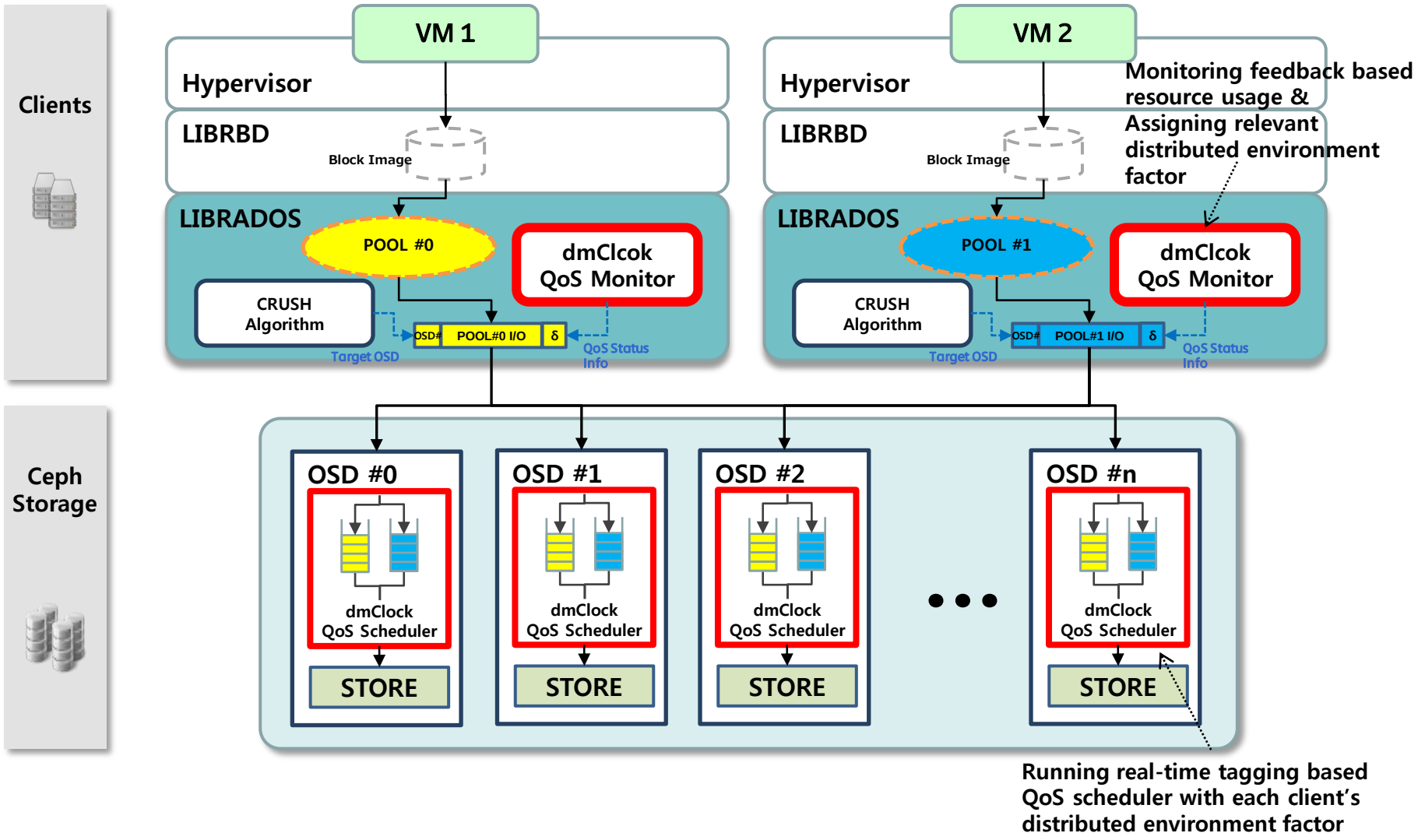
- Simulator stabilization and convenience improvement (https://github.com/ceph/dmclock/pull/)
  - File-based simulator settings
  - High performance setup and fixed simulation error reporting error
  - Fixed server node selection error

- Delivery of mClock distribution parameters (delta, rho and phase) + Enabling client QoS tracker (https://github.com/ivancich/ceph/pull/1), (In Progress)

- Hard Limit (Preparing PR)
  - Allow dmClock hard limit feature in Ceph

- Pool based dmClock Queue (Picture in next page, Preparing PR) (http://marc.info/?l=ceph-devel&m=148044600301927&w=2)
  - Develop mClockPoolQueue to support pool unit QoS in Ceph

QoS on SKT: Per-pool QoS Service in Ceph Cluster



QoS on SKT: In Progress & Plan

- Adaptive throttle with flash device aware QoS revision mechanism
  - Overcoming limitations of dmClock algorithm
- Fine-grained QoS support
  - Subdivided by RBD in current Pool-level QoS
- Ceph QoS support upon OpenStack
- New distributed environment QoS algorithm development and Ceph application
- Completion of the remaining parts of Ceph QoS

References

- All-Flash Ceph 구성과 최적화
  - OpenStack Days in Korea 2016
    - https://www.slideshare.net/openstack_kr/openstack-days-korea-2016-track1-all-flash-ceph
- AF Ceph: Ceph Performance Analysis & Improvement on Flash
  - Ceph Day Sunnyvale 2016
    - https://www.slideshare.net/Inktank_Ceph/af-ceph-ceph-performance-analysis-and-improvement-on-flash
- AFCeph: Problem, Progress and Plan
  - Ceph Day Seoul 2016
    - https://www.slideshare.net/Inktank_Ceph/ceph-day-seoul-afceph-skt-scale-out-storage-ceph
- Performance Optimization for All Flash Scale-out Storage
  - 2016 IEEE Cluster Computing
    - http://ceph.com/wp-content/uploads/2017/03/performance_optimization_for_all_flash_scale-out_storage-SK_Telecom.pdf
- CEPH Optimization on SSD
  - Deview 2016
    - https://deview.kr/2016/schedule#session/140