SDC
STORAGE DEVELOPER CONFERENCE
SNIA SANTA CLARA, 2015

Big Data Analytics on Object Storage
-- Hadoop over Ceph* Object Storage with SSD Cache

David Cohen (david.e.cohen@intel.com)
Yuan Zhou (yuan.zhou@intel.com)
Jun Sun (jun.sun@intel.com)
Weiting Chen (weiting.chen@intel.com)
Sep 2015

Agenda

- Big Data Analytics over Cloud
- Deployment considerations
- Design details of BDA over Ceph* Object Storage
- Sample Performance testing and results

Introduction

- Intel Cloud computing and Big Data Engineering Team
- Global team, local focus
- Open source @ Spark, Hadoop, OpenStack, Ceph, NoSQL etc.
- Working with community and end customers closely
- Technology and Innovation oriented
  - Real-time, in-memory, complex analytics
  - Structure and unstructured data
  - Agility, Multitenancy, Scalability and elasticity
  - Bridging advanced research and real-world applications

Agenda

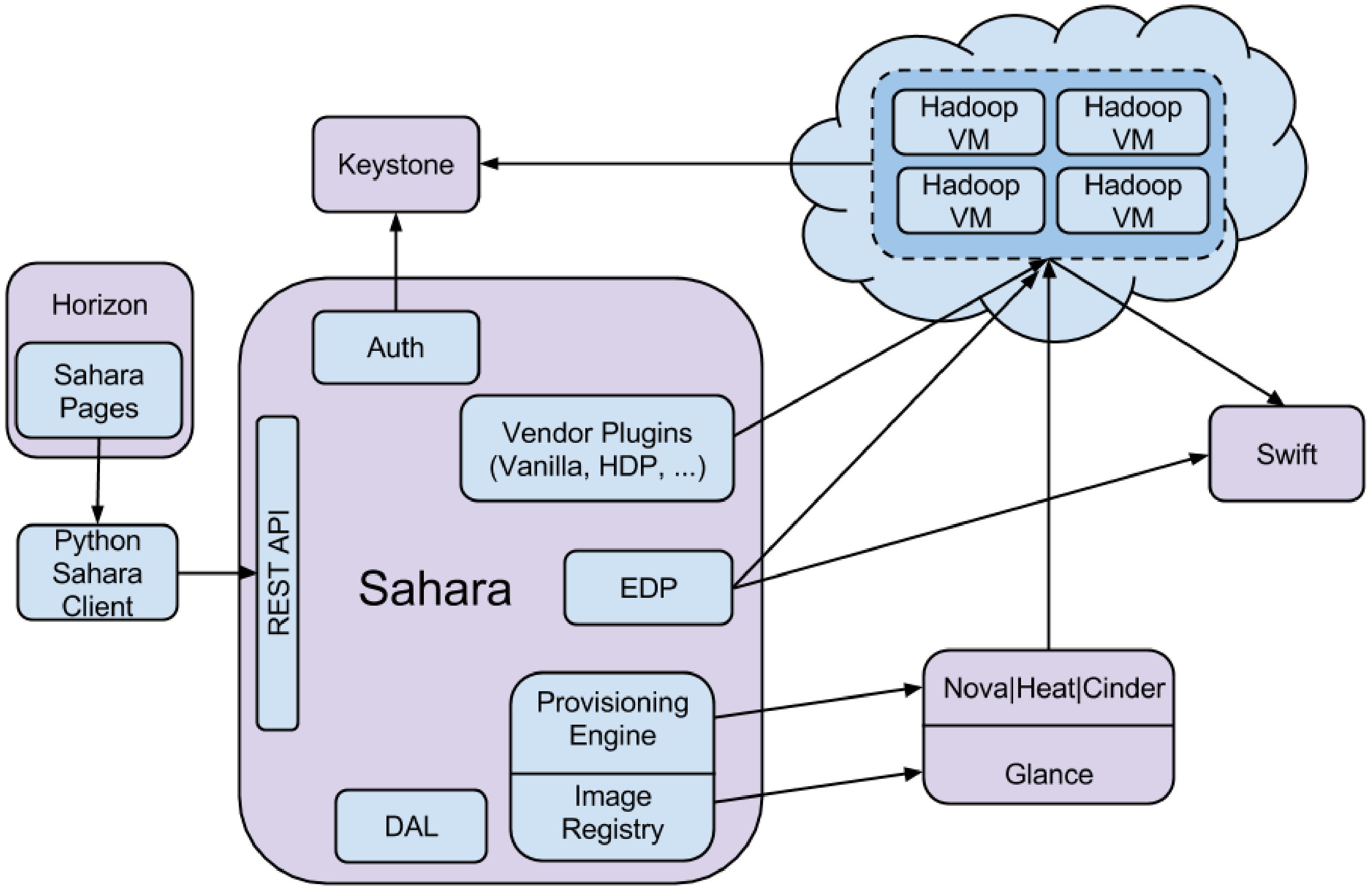
- **Big Data Analytics over Cloud**
- Deployment considerations
- Design details of BDA over Ceph* Object Storage
- Sample Performance testing and results

Big Data Analytics over Cloud

- You or someone at your company is using AWS, Azure, or Google cloud platform
- You're probably doing it for easy access to OS instances, but also the modern application features, e.g. AWS' EMR or RDS or Storage
- Migrating to, or even using, OpenStack infrastructure for workloads means having application features, e.g. Sahara & Trove
- Writing applications is complex enough without having to manage supporting (non-value-add) infrastructure
- **Things are also happening on the cloud computing side**

Big Data Analytics over Cloud: OpenStack Sahara

- Repeatable cluster provisioning and management operations
- Data processing workflows (EDP)
- **Cluster scaling (elasticity), Storage integration (Swift, Cinder, HCFS)**
- Network and security group (firewall) integration
- Service anti-affinity (fault domains & efficiency)



Big data analytics on Object Storage

- Object storage provides restful/http access which is a good choice for archival storage
- It's just like a K-V storage and be able to build on commodity hardware
- Currently Hadoop solution usually relies on HDFS
- Hadoop over Swift (SwiftFS) provides an much easier way
  - Which saves much efforts on the copying between HDFS and local storage

Agenda

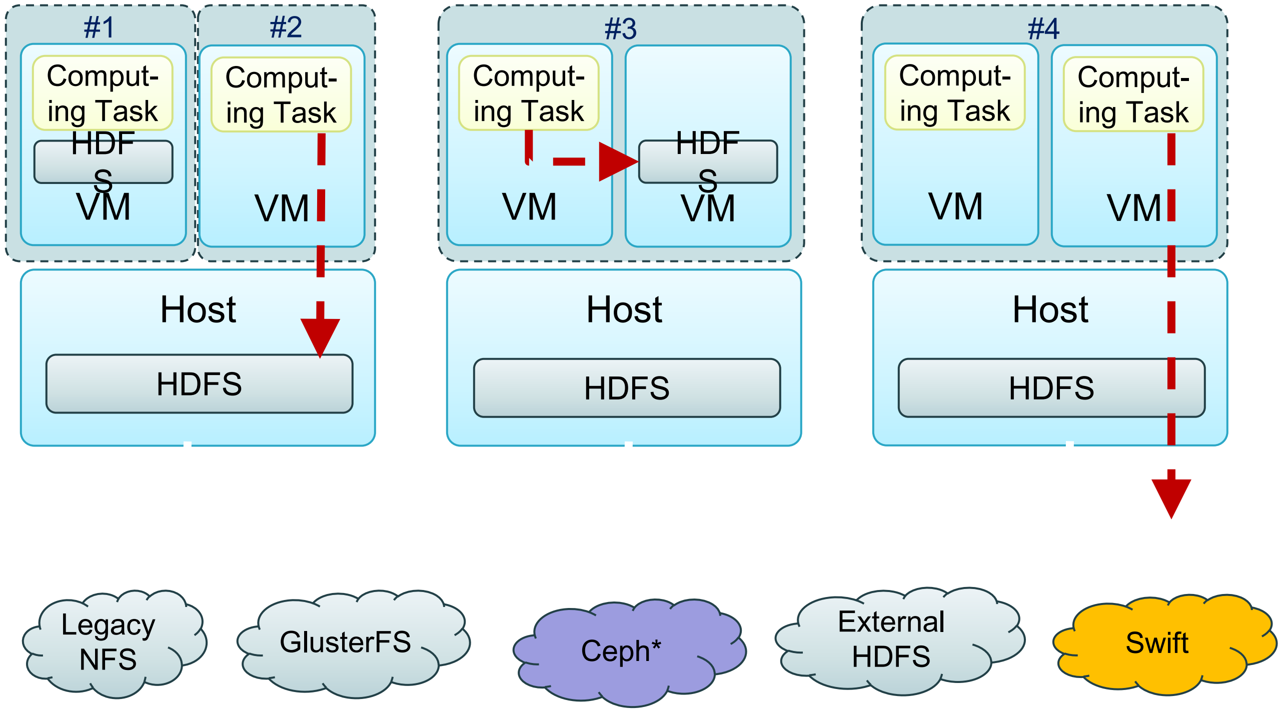
- Big Data Analytics over Cloud
- **Deployment considerations**
- Design details of BDA over Ceph* Object Storage
- Sample Performance testing and results

Deployment Consideration Matrix

Data Processing API	Traditional	EDP (Sahara native)	*3rd party APIs*			
Distro/Plugin	Vanilla	Spark	Storm	CDH	HDP	MapR
Compute	VM	Container	Bare-metal			
Storage	Tenant vs. Admin provisioned	Disaggregated vs. Collocated	HDFS vs. other options			

Storage Architecture

- Tenant provisioned (in VM)
  - HDFS in the same VMs of computing tasks vs. in the different VMs
  - Ephemeral disk vs. Cinder volume
- Admin provided
  - Logically disaggregated from computing tasks
  - Physical collocation is a matter of deployment
  - For network remote storage, Neutron DVR is very useful feature
- A disaggregated (and centralized) storage system has significant values
  - No data silos, more business opportunities
  - Could leverage Manila service
  - Allow to create advanced solutions (.e.g. in-memory overlayer)
  - More vendor specific optimization opportunities



Scenario #1: computing and data service collocate in the VMs
Scenario #2: data service locates in the host world
Scenario #3: data service locates in a separate VM world
Scenario #4: data service locates in the remote network

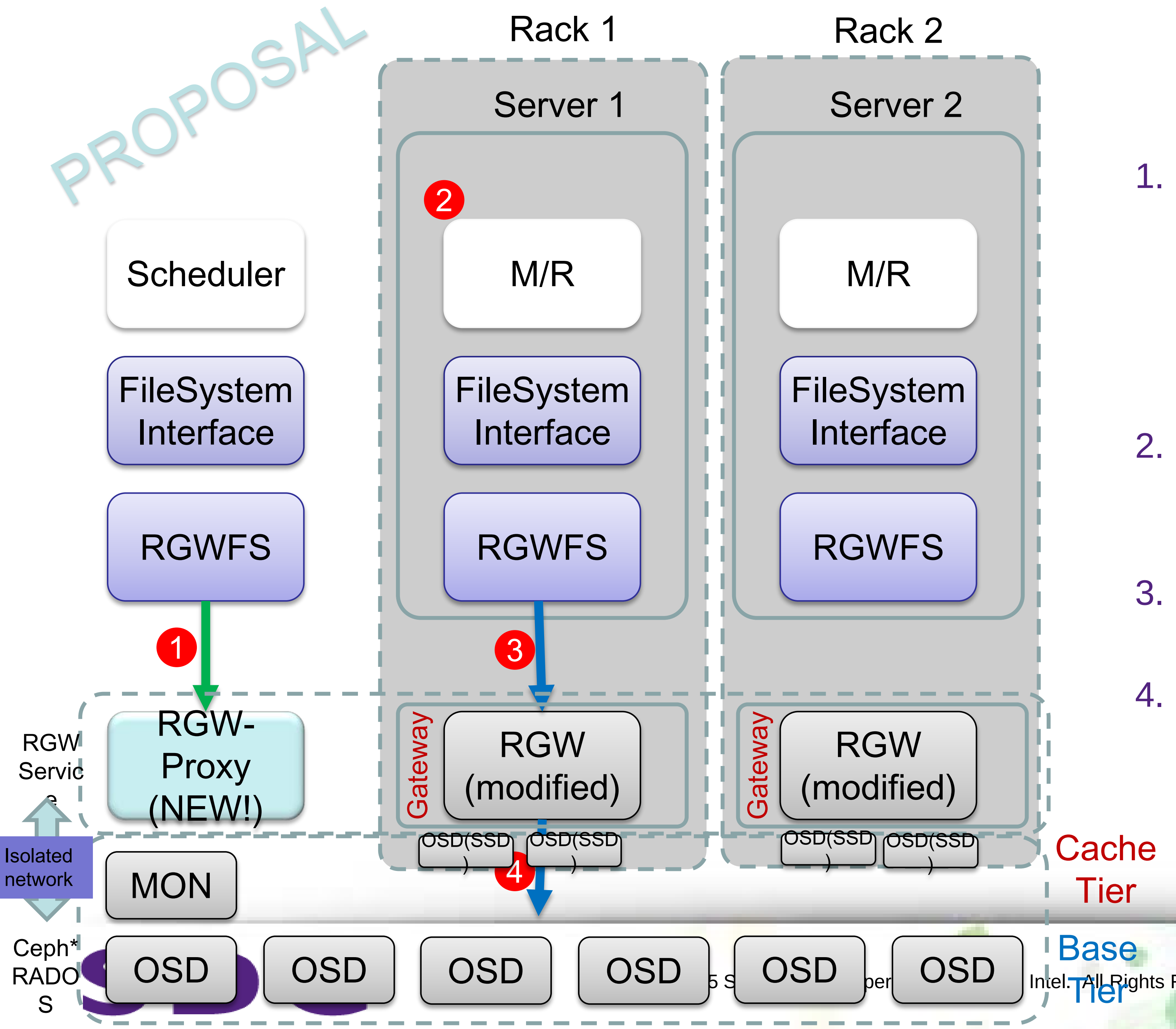
Agenda

- Big Data Analytics over Cloud
- Deployment considerations
- **Design details of BDA over Ceph* Object Storage**
- Sample Performance testing and results

Big data analytics on Ceph Object Storage

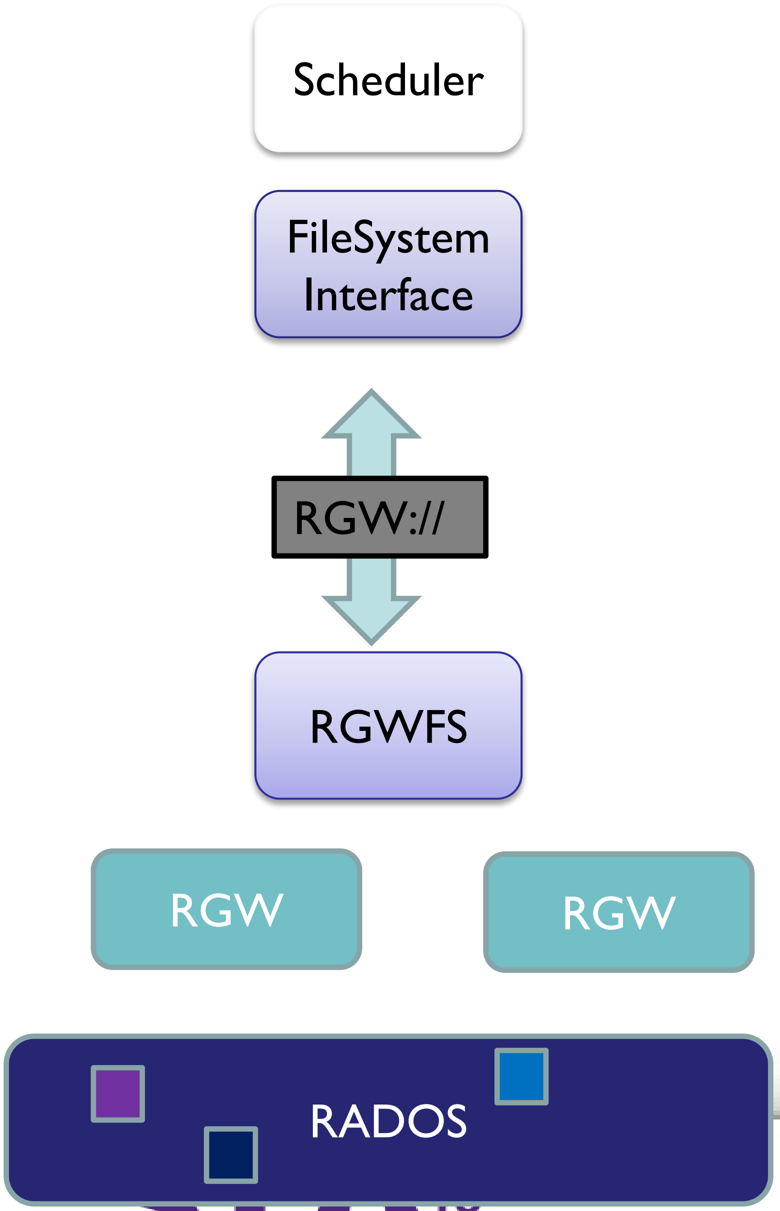
- Ceph provides a unified storage solution, which saves more man power to maintain another different setup for storage.
- We plan to build a reference solution on Hadoop over multiple Ceph* RGW with SSD cache, similar with Hadoop over Swift.
- In this solution there's a requirement that all the storage servers are in a isolated network with the Hadoop cluster. The RGW instances will play as the connectors of these two networks.
- We'll leverage Ceph* Cache Tier technology to cache the data in each RGW servers.
- The reason for not using CephFS:
  - CephFS is not so mature comparing with RGW
  - Existing deployment in DC may require a isolated network between storage and compute nodes

Ceph* RGW with SSD cache



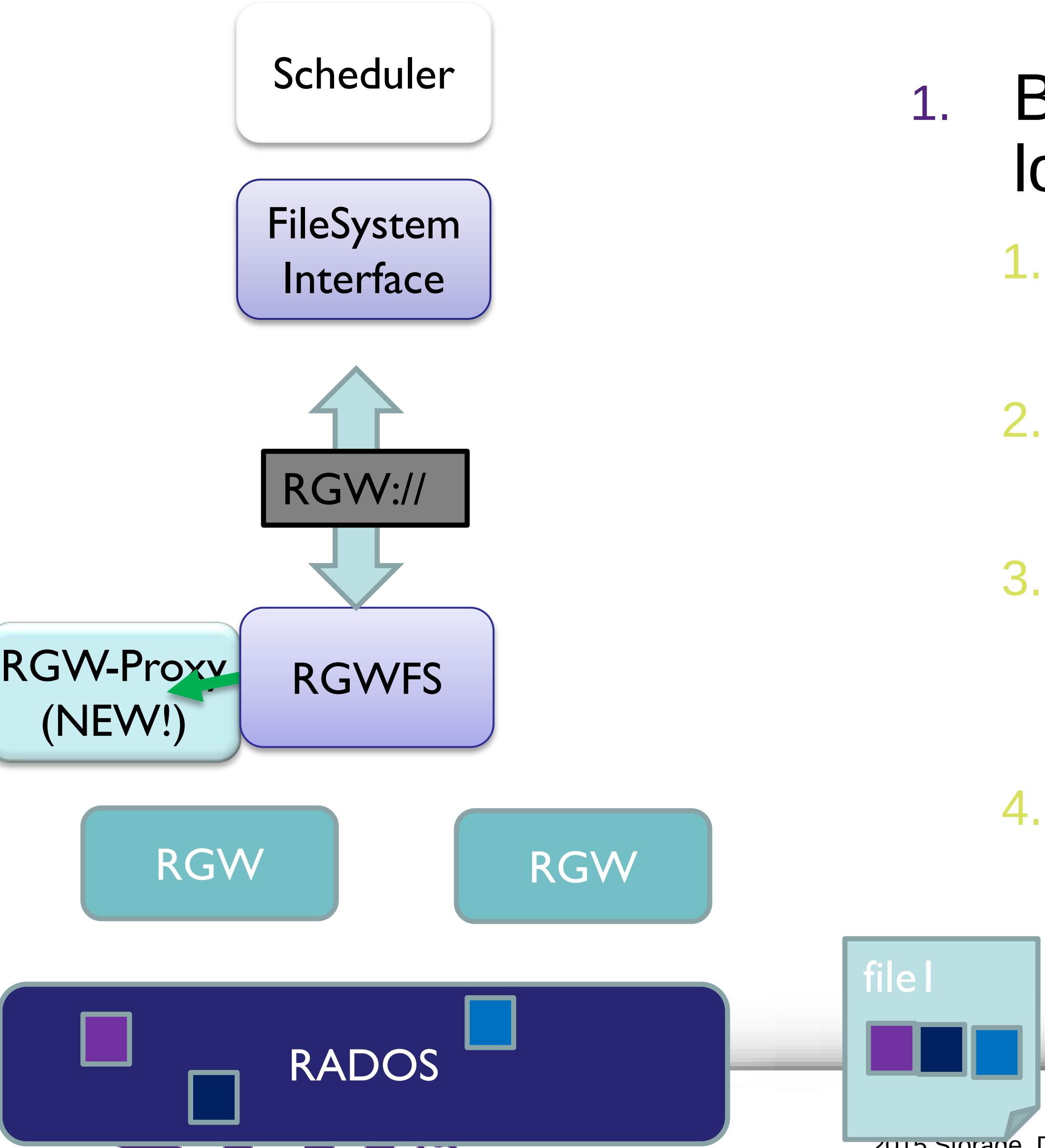
1. Scheduler ask RGW service where a particular block locates (control path)
   - RGW-Proxy returns the closest active RGW instance(s)
2. Scheduler allocates a task on the server that is near to the data
3. Task access data from nearby (data path)
4. RGW get/put data from the CT, and CT would get/put data from BT if necessary (data path)

RGWFS – a new adaptor for HCFS



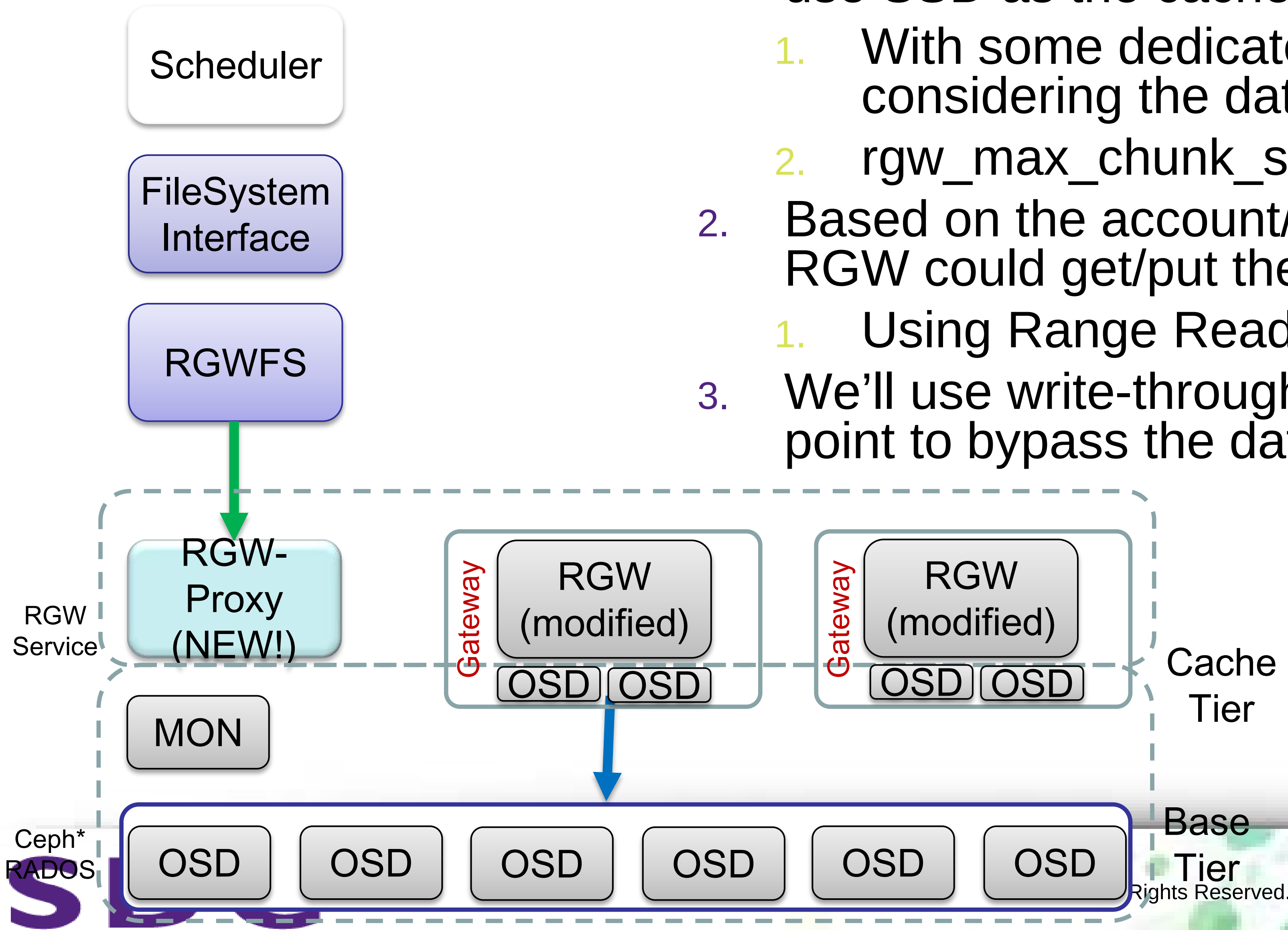
- New filesystem URL rgw://
  1. Forked from Hadoop-8545(SwiftFS)
  2. Hadoop is able to talk to a RGW cluster with this plugin
  3. A new ‘block concept’ was added since Swift doesn’t support blocks
     - Thus scheduler could use multiple tasks to access the same file
     - Based on the location, RGWFS is able to **read** from e closest RGW through range GET API

RGW-Proxy – Give out the closest RGW instance



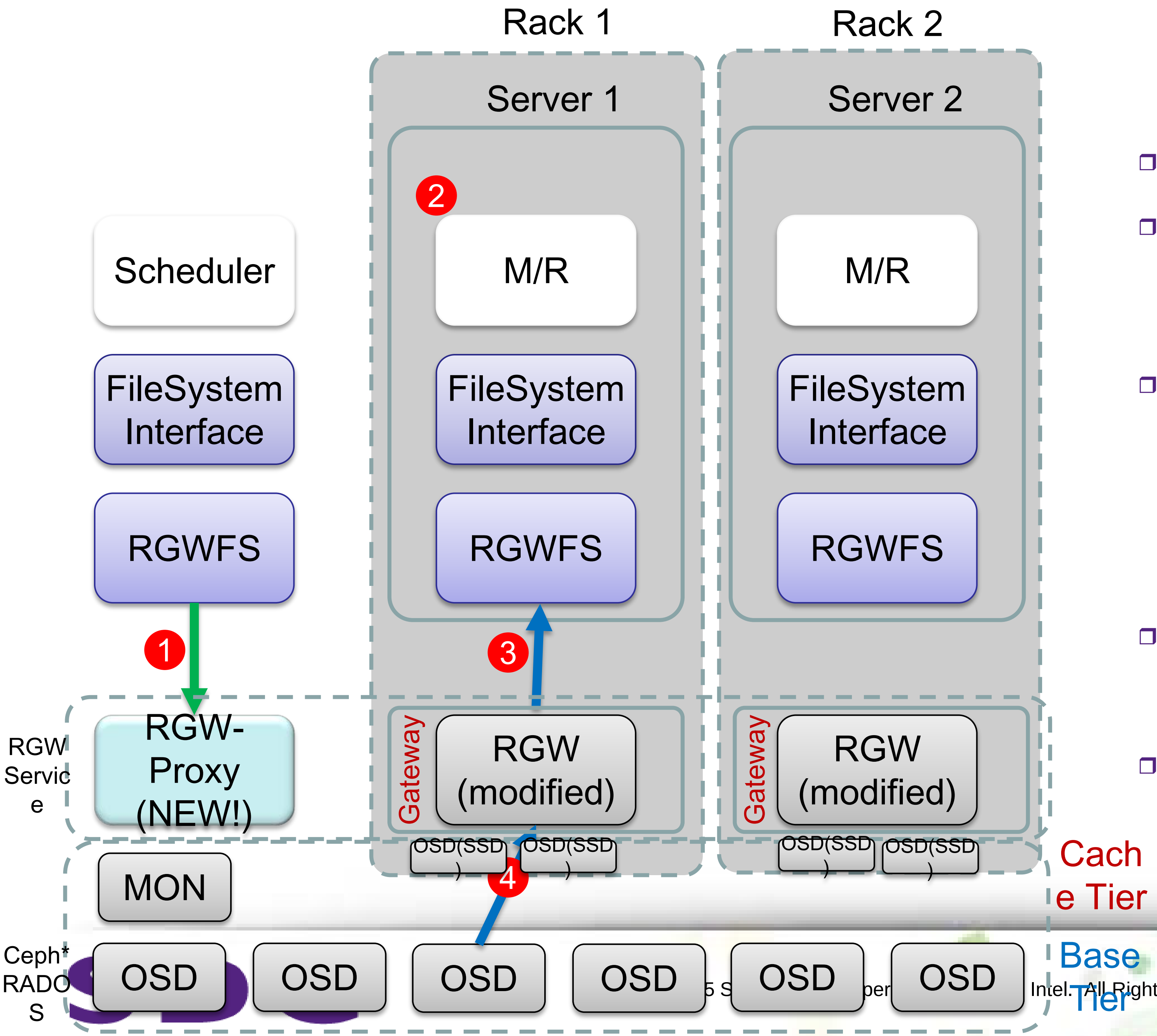
1. Before get/put, RGWFS would try to get the location of each block from RGW-Proxy
    1. One topology file of the cluster is generated
    2. RGW-Proxy would get the manifest from the head object first(librados + getxattr)
    3. Then based on the crushmap RGW-proxy can get the location of each object block(ceph osd map)
    4. RGW-proxy could get the closest RGW the data osd info and the topology file(simple lookup in the topology file)

RGW – Serve the data requests



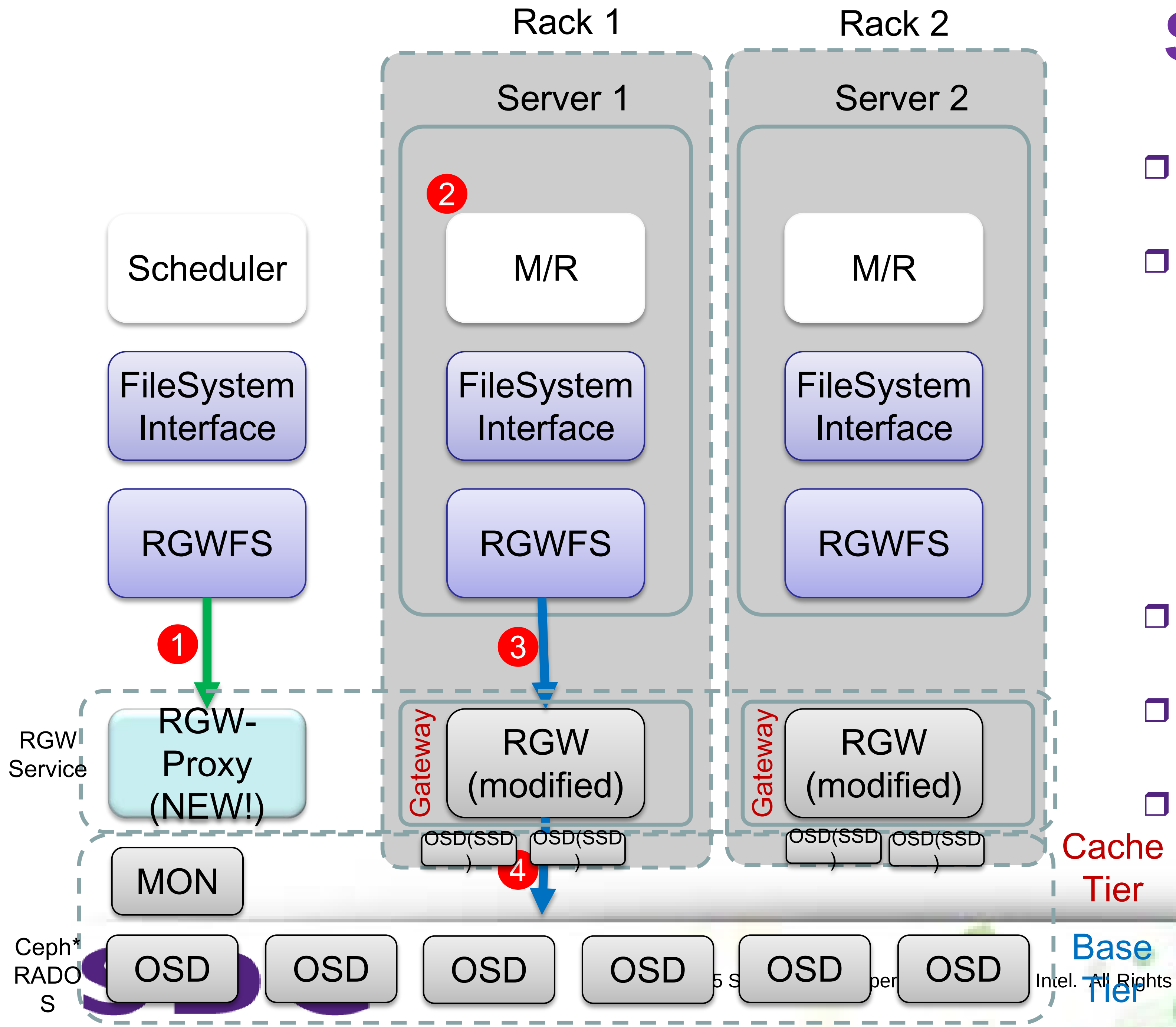
1. Setting up RGW on a Cache Tier thus we could use SSD as the cache.
   1. With some dedicated chunk size: e.g., 64MB considering the data are quite big usually
   2. rgw_max_chunk_size, rgw_obj_strip_size
2. Based on the account/container/object name, RGW could get/put the content.
   1. Using Range Read to get each chunk
3. We'll use write-through mode here as a start point to bypass the data consistency issue.

Sample Get(mapper)



- Scheduler asks RGW-Proxy for the location of a big data file
- RGW-Proxy looks for this file in a specified container (configured by operator) with a HEAD request to RGW
  - RGW then returns the RADOS objects info in the manifest
- RGW-Proxy looks for these objects in RADOS using Crush
  - Get the target OSD in CT for each object. And with the OSD info, we know which RGW instance is the closest.
  - RGW-Proxy would also monitors these RGW instances and do the failover by returning the active RGW instances
- Scheduler allocates a task on the server that is near to the data(RGW instance)
  - RGWFS would issue a range read to get the block
- Ceph* CT would automatically handle the consistency here.

Sample Put(reducer)

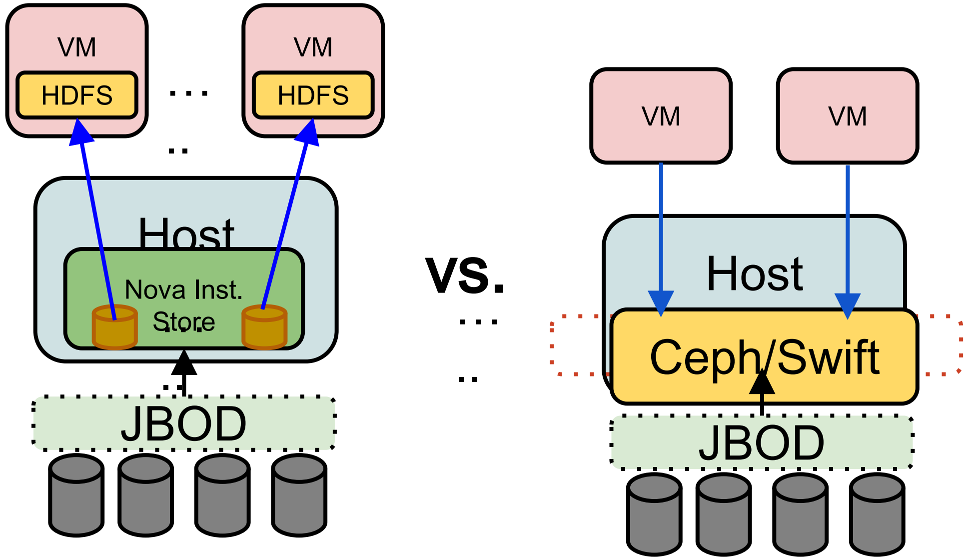


- Scheduler asks RGW-Proxy for the location of a big data file
- RGW-Proxy monitors the RGW instances and returns an active RGW instances list, with the closest RGW instance at the first place and several other RGW instances
  - e.g., [ip1:/a/c/o(same rack), ip2:/a/c/o, ip3:/a/c/o]
- Reducer writes the output to RGW
- RGW would split the object into multiple RADOS objects
- Ceph* CT would automatically handle the consistency here.

Agenda

- Big Data Analytics over Cloud
- Deployment considerations
- Design details of BDA over Ceph* RGW
- **Sample Performance testing and results**

Performance testing results



RGWFS VS SWIFTFS VS HDFS

- 4 Hosts hadoop cluster
- 32 maps 4 reds
- Workload : Sort

Dataset	Phase	RGWFS	SWIFTFS	HDFS
4GB	Run	134s	269s	81s
8GB	Run	248s	369s	142s
16GB	Run	403s	1449s	215s
32GB	Run	820s	Not Test	393s
64GB	Run	1876s	Not Test	737s

- RGWFS is Better than SwiftFS but worse than HDFS
  - Big gap with HDFS
- Profiling data shows the biggest overhead comes from ‘rename’
  - Object stores use ‘copy’ + ‘delete’, which is quite heavy

Rename in Reduce Task

- The output of the reduce function is written to a temporary location in HDFS. After completing, the output will automatically renamed from its temporary location to its final location.

- Object storage cannot support rename, “copy and delete” are used for rename function.
  HDFS Rename -> Change METADATA in Name Node
  Swift Rename -> Copy new object and Delete the older one
  RGW Rename -> Copy new header file and Delete the old header file

Summary

- Big data analytics running on cloud will ease your work
  - However each deployment has Pros and Cons
- Performance penalty is big, optimization needed
  - Comparing with local HDFS setup, performance is 50% worse. The biggest gap is on the ‘rename’ part.
- Intel is optimizing CEPH for Big data world

Next Step

- Finish the development(80% done) and complete the performance testing work
- Open source code repo(TBD)
    - https://github.com/intel-bigdata/MOC
- Optimize the ‘rename’ part

Q&A

Legal Notices and Disclaimers

- Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at intel.com, or from the OEM or retailer.
- No computer system can be absolutely secure.
- Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit http://www.intel.com/performance.
- Cost reduction scenarios described are intended as examples of how a given Intel-based product, in the specified circumstances and configurations, may affect future costs and provide cost savings. Circumstances will vary. Intel does not guarantee any costs or cost reduction.
- This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.
- Statements in this document that refer to Intel's plans and expectations for the quarter, the year, and the future, are forward-looking statements that involve a number of risks and uncertainties. A detailed discussion of the factors that could affect Intel's results and plans is included in Intel's SEC filings, including the annual report on Form 10-K.
- The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.
- No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

- Intel does not control or audit third-party benchmark data or the web sites referenced in this document. You should visit the referenced web site and confirm whether referenced data are accurate.
- Intel, Xeon and the Intel logo are trademarks of Intel Corporation in the United States and other countries.
- *Other names and brands may be claimed as the property of others.
- © 2015 Intel Corporation.

Legal Information: Benchmark and Performance Claims Disclaimers

Software and workloads used in performance tests may have been optimized for performance only on Intel® microprocessors. Performance tests, such as SYSmark* and MobileMark*, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase.

Test and System Configurations: See Back up for details.

For more complete information about performance and benchmark results, visit http://www.intel.com/performance.

Risk Factors

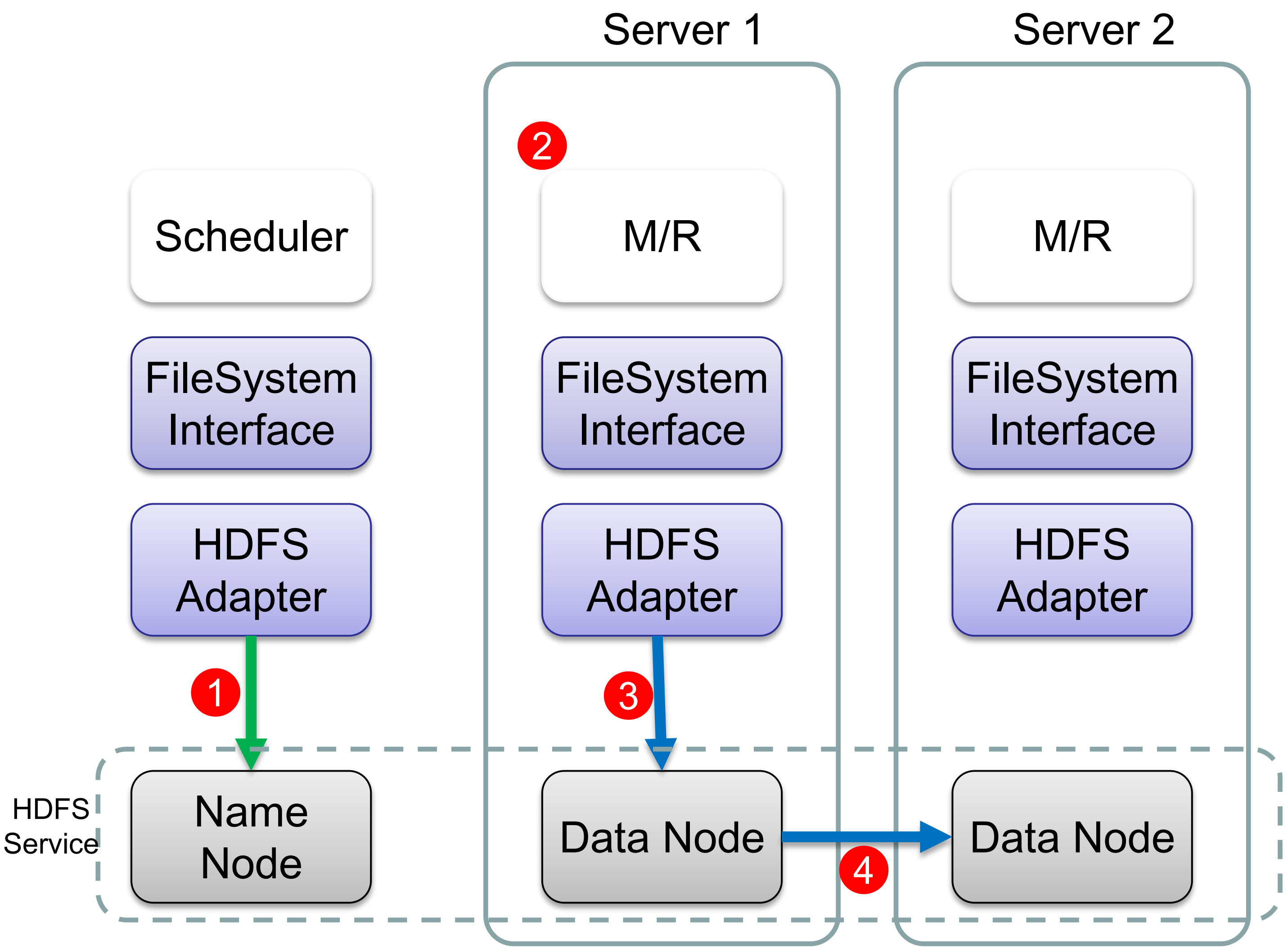
- The above statements and any others in this document that refer to plans and expectations for the first quarter, the year and the future are forward-looking statements that involve a number of risks and uncertainties. Words such as "anticipates," "expects," "intends," "plans," "believes," "seeks," "estimates," "may," "will," "should" and their variations identify forward-looking statements. Statements that refer to or are based on projections, uncertain events or assumptions also identify forward-looking statements. Many factors could affect Intel's actual results, and variances from Intel's current expectations regarding such factors could cause actual results to differ materially from those expressed in these forward-looking statements. Intel presently considers the following to be important factors that could cause actual results to differ materially from the company's expectations. Demand for Intel's products is highly variable and could differ from expectations due to factors including changes in the business and economic conditions; consumer confidence or income levels; customer acceptance of Intel's and competitors' products; competitive and pricing pressures, including actions taken by competitors; supply constraints and other disruptions affecting customers; changes in customer order patterns including order cancellations; and changes in the level of inventory at customers. Intel's gross margin percentage could vary significantly from expectations based on capacity utilization; variations in inventory valuation, including variations related to the timing of qualifying products for sale; changes in revenue levels; segment product mix; the timing and execution of the manufacturing ramp and associated costs; excess or obsolete inventory; changes in unit costs; defects or disruptions in the supply of materials or resources; and product manufacturing quality/yields. Variations in gross margin may also be caused by the timing of Intel product introductions and related expenses, including marketing expenses, and Intel's ability to respond quickly to technological developments and to introduce new features into existing products, which may result in restructuring and asset impairment charges. Intel's results could be affected by adverse economic, social, political and physical/infrastructure conditions in countries where Intel, its customers or its suppliers operate, including military conflict and other security risks, natural disasters, infrastructure disruptions, health concerns and fluctuations in currency exchange rates. Results may also be affected by the formal or informal imposition by countries of new or revised export and/or import and doing-business regulations, which could be changed without prior notice. Intel operates in highly competitive industries and its operations have high costs that are either fixed or difficult to reduce in the short term. The amount, timing and execution of Intel's stock repurchase program and dividend program could be affected by changes in Intel's priorities for the use of cash, such as operational spending, capital spending, acquisitions, and as a result of changes to Intel's cash flows and changes in tax laws. Product defects or errata (deviations from published specifications) may adversely impact our expenses, revenues and reputation. Intel's results could be affected by litigation or regulatory matters involving intellectual property, stockholder, consumer, antitrust, disclosure and other issues. An unfavorable ruling could include monetary damages or an injunction prohibiting Intel from manufacturing or selling one or more products, precluding particular business practices, impacting Intel's ability to design its products, or requiring other remedies such as compulsory licensing of intellectual property. Intel's results may be affected by the timing of closing of acquisitions, divestitures and other significant transactions. A detailed discussion of these and other factors that could affect Intel's results is included in Intel's SEC filings, including the company's most recent reports on Form 10-Q, Form 10-K and earnings release.

Backup

MOC Project

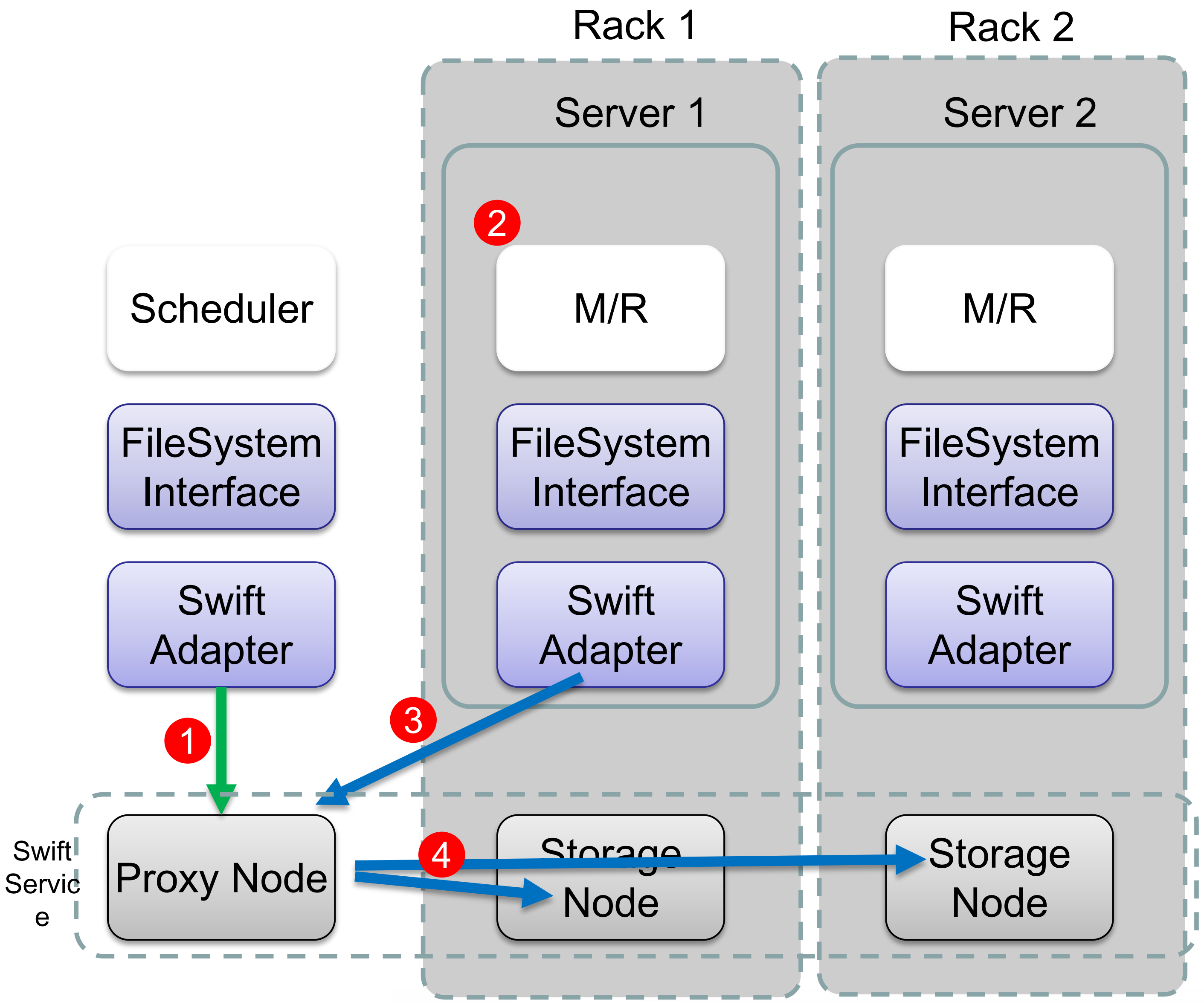
- https://www.openstack.org/assets/presentation-media/MOC-OpenStack.pdf

HDFS



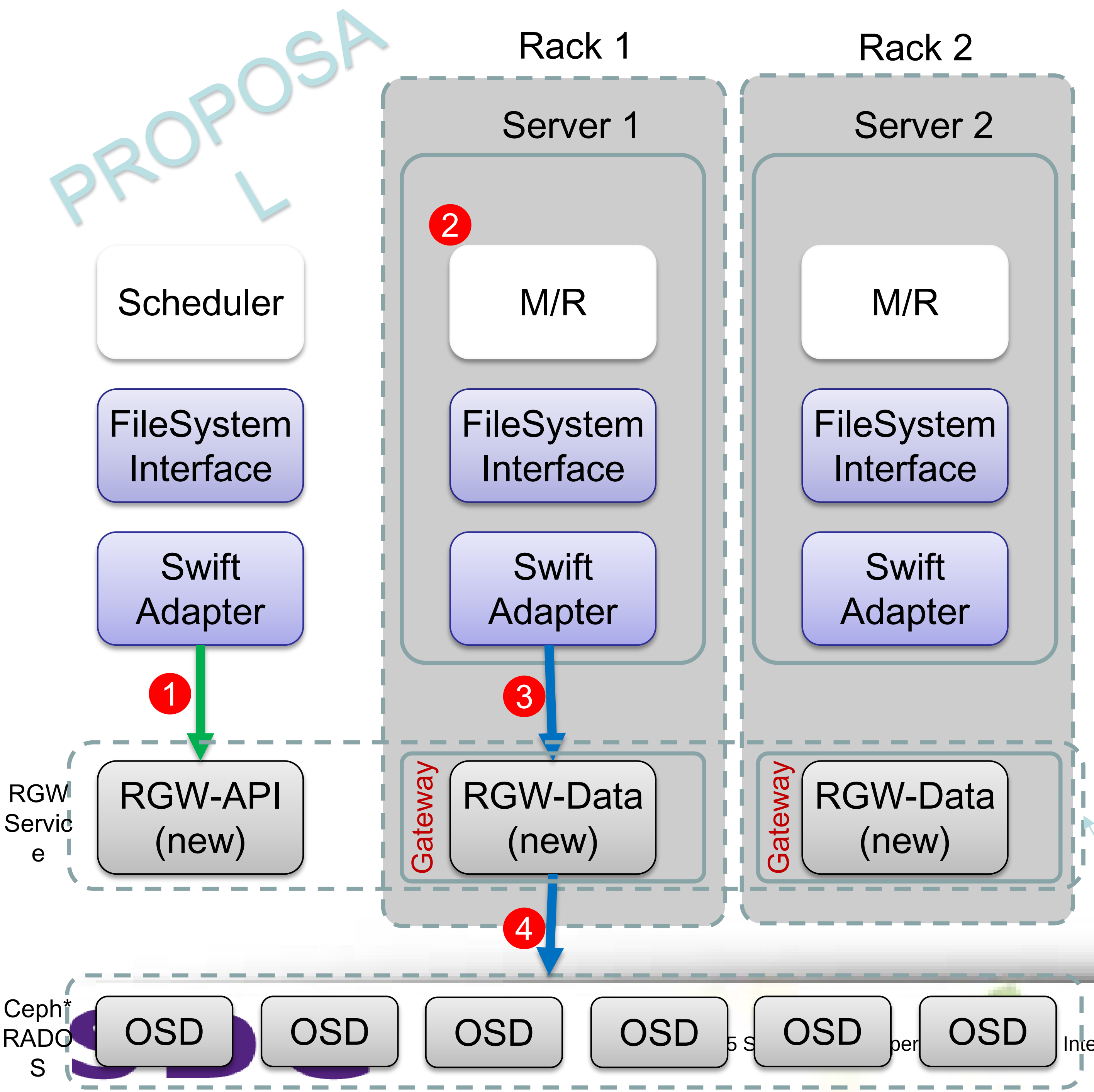
- Scheduler ask HDFS service where a particular block locates (control path)
- Scheduler allocates a task on the server that is near to the data
- Task access data from local (data path)
- In case of write, HDFS replicates the data autonomously (data path)

Swift



1. Scheduler ask Swift service where a particular block locates (control path)
   - Thru regular read API
   - The Swift Proxy service needs to be specially configured so that it will handle the read API in an unusual way
     - return a list of locations instead of data.
2. Scheduler allocates a task on the server that is near to the data
3. Task access data from nearby (data path)
4. In case of write, Proxy will take the responsibility to write all the replicas (data path)

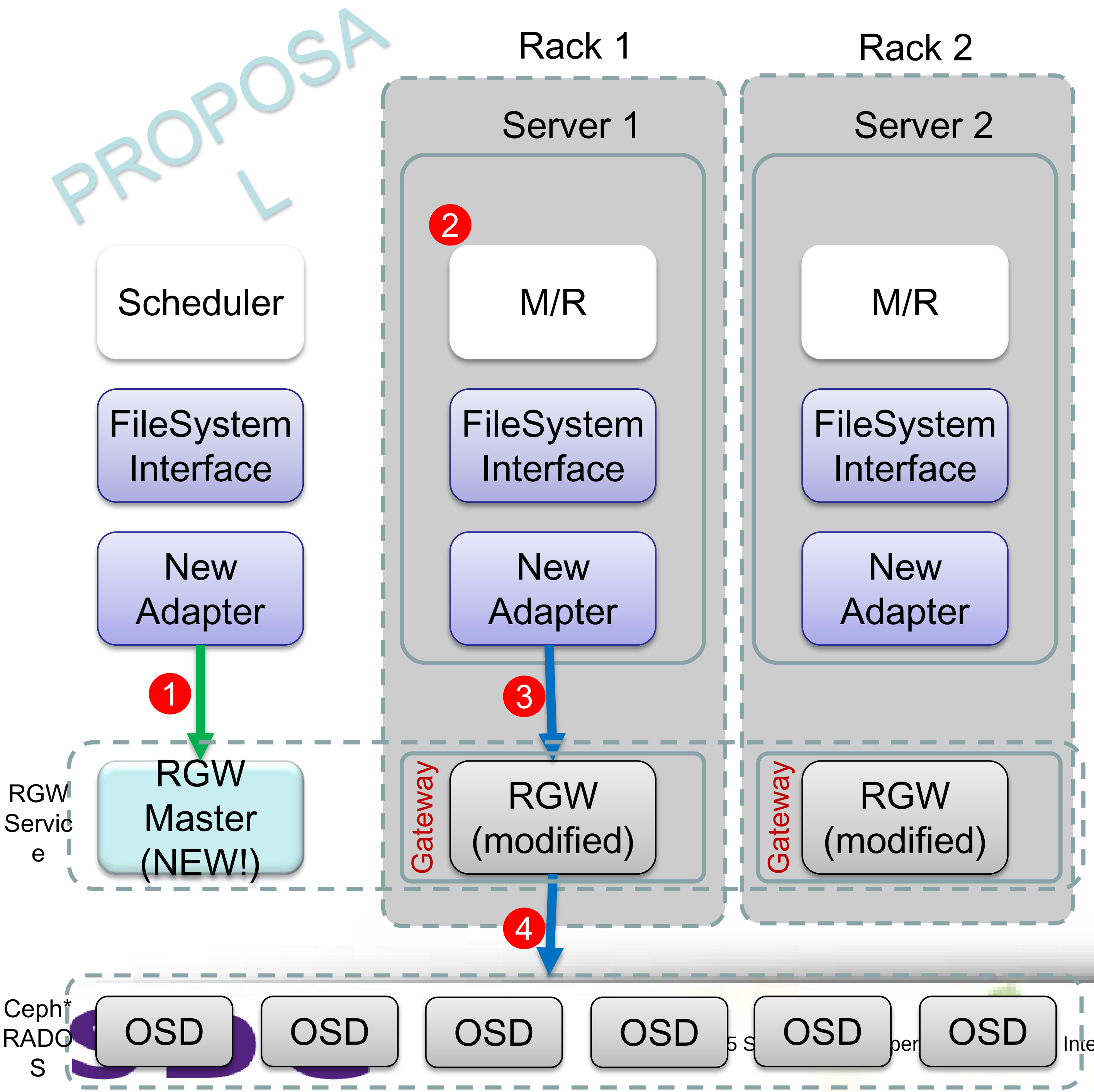
RGW



1. Scheduler ask RGW service where a particular block locates (control path)
   - If the data already existed in one of the gateway, RGW-API returns the location
   - If the data not existed, RGW-API returns a random gateway
2. Scheduler allocates a task on the server that is near to the data
3. Task access data from nearby (data path)
4. RGW-Data relays the request to the backend if necessary (data path)

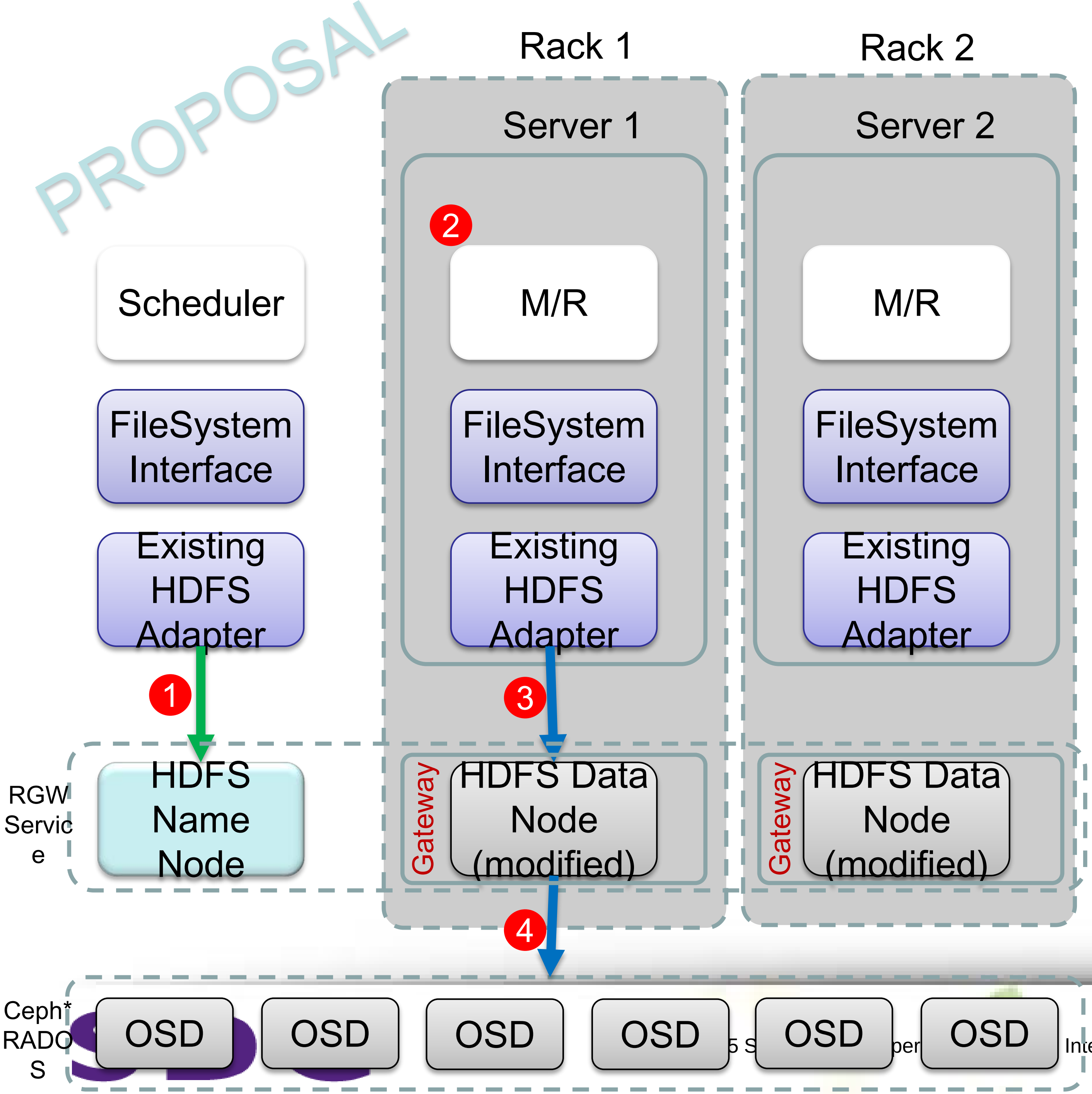
Change to the existing RGW is required!
The rest remains unchanged.

RGW



1. Scheduler ask RGW service where a particular block locates (control path)
   - If the data already existed in one of the gateway, RGW-API returns the location
   - If the data not existed, RGW-API returns a random gateway
2. Scheduler allocates a task on the server that is near to the data
3. Task access data from nearby (data path)
4. RGW-Data relays the request to the backend if necessary (data path)

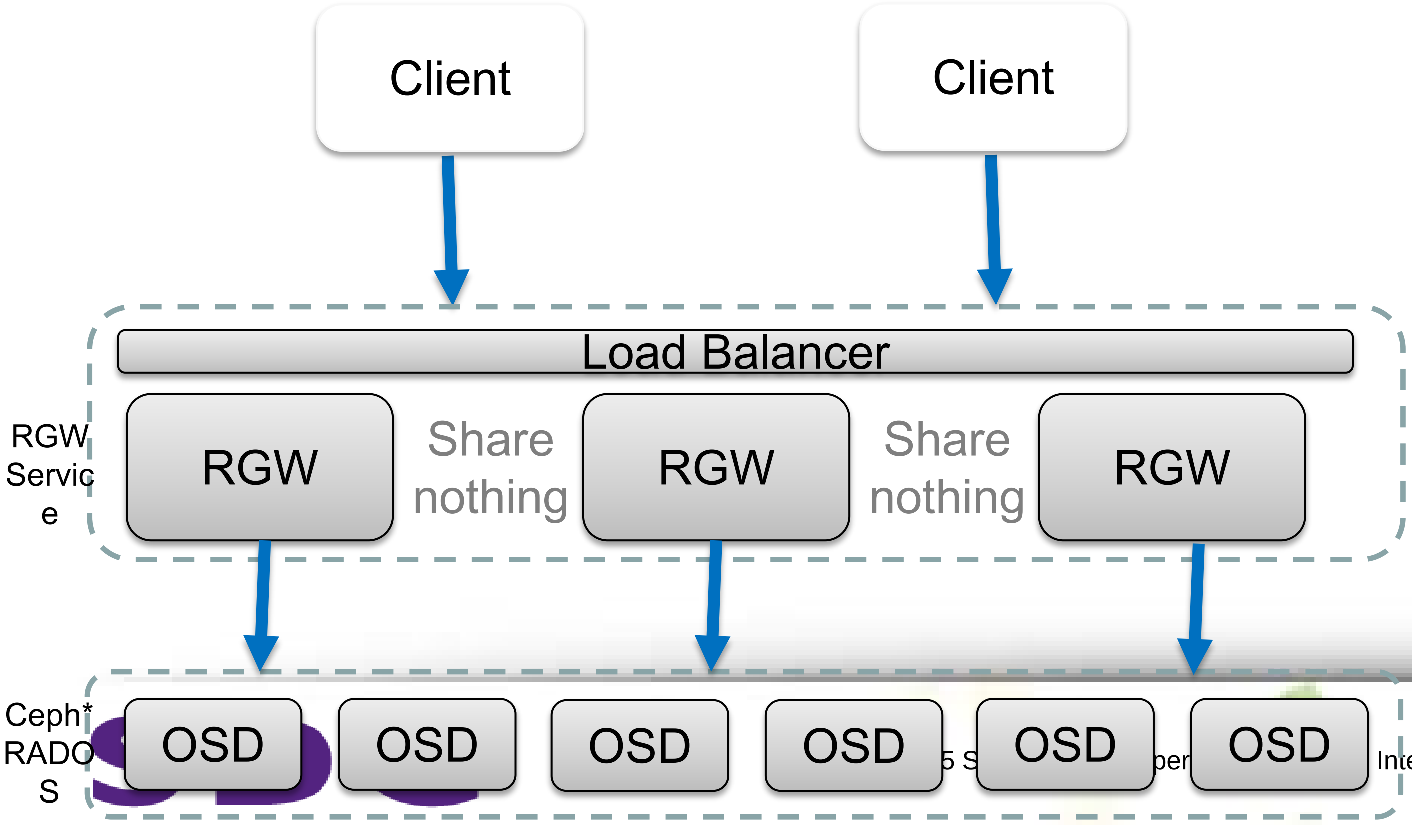
RGW



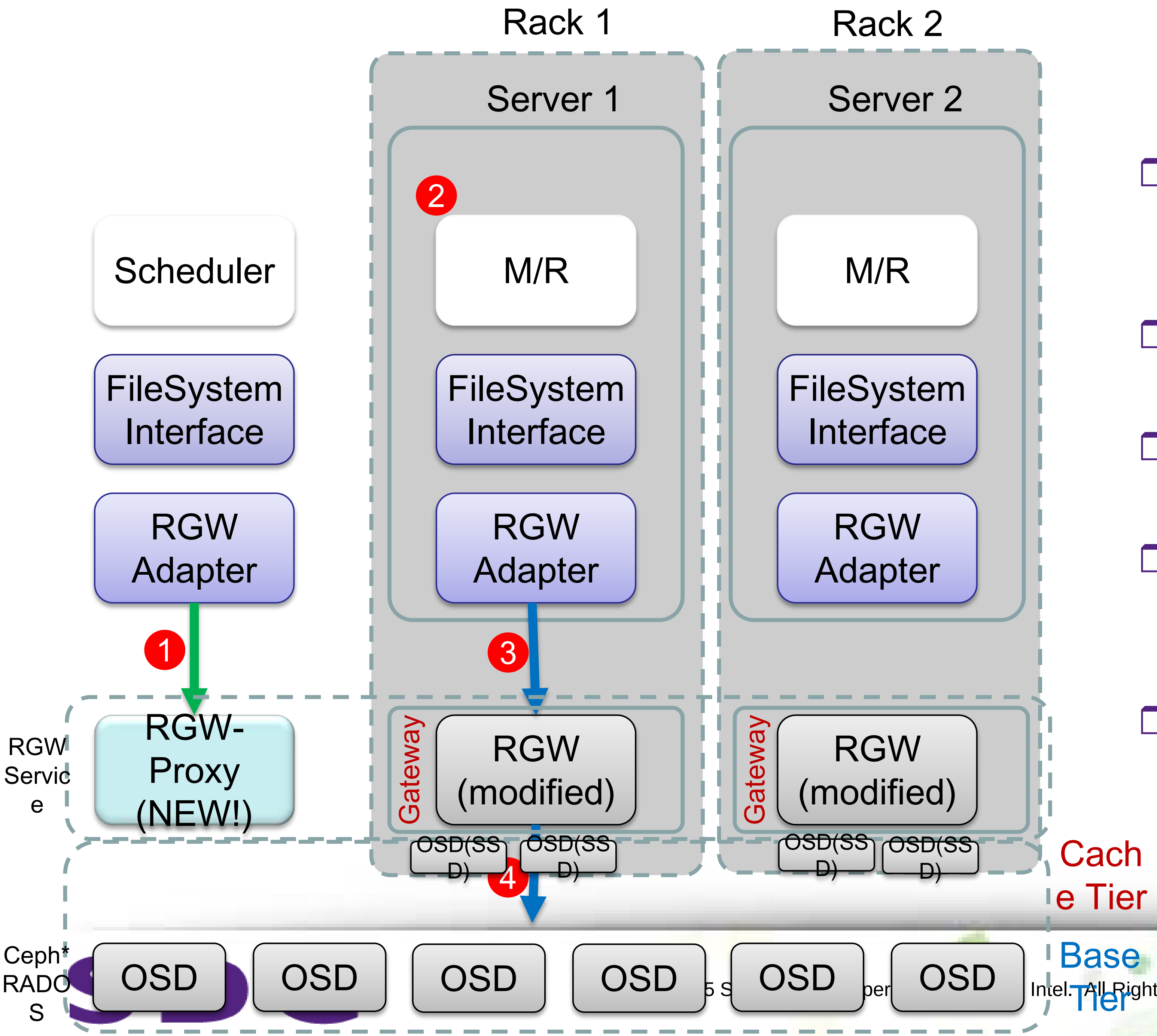
1. Scheduler ask RGW service where a particular block locates (control path)
   - If the data already existed in one of the gateway, RGW-API returns the location
   - If the data not existed, RGW-API returns a random gateway
2. Scheduler allocates a task on the server that is near to the data
3. Task access data from nearby (data path)
4. RGW-Data relays the request to the backend if necessary (data path)

Existing RGW

- Existing RGW implementation works in a different way and can't meet the requirement



Sample Write(reducer) multiple uploads?



- Scheduler asks RGW-Proxy for the location of a big data file
- RGW-Proxy returns the closest RGW instance
- Reducer writes the output to RGW
- RGW would split the object into multiple RADOS objects
- Ceph* CT would automatically handle the consistency here.

Testing Environment

- Host OS: CentOS7
- Guest OS: CentOS7
- Hadoop 2.6.0
- 4-Nodes Cluster
- Baremetal
- OpenStack using KVM
  - qemu-kvm v1.5
- OpenStack using Docker
  - Docker v1.6