Ceph中国社区

CEPHALOCON APAC 2018

THE FUTURE OF STORAGE

22-23 March 2018 | BEIJING

Everything you wanted to know about Radosgw

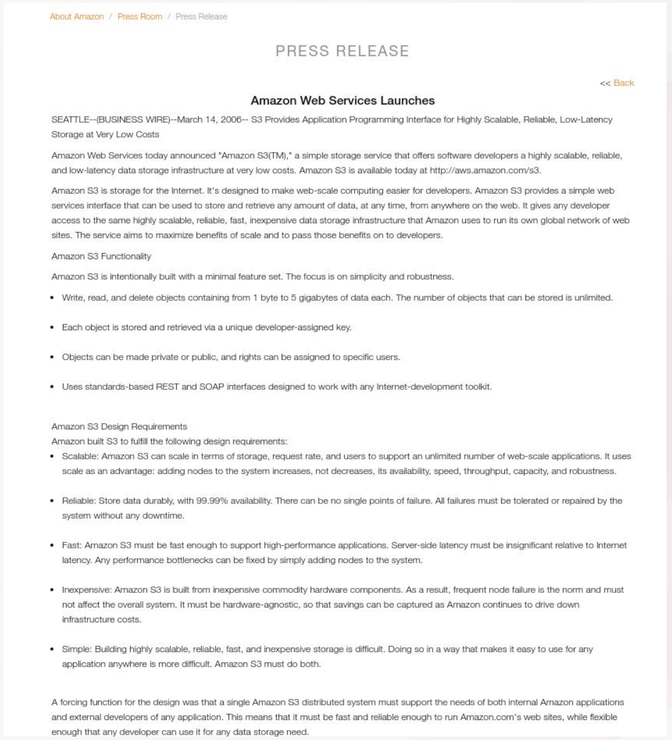
Orit Wasserman Red Hat RGW core developer
Matt Benjamin Red Hat RGW Lead

Object storage

- Flat namespace:
  - Bucket/container
  - Objects
- Metadata:
  - Ownership (Users and tenants)
  - ACL
  - User metadata
- Authentication
- Objects are immutable

Cloud object storage

- Restful API
- Common cloud Protocols:
  - AWS S3
  - Swift (openstack)
  - Google cloud storage
  - Azure blob storage



About Amazon / Press Room / Press Release

PRESS RELEASE

<< Back

Amazon Web Services Launches

SEATTLE--(BUSINESS WIRE)--March 14, 2006-- S3 Provides Application Programming Interface for Highly Scalable, Reliable, Low-Latency Storage at Very Low Costs

Amazon Web Services today announced "Amazon S3(TM)," a simple storage service that offers software developers a highly scalable, reliable, and low-latency data storage infrastructure at very low costs. Amazon S3 is available today at http://aws.amazon.com/s3.

Amazon S3 is storage for the Internet. It's designed to make web-scale computing easier for developers. Amazon S3 provides a simple web services interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the web. It gives any developer access to the same highly scalable, reliable, fast, inexpensive data storage infrastructure that Amazon uses to run its own global network of web sites. The service aims to maximize benefits of scale and to pass those benefits on to developers.

Amazon S3 Functionality

Amazon S3 is intentionally built with a minimal feature set. The focus is on simplicity and robustness.

- Write, read, and delete objects containing from 1 byte to 5 gigabytes of data each. The number of objects that can be stored is unlimited.
- Each object is stored and retrieved via a unique developer-assigned key.
- Objects can be made private or public, and rights can be assigned to specific users.
- Uses standards-based REST and SOAP interfaces designed to work with any Internet-development toolkit.

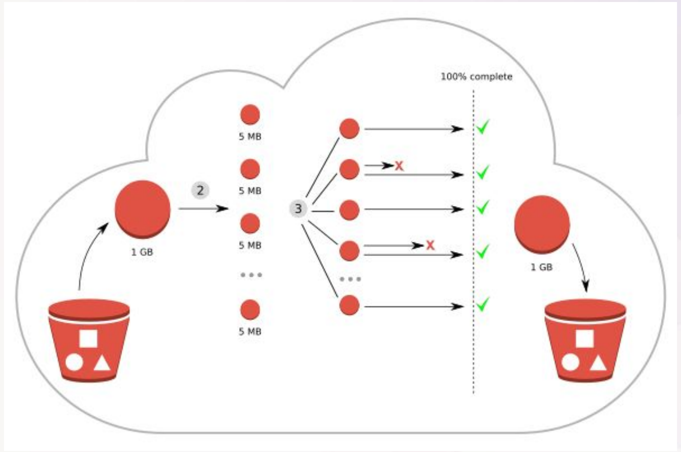
Amazon S3 Design Requirements
Amazon built S3 to fulfill the following design requirements:

- Scalable: Amazon S3 can scale in terms of storage, request rate, and users to support an unlimited number of web-scale applications. It uses scale as an advantage: adding nodes to the system increases, not decreases, its availability, speed, throughput, capacity, and robustness.
- Reliable: Store data durably, with 99.99% availability. There can be no single points of failure. All failures must be tolerated or repaired by the system without any downtime.
- Fast: Amazon S3 must be fast enough to support high-performance applications. Server-side latency must be insignificant relative to Internet latency. Any performance bottlenecks can be fixed by simply adding nodes to the system.
- Inexpensive: Amazon S3 is built from inexpensive commodity hardware components. As a result, frequent node failure is the norm and must not affect the overall system. It must be hardware-agnostic, so that savings can be captured as Amazon continues to drive down infrastructure costs.
- Simple: Building highly scalable, reliable, fast, and inexpensive storage is difficult. Doing so in a way that makes it easy to use for any application anywhere is more difficult. Amazon S3 must do both.

A forcing function for the design was that a single Amazon S3 distributed system must support the needs of both internal Amazon applications and external developers of any application. This means that it must be fast and reliable enough to run Amazon.com's web sites, while flexible enough that any developer can use it for any data storage need.

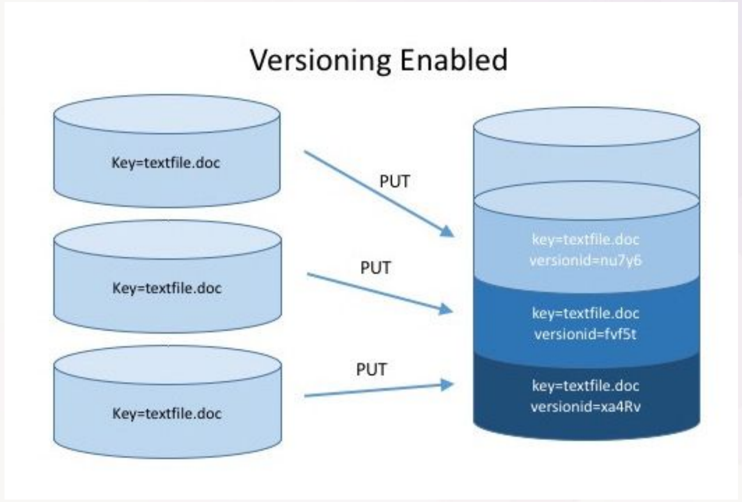
Multipart upload

- upload a single object as a set of parts
- Improved throughput
- Quick recovery from any network issues
- Pause and resume object uploads
- Begin an upload before you know the final object size
- Transactions



Versioning

- Keeps the previous copy of the object in case of overwrite or deletion

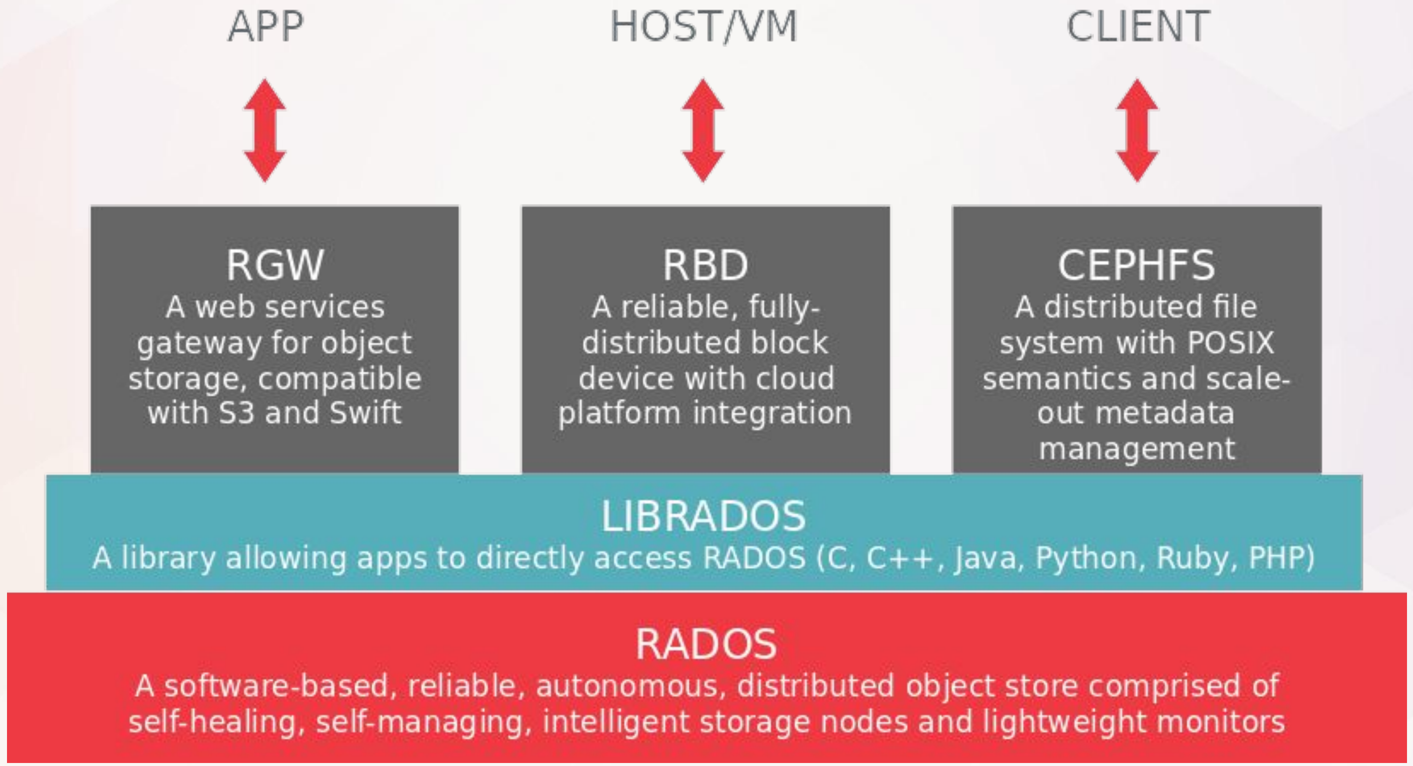


Versioning Enabled
Key=textfile.doc
Key=textfile.doc
Key=textfile.doc
PUT
PUT
PUT
key=textfile.doc versionid=nu7y6
key=textfile.doc versionid=fvf5t
key=textfile.doc versionid=xa4Rv

Life cycle

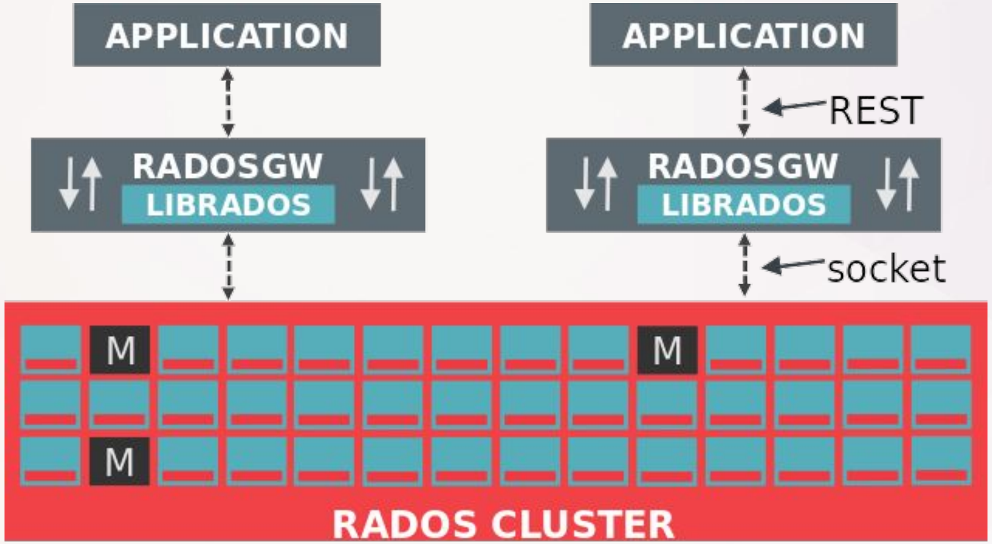
- Configure automatic object transition:
  - Expiration
  - Tiering

Ceph


APP
HOST/VM
CLIENT
RGW
A web services gateway for object storage, compatible with S3 and Swift
RBD
A reliable, fully-distributed block device with cloud platform integration
CEPHFS
A distributed file system with POSIX semantics and scale-out metadata management
LIBRADOS
A library allowing apps to directly access RADOS (C, C++, Java, Python, Ruby, PHP)
RADOS
A software-based, reliable, autonomous, distributed object store comprised of self-healing, self-managing, intelligent storage nodes and lightweight monitors

Radosgw

A web services gateway for object storage, compatible with S3 and Swift



RGW vs RADOS object

- RADOS
  - Limited object sizes (4M)
  - Mutable objects
  - Not indexed
  - per-pool ACLs

- RGW
  - Large objects (TB)
  - Immutable objects
  - Sorted bucket listing
  - per object ACLs

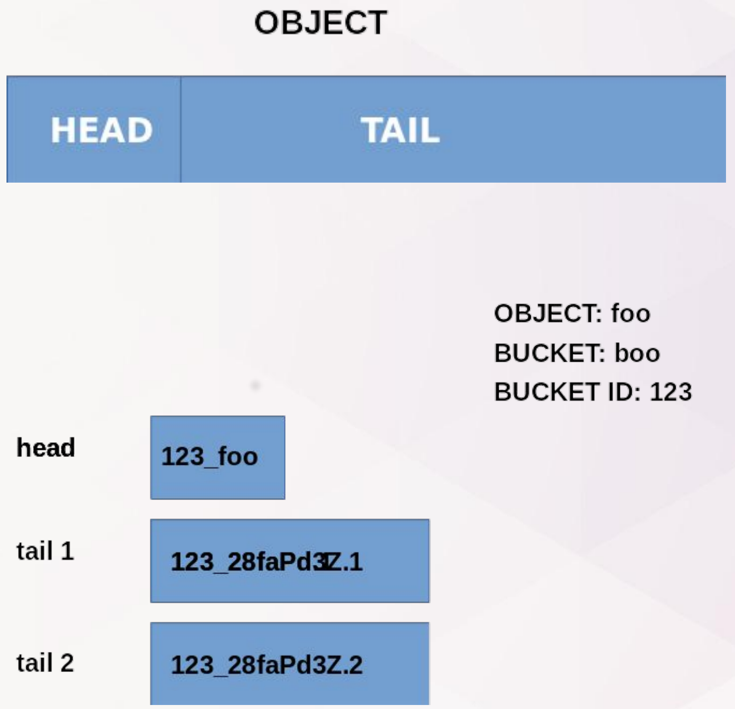
RGW Objects

Head:

- Single rados object
- Object metadata (acls, user attributes, manifest)
- Optional start of data

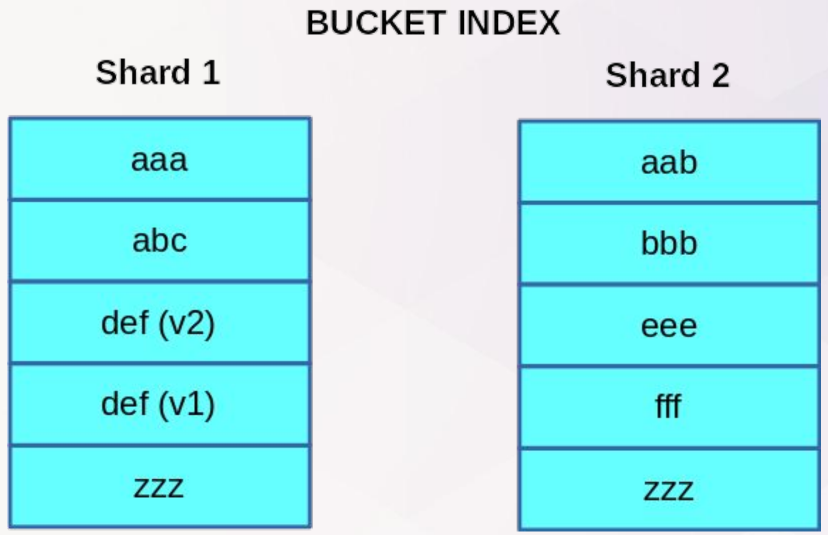
Tail:

- Striped data
- 0 or more rados objects



Bucket index

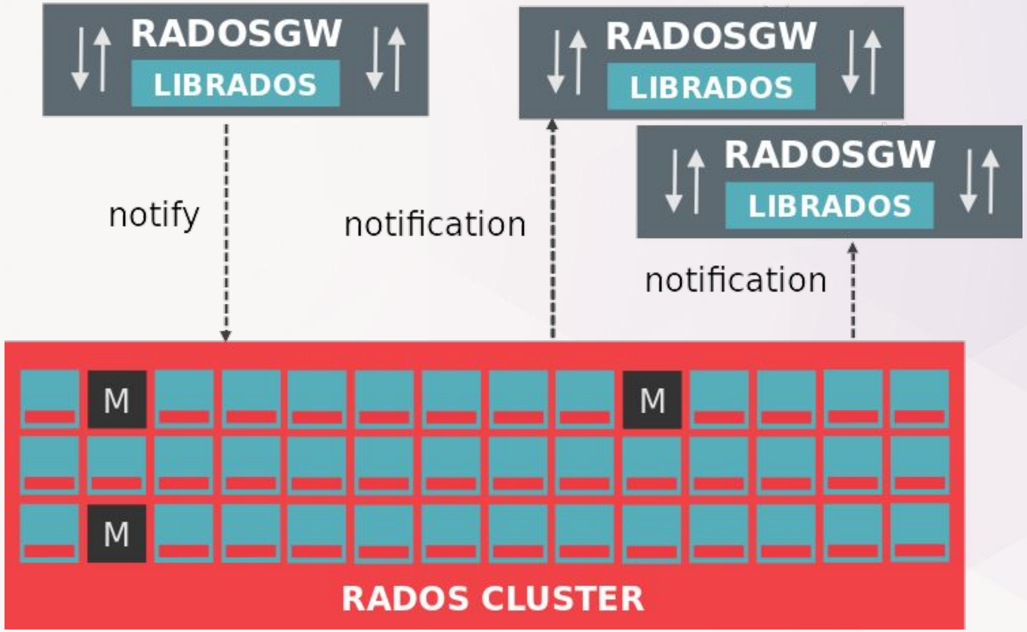
- Bucket index is sharded for scaling
- Offline resharding (Jewel)
- Online resharding (new in Luminous)
- Dynamic resharding (new in Luminous)



Metadata cache

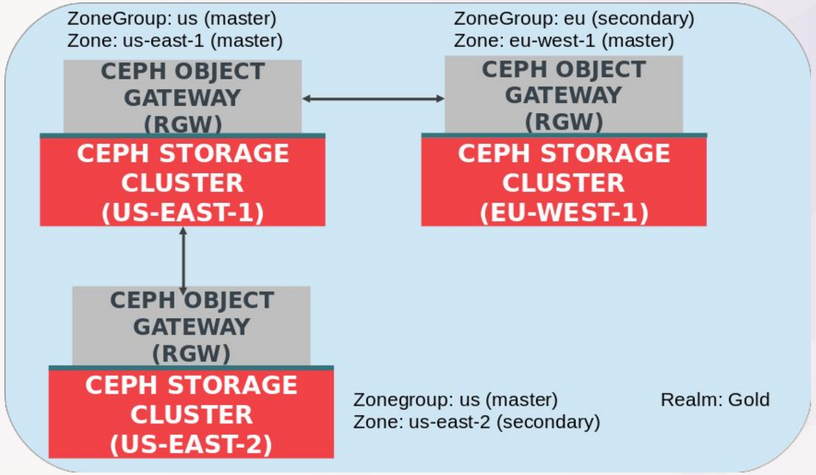
For each request:

- User Info
- Bucket Entry Point
- Buck Instance Info



Geo replication/ Multisite

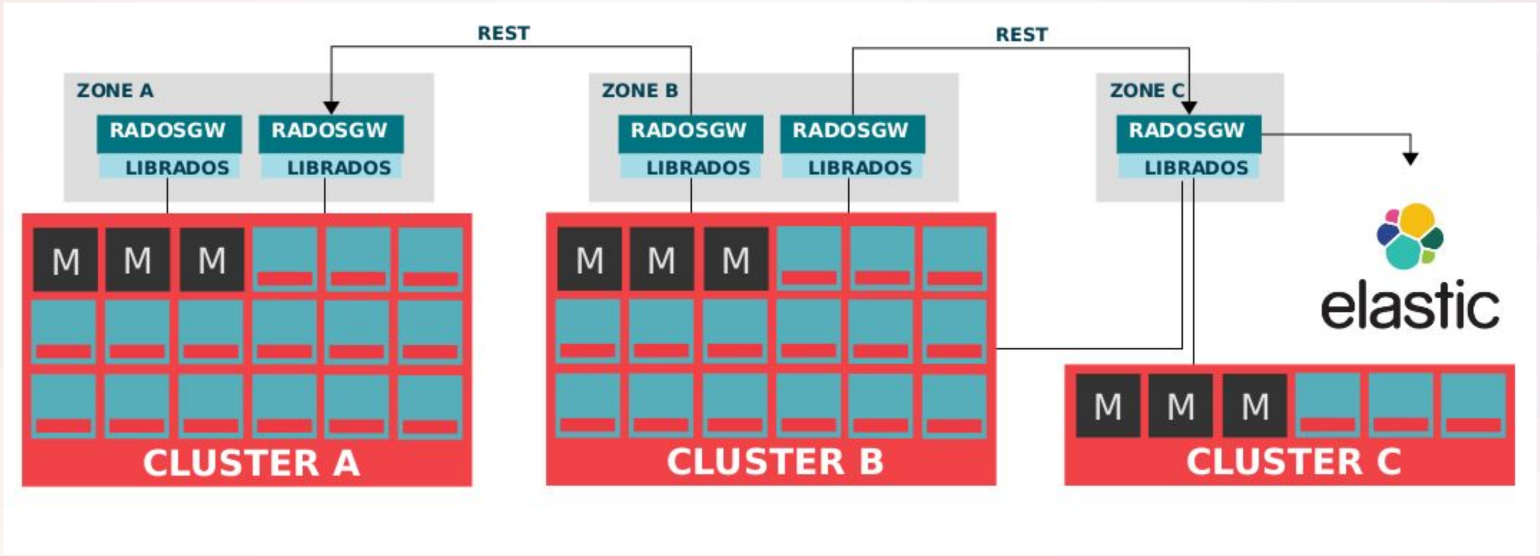
- Global object storage clusters with a single namespace
- Enables deployment of clusters across multiple geographic locations
- Clusters synchronize, allowing users to read from or write to the closest one
- Disaster recovery in case of a zone failure



ZoneGroup: us (master)
Zone: us-east-1 (master)
CEPH OBJECT GATEWAY (RGW)
CEPH STORAGE CLUSTER (US-EAST-1)
ZoneGroup: eu (secondary)
Zone: eu-west-1 (master)
CEPH OBJECT GATEWAY (RGW)
CEPH STORAGE CLUSTER (EU-WEST-1)
CEPH OBJECT GATEWAY (RGW)
CEPH STORAGE CLUSTER (US-EAST-2)
Zonegroup: us (master)
Zone: us-east-2 (secondary)
Realm: Gold

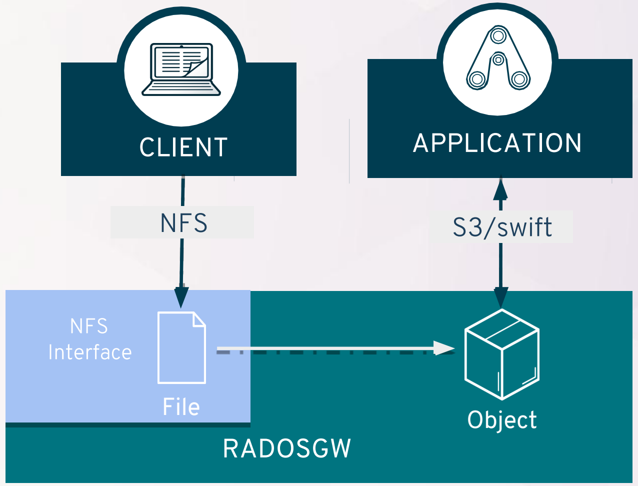
Metadata search

- API to for queries based on object metadata
- Integration with ElasticSearch



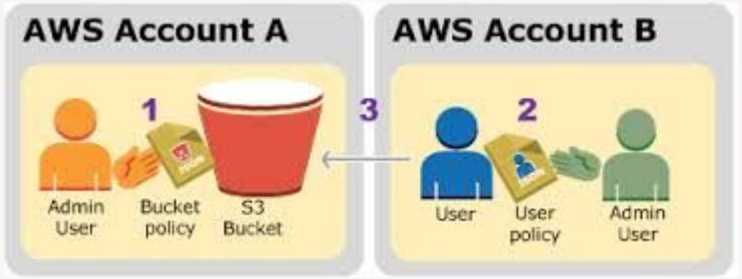
NFS on RGW

- NFS V3/V4 file access on RGW object API
- Hosts can mount and access object namespace using NFS mount
- Real-time translation between NFS and RGW object semantics
- Extended lifetime of existing legacy NFS applications and better ROI



Bucket and Users policy

- Access policies for users and buckets
- Examples:
  - Grant access from multiple accounts
  - Cross account permission
  - Read only for anonymous users
  - Restricting access to a IP specific



More

- Encryption
- Compression
- Object tagging

Mimic and beyond

Cloud sync

- Allows users to archive/backup their data to an external cloud
- Users can map buckets in the source to different buckets in the destination
- Next: bidirectional sync

Beast

- Asynchronous HTTP frontend
- Part of Boost library
- Move to use boost::asio to make radosgw networking I/O asynchronous
- Experimental in Luminous (only frontend)

Bucket index improvements

- Unsorted bucket listing
- Different hashing algorithms
- Caching
- Omap improvements

STS

- Security Token Server
- Amazon “kerberos” https://docs.aws.amazon.com/STS/latest/APIReference/Welcome.html

Ceph中国社区

IT大咖说 知识共享平台

ceph

THANK YOU