One Ring Cannot Rule Them All

Gary Ogasawara
Cloudian Inc.
Motivation
Why Object Storage?

To the application user, the logical “object” matters, Not how it’s physically stored (e.g., pieces, versions, location).
How Object Storage?

To optimize overall system need to be able to have fine-grained and dynamic control inputs.
HyperStore Policy Engine

Store and tier objects intelligently using rules and current information →

*Dynamic storage policies optimized for each object.*

- Cassandra vs. Replicas vs. Erasure Coding
- Consistency Level
- Quality of Service
- Tiering Policy
System Overview
System Overview

1. **S3 Compatible** – Basic and advanced Amazon S3 features implemented.
2. **Optimized** for any workload
3. **Small Start/Elastic:** True P2P Architecture, 10TB to Exabytes
4. **Reliable:** True Multi-DC, Dynamic Consistency, AWS Location Constraint
5. **Turnkey:** Rich, flexible management features (QoS, Reporting, Chargeback, Admin GUI & API)
High-level system view

https://myobjectstorage.com

DNS Server
and/or Load Balancer

Object Storage Cluster
Elastic, distributed and reliable

Logical ring

NOSQL database distributes and replicates data

Data load can be rebalanced when a node is added or removed.

Location of data can be designated, for instance, to multiple datacenters and per rack.

Data is automatically replicated to multiple nodes.
Logical architecture

Cloudian Software Package

Admin Server

Authentication

User Management

Reports

Data Explorer

HTTP/S

S3 Server

Authentication

Account & QoS

Reports

HyperStore® Manager

Data Store (NOSQL)

Data Store (File System)

Data Store (Erasure Coding)

HTTP/S (S3)

Cloudian Mgmt Console

Applications

Web Browser

HTTP/S

© 2014 Cloudian, Inc. All rights reserved.
Multi data center

Replicas can span across data centers

Replicas with location constraint
HyperStore: Storage Type
Policy: Storage Type

- Policies tailored for different object types
- Large object support
- 30% faster reads
- 400% faster writes
- 4x more TPS
- 1.6x better disk utilization
• Optimal storage type selection depends on object size.
Better performance with less variance

**Single Storage Type**

**Multiple Storage Types**

- iostat % utilization
- io read/write (MB)
Replication

One object is replicated to multiple servers

Even in the case of a server failure, client can get the object

Put “object”

Get “object”
Erasure coding

- Divide one object into
  - k data fragments
  - m coding fragments

Spread object across k+m servers

Configurable
- Storage type per-bucket and per-object: {EC, Replicas, Cassandra}
- Number of data & coding fragments: k and m.
- Consistency level: {QUORUM(k), ALL}
- Fragment placement on unique nodes: {T, F}

http://soacloud.ulitzer.com/node/2702747
Large object support

• **Chunking**
  – Break single objects into smaller chunks when storing
  – Distribute chunks across the cluster for better performance

• **Multi-part**
  – Parts uploaded independently and in any order
  – Single parts can be re-transmitted
  – After all parts are uploaded, then presented as a single object
  – All parts are stored in file system, regardless of threshold configuration
HyperStore: Consistency Level
Consistency Level Example

CL=ALL

After “all” objects are replicated, Client receives acknowledgement

- Data is always consistent

CL=QUORUM

After “some” objects are replicated, Client receives acknowledgement

- Data is consistent by QUORUM
# Consistency levels

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ONE</td>
<td>A write has been written to at least 1 replica's commit log and memory table before responding to the client.</td>
</tr>
<tr>
<td>QUORUM</td>
<td>Ensure that the write has been written to ( N / 2 + 1 ) replicas before responding to the client.</td>
</tr>
<tr>
<td>LOCAL_QUORUM</td>
<td>Similar to QUORUM but replicas are in same data center as coordinator node.</td>
</tr>
<tr>
<td>EACH_QUORUM</td>
<td>Ensure that the write has been written to a quorum of replicas in each datacenter in the cluster before responding to the client</td>
</tr>
<tr>
<td>ALL</td>
<td>All replicas must have received the write, otherwise the operation will fail.</td>
</tr>
</tbody>
</table>

## WRITE

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ONE</td>
<td>Returns the response from the first replica causing a consistency check in a background thread.</td>
</tr>
<tr>
<td>QUORUM</td>
<td>Returns the record with the most recent timestamp once ( (N/2 + 1) ) replicas has responded.</td>
</tr>
<tr>
<td>LOCAL_QUORUM</td>
<td>Returns the record with the most recent timestamp once ( (N/2 + 1) ) replicas in the same data center as the coordinator node has reported.</td>
</tr>
<tr>
<td>ALL</td>
<td>Returns the record with the most recent timestamp once all replicas have responded. The read operation will fail if a replica does not respond.</td>
</tr>
</tbody>
</table>

## READ
Configurable consistency across DCs

**Multi-DataCenter, Single Region**
- 2 DCs, 1 Region, 5 replicas
- # of replicas per DC
- Consistency level: {QUORUM, LOCAL_QUORUM, EACH_QUORUM, ONE, ALL}

**Multi-DataCenter, Multi Region**
- 2 DCs, 2 Regions, 3 Replicas
- Location constraint at bucket level.
- Objects not shared across regions.
- Span buckets across regions (virtual buckets)
- QoS & Billing based on region
- In each region, standard multi-DC configuration.
Problem: CL=ALL for Multi-DC

- “All” or “QUORUM” cannot be completed in the case of failure
- “All” or “QUORUM” is slower for clients/applications

Assumptions:
- 3 Replications
- 2 in DC1
- 1 in DC2

Data Center 1

Data Center 2

Client

No “ack”
Problem: CL=LOCAL_QUORUM for Multi-DC

“Local _Quorum” cannot provide “data consistency” in the whole system

Assumptions:
- 3 Replications
- 2 in DC1
- 1 in DC2

Data is consistent only when object is read from DC1
Automatic dynamic consistency level

**Basic CLs**
- Synchronous Replication
  - + No Data Loss
  - - Failures if DR site not reachable
- Asynchronous Replication
  - + Not dependent on availability of DR site
  - - Data Loss in case of Disaster

**Advanced CLs**
- Synchronous Replication
- Each Quorum
- Local Quorum
- Quorum
- Asynchronous Replication

**Advanced CLs with Automatic Change**
- Synchronous Replication #1
- Each Quorum
- Local Quorum #2
- Quorum
- Asynchronous Replication #3
HyperStore: QoS
Quality of Service (QoS)

Operator:
• Prevent a user or group from impacting other users/groups.
• Guarantee a specific level of performance to a user/group.

User/Group:
• Don’t exceed a preset limit.
Quality of Service (QoS) Management

- Configurable maximum limits per-region at per-user, per-group, system level.
  - Requests/minute
  - Storage bytes
  - Storage objects
  - Data Bytes Inbound
  - Data Bytes Outbound
- While limit is reached, requests are rejected.

Storage Byte Limiter

Inbound/Outbound Requests Limiter

Inbound/Outbound Data Byte Limiter
HyperStore: Tiering
Tiering: Hybrid private/public

• Some data on-premise, some off-premise
• S3 bucket lifecycle policies (e.g., age) to migrate data to Amazon S3 and Glacier (or any S3 system)

• Read options:
  – Streaming
  – HTTP redirect
  – Restore

• Consolidated reports and bill
Cloudian products

HyperStore Software

- The software only version
- Runs on commodity hardware
- Runs on commodity software: Linux, POSIX filesystem

HyperStore Appliance

- Sold as an appliance by Cloudian or by a Cloudian Partner
- End user gets complete hardware/software solution
- No software installation needed
- 3 Models
  - HSA1024: 1U, 32GB RAM, 4xGigE NIC, 24 TB,
  - HSA1048: 1U, 32GB RAM, 4xGigE NIC, 48 TB
  - HSA2048: 2U, 64GB RAM, 2x1 GigE + 2x10GigE NIC, 48 TB, Flash Optimized
Closing

Optimize overall system
by providing fine-grained controls,
both manually and automatically changed.

More Info, free trial, demo, PoC:
● www.cloudian.com
● @CloudianStorage, @go10
● www.facebook.com/cloudian.cloudstorage
Backup
About Cloudian

- Object storage startup in Silicon Valley
- Production hardened product
- Target market: mid- to large-enterprises & regional service providers

CLOUDIAN PARTNERS
Objects as a higher layer of abstraction

Abstraction Level

<table>
<thead>
<tr>
<th>Application Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS User Level</td>
</tr>
<tr>
<td>OS Kernel Level</td>
</tr>
<tr>
<td>Abstraction Level</td>
</tr>
</tbody>
</table>

OBJECTS

FILES

BLOCKS

HTTP (S3)

NAS (NFS, CIFS)

SAN (iSCSI)
Block & File vs. Object
Emergence of a two-tier enterprise storage architecture

**Faster**
- For ‘hot’ data
- Flash-optimized
- IOPS-centric
- VM/VDI optimized
- Variety of approaches

**Bigger**
- For cool/cold data
- Object-based
- Scale-out (multi-PB)
- Software-centric
- Cloud-compatible
Object Storage Use Cases

- Enterprise backup
- Long term archiving
- Sync and share
- Remote office file storage
- Cloud storage (PaaS)
S3 advanced features:

- Multi-part uploads: allows uploading large objects in multiple parts
- Versioning: multiple versions of same object
- Bucket Lifecycle: auto-expiration using rules
- Server side encryption: enhance confidentiality
- Location constraint: Assign data to specific region (eg for HIPAA Compliance)
- Bucket Website: Create buckets as websites to host web content
- Access control lists (ACLs) define access rights to bucket and object
- And more...