Scaling and High Performance Storage System

Yosuke Hara - @yosukehara
A Researcher of R.I.T. and Tech Lead LeoFS
with Hiroki Matsue, LeoFS Support and Rakuten Software Engineer
LeoFS is an Unstructured Object Storage for the Web and a highly available, distributed, eventually consistent storage system.

LeoFS was published as OSS on July of 2012

leo-project.net/leofs
Overview

Brief Benchmark Report

Multi Data Center Replication

LeoFS Administration at Rakuten

Future Plans

“NFS” Support and more
Overview
LeoFS
The Lion of Storage Systems

HIGH Availability
LeoFS Non Stop

3 Vs in 3 HIGHLs

Velocity: Low Latency
Minimum Resources

Volume: Petabyte / Exabyte
Variety: Photo, Movie, Unstructured-data

HIGH Cost
Performance Ratio

HIGH Scalability
LeoFS Overview

Request from Web Applications / Browsers w/HTTP over REST-API / S3-API

Load Balancer

Gateway

Manager

Storage

( Erlang RPC)

( TCP/IP, SNMP )

Monitor

GUI Console

Keeping High Availability
Keeping High Performance
Easy Administration
Gateway
LeoFS Overview - Gateway

HTTP Request and Response

Built in *Object Cache Mechanism*

REST-API / S3-API

Stateless Proxy + Object Cache

[Memory Cache, Disc Cache]

Use Consistent Hashing for decision of a primary node

Storage Cluster

Clients

Gateway(s)

Storage Cluster

Fast HTTP Server - Cowboy
API Handler
Object Cache Mechanism

Rakuten
Storage
LeoFS Overview - Storage

WRITE: Auto Replication
READ: Auto Repair of an Inconsistent Object with Async

Use "Consistent Hashing" for Data Operation in the Storage Cluster

Choosing Replica Target Node(s)
- KEY = "bucket/leofs.key"
- Hash = md5(Filename)
- RING 2^128 (MD5)
- # of replicas = 3

"P2P"
LeoFS Overview - Storage

Storage consists of **Object Storage** and **Metadata Storage**
Includes **Replicator** and **Recoverer** for the eventual consistency
LeoFS Overview - Storage - Data Structure

<Metadata>
- Key
- KeySize
- Custom Meta Size
- File Size
- Offset
- Version
- Time-stamp
- Checksum

<Needle>
- Checksum
- KeySize
- User-Meta Size
- DataSize
- Offset
- Version
- Time-stamp
- Key
- User-Meta
- Actual File
- Footer

<Object Container>
- Super-block
- Needle-1
- Needle-2
- Needle-3
- Needle-4
- Needle-5

Robust and High Performance Necessary for GC
LeoFS Overview - Storage - Large Object Support

To Equalize Disk Usage in Every Storage Node
To Realize High I/O efficiency and High Availability

[ WRITE Operation ]

An Original Object’s Metadata

Client(s) | Gateway | Storage Cluster
---|---|---

Chunked Objects

Every chunked object and metadata are replicated in the cluster
Manager
LeoFS Overview - Manager

Operate LeoFS - Gateway and Storage Cluster
"RING Monitor" and "NodeState Monitor"

Manager(s)

Gateway(s)

Storage Cluster

Monitor
RING, Node State

Operate
status, suspend, resume, detach, whereis, ...
Brief Benchmark Report
Brief Benchmark Report

Summary of the benchmark results

LeoFS kept in a stable performance through the benchmark

Bottleneck is Disk I/O

The cache mechanism contributed to reduce network traffic between Gateway and Storage
Brief Benchmark Report

1st Case:

Group of Value Ranges (HDD)

Storage: 5, Gateway: 1, Manager: 2

R:W = 9:1

source: https://github.com/leo-project/notes/tree/master/leofs/benchmark/leofs/20140605/tests/1m_r9w1_240min

2nd Case:

Group of Value Ranges (HDD)

Storage: 5, Gateway: 1, Manager: 2

R:W = 8:2

source: https://github.com/leo-project/notes/tree/master/leofs/benchmark/leofs/20140605/tests/1m_r8w2_120min
### Brief Benchmark Report

#### Server Spec - Gateway:

<table>
<thead>
<tr>
<th>Component</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Intel(R) Xeon(R) CPU X5650 @ 2.67GHz * 2 (12 cores / 24 threads)</td>
</tr>
<tr>
<td>Memory</td>
<td>96GB</td>
</tr>
<tr>
<td>Disk</td>
<td>HDD - 240GB RAID0</td>
</tr>
<tr>
<td>Network</td>
<td>10G-Ether</td>
</tr>
</tbody>
</table>

#### Server Spec - Storage x5:

<table>
<thead>
<tr>
<th>Component</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Intel(R) Xeon(R) CPU X5650 @ 2.67GHz * 2 (12 cores / 24 threads)</td>
</tr>
<tr>
<td>Memory</td>
<td>96GB</td>
</tr>
<tr>
<td>Disk</td>
<td>HDD - 240GB RAID0 (System)</td>
</tr>
<tr>
<td></td>
<td>HDD - 2TB RAID0 (Data)</td>
</tr>
<tr>
<td>Network</td>
<td>10G-Ether</td>
</tr>
</tbody>
</table>
**Brief Benchmark Report - 1st Case (HDD / R:W=9:1)**

**Environment:**

<table>
<thead>
<tr>
<th>Component</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network</td>
<td>10Gbps</td>
</tr>
<tr>
<td>OS</td>
<td>CentOS release 6.5 (Final)</td>
</tr>
<tr>
<td>Erlang</td>
<td>OTP R16B03-1</td>
</tr>
<tr>
<td>LeoFS</td>
<td>v1.0.2</td>
</tr>
</tbody>
</table>

**System Consistency Level:** [N:3, W:2, R:1, D:2]

**Benchmark Configuration:**

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Duration</td>
<td>4.0h</td>
</tr>
<tr>
<td>R:W</td>
<td>9:1</td>
</tr>
<tr>
<td># of Concurrent Processes</td>
<td>64</td>
</tr>
<tr>
<td># of Keys</td>
<td>100,000</td>
</tr>
</tbody>
</table>

**Value Size**

<table>
<thead>
<tr>
<th>Range (byte)</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1024</td>
<td>10240</td>
</tr>
<tr>
<td>10241</td>
<td>102400</td>
</tr>
<tr>
<td>10241</td>
<td>819200</td>
</tr>
<tr>
<td>819201</td>
<td>1572864</td>
</tr>
</tbody>
</table>
Brief Benchmark Report - 1st Case (HDD / R:W=9:1)

1,500 ops
No Errors
Brief Benchmark Report - 1st Case / Network Traffic

- 10.0 Gbps
- 7.0 Gbps
- 6.0 Gbps
- 5.0 Gbps

Storage

Gateway

60%
Brief Benchmark Report - 1st Case / Memory and CPU

Memory Usage

- gateway
- storage-1
- storage-2
- storage-3
- storage-4
- storage-5

CPU Load 5min

- gateway
- storage-1
- storage-2
- storage-3
- storage-4
- storage-5

Rakuten
Brief Benchmark Report - 2nd Case (HDD / R:W=8:2)

Environment:

<table>
<thead>
<tr>
<th>Network</th>
<th>10Gbps</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS</td>
<td>CentOS release 6.5 (Final)</td>
</tr>
<tr>
<td>Erlang</td>
<td>OTP R16B03-1</td>
</tr>
<tr>
<td>LeoFS</td>
<td>v1.0.2</td>
</tr>
</tbody>
</table>

System Consistency Level: [ N:3, W:2, R:1, D:2 ]

Benchmark Configuration:

<table>
<thead>
<tr>
<th>Duration</th>
<th>2.0h</th>
</tr>
</thead>
<tbody>
<tr>
<td>R:W</td>
<td>8:2</td>
</tr>
<tr>
<td># of Concurrent Processes</td>
<td>64</td>
</tr>
<tr>
<td># of Keys</td>
<td>100,000</td>
</tr>
</tbody>
</table>

Value Size

<table>
<thead>
<tr>
<th>Range (byte)</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1024</td>
<td>10240</td>
</tr>
<tr>
<td>10241</td>
<td>102400</td>
</tr>
<tr>
<td>10241</td>
<td>819200</td>
</tr>
<tr>
<td>819201</td>
<td>1572864</td>
</tr>
</tbody>
</table>
Brief Benchmark Report - 2nd Case (HDD / R:W=8:2)

60-70ms

80-90ms

1,000ops

No Errors
Compare 1st case with 2nd case
Brief Benchmark Report

1st Case - Network Traffic

2nd Case - Network Traffic

minus 0.7Gbps
**Brief Benchmark Report**

**1st Case - Disk util%**

**2nd Case - Disk util%**

1.8x high
Brief Benchmark Report

1st Case - CPU Load 5min

2nd Case - CPU Load 5min

1.00

1.6x high

1.00
Conclusion:

LeoFS kept in a stable performance through the benchmark

Bottleneck is Disk I/O

The cache mechanism contributed to reduce network traffic between Gateway and Storage
Multi Data Center Replication

Hokkaido, Japan
Multi Data Center Replication

HIGH-Scalability  +  Easy Operation for Admins
HIGH-Availability

US  |  Europe  |  Tokyo  |  Singapore

NO SPOF
NO Performance Degradation
Multi Data Center Replication

Designed it as simple as possible

1. Easy Operation to build **multi clusters**.
2. **Asynchronous data replication** between clusters
   
   Stacked data is **transferred** to remote cluster(s)
3. **Eventual consistency**
Multi Data Center Replication

Preparing MDC Replication

Executing “Join Cluster” on Manager Console

"join cluster DC-2 and DC-3"

DC-1
[# of replicas:3]

DC-2
[# of replicas:1]

DC-3
[# of replicas:1]

Monitors and Replicates each “RING” and “System Configuration”

"Leo Storage Platform"
Multi Data Center Replication

Stacking objects

Application(s)
Request to the Target Region

DC-1
[# of replicas:3]

DC-2
[# of replicas:1]

DC-3
[# of replicas:1]

Temporally Stacking objects
- One container's capacity is *32MB
- When capacity is full, send it to remote cluster(s)

* 32MB: default capacity - able to set optional value

Monitors and Replicates each “RING” and “System Configuration”

"Leo Storage Platform"
Multi Data Center Replication

Transferring stacked objects

Application(s)

Request to the Target Region

DC-1

Stacked objects

Compress it with LZ4

DC-2

Replicated an object

DC-3

Monitors and Replicates each “RING” and “System Configuration”

“Leo Storage Platform”
Multi Data Center Replication

Investigating stored objects

1) Receive metadata of stored objects
2) Compare them at the local cluster
3) Fix inconsistent objects

Monitor and Replicate each “RING” and “System Configuration”

"Leo Storage Platform"
LeoFS Administration at Rakuten

Presented by Hiroki Matsue
Rakuten Software Engineer

Kyoto, Japan
LeoFS Administration at Rakuten

Storage Platform
File Sharing Service
Others

Portal Site
Photo Storage
Background Storage of OpenStack
Storage Platform
Storage Platform - Scaling the Storage Platform

- Reduce Costs
- High Reliability
- Easy to Scale
- S3-API
Storage Platform - Scaling the Storage Platform

Using Various Services

Total Usage: 450TB

# of Files: 600 Million

Daily Growth: 100GB

Daily Reqs: 13 Million

E-Commerce

Blog

Review

Photo share

Portal & Contents (Movie)

Recruiting

Bookmark

Insurance

Calendar

Storage Platform

Rakuten
Storage Platform - System Layout

Requests from Web Applications / Browsers w/HTTP over S3-API

Load Balancer / Cache Servers

Gateway x 4

Storage x 14

Manager x 2

Total disk space: 600TB
Number of Files: 600Million
Access Stats:
800Mbps (MAX)
400Mbps (AVG)

Manager

Monitor

GUI Console

TCP/IP, SNMP

Erlang RPC

Erlang RPC

TCP/IP, SNMP

Web Applications / Browsers w/HTTP over S3-API
Storage Platform - Monitor

Status Collection *(Ganglia)*
Status Check *(Nagios)*
Port + Threshold Check

Ganglia and Nagios Agent

Send Mail Alert

Gateway x 4

Storage x 14

Manager x 2

Monitor

GUI Console

TCP/IP, SNMP

*Erlang RPC*

(Nagios) Status Collection
(Nagios) Status Check

Port + Threshold Check

Send Mail Alert
Storage Platform - Spreading Globally

Covering All Services with Multi DC Replication
File Sharing Service

https://owncloud.com/
File Sharing Service - Required Targets

Reduce Costs
Handle Confidential Files
Store Large Files
Scale Easily
File Sharing Service - Usage

Share **Docs** and **Movies** with Group Companies
Over **20** Companies, Over **10** Countries
Over **4,000** Users, Over **10,000** Teams
File Sharing Service - System Layout

Web GUI File Browser

Authenticate Users

LDAP

Manager x 2

(KVS)

Web GUI File Browser

Manage Configurations

Manage Login Session

(Rakuten)

(KVS)
File Sharing Service - Future Plans

Cover 25 Countries/Regions
Over 20,000 Users
Empowering the Services and the Users Through the Cloud Storage
Future Plans
Future Plans

NFS Support

Data-HUB: Centralize unstructured data in LeoFS
Future Plans

SavannaDB for Statistics Data

LeoInsight

Operate LeoFS

REST-API (JSON)

LeoCenter

SavannaDB's Agent
Insight LeoFS

Retrieve metrics and stats from SavannaDB's Agents

Notify a message of over # of req threshold

LeoFS
The Lion of Storage Systems

Storage Cluster
Manager
Gateway

openstack™
Set Sail for “Cloud Storage”

Website: leo-project.net
Twitter: @LeoFastStorage
Facebook: www.facebook.com/org.leofs