AMBRY: LinkedIn’s Immutable Blob Storage System

Arjun Shenoy
Data SRE
Motivation

- High MTTR
- Operational Overhead
- Scalability issue
- Not Cost effective
Motivation

- Reliability
- Scalability
- Low MTTR
- Active - Active
- Low Cost
Evaluation
Inspiration

Facebook

Haystack

#blobstore
Handle Based  Horizontally Scalable  Petabyte Scalable

Metadata Support  Active-Active  Large Blob Support
ARCHITECTURE
Data Center 1

Ambry-Frontend

Clustermap Manager

Ambry-Server

Clients

Cross-Colo Get

Replication

Data Center 2

Ambry-Frontend

Clustermap Manager

Ambry-Server

Clients

CDN
COMPONENTS
Ambry-Frontend

Coordination

Security

Non Blocking
Ambry-Front End

Ambry-frontend (Non-Blocking Model)

- NIO Layer
- Scaling Layer
- Remote Service Layer
Ambry-Server

Responsible

Storage

Replication

Built on

JBOD
- HardwareLayout file
- Node State
- Disk Capacity and State
- PartitionLayout file
- Mapping to nodes
- Partition State

```json
{
  "clusterName": "TestCluster",
  "version": 1,
  "partitions": [
    {
      "id": 0,
      "partitionState": "READ_WRITE",
      "replicaCapacityInBytes": 10737418240,
      "replicas": [
        {
          "hostname": "*****",
          "mountPath": "/tmp/x001/ambydata/",
          "port": 15088
        }
      ]
    }
  ]
}
```
HOW DO THEY INTERACT?
PutBlob WorkFlow

1. PutBlob
2. Choose Partition
3. Generate BlobId
4. Send PUT Request in Parallel to all 3 replicas
5. Wait for 2 acks
6. Return BlobId

Sample Blob-Id: /AAEAAQAAAAAADDFAAAAAAJDMyYWZiOTJmLTBkNDYtNDQyNS1iYzU0LWEwMWQ1Yzg3OTJkZQ.gif
PutBlob Statistics

- **Average QPS:** 370

- **Latencies (95\(^{th}\) Percentile):**
  - SmallBlob (<100KB): <10ms
  - MediumBlob (100KB - 4MB): <10ms (Goes up to 20ms extreme cases)
  - LargeBlob (>4MB): <50ms (Goes up to 80ms extreme cases)
GetBlob Workflow

1. GET(/AAAEQ...zU0wMWQ1Yzg3OTJkZQ.gif)
2. Determination of Partition Based on the Blob Id
3. Send Requests to Replicas in Parallel
4. Wait for at least 1 successful response
5. Return Blob
GetBlob Workflow

DC1

Client

Frontend

DC2

DataNode

Response

GET

DataNode

Cross-colo Get

404
GetBlob Statistics

- Average QPS: 24K

- Latencies (95th Percentile):
  - SmallBlob: <15ms
  - MediumBlob: <75ms (Goes up to 700ms extreme cases)
  - LargeBlob: <200ms (Goes up to 1.5s extreme cases)
Replication Workflow

1. GetBlobSince(Offset)
2. GetBlobSince(Offset)
3. BlobSinceContext
4. BlobSinceContext
5. Find BlobInfo
6. Context
Replication Workflow

- Inter DC Replication Time: 20-100ms
- Intra DC Replication Time: <10ms
SOME STATISTICS
Total Capacity: 900TB        Serving: 300TB

Growth Rate: 1160GB per day

Use-cases: Images, PDFs, Static files, Videos so on...
We’re Open Source!

Git Link: github.com/linkedin/ambry/wiki

Blog Post: engineering.linkedin.com/blog/2016/05/introducing-and-open-sourcing-ambry---linkedins-new-distributed-
Thank You!!
The information in this presentation was compiled from sources believed to be reliable for informational purposes only. It does not constitute legal or professional advice. The views and opinions expressed in this presentation are those of the authors and do not necessarily represent official policy or position of Linkedin.

All product and company names are trademarks™ or registered® trademarks of their respective holders.