How to manage your OpenStack Swift Cluster using Swift Metrics

Sreedhar Varma
Vedams Inc.
What is OpenStack Swift Cluster?

- Cluster of Storage Server Nodes, Proxy Server Nodes and Storage Devices

OpenStack Swift Object Storage Setup
Data Path Software Servers

- Data Path consists of 4 software servers
- Proxy Server
  - Responsible for accepting HTTP requests from user
  - Lookup storage server(s) where request needs to be routed
  - Performs read/write affinity – by sending requests to the same region
  - Account for failures – lookup handoff nodes
Data Path Software Servers

- Account Server
  - Tracks the names of containers in a particular account – handles listing of containers
  - SQLite database used to store data
  - Track statistics, but does not have location information about containers
  - Location information determined by proxy server based on ring

- Container Server
  - Deals with Object names in a particular container – handle listing of objects
Data Path Software Servers

- Object Server
  - Simply stores, retrieves and deletes objects stored on disk’s filesystems
- Usually account, container and object servers are put on one physical server – storage server
- Servers are distributed across zones and regions
Uploading Objects to Swift Cluster

- Request is sent via an HTTP PUT API call to a proxy server.
- Proxy server interacts with the ring to get a list of disks and associated object servers to write data to.
- Once a majority of disks acknowledge the write, the operation is returned as being successful.
Downloading Objects from Swift Cluster

- Read request is sent via an HTTP GET API call to a proxy server
- Proxy server interacts with the ring to get a list of disks and associated object servers
- Read request is issued to object servers in the same region as the proxy server - read affinity
- Multi-Region: Read object with latest timestamp
  - Proxy servers first request the time-stamp from all the object servers and read from the one with the newest copy
- Similar to the write case, in the case of a failure, handoff devices may be requested
How do you know if your Swift Cluster is healthy?

- Routine management – CPU Utilization, Memory, Disk usage, etc
- Swift stack monitoring – Proxy services, Storage server services, replicator, auditor, etc
- Tools:
  - Swift Recon
  - StatsD
  - Swift Dispersion
  - Swift Informant
Swift Recon

- Middleware software that is configured on the object server node and sits in the data path

- Metrics that are tracked include:
  - Load averages
  - The /proc/meminfo data
  - Mounted filesystems
  - Unmounted drives
  - Socket statistics
  - MD5 checksums of account, container, and object ring
  - Replication information
  - Number of quarantined accounts, containers, and objects
Swift Informant

- Middleware software that gives insight into client requests to the proxy server
- Software sits in the proxy server's data path and provides the following metrics to the StatsD server:
  - Status code for requests to account, container, or object
    - GET.200, PUT.201, POST.204, DELETE.204, PUT.404, etc.
  - Duration of the request and time until start_response metric was seen
  - Bytes transferred in the request
Swift StatsD metrics

- Swift services have been instrumented to send statistics (counters and logs) directly to a StatsD server that is configured
- StatsD metrics are provided in real time
- Configuration files containing the following parameters should be set in the Swift configuration files to enable StatsD logging:
  - log_statsd_host
  - log_statsd_port
  - log_statsd_default_sample_rate
  - log_statsd_sample_rate_factor
  - log_statsd_metric_prefix
Real Time Monitoring - Twister
Real Time Monitoring - using Swift StatsD metrics

<table>
<thead>
<tr>
<th>Create/PUT</th>
<th>Read/GET</th>
<th>Update/POST</th>
<th>Delete</th>
</tr>
</thead>
<tbody>
<tr>
<td>account-server. PUT.errors. timing</td>
<td>account-server. GET.errors. timing</td>
<td>account-server. POST.errors. timing</td>
<td>account-server. DELETE.errors. timing</td>
</tr>
<tr>
<td>account-server. PUT.timing</td>
<td>account-server. GET.timing</td>
<td>account-server. POST.timing</td>
<td>account-server. DELETE.timing</td>
</tr>
<tr>
<td>container-server.PUT.errors.timing</td>
<td>container-server.GET.errors.timing</td>
<td>container-server.POST.errors.timing</td>
<td>container-server.DELETE.errors.timing</td>
</tr>
<tr>
<td>container-server.PUT.timing</td>
<td>container-server.GET.timing</td>
<td>container-server.POST.timing</td>
<td>container-server.DELETE.timing</td>
</tr>
<tr>
<td>object-server.PUT.errors.timing</td>
<td>object-server.GET.errors.timing</td>
<td>object-server.POST.errors.timing</td>
<td>object-server.DELETE.errors.timing</td>
</tr>
</tbody>
</table>
Swift StatsD metrics Processing Engine

- StatsD Engine is configured in the Swift Cluster to receive Swift metrics and process them
- Various Algorithms used to learn Swift cluster behavior and predict failures
- Anomalies detected in the Swift Cluster reported as alerts to Administrator
- Administrator takes action and resets the alert to indicate that the problem is fixed
- Engine continues to track and predict Swift Cluster health
Swift StatsD metrics Processing Engine/UI

- Dynamically determine and map the Swift cluster components to UI
- Report Warnings, Errors in real time
- Pin point failures
  - Drive Failures
  - Swift Services Failures
  - Performance degradation with Upload/Download
- Bandwidth monitoring
- Dynamic Graphs
Swift Cluster – Good Health

Dashboard Cluster Overview

Proxy Server Status

Storage Node 1 Status

Storage Node 2 Status

Storage Node 3 Status

Storage Node 4 Status

Drive 1 Status

Drive 1 Status

Drive 1 Status

Drive 1 Status

Drive 2 Status

Drive 2 Status

Drive 2 Status

Drive 2 Status

Cluster Health: OK
Swift Cluster – I/O Bandwidth Monitoring
Swift Cluster – Drive Failure

Dashboard Cluster Overview

Cluster Health: DRIVE DOWN

Proxy Server Status

Storage Node 1 Status
Storage Node 2 Status
Storage Node 3 Status
Storage Node 4 Status

Drive 1 Status
Drive 1 Status
Drive 1 Status
Drive 1 Status

Drive 2 Status
Drive 2 Status
Drive 2 Status
Drive 2 Status

Network Status

Copyright © 1999-2014 Vedams Software Solutions Pvt Ltd. All rights reserved.
Swift Cluster – Swift Service Down

Dashboard Cluster Overview

Cluster Health: SERVICE DOWN

Proxy Server Status

Storage Node 1 Status
Storage Node 2 Status
Storage Node 3 Status
Storage Node 4 Status

Drive 1 Status
Drive 1 Status
Drive 1 Status
Drive 1 Status

Drive 2 Status
Drive 2 Status
Drive 2 Status
Drive 2 Status

Network Status
Implementing Cloud Storage With OpenStack Swift

- Chapter 5 – Managing Swift
Thank You

Questions?