November 7, 2013

OpenStack Summit Hong Kong

DAN WILSON
Global Operations Architecture, Concur
dan.wilson@concur.com
@tweetdanwilson

JOE ARNOLD
CEO, SwiftStack
joe@swiftstack.com
@joearnold

SwiftStack
Introduction (Concur / SwiftStack)

OpenStack Swift: Introduction

Concur User Story
Swift + Global Replication

OpenStack Swift
Global Replication in Swift
An Introduction to Concur

• Leader in integrated travel and expense management
  – Founded in 1993, headquartered in Redmond, Washington, U.S.
  – >50% of Fortune 500 use Concur
  – >$50B in T&E spend
    • Some of you in the audience will be expensing this trip with us!
What is OpenStack Swift?

Storage created specifically for heavy use and unstructured data

Reliable
Highly scalable
Hardware proof
What is SwiftStack?
Flexible, private cloud storage for today’s applications

- Start fast, operate easy
- Simplicity at scale
- No lock-in

SwiftStack Nodes

1. **Foundation**
   - Operates, manages & monitors hardware, storage software & middleware

2. **Integration**
   - OpenStack Swift
     - Released and supported by SwiftStack

3. **Control**
   - Standard Linux Distribution
     - Any Linux admin can install Ubuntu, Red Hat, CentOS
   - Standard Hardware
     - Multiple channels hardware procurement

Integrations & Interfaces
- End-user web UI, legacy interfaces, authentication, utilization API, etc.

Swift Runtime
- Integrated storage engine with all node components
Introduction (Concur / SwiftStack)

OpenStack Swift: Introduction

Concur User Story
Swift + Global Replication

OpenStack Swift
Global Replication in Swift
http://example.com/v1/account/container/object

**Swift’s Design Goals**

<table>
<thead>
<tr>
<th>Reliable</th>
<th>Configurable replica model with zones &amp; regions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Easy to use HTTP API – Developers don’t shard</td>
</tr>
<tr>
<td></td>
<td>High Concurrency (Supports lots of users)</td>
</tr>
<tr>
<td>Highly scalable</td>
<td>Multi-tenant – each account has own namespace</td>
</tr>
<tr>
<td></td>
<td>Tier &amp; scale any component in the system</td>
</tr>
<tr>
<td>Hardware proof</td>
<td>No Single Point of Failure (High Availability)</td>
</tr>
<tr>
<td></td>
<td>Assumes unreliable hardware</td>
</tr>
<tr>
<td></td>
<td>Mix &amp; match hardware vendors</td>
</tr>
</tbody>
</table>
OpenStack Swift

Users

Direct to Clients
Data is sent directly over HTTP to a browser, mobile device, etc.

Broad Language Support
- Java
- Javascript
- Python
- C#
- Ruby
- PHP
- Node.js

Services & Software
- Filesystem Gateways
- Desktop Clients
- Backup Software
- Archive Software
- Filesharing Software

In the data center
- Servers contact the storage system
- Application Data
- Backups / Archive
- Log Data
OpenStack Object Storage
(code named Swift)

- Object storage system similar to Amazon S3
- Apache 2 License

DEVELOPERS

136+

2009
2010
2011
2012
2013
2014

MATURE, PRODUCTION-GRADE STORAGE SYSTEM
Introduction (Concur / SwiftStack)

OpenStack Swift: Introduction

Concur User Story
Swift + Global Replication

OpenStack Swift
Global Replication in Swift
Scale = Tens of Millions of Images Monthly
Concur Imaging Application: Previous Workflow

Images processed first, then stored
Extract data, resize, etc.

Core part of the business
Disaster tolerance a must

Handling objects vs. files
Minimal difference for developers
Concur Imaging Infrastructure

Linear image processing
Mixed stack including Microsoft

Designed for flexibility
Abstracted services

3-tier architecture
Web, application, database

Images encrypted individually
Flexibility to use any storage
Challenges with Traditional SAN/NAS

**Availability**
- Handling failures of a single-node

**Performance**
- Unbalanced performance
- Each application server goes to the same node when reading/writing data

**Cost**
- Tied to specific vendor pricing

**Black box when troubleshooting**
- Limited visibility, reliant on vendor
Global Cluster Setup

Availability

- Two site replication strategy
- Designed for failure

Performance

- Quickly acknowledged writes
- Local Reads

Region 1

Zone 1

Zone 2

Region 2

Zone 1

Zone 2
Visibility under the hood

Open Source
- Storage functions entirely open source
- Go under the hood

Flexibility
- Can choose private, public
- Future proof infrastructure

Open Stack
- Open community
- Give back

Ecosystem Partners
- Many choices in the ecosystem
**Hardware Configuration**

**Scalability**
- Standard Server Hardware
- Scale by adding nodes

**36-drive 4U Storage Node**
- Intel Xeon Processor
- Seagate Constellation SATA or SAS Disks
- LSI HBA Controller + Expanders
- 96 GB RAM
- Mirrored boot drives
- Dual 10GbE Intel NIC

**1U Proxy Node**
- Dual Intel Xeon Processor
- 48 GB RAM
- Mirrored boot drives
- Dual 10GbE Intel NIC
TCO - Less Expensive than Public Cloud

$0.027 per GB / Mo.

- 4 replicas for redundancy (2 in each data center)
- All the power / space / cooling costs
- Our management costs
- SwiftStack + other license costs

Less expensive than public cloud.
Less expensive than traditional storage.
Ready for BILLIONS of Images in the Future!

What this means for the business
- Best experience for our users as possible.
- Amass data in a cost-effective way

What this means for our architecture
- Business growth forced finding a better way
- Architecture can now support growth

OpenStack Swift + global cluster functionality, enables us to achieve these goals.
Introduction (Concur / SwiftStack)

OpenStack Swift: Introduction

Concur User Story
Swift + Global Replication

OpenStack Swift
Global Replication in Swift
Global Clusters

Multi-Site with OpenStack Swift
Global Clusters

Data placement: as unique as possible

**Single Node Cluster**
Disks are “as-unique-as-possible”

**Small Cluster**
Storage Nodes are “as-unique-as-possible”

**Large Cluster**
Storage Racks are “as-unique-as-possible”

**Multi-Region**
Distributed data centers are “as-unique-as-possible”
Data Center Zones

Zone 1

Zone 2
Regions & Zones

Region 1
Zone 1
Zone 2

Region 2
Zone 1
Zone 2
Global Clusters

Distribute data across data center regions and zones

- Software automatically routes around storage or network failures
- Tolerate failures across regions and data center zones
- Utilize all available capacity to serve users vs. Traditional DR
- Send requests to ‘closest’ region
Global Clusters

Affinity Write

Region A

Region B

Region C
Proxy Service

Write
Proxy streams writes to each storage location simultaneously
There is not a 'master' object
Client receives 'OK' if quorum write successful

Client Request

Read
Proxy contacts a single available storage node and streams back data
Will try alternate replicas when there is failures
Proxy Read Affinity

Proxy Prioritizes “Nearby”
Zone / Region, Where am I?
Latency to Storage Node

Global DNS Routes User
Each proxy pool has it’s own hostname
Routes users to closest region

Global DNS

Region A
Region B
Separate Replication Network
Next: Storage Policies

Across Regions
Various Media
Number of Replicas

Replica Count

Media

Regions

High throughput
East Coast

SSD

West Coast
APAC

East Coast

HDD

APAC + West

2

3

4

5
Next: Erasure Codes

Reduce Storage Footprint
Pluggable encoding schemes


Replica Count

Media

Regions

Erasure Codes

HDD

SSD

West Coast

APAC

East Coast

Intel isa-l

Reed Solomon 6+3

Reduced Storage Footprint
Thank You!

OpenStack Summit Hong Kong