Deploying SwiftStack Object Storage for OpenStack Glance and Cinder Backups

February 2015
Table of Contents

Table of Contents
Introduction
   What is Swift and SwiftStack?
   Where do I use it?
Preparation
Using SwiftStack for Glance
Using SwiftStack for Cinder Backups
Introduction

What is Swift and SwiftStack?
As its name implies, OpenStack Swift is a key part of the OpenStack community, and as a founding project, it has been tightly integrated in small to massive clouds from the early days of OpenStack. Swift provides durable and scalable object storage (for unstructured data) alongside Cinder block storage (for virtual machines), yet as well-proven and reliable as Swift is—and there are hundreds and hundreds of petabytes of Swift deployed in public and private clouds around the world, the tasks and integrations necessary to deploy, manage, and monitor it—especially at a large scale—are tedious and complicated. Fortunately, SwiftStack—the leading contributor to the open-source Swift project in OpenStack—has developed incredibly simple-to-deploy, simple-to-manage, and simple-to-monitor “controller” software for OpenStack Swift (you might say, SwiftStack makes Swift “enterprise-ready”). With SwiftStack’s software, you can have Swift object storage ready to use in minutes, and you get OpenStack Swift plus simple management software, enterprise integration tools (e.g., LDAP and AD authentication, Nagios plugin), and support from the world’s recognized Swift experts.

Where do I use it?
SwiftStack object storage is typically used in three ways in OpenStack deployments:

1. **Glance virtual machine images**: The OpenStack Glance service can store its virtual machine image files in Swift object storage.

2. **Cinder volume backups**: The OpenStack Cinder service can be configured to backup its block storage volumes to Swift object storage.

3. **Native application storage**: With the exponentially increasing growth of unstructured data, modern application developers are rapidly developing against the Swift API to give their applications access to massively scalable and geographically distributed Swift object storage systems.
This document describes the steps necessary to point Glance and Cinder backups in a simple OpenStack environment to use a SwiftStack object storage system.

**Preparation**

For the purposes of this document, OpenStack was deployed using DevStack on a single virtual machine, and three SwiftStack storage nodes were deployed as virtual machines as well. This is acceptable for a test and development environment, but please note that production deployments of both OpenStack and SwiftStack run on bare-metal hardware and typically include many (tens to thousands) of nodes. Refer to the DevStack and SwiftStack documentation for initial installation instructions. Except where noted, the steps described here apply to both a small test/dev environment and to a massive cloud deployment. In addition, a Cinder backend was preconfigured for OpenStack to use when creating volumes.

**Using SwiftStack for Glance**

The Glance service in OpenStack primarily provides images from which virtual machines can boot. By default, these images are stored in local storage, but Glance can be easily redirected to utilize a Swift object storage cluster instead.

Basic configuration is simple, and details are included in the OpenStack Glance documentation: [http://docs.openstack.org/developer/glance/configuring.html#configuring-glance-storage-backends](http://docs.openstack.org/developer/glance/configuring.html#configuring-glance-storage-backends)

Specifically, in the `glance-api.conf` file edit the `default_store` and `stores` lines as shown here:
In the same file, edit `swift_store_auth_version`, `swift_store_auth_address`, `swift_store_user`, and `swift_store_key`; details for these and other parameters are included in the documentation:

```bash
# Swift Store Options
# Version of the authentication service to use
# Valid versions are '2' for keystone and '1' for swauth and rackspace
swift_store_auth_version = 1

# Address where the Swift authentication service lives
# Valid schemes are 'http://' and 'https://'
# If no scheme specified, default to 'https://'
# For swauth, use something like '127.0.0.1:8000/v1.0/
swift_store_auth_address = http://192.168.57.41/auth/v1.0

# User to authenticate against the Swift authentication service
# If you use Swift authentication service, set it to 'account': 'user'
# where 'account' is a Swift storage account and 'user'
# is a user in that account
swift_store_user = user1

# Auth key for the user authenticating against the
# Swift authentication service
swift_store_key = password

# Container within the account that the account should use
# for storing images in Swift
swift_store_container = glance

# Do we create the container if it does not exist?
swift_store_create_container_on_put = True
```

Note, the authentication address for the Swift cluster can be found in the SwiftStack console:

![SwiftStack Console](image)

After saving the changes in `glance-api.conf`, restart the `glance-api` service. In DevStack, that is done by killing and restarting the `g-api` screen. Then, test the integration by creating a new image and verifying it is successfully stored in Swift:
Using SwiftStack for Cinder Backups

The Cinder service in OpenStack provides block storage most often used for virtual machines. To back up those volumes, a Cinder backup service is available that can be configured to utilize Swift object storage to store those backup files.

Basic configuration is simple, and details are included in the OpenStack Glance documentation: http://docs.openstack.org/trunk/config-reference/content/swift-backup-driver.html
In order to use the Cinder backup service, you must enable the service. In DevStack, that is done by adding `enable_service c-bak` to `~/devstack/localrc` and rerunning the `stack.sh` script.

To configure the backup service, add the following items to `/etc/cinder/cinder.conf` as in the following screenshots:

```
backup_driver
backup_swift_url
backup_swift_auth
backup_swift_user
backup_swift_key
backup_swift_container
```

With those changes saved, restart the Cinder backup service, and use commands like the following:

```
cinder service-list
cinder list
cinder backup-create <volume ID>
cinder backup-list
cinder backup-show <backup ID>
cinder backup-restore <backup ID>
```
To test the integration, perform a backup, and verify that the backup is populated in the volumebackups container on SwiftStack: