Storage in Openstack:
Block Storage, Ephemeral Storage, Object Storage

Marica Antonacci - INFN Bari

Scuola di Cloud Storage
Bari, 9-11 Dicembre 2015
Outline

• Openstack intro

• Backend storage for Openstack components

• Demo

  ‣ Configure Cinder, Nova, Glance to use Ceph
Openstack Architecture
Conceptual architecture
Main components

- Keystone - Identity Service
- Nova - Compute Service
- Glance - Image Service
- Cinder - Block Storage Service
- Swift - Object Storage Service
- Neutron - Networking Service
Backend Storage for Openstack components
Openstack Storage

1. **Ephemeral** storage with Nova
2. **Persistent Block** Storage with Cinder
3. **Object** Storage with Swift
4. [File Share Service with Manila]
Ephemeral storage is allocated for an instance and is deleted when the instance is deleted.

- used to run the operating system and scratch space

By default, Compute stores ephemeral drives as files on local disks on the Compute node

- /var/lib/nova/instances
- only VM migration moves the disk image to another compute node (Nova copies it via SSH)

**Shared Filesystem**

- enables live migration
  - /var/lib/nova/instances located on DFS and exported to the compute nodes
  - RBD driver
Block Storage

- Add additional persistent storage to a virtual machine
- It is accessed through a block device that can be partitioned, formatted, and mounted
- Can be resized
- Persists until the user deletes it
- Can be encrypted
- Use case: provide persistent storage for long-running services that require strong consistency and low-latency connectivity (e.g. databases)
Cinder

- Block data for volumes
- Stored in one or more backend storage devices
- Multi-backend support
- QoS support

Openstack Block Storage Drivers Support Matrix: https://wiki.openstack.org/wiki/CinderSupportMatrix
Object Storage

- Stores unstructured data, including VM images
- Eventually consistent
- Highly available. Can be replicated across different data centers
- Provides REST APIs (native and standard, e.g. S3, CDMI) and offer simple web services interfaces for access
- Use-cases: Storage for backup files database dumps, and log files; Large data sets (e.g. multimedia files); backend storage of the Image Service
The primary objective of Glance is to publish a catalog of virtual machine images.

Main components:

- **glance-api**: accepts Image API calls for image discovery, retrieval and storage

- **glance-registry**: stores, processes, and retrieves metadata for images

- **storage backend** (filesystem, rbd, swift, s3, cinder, etc.)
Glance images

- Image block data
- Read-only
- Can be massive file sizes (100+ GB for some Windows images)
- Huge array of backend store drivers
  - Worst option: filesystem (unless it’s a shared filesystem)
  - Better options: rbd, sheepdog, swift and s3
- These are distributed storage systems with built-in redundancy
- Choose one based on degree of familiarity, size of deployment
Ceph: de-facto storage backend for Openstack

- Storage consolidation:
  - Glance image storage in RADOS
  - Cinder provisioning of persistent RBD volumes
  - Nova provisioning of ephemeral RBD volumes
  - Swift and Keystone compatible RADOS
Dedicated pools and users

- Three different pools: images, volumes, vms, [backups]

```
ceph osd pool create volumes 128
ceph osd pool create images 128
ceph osd pool create vms 128
```

- Dedicated and right-limited user to access the pools

  - prior Icehouse, we had to use client.admin in libvirt to authenticate and interact with the Ceph cluster

```
ceph auth get-or-create client.cinder mon 'allow r' osd 'allow class-read object_prefix rbd_children, allow rwx pool=volumes, allow rwx pool=vms, allow rx pool=images'

ceph auth get-or-create client.glance mon 'allow r' osd 'allow class-read object_prefix rbd_children, allow rwx pool=images'
```
✓ Decouple VM storage from hypervisors

✓ Images stored in RADOS

✓ Snapshots

✓ Live migration

✓ Thin provisioning

✓ Copy on write cloning

✓ Images striped across storage pool
Configure RBD backend for nova-compute

- **Configure Libvirt**

```plaintext
uuidgen
d38c68b3-53d3-4a4f-8f36-10d3b37ca4eb

cat > secret.xml <<EOF
<secret ephemeral='no' private='no'>
  <uuid> d38c68b3-53d3-4a4f-8f36-10d3b37ca4eb</uuid>
  <usage type='ceph'>
    <name>client.cinder secret</name>
  </usage>
</secret>
EOF

sudo virsh secret-define --file secret.xml
Secret d38c68b3-53d3-4a4f-8f36-10d3b37ca4eb created
sudo virsh secret-set-value --secret d38c68b3-53d3-4a4f-8f36-10d3b37ca4eb --base64 $(cat client.cinder.key) && rm client.cinder.key secret.xml
```

- **Edit /etc/nova/nova.conf, add:**

```plaintext
[libvirt]
images_type = rbd
images_rbd_pool = vms
images_rbd_ceph_conf = /etc/ceph/ceph.conf
rbd_user = cinder
rbd_secret_uuid = d38c68b3-53d3-4a4f-8f36-10d3b37ca4eb
```
Domain definition

• VM disk on Filesystem

```xml
<disk type='file' device='disk'>
  <driver name='qemu' type='qcow2' cache='none'/>
  <source file='/var/lib/nova/instances/7b972a35-3ee5-4931-8d83-b83cb6e42ef2/disk'/>
  <target dev='vda' bus='virtio'/>
  <alias name='virtio-disk0'/>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x04' function='0x0'/>
</disk>
```

• VM disk on RBD

```xml
<disk type='network' device='disk'>
  <driver name='qemu' type='raw' cache='none'/>
  <auth username='ubuntu'>
    <secret type='ceph' uuid='457eb676-33da-42ec-9a8c-9293d545c337'/>
  </auth>
  <source protocol='rbd' name='vms/197d07bf-670e-401a-8597-12099f1911b5_disk'>
    <host name='90.147.102.76' port='6789'/>
  </source>
  <target dev='vda' bus='virtio'/>
  <alias name='virtio-disk0'/>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x04' function='0x0'/>
</disk>
```
Limitations

• Ceph doesn’t support **QCOW2** for hosting virtual machine disk

• nova-compute checks the image format before booting the machine
  
  • QCOW2 images are converted to RAW

• Snapshotting instances still takes a long journey getting written to local disk then pushed back up to glance.
  
  • Liberty partially implements RBD snapshots instead of QEMU snapshots
Glance configuration example

- Edit /etc/glance/glance-api.conf
  - [glance_store]/default_store
  - [glance_store]/stores

```
[glance_store]
  ..
  rbd_store_pool = images
  rbd_store_user = glance
  rbd_store_ceph_conf = /etc/ceph/ceph.conf
  rbd_store_chunk_size = 8
```

```
[glance_store]
 ..
  swift_store_auth_address = 127.0.0.1:5000/v2.0/
  swift_store_user = jdoe:jdoe
  swift_store_container = glance
  swift_store_large_object_size = 5120
```

- Use --store option with `glance image-create` to specify where the image will be stored in case of multiple backends. If not provided the default store will be used.
Enable Copy-on-Write clones

• Use RAW images (prior Liberty)
  • **Liberty**: qemu-img convert -O raw rbd:$pool/$uuid rbd:$pool/$uuid

• Expose image URL (/etc/glance/glance-*.conf)
  ```
  [DEFAULT]
  ...
  show_image_direct_url = True
  ```

• Disable glance cache:
  ```
  [paste_deploy]
  flavor = keystone+cachemanagement
  ```

```bash
$sudo rbd info vms/af0c0e38-8b7f-4fa7-81b2-2ab604623d61_disk
rbd image 'af0c0e38-8b7f-4fa7-81b2-2ab604623d61_disk':
  size 1024 MB in 128 objects
  order 23 (8192 kB objects)
  block_name_prefix: rbd_data.62136607a448
  format: 2
  features: layering
  flags:
  parent: images/e532032f-d46a-45b7-98e5-0404694dd365@snap
  overlap: 40162 kB
```
Cinder: rbd driver

```
[ceph]
volume_driver = cinder.volume.drivers.rbd.RBD_driver
rbd_pool = volumes
glance_api_version = 2
rbd_user = cinder
rbd_secret_uuid = 925560f4-ae0d-40a6-805f-dc628d63cef8
volume_backend_name=CEPH
rbd_ceph_conf=/etc/ceph/ceph.conf
rbd_flatten_volume_from_snapshot=false
rbd_max_clone_depth=5
```
Cinder backup

• Un backup è una copia del volume archiviata nell’Object Store

• Managed by a separate service: **cinder-backup** (not installed in the default configuration)

• Configurable drivers:
  
  ➪ Ceph
  
  ➪ Swift
  
  ➪ NFS (since Kilo)
  
  ➪ IBM Tivoli Storage Manager
Backup driver Swift

Edit cinder.conf - DEFAULT section

```ini
backup_driver=cinder.backup.drivers.swift

# The URL of the Swift endpoint (string value)
backup_swift_url=http://localhost:8080/v1/AUTH_

# Swift authentication mechanism (string value)
backup_swift_auth=per_user

# Swift user name (string value)
#backup_swift_user=<None>

# Swift key for authentication (string value)
#backup_swift_key=<None>

# The default Swift container to use (string value)
backup_swift_container=volumebackups

# The size in bytes of Swift backup objects (integer value)
backup_swift_object_size=52428800

# The number of retries to make for Swift operations (integer value)
#backup_swift_retry_attempts=3

# The backoff time in seconds between Swift retries (integer value)
#backup_swift_retry_backoff=2

# Compression algorithm (None to disable) (string value)
#backup_compression_algorithm=zlib
```
Backup driver Swift

Cinder backup create:

```
root@wn-recas-uniba-30:~# cinder backup-create --display-name test-bck 4b849af0-f989-4e95-9d79-60aede80a4ca
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>0542b982-45c5-4b39-8caf-930c05c12654</td>
</tr>
<tr>
<td>name</td>
<td>test-bck</td>
</tr>
<tr>
<td>volume_id</td>
<td>4b849af0-f989-4e95-9d79-60aede80a4ca</td>
</tr>
</tbody>
</table>

...we find it in Swift:

```
root@wn-recas-uniba-30:~# swift list volumebackups
```

Nota: cinder-backup allow to create replicated volume backups (exploiting the Object Storage capabilities). Disaster-recovery can be implemented if the Object Store is geographically distributed.
Backup driver Ceph

Modificare il file cinder.conf - sezione DEFAULT

backup_driver=cinder.backup.drivers.ceph

# Ceph configuration file to use. (string value)
backup_ceph_conf=/etc/ceph/ceph.conf

# The Ceph user to connect with. Default here is to use the
# same user as for Cinder volumes. If not using cephx this
# should be set to None. (string value)
backup_ceph_user=cinder-backup

# The chunk size, in bytes, that a backup is broken into
# before transfer to the Ceph object store. (integer value)
#backup_ceph_chunk_size=134217728

# The Ceph pool where volume backups are stored. (string
# value)
backup_ceph_pool=backups

# RBD stripe unit to use when creating a backup image.
# (integer value)
#backup_ceph_stripe_unit=0

# RBD stripe count to use when creating a backup image.
# (integer value)
#backup_ceph_stripe_count=0

# If True, always discard excess bytes when restoring volumes
# i.e. pad with zeroes. (boolean value)
#restore_discard_excess_bytes=true
Ceph backup service

- Ceph driver allows backing up volumes of any type to a Ceph object store

- Ceph driver is also capable of detecting if the source volume is stored on the same kind of backend, i.e. Ceph RBD

- In this case, it attempts to perform an incremental backup, falling back to full backup/copy if the former fails.

- It also supports backing up...
  - within the same pool (not recommended)
  - between two different pools
  - between two different Ceph clusters
Ceph backup: under the hood

Workflow executed for the first backup of a volume

1. Create a base backup image used for storing differential exports
2. Snapshot source volume to create a new point-in-time
3. Perform differential transfer:

   ```
   rbd export-diff --id cinder --conf /etc/ceph/ceph.conf --pool volumes volumes/volume-afa33905-0d87-42ff-ad36-9c75f9cf09ba@backup.4e50e949-3dcd-4ff1-89e0-a6a9c1beb5c1.snap.1418722200.64 -
   rbd import-diff --id cinder-backup --conf /etc/ceph/ceph.conf --pool backups - backups/volume-afa33905-0d87-42ff-ad36-9c75f9cf09ba.backup.base
   ```

Results in rbd:

```
# rbd -p volumes ls -l
NAME                                                                                                         SIZE PARENT FMT
volumes/afa33905-0d87-42ff-ad36-9c75f9cf09ba                                                                10240M          2
volumes/afa33905-0d87-42ff-ad36-9c75f9cf09ba@backup.4e50e949-3dcd-4ff1-89e0-a6a9c1beb5c1.snap.1418722200.64 10240M          2
# rbd -p backups ls -l
NAME                                                                                                                     SIZE PARENT FMT
backups/volume-afa33905-0d87-42ff-ad36-9c75f9cf09ba.backup.base                                                                10240M
backups/volume-afa33905-0d87-42ff-ad36-9c75f9cf09ba.backup.base@backup.4e50e949-3dcd-4ff1-89e0-a6a9c1beb5c1.snap.1418722200.64 10240M
```

Ceph backup: under the hood (2)

Workflow executed for the next backups

1. Snapshot source volume to create a new point-in-time

2. Perform differential transfer using --from-snap:

   rbd export-diff --id cinder --conf /etc/ceph/ceph.conf --pool volumes --from-snap backup.4e50e949-3dcd-4ff1-89e0-a6a9c1beb5c1.snap.1418722200.64 volumes/volume-afa33905-0d87-42ff-ad36-9c75fdec09a@backup.c255e3ca-f01b-4fe6-ad9f-af0524a7b531.snap.1418725945.25

   rbd import-diff --id cinder-backup --conf /etc/ceph/ceph.conf --pool backups - backups/volume-afa33905-0d87-42ff-ad36-9c75fdec09ba.backup.base

Results in rbd:

```
# rbd -p volumes ls -l
NAME |
PROT |
LOCK |
------
volume-afa33905-0d87-42ff-ad36-9c75fdec09ba | 10240M |
volume-afa33905-0d87-42ff-ad36-9c75fdec09ba@backup.c255e3ca-f01b-4fe6-ad9f-af0524a7b531.snap.1418725945.25 | 10240M |

# rbd -p backups ls -l
NAME |
PARENT |
FMT |
 PROT |
 LOCK |
------|
------|
------|
volume-afa33905-0d87-42ff-ad36-9c75fdec09ba.backup.base | 10240M |
volume-afa33905-0d87-42ff-ad36-9c75fdec09ba.backup.base@backup.4e50e949-3dcd-4ff1-89e0-a6a9c1beb5c1.snap.1418722200.64 | 10240M |
volume-afa33905-0d87-42ff-ad36-9c75fdec09ba.backup.base@backup.c255e3ca-f01b-4fe6-ad9f-af0524a7b531.snap.1418725945.25 | 10240M |
```
Demo!
Demo Environment

- Openstack installation using DevStack
  - ALL-IN-ONE Single VM
  - Openstack deployment & cloud-init
- Ceph docker
#cloud-config

users:
- default
- name: stack
  lock_passwd: False
  sudo: ["ALL=(ALL) NOPASSWD:ALL\nDefaults:stack !requiretty"]
  shell: /bin/bash

write_files:
- content: |
  #!/bin/sh
  DEBIAN_FRONTEND=noninteractive sudo apt-get -qqy update || sudo yum update -qy
  DEBIAN_FRONTEND=noninteractive sudo apt-get install -qqy git || sudo yum install -qy git
  sudo chown stack:stack /home/stack
  cd /home/stack
  git clone https://git.openstack.org/openstack-dev/devstack -b stable/kilo
  cd devstack
  echo '[[local|localrc]]' > local.conf
  echo ADMIN_PASSWORD=D3moTutorial.2015 >> local.conf
  echo DATABASE_PASSWORD=D3moTutorial.2015 >> local.conf
  echo RABBIT_PASSWORD=D3moTutorial.2015 >> local.conf
  echo SERVICE_PASSWORD=D3moTutorial.2015 >> local.conf
  echo SERVICE_TOKEN=tokenD3moTutorial.2015 >> local.conf
  echo "enable_service n-cauth" >> local.conf
  ./stack.sh
  path: /home/stack/start.sh
  permissions: 0755

runcmd:
- su -l stack ./start.sh

Gist: https://goo.gl/qimDuH

Choose a branch with -b or the HEAD without -b
Devstack Usage

• The output of the command stack.sh

  Horizon is now available at http://192.168.1.15/
  Keystone is serving at http://192.168.1.15:5000/v2.0/
  Examples on using novaclient command line is in exercise.sh
  The default users are: admin and demo
  The password: D3moTutorial.2015
  This is your host ip: 192.168.1.15

• Useful commands:
  
  • ./unstack.sh, ./rejoin-stack.sh, screen -x

  • ALT+a+" : allow to list of service tabs and choose one of them

  • ALT+a+d : detach from screen

• You can modify the configuration of a service and then restart the service
  entering the corresponding screen tab and typing CRTL+C and then launching
  the last command in the history with ARROW UP
The container provides all the Ceph daemons, so you can rapidly start playing with Ceph.

```
docker run -d --net=host -v /etc/ceph:/etc/ceph \
  -e MON_IP=90.150.10.76 \
  -e CEPH_NETWORK=90.150.10.0/24 ceph/demo
```

/`etc/ceph` bound-mounted to `/etc/ceph` in the container. Copy this folder on any client machine to connect to the cluster.