OpenStack

Hung Pham
Minh Tri Tran
Quang Pham

FRANKFURT
UNIVERSITY
OF APPLIED SCIENCES

Frankfurt University of Applied Sciences

January 25, 2019

Overview

1. OpenStack Introduction
2. OpenStack Implementation
3. Conclusion

What is OpenStack?

Open source software for creating private and public clouds

OpenStack controls large pools of:

- Compute
- Storage
- Networking resources

OpenStack lets users deploy virtual machines and other instances that handle different tasks

OpenStack Main Components

- Compute Service – nova
- Image Service – glance
- Networking Service – neutron
- Identity Service – keystone
- Block Storage Service – cinder
- Dashboard – horizon

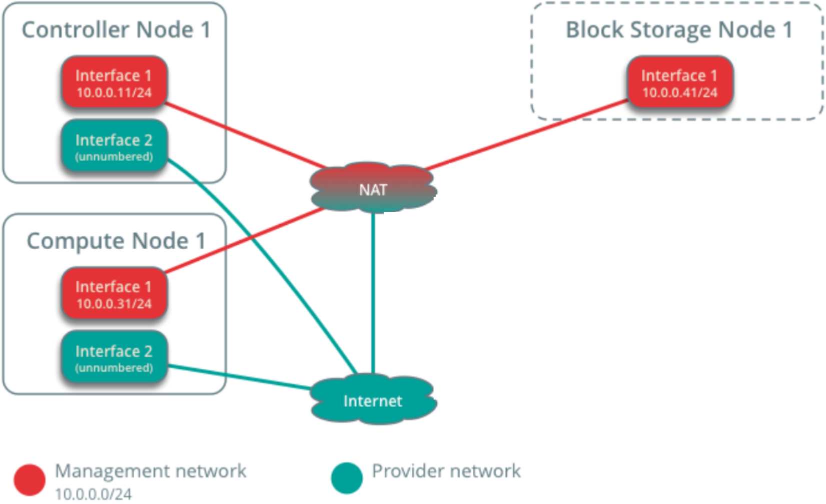
Preparation

Physical machines:

- Controller node
- Compute node
- Storage node

- Cores: 4
- Threads per Core: 8
- Memory: 16GB
- Storage: 1000GB
- OS: RHEL 7.6

Preparation



Preparation

Physical machine: Solution 2: 1 machine server

- Cores: 12
- Threads per Core: 2
- Memory: 40GB
- Storage: 280GB
- OS: CentOS 7.6

Preparation

MariaDB: Database

- bind-address = controller_node_ipaddress
  or
- bind-address = 127.0.0.1

Preparation

RabbitMQ: Message queue

- user: openstack
- permissions: configure, read, write

Preparation

Memcached: Cache tokens for Identity service

Etcd: a distributed reliable key-value store for distributed key locking, storing configuration, keeping track of service liveness

Identity Service – keystone

Provides:

- API client authentication
- Service discovery
- Distributed multi-tenant authorization

Implemented with OpenStack's Identity API

Identity Service – keystone

Steps:

- Create database and database user for keystone
- Bootstrap the Identity service
  - admin password
  - admin url
  - internal url
  - public url
  - region
- Bring up httpd for HTTP API server

Image Service – glance

Provides: provides a service where users can upload, discover, registering, and retrieving virtual machine (VM) images and metadata definitions.

RESTful API that allows querying of VM image metadata as well as retrieval of the actual image

Image Service – glance

Components:

- glance-api: accepts Image API calls for image discovery, retrieval, and storage
- glance-registry: stores, processes, and retrieves metadata about images
- Database: stores image metadata
- Storage repository for image files
- Metadata definition service: common API for vendors, admins, services, and users to meaningfully define their own custom metadata

Image Service – glance

Steps:

- Create database and database user for glance
- Create user and service in OpenStack Identity service - keystone
- Add endpoint of glance api to keystone
- Install package openstack-glance
- Configure glance to use keystone and mariadb
- Start API server and registry service

Compute Service – nova

- Concept: To provide compute instances (virtual server) on demand.
- Interaction with:
  - Hypervisor and VM managers (e.g: Qemu-kvm + libvirt, Xen...)
  - Identity service.
  - Image service
  - Networking service
  - Dashboard service

Compute Service – nova

Prerequisite:

- Keystone.
- Glance.
- Neutron.

General setup step:

- Setup appropriate database and credentials.
- Configure the components.

Details for installing the service on different OS can be found on the documentation.

Networking Service – neutron

Concept:

- To provide a method to attach different devices from other OpenStack services to networks.
- Provide a way to create virtual network topologies, firewall, load balancer and VPN.
- At least one external network on any given network setting.
- VMs in the same internal network can interact via the network directly.

Networking Service – neutron

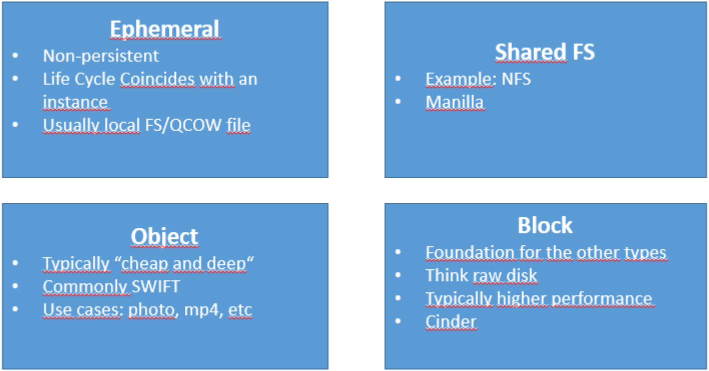
Installation:

- Configure network between nodes.
- Setup appropriate database and credentials.
- Configure the components on each nodes accordingly.

Details for installing the service can be found on the documentation.

Cinder Definition

Different types of storage in OpenStack



Ephemeral

- Non-persistent
- Life Cycle Coincides with an instance
- Usually local FS/QCOW file

Shared FS

- Example: NFS
- Manilla

Object

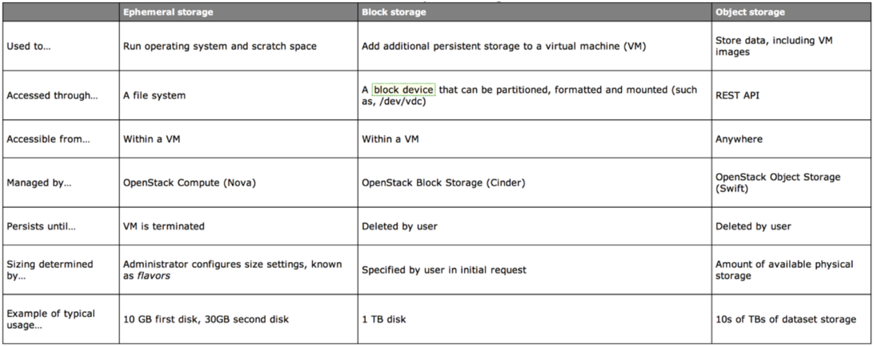
- Typically "cheap and deep"
- Commonly SWIFT
- Use cases: photo, mp4, etc

Block

- Foundation for the other types
- Think raw disk
- Typically higher performance
- Cinder

Cinder Definition

Provides instances with block storage volumes that persist even when the instances they are attached to are terminated



	Ephemeral storage	Block storage	Object storage
Used to...	Run operating system and scratch space	Add additional persistent storage to a virtual machine (VM)	Store data, including VM images
Accessed through...	A file system	A block device that can be partitioned, formatted and mounted (such as, /dev/vdc)	REST API
Accessible from...	Within a VM	Within a VM	Anywhere
Managed by...	OpenStack Compute (Nova)	OpenStack Block Storage (Cinder)	OpenStack Object Storage (Swift)
Persists until...	VM is terminated	Deleted by user	Deleted by user
Sizing determined by...	Administrator configures size settings, known as *flavors*	Specified by user in initial request	Amount of available physical storage
Example of typical usage...	10 GB first disk, 30GB second disk	1 TB disk	10s of TBs of dataset storage

Cinder Definition

Use case

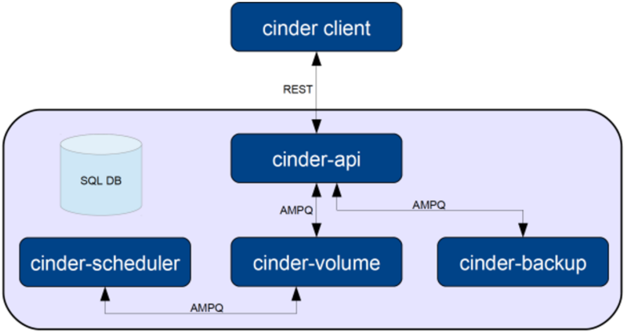
- Create volume
- Create Snapeshot
- Backup
- Attack/Detach

Cinder provides

- Commands and API‘s to interact with vendors‘ storage backend
- Persistent storage to VM‘s

Expose vendors storage hardware to the cloud

Cinder Architecture



Installation

Use case

- Controller node
- Storage node

Prerequisite

Create database, service credentials, API endpoints

Create database:

- Use the database access client to connect to the database server as the root user: mysql –u root -p
- Create cinder database: MariaDB [(none)]> CREATE DATABASE cinder;
- Grant proper access to cinder database: MariaDB [(none)]> GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'localhost' IDENTIFIED BY 'CINDERDBPASS';

Exit the database access client

Prerequisite

Source the admin credentials to gain access to admin-only CLI commands: . admin-openrc

To create the service credentials, complete these steps:

- Create a cinder user: openstack user create –domain default –password-prompt cinder
- Add the admin role to the cinder user: openstack user create –domain default –password-prompt cinder
- Create the cinderv2 and cinderv3 service entities:
  - openstack service create –name cinderv2 –description "OpenStack Block Storage" volumev2
  - openstack service create –name cinderv3 –description "OpenStack Block Storage" volumev3

Prerequisites

Create the Block Storage service API endpoints:

- openstack endpoint create –region RegionOne volumev2 public http://controller:8776/v2/
- openstack endpoint create –region RegionOne volumev2 internal http://controller:8776/v2/
- openstack endpoint create –region RegionOne volumev2 admin http://controller:8776/v2/

Install and configure components

- openstack endpoint create –region RegionOne volumev3 public http://controller:8776/v3/
- openstack endpoint create –region RegionOne volumev3 internal http://controller:8776/v3/
- openstack endpoint create –region RegionOne volumev3 admin http://controller:8776/v3/

Install and configure components

Instal the packages: yum install openstack-cinder

Edit the /etc/cinder/cinder.conf file and complete the following actions:

- In the [database] section, configure database access: connection = mysql+pymysql://cinder:CINDERmotlaso2lachu@controller/cinder
- In [DEFAULT], configure RabbitMQ message queue access: transport_url = rabbit://openstack:RABBITmothaibABA4444nam@controller

Install and configure components

In the [DEFAULT] and [keystone_authtoken] sections, configure Identity service access:

auth_strategy = keystone
[keystone_authtoken]
auth_uri = http://controller:5000 auth_url = http://controller:5000
memcached_servers = controller:11211 auth_type = password
project_domain_id = default user_domain_id = default project_name =
service username = cinder password = mothaibe4lam

Install and configure components

In the [DEFAULT] section, configure the my_ip option to use the management interface IP address of the controller node:

```
[DEFAULT]
my_ip = 10.0.0.11
```

Install and configure components

In the [oslo_concurrency] section, configure the lock path
[oslo_concurrency] lock_path = /var/lib/cinder/tmp

Populate the Block Storage database: # su -s /bin/sh -c "cinder-manage db sync" cinder

Configure Compute to use Block Storage

Edit the /etc/nova/nova.conf file and add the following to it:
[cinder] os_region_name = RegionOne

Finalize installation

Restart the Compute API service: # systemctl restart openstack-nova-api.service

Start the Block Storage services and configure them to start when the system boots:

- # systemctl enable openstack-cinder-api.service openstack-cinder-scheduler.service
- # systemctl start openstack-cinder-api.service openstack-cinder-scheduler.service

Install and configure a storage node: Prerequisites

Install the LVM packages: # yum install lvm2 device-mapper-persistent-data

Start the LVM metadata service and configure it to start when the system boots:

- # systemctl enable lvm2-lvmetad.service
- # systemctl start lvm2-lvmetad.service

Prerequisites

Create the LVM physical volume /dev/sdb: # pvcreate /dev/sdb

Create the LVM volume group cinder-volumes: # vgcreate cinder-volumes /dev/sdb

The Block Storage service creates logical volumes in this volume group.

Install and configure components

Install the packages: # yum install openstack-cinder targetcli python-keystone

In the [database] section, configure database access:

connection = mysql+pymysql://cinder:CINDER_DBPASS@controller/cinder

Install and configure components

In the [DEFAULT] section, configure RabbitMQ messaage queue access: [DEFAULT] transport_url = rabbit://openstack:RABBIT_PASS@controller Replace RABBIT_PASS with the password you chose for the openstack account in RabbitMQ.

Install and configure components

In the [lvm] section, configure the LVM back end with the LVM driver, cinder-volumes volume group, iSCSI protocol, and appropriate iSCSI service. If the [lvm] section does not exist, create it:

[lvm] volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = cinder-volumes iscsi_protocol = iscsi iscsi_helper =
lioadm

Install and configure components

In the [DEFAULT] section, enable the LVM back end: [DEFAULT] enabled_backends = lvm

In the [DEFAULT] section, configure the location of the Image service API: [DEFAULT] glance_api_servers = http://controller:9292

In the [oslo_concurrency] section, configure the lock path: [oslo_concurrency] lock_path = /var/lib/cinder/tmp

Final installation

Start the Block Storage volume service including its dependencies and configure them to start when the system boots:

- # systemctl enable openstack-cinder-volume.service target.service
- # systemctl start openstack-cinder-volume.service target.service

Conclusion

Advantages:

- Short deployment time (a few hours for a small-scale system)
- Scalable services

Disadvantages:

- Not too easy to implement

Thank You!

References

OpenStack
OpenStack Documentation
[Online].Available: https://docs.openstack.org/rocky/