Chapter 1

Introduction to OpenStack and Its Components

1.1 Cloud Computing

Cloud computing is a computing model, where resources such as computing power, storage, network and software are abstracted and provided as services on the internet in a remotely accessible fashion. Billing models for these services are generally similar to the ones adopted for public utilities. On-demand availability, ease of provisioning, dynamic and virtually infinite scalability are some of the key attributes of cloud computing.

An infrastructure setup using the cloud computing model is generally referred to as the "cloud". The following are the broad categories of services available on the cloud:

- Infrastructure as a Service (IaaS)
- Platform as a Service (PaaS)
- Software as a Service (SaaS)

Infrastructure as a service refers to the hardware components and the operating system provided as a service. This is similar to a virtual private server offerings where the virtual server and the operating system are managed by the vendor and the user handles installation of applications and other software.

Platform as a service refers to a system where the hardware and the necessary software required to run an application. For instance, if we have to run a PHP web application, the PaaS platform provides us the PHP processor, allied libraries required for our application and the hardware. Platform as a service also lets the application scale in response to increase in load. All of these components are managed by the service provider, thus letting the user focus on development and testing of the application.

Software as a service is any application provided as a service. Usually, applications are purchased as a product. Like an image processing software or a spreadsheet software. But in SaaS model, the software is not purchased, it is rather used as a service over the browser. Examples are Google documents, and Microsoft 360!

1.2 OpenStack

OpenStack is a collection of open source software projects that enterprises/service providers can use to setup and run their cloud compute and storage infrastructure. Rackspace and NASA are the key initial contributors to the stack. Rackspace contributed their "Cloud Files" platform (code) to power the Object Storage part of the OpenStack, while NASA contributed their "Nebula" platform (code) to power the Compute part. OpenStack consortium has managed to have more than 150 members including Canonical, Dell, Citrix etc.

There are 6 main service families under OpenStack

- Nova - Compute Service

- Swift - Storage Service
- Glance - Imaging Service
- Keystone - Identity Service
- Neutron - Networking service
- Cinder - Volume Service
- Horizon - Web UI Service

1.2.1 Open Stack Compute Infrastructure (Nova)

Nova is the Computing Fabric controller for the OpenStack Cloud. All activities needed to support the life cycle of instances within the OpenStack cloud are handled by Nova. This makes Nova a Management Platform that manages compute resources, networking, authorization, and scalability needs of the OpenStack cloud. But, Nova does not provide any virtualization capabilities by itself; instead, it uses libvirt APIs to interact with the supported hypervisors. Nova exposes all its capabilities through a web services API that is compatible with the EC2 API of Amazon Web Services.

1.2.1.1 Functions and Features:

- Instance life cycle management
- Management of compute resources
- Networking and Authorization
- REST-based API
- Asynchronous eventually consistent communication
- Hypervisor agnostic : support for Xen, XenServer/XCP, KVM, UML, VMware vSphere and Hyper-V

1.2.1.2 Components of OpenStack Compute

Nova Cloud Fabric is composed of the following major components:

- API Server (nova-api)
- Message Queue (rabbit-mq server)(TODO)
- Compute Workers (nova-compute)
- Volume Worker (nova-volume)
- Scheduler (nova-scheduler)

API Server (nova-api)

The API Server provides an interface to the outside world to interact with the cloud infrastructure. API server is the only component that the outside world uses to manage the infrastructure. The management is done through web services calls using EC2 API. The API Server then, in turn, communicates with the relevant components of the cloud infrastructure through the Message Queue. As an alternative to EC2 API, OpenStack also provides a native API called "OpenStack API".

Compute Worker (nova-compute)

Compute workers deal with instance management life cycle. they receive the requests for life cycle management via the Message Queue and carry out operations. There are several Compute Workers in a typical production cloud deployment. An instance is deployed on any of the available compute worker based on the scheduling algorithm used.

Scheduler (nova-scheduler)

The scheduler maps the nova-API calls to the appropriate openstack components. It runs as a daemon named nova-schedule and picks up a compute server from a pool of available resources depending upon the scheduling algorithm in place. A scheduler can base its decisions on various factors such as load, memory, physical distance of the availability zone, CPU architecture, etc. The nova scheduler implements a pluggable architecture.

Currently the nova-scheduler implements a few basic scheduling algorithms:

- chance: In this method, a compute host is chosen randomly across availability zones.
- availability zone: Similar to chance, but the compute host is chosen randomly from within a specified availability zone.
- simple: In this method, hosts whose load is least are chosen to run the instance. The load information may be fetched from a load balancer.

1.2.2 OpenStack Imaging Service (Glance)

OpenStack Imaging Service is a lookup and retrieval system for virtual machine images. It can be configured to use any one of the following storage backends:

- Local filesystem (default)
- OpenStack Object Store to store images
- S3 storage directly
- S3 storage with Object Store as the intermediate for S3 access.
- HTTP (read-only)

1.2.2.1 Functions and Features

- Provides imaging service

1.2.2.2 Components of Glance

- Glance-api
- Glance-registry

1.2.3 OpenStack Storage Infrastructure (Swift)

Swift provides a distributed, eventually consistent virtual object store for OpenStack. It is analogous to Amazon Web Services - Simple Storage Service (S3). Swift is capable of storing billions of objects distributed across nodes. Swift has built-in redundancy and failover management and is capable of archiving and media streaming. It is extremely scalable in terms of both size (several petabytes) and capacity (number of objects).

1.2.3.1 Functions and Features

- Storage of large number of objects
- Storage of large sized objects
- Data Redundancy
- Archival capabilities - Work with large datasets
- Data container for virtual machines and cloud apps
- Media Streaming capabilities
- Secure storage of objects
- Backup and archival
- Extreme scalability

1.2.3.2 Components of Swift

- Swift Account
- Swift Container
- Swift Object
- Swift Proxy
- The RING

1.2.3.3 Swift Proxy Server

The consumers interact with the Swift setup through the proxy server using the Swift API. The proxy server acts as a gatekeeper and recieves requests from the world. It looks up the location of the appropriate entities and routes the requests to them.

The proxy server also handles failures of entities by rerouting requests to failover entities (handoff entities)

1.2.3.4 Swift Object Server

The Object server is a blob store. It's responsibility is to handle storage, retrieval and deletion of objects stored in the local storage. Objects are typically binary files stored in the filesystem with metadata contained as extended file attributes (xattr).

Note: xattr is supported in several filesystems such as ext3, ext4, XFS, Btrfs, JFS and ReiserFS in Linux. But it is known to work best under XFS, JFS, ReiserFS, Reiser4, and ZFS. XFS is considered to be the best option.

1.2.3.5 Swift Container server

The container server lists the objects in a container. The lists are stored as SQLite files. The container server also tracks the statistics like the number of objects contained and the storage size occupied by a container.

1.2.3.6 Swift Account Server

The account server lists containers the same way a container server lists objects.

1.2.3.7 The Ring

The ring contains information about the physical location of the objects stored inside Swift. It is a virtual representation of mapping of names of entities to their real physical location. It is analogous to an indexing service that various processes use to lookup and locate the real physical location of entities within the cluster. Entities like Accounts, Containers, Objects have their own seperate rings.

1.2.4 OpenStack Identity Service (Keystone)

Keystone provides identity and access policy services for all components in the OpenStack family. It implements it’s own REST based API (Identity API). It provides authentication and authorization for all components of OpenStack including (but not limited to) Swift, Glance, Nova. Authentication verifies that a request actually comes from who it says it does. Authorization is verifying whether the authenticated user has access to the services he/she is requesting for.

Keystone provides two ways of authentication. One is username/password based and the other is token based. Apart from that, keystone provides the following services:

- Token Service (that carries authorization information about an authenticated user)
- Catalog Service (that contains a list of available services at the users’ disposal)
- Policy Service (that let’s keystone manage access to specific services by specific users or groups).

1.2.4.1 Components of Identity Service

- Endpoints - Every OpenStack service (Nova, Swift, Glance) runs on a dedicated port and on a dedicated URL(host), we call them endpoints.
- Regions - A region defines a dedicated physical location inside a data centre. In a typical cloud setup, most if not all services are distributed across data centers/servers which are also called regions
- User - A keystone authenticated user.
- Services - Each component that is being connected to or being administered via keystone can be called a service. For example, we can call Glance a keystone service.
- Role - In order to maintain restrictions as to what a particular user can do inside cloud infrastructure it is important to have a role associated.
- Tenant - A tenant is a project with all the service endpoint and a role associated to user who is member of that particular tenant.

1.2.5 OpenStack Network Service (Neutron)

Neutron is OpenStack’s networking component. It is responsible for the following

- Providing api for users to configure networking
- Provide api for OpenStack compute services
- Provide L2/L3 and security services for OpenStack Instances

Neutron is highly modular and its functionality is distributed among the several independent components. Each component except the ‘neutron-server’ is pluggable and can be replaced with another that offers the same service.

1.2.5.1 Neutron Server

The ‘neutron-server’ is the process that exposes the Neutron API to users and other OpenStack services. The ‘neutron-server’ is also responsible for storing the network state and user configuration in database for use by the other services.

1.2.5.2 Neutron OpenvSwitch Agent

The ‘neutron-openvswitch-agent’ runs on every compute node and the network node and is responsible for providing L2 connectivity between Instances running on various compute servers and the agents that offer other services. The ‘neutron-openvswitch-agent’ accepts user configuration from neutron’s config files and overlays a virtual network atop the physical(data) network.

1.2.5.3 Neutron DHCP Agent

The ‘neutron-dhcp-agent’ runs on the network node and is responsible for offering IP configuration for OpenStack Instances. The ‘neutron-dhcp-agent’ configures a ‘dnsmasq’ Instance for every subnet created in Neutron, obeying the subnet parameters like ‘Allocation Pools’ and ‘DNS Name Servers’. The ‘dnsmasq’ instance is configured to bind to an appropriate port configured by the L2 agent on the network node and offer dhcp lease addresses for the Instances it its subnet.

1.2.5.4 Neutron L3 Agent

The ‘neutron-l3-agent’ runs on the network node and is responsible for L3 connectivity, FloatingIP provisioning and enforcing security groups. The ‘neutron-l3-agent’ relies on the network nodes IP forwarding capabilities to perform routing and iptables rules to provision FloatingIP and enforce security groups. Bot the the L3 agent and DHCP agent make use of linux networking namespaces to make sure tenants are allowed to share subnet CIDR range.

1.2.5.5 Neutron Metadata Agent

The ’neutron-metadata-agent’ provides Instance metadata for all OpenStack Instances. The Instances can pull the metadata using the url ’http://169.254.169.254:80’ and use them for early Initialization, like for example

- Update the Instance’s ssh authorized keys for password less ssh login
- Create default users
- Refresh package cache and system update, etc.

1.2.6 Openstack Volume Service (Cinder)

Volume services are used for management of LVM-based instance volumes. Volume services perform volume related functions such as creation, deletion, attaching a volume to an instance, and detaching a volume from an instance. Volumes provide a way of providing persistent storage for the instances, as the root partition is non-persistent and any changes made to it are lost when an instance is terminated. When a volume is detached from an instance or when an instance, to which the volume is attached, is terminated, it retains the data that was stored on it. This data can be accessed by re-attaching the volume to the same instance or by attaching it to other instances.

Critical data in an instance must always be written to a volume, so that it can be accessed later. This typically applies to the storage needs of database servers etc.

Cinder services include

- Cinder API
- Cinder Scheduler
- Cinder Volume

1.2.6.1 Cinder API

Cinder API provides an interface to the external world other services like nova to interact with other Cinder services.

1.2.6.2 Cinder Scheduler

When there are 2 or more machines running Cinder Volume service, scheduler decides the machine in which the new volume is to be created.

1.2.6.3 Cinder Volume

In a default setup, Cinder Volume interacts with Logical Volume Manager(LVM)to create, delete and other volume related functions

1.2.7 Openstack Administrative Web-Interface (Horizon)

Horizon the web based dashboard that can be used to manage /administer OpenStack services. It can be used to manage instances and images, create keypairs, attach volumes to instances, manipulate Swift containers etc. Apart from this, dashboard even gives the user access to instance console and can connect to an instance through VNC. Overall, Horizon features the following:

- Instance Management - Create or terminate instance, view console logs and connect through VNC, Attaching volumes, etc.
- Access and Security Management - Create security groups, manage keypairs, assign floating IPs, etc.
- Flavor Management - Manage different flavors or instance virtual hardware templates.
- Image Management - Edit or delete images.
- View service catalog.
- Manage users, quotas and usage for projects.
- User Management - Create user, etc.
- Volume Management - Creating Volumes and snapshots.
- Object Store Manipulation - Create, delete containers and objects.
- Downloading environment variables for a project.

1.2.8 Message Queue (Rabbit MQ Server)

The OpenStack Cloud Controller communicates with other nova components such as the Scheduler, Network Controller, and Volume Controller via AMQP(Advanced Message Queue Protocol). Nova uses asynchronous calls for request response, with a call-back that gets triggered once a response is received. Since asynchronous communication is used, none of the user actions get stuck for long in a waiting state. This is especially true since many actions expected by the API calls such as launching an instance or uploading an image are time consuming.

1.2.9 SQL Backend (MySQL)

OpenStack services persist their configuration in database. You can use MySQL, PostgreSQL or SQLite as your database server. Depending upon your choice of database, you will need to install the necessary packages and configure the database server. In this guide we will use MySQL.

1.2.10 Software Switch (OpenvSwitch)

OpenvSwitch is a production quality, multilayer software switch. In OpenStack, OpenvSwitch is used to create virtual(software) switches, with connected ports to which the VMs attach. The OpenvSwitch is configured by Neutron in way such as to ensure VM Connectivity with each other and with the external networks.