Introduction to OpenStack and its Components

Sruthi Paramkusham
Department of Computer Science and Engineering
Opentext Technologies India Pvt Ltd Hyderabad, India

Abstract— This paper introduces overview of OpenStack and its components, good familiarity with virtualization is required as OpenStack problems requires knowledge on virtualization. This paper also introduces few commands which are useful for Installation and Configuration of OpenStack components and a brief description of Image management, Instance Management, Storage Management, Network Management etc.

Key words: OpenStack Dashboard, Virtualization, Cloud Computing, Nova, Glance, Swift, Keystone, Neutron, Cinder

I. INTRODUCTION

Cloud Computing is a computing model, where resources such as computing power, storage, network and software are abstracted and provided as services over the internet in a remotely accessible fashion. An infrastructure setup using the cloud computing model is generally referred to as the Cloud. The categories of services available on the cloud are Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).

II. WHAT IS 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 OpenStack, While NASA contributed their "Nebula" Platform (code) to power the compute part.The service families in OpenStack Icehouse includes:

1) Nova – Compute Service
2) Swift – Storage Service
3) Glance – Imaging Service
4) Keystone – Identity Service
5) Neutron – Networking Service
6) Cinder – Volume Service
7) Horizon – Web UI Service

III. COMPONENTS OF OPENSTACK

A. *OpenStack Compting Service*: Nova

Nova is the computing fabric controller for the OpenStack cloud. All activities needed to supportthe lifecycle of instances within the OpenStack cloud are handled by Nova.This makes Nova a management platform that manages compute resources, networking, authorizationand scalability needs of openstack cloud. But, Nova does not provide any virtualization capabilities by itself; instead it useslibvirt API's to interact with the supported hypervisors. Nova exposes all its capabilities through a web service API that is compatible with the EC2 API of Amazon webservices.

1) Function 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,Xen Server/XCP, KVM, UML,VMWare vSphere and Hyper-V

2) Components of OpenStack Compute:

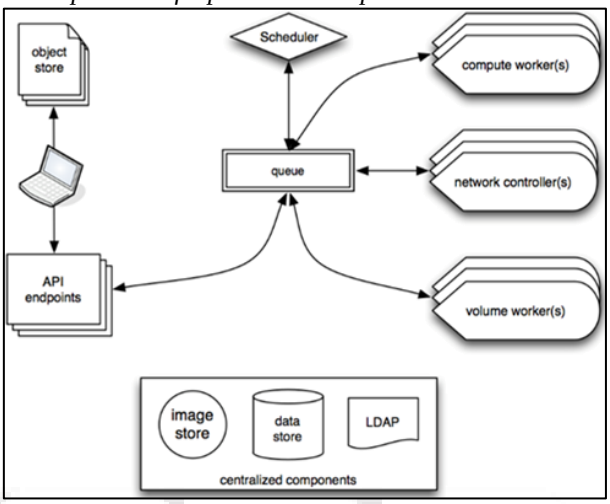

Fig. 1: Components of OpenStack Compute

3) Nova Cloud Fabric is composed of the following major components:

– API Server (nova-api)
– Compute Workers(nova-compute)
– Scheduler (nova- scheduler)

B. *OpenStack Imaging Service: Glanc*

OpenStack Imaging Service is a looku up and retrieval system for virtual machine images. It can be configured to use any one of the following atorage back ends.

– Local file system (default)
– OpenStack object store to store images
– S3 Storage directly
– S3 Storage with object store as intermediate for S3 success
– HTTP (Read-only)

1) Function and Features:

– Provides Imaging Service

2) Components of OpenStack Glance:

– Glance API
– Glance Registry

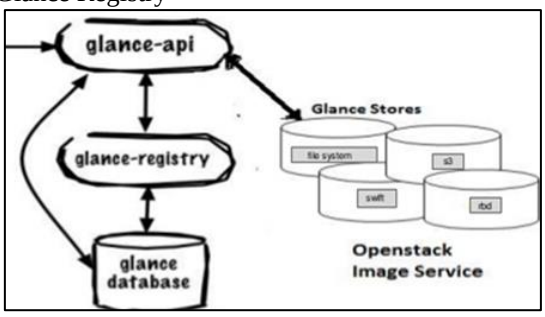

Fig. 3: OpenStack Imaging Service: Glanc

C. OpenStack Storaging Service – Swift

Swift provides a distributed, eventually consistent virtual object store for OpenStack. It is analogous to Amazon Web services Simple StorageService (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) Function and Features:

- Storage of large number of objects.
- Storage of large sized objects
- Data Redundancy
- Archival Capabilities – Working with large datasets
- Data Container for virtual machines and cloud apps
- Media Streaming Capabilities
- Secure storage of objects
- Backup and Archival
- Extreme Scalability

2) Components of Swift:

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

D. OpenStack Identity Service - KeyStone

Keystone provides identity and access policy services for all components in the OpenStack family. It implements its own REST based API (Identity API). It provides authentication and authorization for all components of OpenStack including Swift, Glance, and Nova. Authentication verifies that a request actually comes who it says it does. Authorization is verifying whether the authenticated user has the access to the service he/she is requesting for.

Keystone provides 2 ways of authentication. One is Username/Password based and the other is token based. Apart from 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 user's disposal)
- Policy Service (that let's keystone manage access to specify services by specific user's or groups)

1) Components of Identity Service

- Endpoints: Every OpenStack service (Nova, Swift, and Glance) runs on a dedicated port and on a delicate URL (host), we call them endpoints.
- Region: A Region defines a dedicated physical location inside a data center. In a typical cloud setup, most if not all services are distributed across datacenters/servers which are called as regions.
- User: A Keystone authenticated user.
- Services: Each component that is being connected to or being administered via keystone can be called as 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.

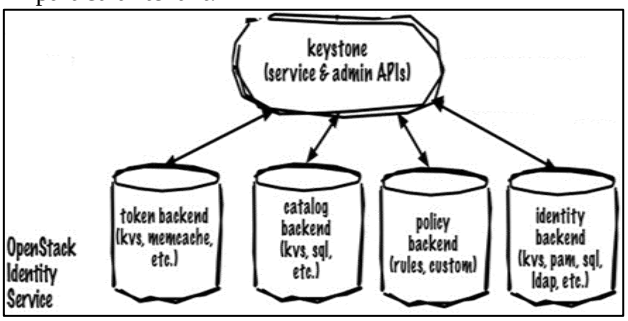

Fig. 4: Components of Identity Service

E. OpenStack Network Service – Neutron

Neutron is Open Stack's networking component. It is responsible for the following

- Provide 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.

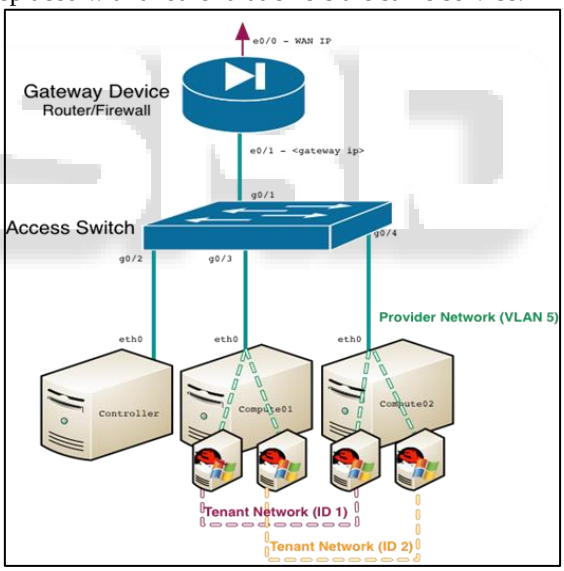

Fig. 5: OpenStack Network Service – Neutron

Of feature provided by default in RDBMS is transactions, our development methods are so used to this feature that we have stopped thinking about what would happen when the database does not provide transactions. Most NoSQL databases do not provide transaction support by default, which means the developers have to think how to implement transactions, does every write have to have the safety of transactions or can the write be segregated into "critical that they succeed" and "it's okay if I lose this write" categories. Sometimes deploying external transaction managers like Zookeeper can also be a possibility.

F. 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 provides a way of providing persistent storage for the instance, as the root partition is non-persistent and any changes made to it are lost when an instance 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 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.

1) Cinder Services include

– Cinder API
– Cinder Scheduler
– Cinder Volume

G. 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, creates key pairs, attach volumes to instances, and manipulates 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 key pairs, assign floating IP's, 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 u ser , etc.
– Volume Management: Creating volumes and snapshots.
– Object Store Manipulation – Create, delete containers and objects.
– Downloading environment variables for a project

IV. OPENSTACK SERVICES

OpenStack services can broadly be categorized into below mentioned types.

Service	Project name	Description
Orchestration	*Heat*	Orchestrates multiple composite cloud applications by using either the native *HOT* template format or the AWS CloudFormation template format, through both an OpenStack-native REST API and a CloudFormation-compatible Query API.
Database Service	*Trove*	Provides scalable and reliable Cloud Database-as-a-Service functionality for both relational and non-relational database engines.
Dashboard	*Horizon*	Provides a web-based self-service portal to interact with underlying OpenStack services, such as launching an instance, assigning IP addresses and configuring access controls.
Compute	*Nova*	Manages the lifecycle of compute instances in an OpenStack environment. Responsibilities include spawning, scheduling and decommissioning of virtual machines on demand.
Networking	*Neutron*	Enables Network-Connectivity-as-a-Service for other OpenStack services, such as OpenStack Compute. Provides an API for users to define networks and the attachments into them. Has a pluggable architecture that supports many popular networking vendors and technologies.
Storage		
Object Storage	*Swift*	Stores and retrieves arbitrary unstructured data objects via a *RESTful*, HTTP based API. It is highly fault tolerant with its data replication and scale out architecture. Its implementation is not like a file server with mountable directories.
Block Storage	*Cinder*	Provides persistent block storage to running instances. Its pluggable driver architecture facilitates the creation and management of block storage devices.
Shared services		
Identity service	*Keystone*	Provides an authentication and authorization service for other OpenStack services. Provides a catalog of endpoints for all OpenStack services.
Image Service	*Glance*	Stores and retrieves virtual machine disk images. OpenStack Compute makes use of this during instance provisioning.
Telemetry	*Ceilometer*	Monitors and meters the OpenStack cloud for billing, benchmarking, scalability, and statistical purposes.
Higher-level services		

V. CONCLUSION

OpenStack is designed to allow administrators and researchers to deploy IaaS infrastructure and provide tools for creating and managing virtual machines on top of existing resources.

This work aims to illustrate that the system OpenStack has filled an important niche in the design space of cloud computing by providing an easy to deploy over the existing Resources, easy to use in experimentation by being modular, and most importantly forms open source and provides powerful features while following emerging open standards. Currently, we deployed the entire system. This system will be used in future works to study and evaluate its performances in dynamic reconfiguration in an IaaS Cloud Computing.

REFERENCES

[1] OpenStack Beginners Guide for Ubuntu, Cloud Software
[2] B. Sotomayor, R.S. Montero, I.M. Llorente, I. Foster, Virtual infrastructure management in private and hybrid clouds IEEE Internet Comput. 13 (2009) 1422.doi:10.1109/MIC.2009.119.
[3] Citrix Systems. (2010). Xen Cloud Platform – Advanced Virtualization Infrastructure for the Clouds. URL: http://www.xen.org/products/cloudxen.html.
[4] Elastic IP and Security Groups using OpenFlow URL :http://blog.opennebula.org/?p=2695

