Private Cloud using OpenStack for Knowledge Sharing System

Abhijeet Kohakade
Information Technology
Pune Vidyarthi Griha's COET
Pune, India

Abhishek Pakhale
Information Technology
Pune Vidyarthi Griha's COET
Pune, India

Omkar Joshi
Information Technology
Pune Vidyarthi Griha's COET
Pune, India

Unmesh Deshmukh
Information Technology
Pune Vidyarthi Griha's COET
Pune, India

ABSTRACT

OpenStack is an open source cloud computing platform which is a joint venture by Rackspace and NASA. OpenStack consists of many projects such as Nova (Cpute), Swift (Storage), Neutron (Network), Glance (VM Images) etc. Swift is used as Object storage. Amazon Web Services provide way of managing cloud instances; the similar purpose can be achieved by OpenStack at virtually no cost.

System aims at developing a private cloud using OpenStack and building a website for the purpose of knowledge sharing. This system consists of firstly, building a private cloud using OpenStack for object based storage as well as compute node and secondly a website to implement knowledge sharing through the medium of web. Motivation of this system is to fill the gap between the growing technology and its use in academia. Key features for this system include abstraction of complexity, simplicity and fault tolerance of object storage and managing of cloud instances.

Keywords

Cloud, Instances, OpenStack, REST, Swift

1. INTRODUCTION

OpenStack is a software community which provides open source cloud computing platforms for public, private as well as hybrid clouds. OpenStack is an IAAS (Infrastructure as a Service) cloud computing project that is free open source project under terms and conditions of Apache License. OpenStack mission was started jointly by Rackspace and NASA Hosting in 2010 July by combining code from their cloud files and Nebula platform respectively. Basic principles on which OpenStack believes to provide cloud offering include simple, elastic, consistent, massively scalable services. All the cloud services offered by OpenStack are open source and anybody can contribute to the project.

This project targets to deliver solutions to all types of clouds (private, public as well as hybrid) and simultaneously providing simple implementation and massive scalability. This is not a single technology but set of interrelated components used to deliver solutions.

Initially OpenStack operated at 3 month release cycle but later on further stable releases were released at time intervals of 6 months. Following are the release details:

Austin - Oct, 2010

Bexar - Feb, 2011

Cactus - Apr, 2011

Diablo - Sep, 2011

Essex -Apr, 2012

Folsom - Sep, 2012

Grizzly – Apr, 2013

Havana – Oct, 2013

Icehouse – Apr,2014 [3]

With every successive release additional components are introduced. Different components of OpenStack project according to Grizzly architecture are:

OpenStack Compute (Nova)

OpenStack Compute (Nova) is used to provide and manage large networks of virtual machines. It provides computing power through virtual machine and network management on demand. It gives the software, control panels and APIs required for managing a cloud, including running instances, networks and controlled access through users and projects. Compute architecture is horizontally as well as vertically scalable and supports multi-tenancy and authentication at various levels which are key features of the project. (See figure 1)

OpenStack Image (Glance)

Glance provides discovery, storage and retrieval of virtual machine images which can also be used by Nova [3]. It provides a standard REST interface for querying information about virtual machine images stored in various backend stores including OpenStack Object Storage (Swift).It allows uploading of public as well as private images in many different formats such as Raw, VHD (Hyper-V), VDI (Virtual Box) etc.

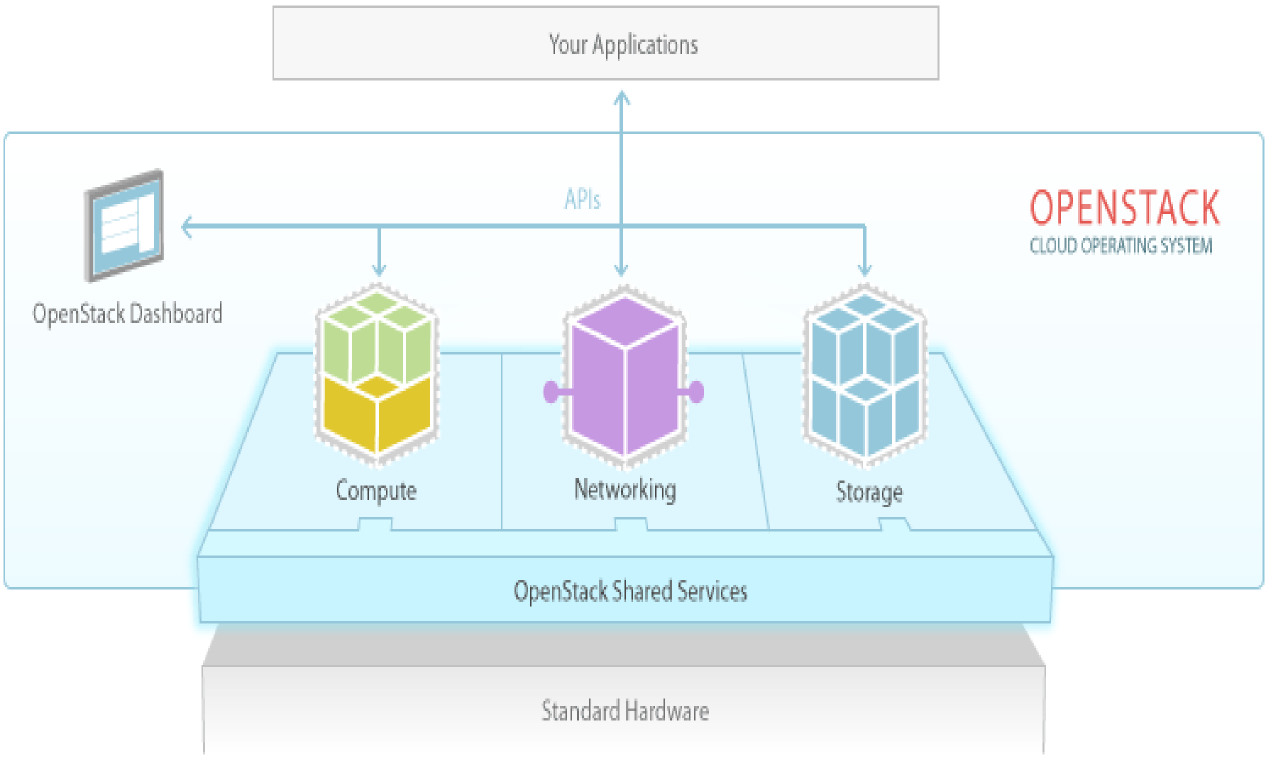

FIG1. OpenStack Basic Architecture

OpenStack Dashboard (Horizon)

Horizon was initially started as app to manage all OpenStack compute project. So initially it comprised of set of views, templates and API calls. As OpenStack projects grew, new projects were added to it and then additional features were included in Horizon as well. [3] Now, Horizon is a Django-based project aimed to provide a complete OpenStack Dashboard along with extensible framework for building new components. It provides web based interface to all OpenStack projects. The extensible design helps to expose third party services such as billing and additional management tools.

OpenStack Identity (Keystone)

Identity provides Token, Catalog and Policy services for various components of OpenStack. It provides authentication and high level authorization. It currently supports token based authentication.

OpenStack Networking (Neutron)

This OpenStack project was added in Folsom release. It is a core OpenStack project that provides network connectivity abstraction layer to other OpenStack projects. It is a pluggable, scalable and API driven system for managing IP addresses and eventually networks [3]. Like other cloud operators it can be used by administrators and users both. It allows static IP’s as well as DHCP.

OpenStack Block Storage (Cinder)

This OpenStack project was added in Folsom release. This was added to separate the existing nova block volume storage into a separate project [3]. As size of nova is growing exponentially it will be difficult in future to manage that large amount of data and add new functionality to them, therefore a new block storage project that is Cinder was started. Cinder provides “Block Storage As A Service”. It supports simple Linux server storage and many other storage platforms such as Ceph, NetApp, IBM storage etc.

OpenStack Object Storage (Swift)

In today’s world large amount of data is in unstructured format and cannot be easily stored in block storage as it will result in more wastage of memory. Due to widespread of social networking, amount of unstructured data is increasing by passing day. To store such data object storage is required. OpenStack Swift is multi-tenant, highly scalable redundant object storage system. All objects in swift have a URL by which it can be accessed. All objects are replicated 3 times in cluster. Each object has its own metadata and can be located anywhere in the distributed cluster. To interact with object RESTful HTTP transaction is necessary and all these advantages of swift make it a very useful object store at low cost.

Building blocks of Swift include Proxy Servers, Zones, Rings, Partitions, Accounts, Containers and Objects. Proxy servers are public face of Swift and handle all incoming API requests. They are also responsible for coordinating response, timestamp and handle failures. Ring maps partitions to a specific physical location on disk. Each partition in a ring is replicated thrice which can be used in case of failure. Zones in swift are configured so as to isolate failures. A zone can be a single disk or a large array of disks. As discussed earlier, any object is replicated thrice in Swift. This replication must be achieved in such a way that replicators must be from different zones and not the same zone. Account indicates an individual SQLite or MySQL database and is distributed across the cluster. An account has many containers and objects within itself. Container is similar to directory in file system and contains many objects. Objects store actual data and they are similar to files in file system. A partition is a collection of object, container, and account databases. It is a core part of replication system. [3]

Swift typically runs on Ubuntu machine 10.04 onwards. To install and run swift successfully along with swift following software needs to be installed.

- Python 2.7

Swift has its authorization system named tempAuth. One can use tempAuth for authorization or can also OpenStack keystone for authorization. Keystone has been used for authorization.

2. KNOWLEDGE SHARING SYSTEM

The aim is to inculcate collaborative learning mechanism through the use of cloud computing and storage. The object storage component, namely, OpenStack Swift provides a way of storing files and data into the containers as objects and the containers are replicated thrice over different zones. The purpose of using object storage is to enable storage of unstructured data which is a major building block of today’s web. The traditional forums are built over relational databases and hence are highly redundant. The object storage will enable the easy management of such forums as each thread will be a separate object in the forum container.

Amazon web services provide way of managing cloud instances through web access as well as secure shell (SSH). The similar purpose could be achieved by OpenStack compute and imaging services. Instances can be created, booted into and terminated on demand. This will save big need of having several platforms installed on different host machines. Also, the time required for manually installing and configuring a particular platform is high; this time can be saved by dynamically creating the bootable instances using OpenStack Glance. The instances can be Amazon Machine Images (AMI), cloud images (.img), virtual disk images (.VDI) or iso images.

The file storage of all assignments and work done in a curriculum will essentially build academic portfolio of individual students which can be used showcase their talent. (See figure 2)

3. CONFIGURATION OF CLOUD

As mentioned above OpenStack has been used to build private cloud.

The different components required for the proposed system are

- Nova (Compute)
- Keystone (Identity Service)
- Swift (Object Storage)
- Glance (Imaging Service)
- Horizon (Dashboard)

Before proceeding to the configuration of cloud, user must set up the environment to suit the needs.

First step is install the virtualization software. The preferred one by users is Oracle Virtual Box. Then it must be configured with a Linux distribution to create a sandbox environment. The advantage of using Virtual Box is that it gives us the ability to spin up virtual machines and networks without affecting rest of the working environment. The environment has necessary networking in place to allow us to access the virtual machine from host machine.

MySQL is an essential service to OpenStack as number of services relies on it. Developer needs to create a database for Nova so that the compute installation can be specified with the details. The services that need to be configured are nova-api, nova-compute, nova-network, nova-objectstore, nova-scheduler, novacert, glance-api, and glance-registry. Now with Nova configured, it helps us to accept and respond to end user compute api calls. Also it helps in running and managing instances.

User must proceed with configuration of the most important component of OpenStack, Identity Service which provides services for managing and authenticating user, account, and role information. It underpins authentication and authorization between cloud services. Keystone provides authorization token which is passed between cloud services once validated. With the help of Keystone user can create tenants which in OpenStack are projects. Also the developers can create users, which require a tenant to exist and should have a role defined to it. The developers create SERVICE_ENDPOINT which help them in binding other components to keystone.

For the object storage developer needs to proceed with the configuration of OpenStack Swift service. To use multi-node architecture, one requires different nodes running in synchronization managed by Network Time Protocol. Swift is a component which helps in fulfilling multi-node architecture. It can be configured on different environment with similar specifications to above configured environment. It is bound to keystone with the help of authorization tokens; hence it can communicate smoothly with environment consisting of nova and keystone. The system proposes single node architecture where nova and swift run on the same node. The services that need to be configured are swift, swift-proxy, swift-container, swift-object, and swift-account.

Then the next step is to proceed with the configuration of OpenStack imaging service, Glance. The developer creates a database for Glance similar to nova. OpenStack image service is split into two running services

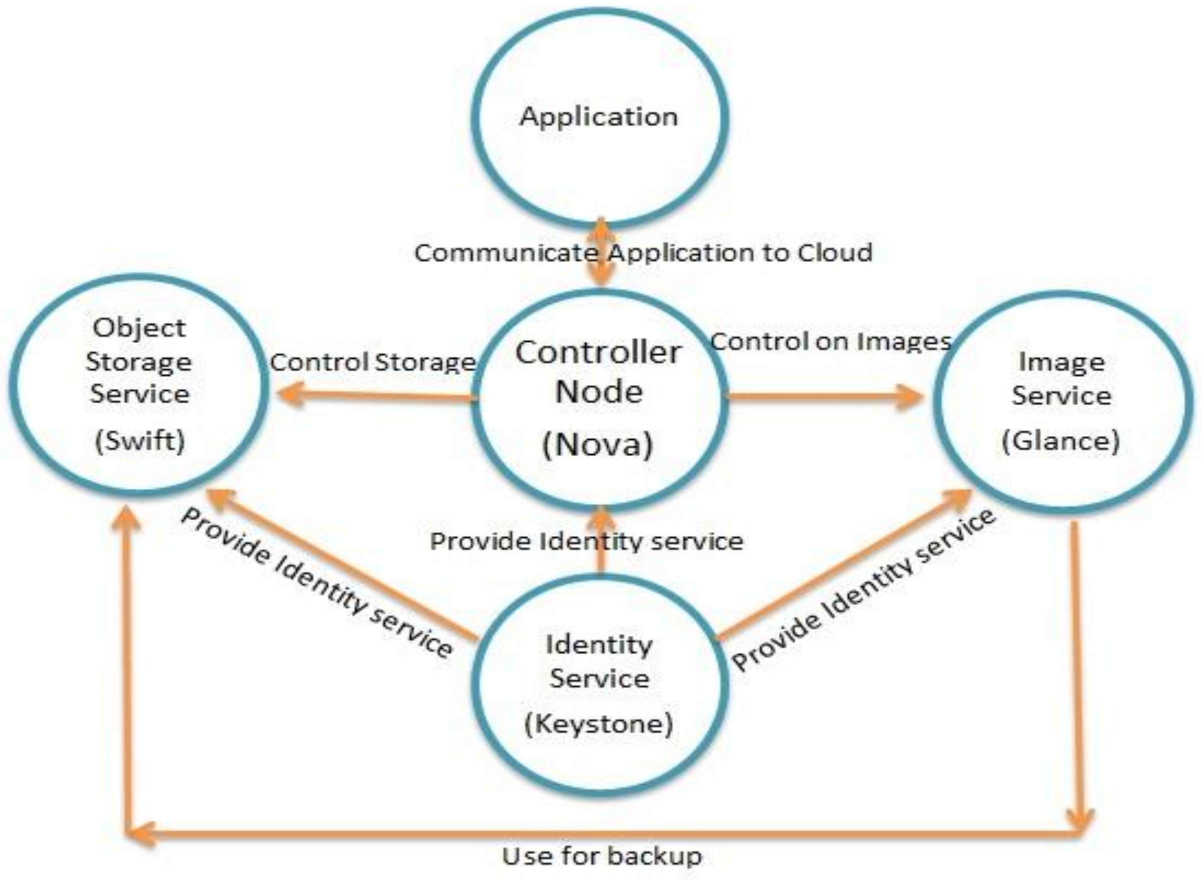

Fig2. System Architecture

Glance-api and glance-registry that connects to database created. Also developers modify the glance registry so that it knows where to connect to database.

Lastly, to complete the cloud set up, the configuration of the OpenStack dashboard, Horizon is necessary. It provides us GUI to perform the administrative task without using command prompt. The Apache2 has been configured to redirect all the traffic on port 80 to the OpenStack Horizon. The System will actually run on ephemeral port.

4. CONCLUSION

The advantages of cloud outnumber the disadvantages of it. Having cloud makes work faster and it is done on the go. Therefore to get the benefits of the cloud the system is developed. The academic institutions are having large amounts of data of students which is difficult to manage without cloud. Additionally the students are at benefit while using the system.

OpenStack is an open source project and it can be used to set up private clouds according to custom requirements and the same is proposed to be used in the academia. The system has been implemented successfully the key features for this system include abstraction of complexity, simplicity and fault tolerance of object storage and managing of cloud instances. The future scope of this system is that it can provide PAAS (Platform As A Service). That is it can provide different platforms catering to the user needs.

5. REFERENCES

[1] Mohd Zamri Murah Teaching and Learning cloud (2011) 59(2012) 157-163 doi:10.1016/j.sbspro.2012.09.260

[2] Ajit Singh, M. Hemlata Cloud for Academic Environment ISSN:2223-4985. http://www.pixelonline.net/edufuture/common/download/Paperpdf

[3] OpenStack Documentation : http://docs.openstack.org

[4] OpenStack Architecture: http://openstack.org