

WEB SERVICES USING OPENSTACK

George Philip C

Associate Professor

Department of Information Science & Engineering
M S Ramaiah Institute of Technology, Bangalore
georgephilipc@gmail.com

Pranab Kumar

Final Year BE Student

Department of Information Science & Engineering
M S Ramaiah Institute of Technology, Bangalore
pranab308@gmail.com

Manish Kumar

Final Year BE Student

Department of Information Science & Engineering
M S Ramaiah Institute of Technology, Bangalore
manish11593@gmail.com

PatanAmanullah Khan

Final Year BE Student

Department of Information Science & Engineering
M S Ramaiah Institute of Technology, Bangalore
pathanamanullakhan@yahoo.co.in

EMC²

Table of Contents

Introduction	3
Objective	4
Virtualization Strategy	4
OpenStack at a glance.....	5
Components of OpenStack	5
Compute-Nova	5
Object Storage-Swift	5
Block Storage-Cinder.....	5
Networking-Neutron.....	5
Dashboard-Horizon	6
Identity-Keystone.....	6
Image Service-Glance.....	6
Architecture of OpenStack.....	7
Interaction among Components of OpenStack	8
Architecture of the Developed System.....	8
Network Topology on Host OS.....	9
Web Application to access Swift Server.....	10
Load Balancing	12
Platform to run applications like MASM.....	13
Highlights of the System	14
Conclusions	14
Scope for Future Work.....	15
Bibliography	16
Glossary.....	18

Disclaimer: The views, processes or methodologies published in this article are those of the authors. They do not necessarily reflect EMC Corporation's views, processes or methodologies.

Introduction

The origin of the term cloud computing remains unclear. The term “cloud” is commonly used in science to represent an extensive set of objects that visually appears from a distance as a cloud and describes any set of things whose details are not examined further in assigned context. In early 1994, the cloud symbol was accepted to represent the Internet, in which servers were shown connected to the cloud externally. The term became popular in 2006 when Amazon introduced the Elastic Compute Cloud.

There are several cloud offerings today. However, there are certain challenges in the selection of any appropriate offering. The cloud contributions available in the market are, for the most part, more restrictive and often are not interoperable. It has become inconvenient for clients to migrate from one cloud to another as it involves huge volumes of data exchange and migration costs. Thus, many customers opt to stay with a provider that may not meet their requirements, just to avoid tediousness or due to lack of awareness about other cloud contributions. This approach is known as “vendor lock-in”.

Due to these restrictions in the available cloud services, there felt a need to have extensive blueprints and APIs for cloud contributions. Open Cloud Computing Interface (OCCI) emerged as a standard to try to provide a solution for this, by establishing interoperable, portable and integration measures. OpenStack began as an independent implementation of OCCI and grants a resilient cloud service. OpenStack is a framework that enables effective and productive supervision and virtualization of all the resources with great versatility and ease.

A Web Service is a standards-based, language-agnostic software entity that accepts specially formatted requests from other software entities on remote machines via vendor- and transport-neutral communication protocols producing application-specific responses. The World Wide Web is frequently being used for information interchanges between applications. The programmatic interfaces made accessible over the web for application-to-application communication are often referred to as web services. There are numerous types of applications that can be counted as web services but interoperability between applications is enhanced most by the use of well-known technologies such as XML and HTTP. These technologies allow applications using different languages and platforms to interface in a familiar way.

Objective

Cloud computing delivers benefits such as ease of use, cost-efficiency, elasticity, scalability, and so forth, and also provides a platform-independent service to the clients. This means the enterprises and individuals must be able to use the cloud outcomes and services as far as possible. The outcomes and services should work together, and there should be minimum effort to consolidate them into the user's systems. It should be feasible to write any user-specific software that might be required, which is based on commonly accessible components that can easily be acquired from multiple suppliers. Portability and interoperability measures enable development of such outcomes and services, and of user-specific software that operates with them. The availability of these outcomes and services in a free business will stimulate the outgrowth of cloud computing and the corporations that use it.

This project is an attempt to build a Cloud using the concepts of virtualization, compute, network, and storage with focus on:

- Building up a cost-effective cloud infrastructure by extending the theory of Open Source.
- Implementing services over the cloud and providing as much storage as possible from the available storage space.
- Implementation of the best Security and Network protocols to deliver secure and dynamic access to the cloud.
- The addition of Load Balancer System to handle network traffic.

Virtualization Strategy

Virtualization is a very useful concept. It allows abstraction and isolation of lower-level functionalities and underlying hardware. This enables portability of higher-level functions and sharing of the physical resources. The virtualization concept has been around since the 1960s (e.g. IBM mainframe systems). The concept has matured considerably and it has been applied to all aspects of computing - memory, storage, processors, software, networks, as well as services that IT offers. It is the combination of the growing needs and the recent advances in the IT architectures and solutions that are now bringing virtualization to the true commodity level. Virtualization, through its economy of scale and its ability to offer very advanced and complex IT services at a reasonable cost is poised to become, along with wireless and highly distributed and pervasive computing devices, such as sensors and personal cell-based access devices, the driving technology behind the next wave of IT growth. Not surprisingly there are dozens of virtualization products and a number of small and large companies that make them. Examples in the operating systems and software applications space are VMware, XEN (an open source Linux-based product developed by XenSource), and Microsoft virtualization products, to mention a few. Major IT players have also shown a renewed interest in the technology. [3] [5]

OpenStack at a glance

OpenStack is a free and open-source cloud computing software platform. Users principally deploy it as an Infrastructure as a Service (IaaS) solution. The technology consists of a series of interrelated projects that control pools of processing, storage, and networking resources throughout a data centre, which users manage through a web-based dashboard, command-line tools, or a Restful API. OpenStack.org released it under the terms of the Apache License. Amongst many architectural features of OpenStack, storage is one of the foundational cloud architecture necessities. Presenting scalable, redundant object storage, OpenStack handles clusters of servers and can store petabytes of data. Through this distributed storage system, OpenStack adds to its feature list another area of scalability, redundancy, and durability. Written to multiple disks across the data centre, data replication is managed and replication ensured. For those that are mindful of budgets, the OpenStack storage solution can write across older, smaller drives as well as newer, faster ones.[7] [1][18][9][15]

Components of OpenStack

Compute-Nova

OpenStack Compute (Nova) is a cloud computing fabric controller, which is the main part of an IaaS system. It is designed to manage and automate pools of computer resources and can work with widely available virtualization technologies, as well as bare metal and high-performance computing (HPC) configurations. KVM and XEN are available choices for hypervisor technology, together with Hyper-V and Linux container technology such as LXC. [1][3][10]

Object Storage-Swift

OpenStack Object Storage (Swift) is a scalable storage system. Swift can store files, web data, video, audio, images, backups, virtual machine snapshots, and other unstructured data on a large scale with high availability and durability. Objects are stored on multiple disk drives spread across servers in the data centre, with the OpenStack software responsible for ensuring data replication and integrity across the cluster. It can be accessed using a REST application programming interface (API) and can be scaled horizontally to store petabytes of data through the addition of nodes. When a server or hard drive fails, OpenStack replicates its content from other active nodes to a newly installed node in the cluster. Since OpenStack uses software logic to ensure data replication and distribution across different devices, inexpensive commodity hard drives and servers can be used. [1][4][6]

Block Storage-Cinder

OpenStack Block Storage (Cinder) provides persistent block-level storage devices for use with OpenStack compute instances. The block storage system manages the creation, attaching, and detaching of the block devices to servers. Block storage volumes are fully integrated into OpenStack Compute and the Dashboard allowing for cloud users to manage their own storage needs. [1] [16]

Networking-Neutron

OpenStack Networking (Neutron) is a service of OpenStack for managing networks and IP addresses. OpenStack Networking ensures the network is not a bottleneck in a cloud deployment. It gives users

self-service ability even over network configurations. OpenStack Networking provides networking models for different applications and user groups. OpenStack Networking manages IP addresses, which can be used for dedicated static IP addresses or DHCP. Floating IP addresses let the incoming requests be dynamically rerouted to any virtual resources in the IT infrastructure, so users can redirect traffic during maintenance or in the case of a failure. Users can create their own networks, control traffic, and connect servers and devices to the networks. Administrators can use Software Defined Networking (SDN) technology like OpenFlow to support high levels of multi-tenancy and on a massive scale. OpenStack Networking provides an extension framework that can deploy and manage additional network services such as Intrusion Detection Systems (IDS), load balancing, firewalls, and virtual private networks (VPN).[1] [17]

Dashboard-Horizon

OpenStack Dashboard (Horizon) provides the graphical interface API for clients as well as the administrator to access the cloud. Using this API they can manage various services such as Cinder, Swift, Glance, etc. Hence, it is one of the several ways users can interact with OpenStack. [1]

Identity-Keystone

OpenStack Identity (Keystone) provides a central directory of users mapped to the OpenStack services they can access. It acts as a common authentication system across the cloud operating system and can integrate with existing backend directory services such as Lightweight Directory Access Protocol (LDAP). It supports multiple forms of authentication including standard username and password credentials, token-based systems and AWS-style (i.e. Amazon Web Services) logins. Additionally, the catalogue provides a list of all of the services deployed in an OpenStack cloud in a single registry that can be queried. Users and third-party tools can programmatically determine which resources they can access. [1][8]

Image Service-Glance

OpenStack Image Service (Glance) provides development, registration, and delivery services for disk and server images. Stored images can be used as a template. It can also be used to store and catalogue an unlimited number of backups. The Image Service can store disk and server images in a variety of backends, including OpenStack Object Storage. The Image Service API provides a standard REST interface for querying information about disk images and lets clients stream the images to new servers. OpenStack image is an operating system installed on a virtual machine (VM). If a developer adds a variation to an image (as configuration job) the result is an instance of that image. Subsequently, that instance is an image that developers can add more variations. Glance "OpenStack's image service module" is a compute module, as it does not store images, variations, or instances—but rather catalogues them and holds their metadata from Swift or a storage backend data store. Other modules must communicate with the images' metadata through Glance. Also, Nova can present information about the images, and configure a variation on an image to produce an instance. However, Glance is the only module that can add, delete, share, or duplicate images. [1]

Architecture of OpenStack

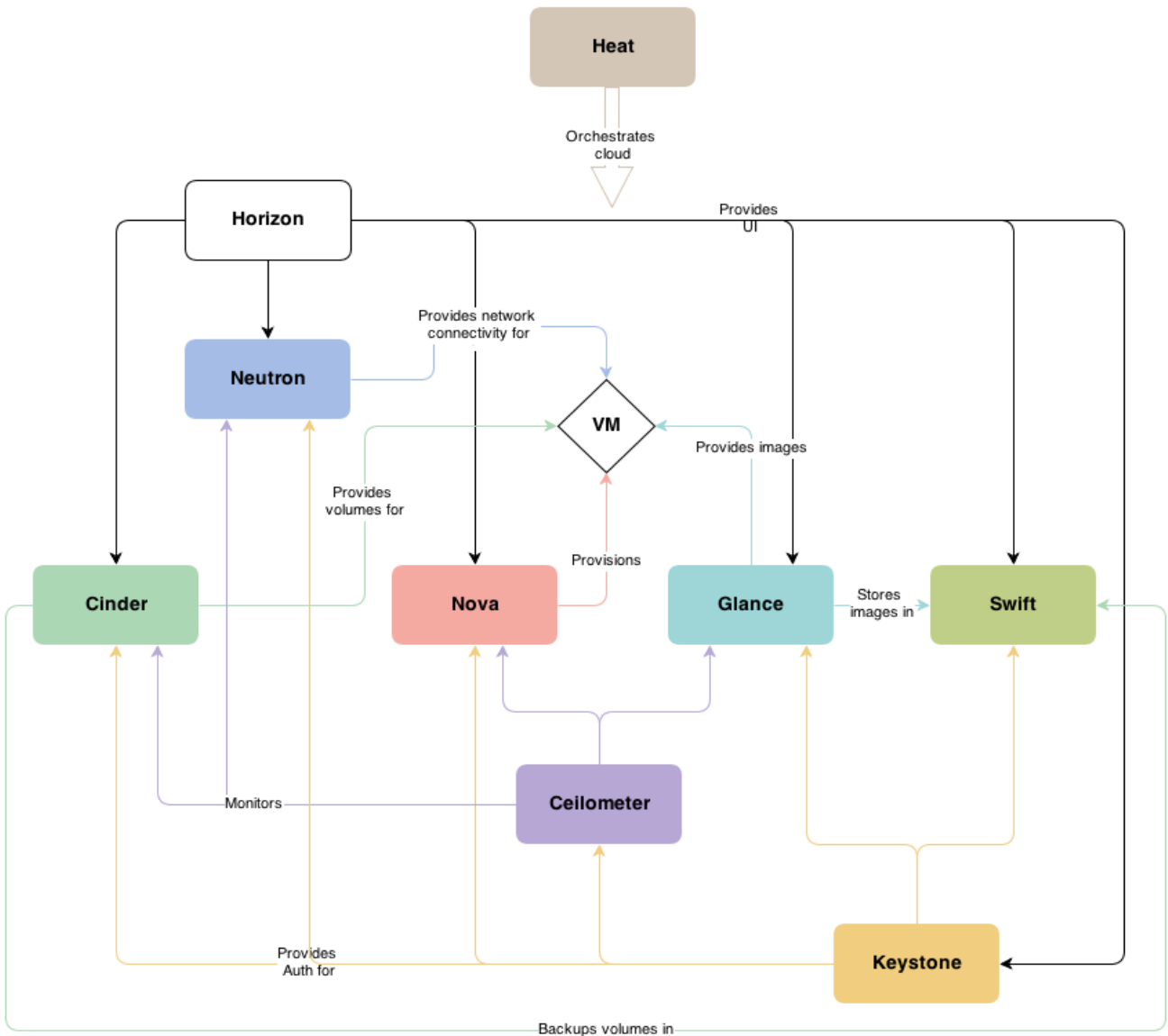


Fig 1: Architecture of OpenStack [1]

Interaction among Components of OpenStack

- Sign in through Horizon.
- Authentication of users is done by Keystone.
- When authentication is successful, Nova creates VM whenever it gets the request from Horizon.
- When users request to launch an instance, it operates under Glance service where users have to select/upload the images (Virtual Creation of Operating System).
- Later, user has to activate Neutron (OpenStack Networking) where they can set a networking protocol.
- Neutron first authenticates the token from Keystone and then checks the localrc file for the network configurations.

Architecture of the Developed System

Figure 2 shows the implementation architecture of the refined system, in which a virtual machine monitor (VMM) has been used to create a virtual environment on a host machine. A virtual machine of Ubuntu server has been created on top of this to host OpenStack and its services. The services can be accessed within the VM as well as outside the virtual machine.

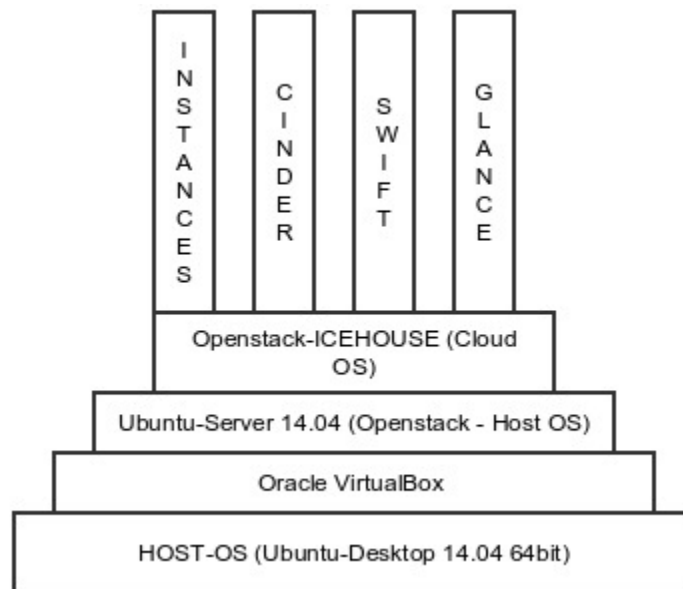


Fig 2: System Architecture

Network Topology on Host OS

Network topologies shown in Figure 3 describe the intercommunication of the host machine with VM and OpenStack server. This is the design which has been executed in the system. A bridge has been created between HOST OS and VM and a router has been added to connect with OpenStack instances from the Host machine. Two routers have been attached to the VM for Private Network within OpenStack and one for Public Network within OpenStack.

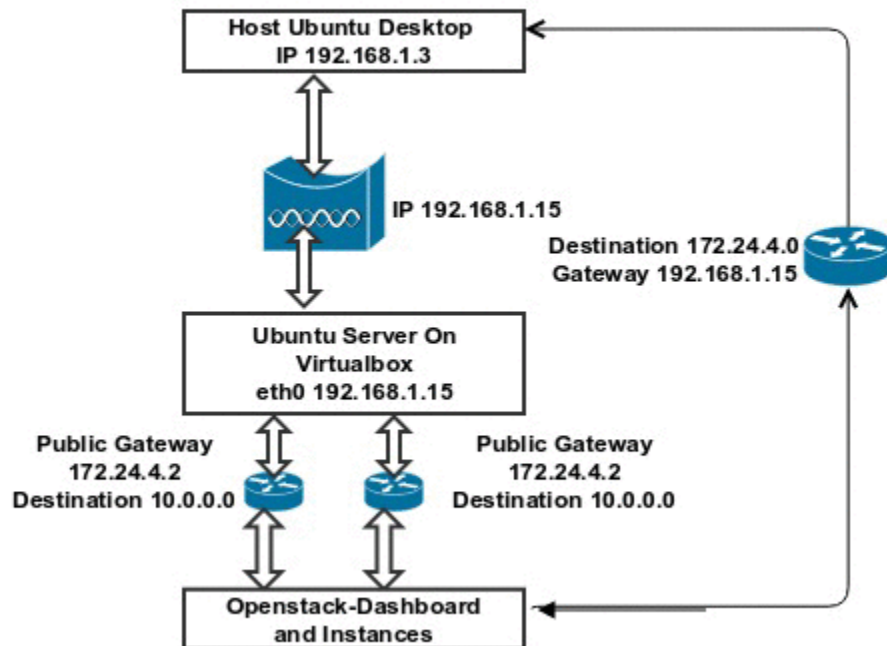


Fig 3: Network Topology on the Host Machine

Web Application to access Swift Server

OpenStack requires users to login to the Dashboard, understand the functionalities of the dashboard – and basically the whole system – to operate on it. This will be a bit difficult for general users to perform basic operations such as upload/download file/Folder and do the basic file operations over the cloud.

To deal with this situation, a simplistic web application has been designed to access OpenStack Swift. This application is useful for those users who just want to use the OpenStack for object storage and do not want to concern themselves with the details of the system. All interaction with the OpenStack Swift has been taken care of by the application. This application provides a good feel, ease of use, platform independence, and easy accessibility to the user.

A new user first needs to register on the website. All the user information is stored in the Keystone database. At the time of login, user credentials are verified from the same database. After successful login, the user is connected to the Swift server and the objects are fetched from the container.

Figure 4 illustrates the working of the web application.

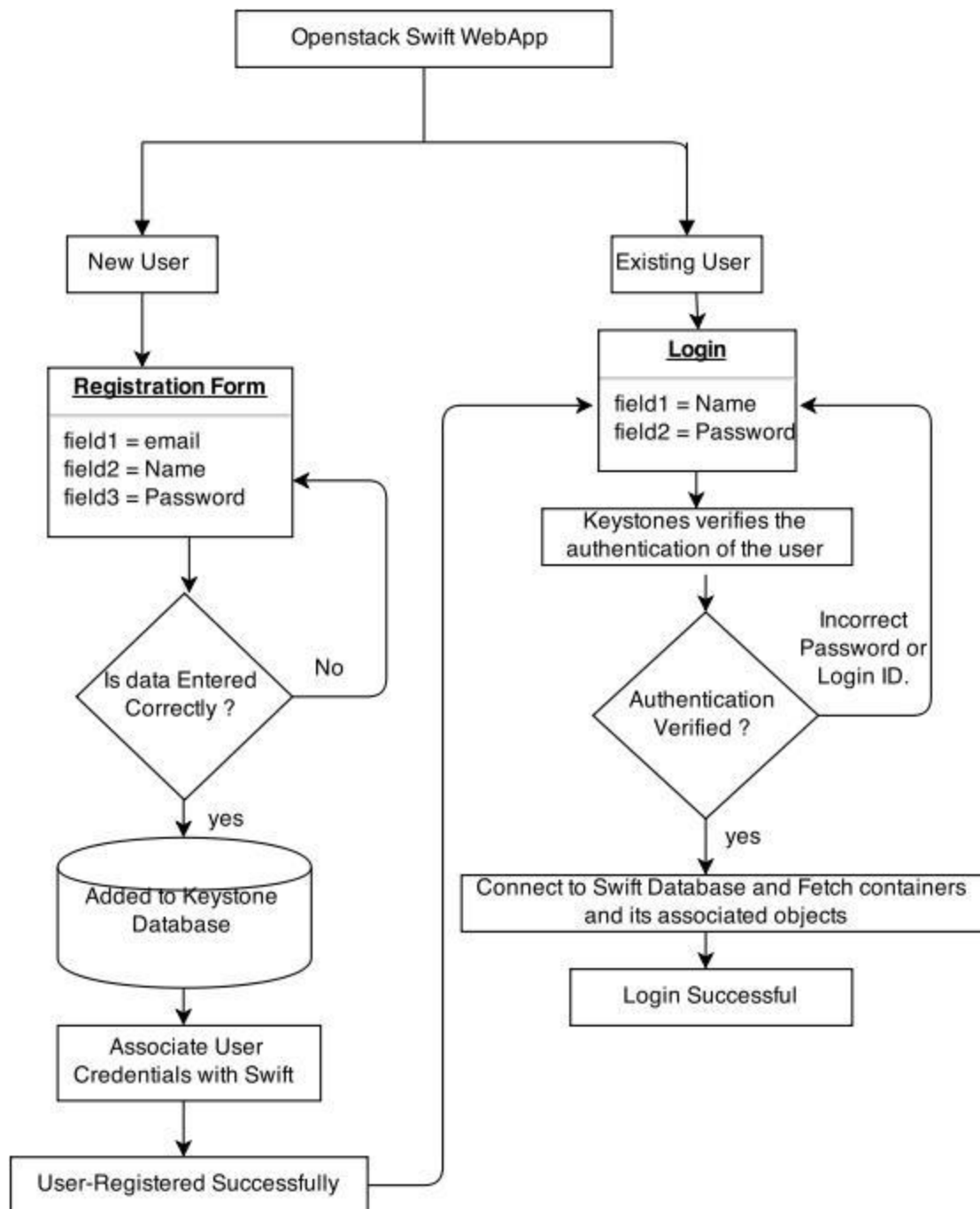


Fig.4: Flow Chart of the Web Application

The Swift server can be accessed using a simple web interface, where the file/folder hierarchy can be seen clearly. Users can upload and download data to/from the servers just by the click of a few buttons. Security mechanisms have been provided while communicating with the server. For this purpose "REST" APIs have been used.

Load Balancing

An enormous number of requests to a single server make it sluggish and strains the system. Requests are queued and the server takes more time to process a particular request. Instead of having a single server, this system has more than one server which can be easily attached and removed to/from the system depending on the demand. The concept of load balancing is incorporated to minimize the burden on a single server and minimize process time of a single request. Figure 5 shows the load balancer implemented in the system. As all the users are provided with a single public IP, there can be high traffic in the network. When multiple users are accessing the servers at the same time, the requests are distributed to the servers present in the pool.

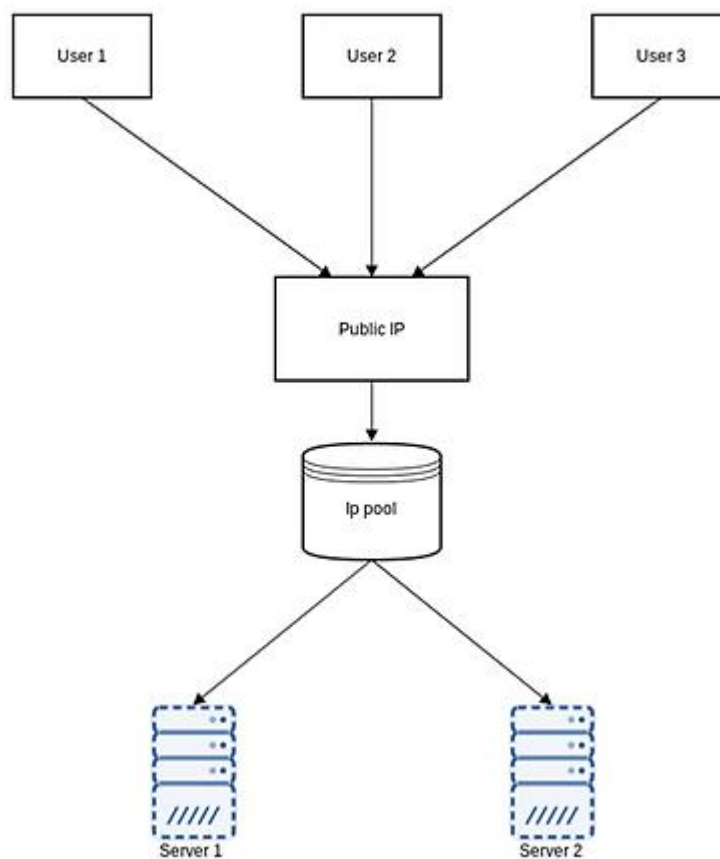


Fig 5: Flow diagram of the Load Balancer

Platform to run applications like MASM

Users of the system would be able to run the operating system on an online platform which can be accessed from everywhere just by simply inserting the URL. Figure 6 shows the DOS operating system running on the OpenStack server of the system. MASM assembler is running on the DOS operating system which is used to execute the assembly level source code.

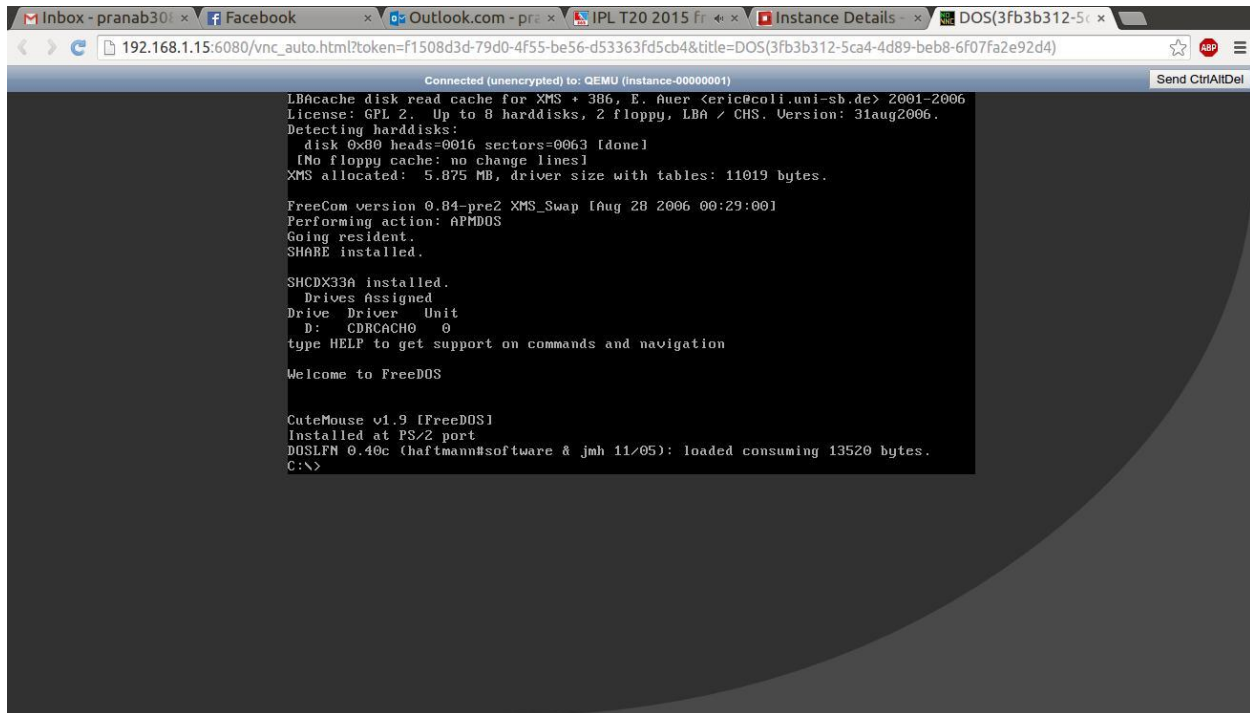


Fig 6: DOS OS Running on the System Server

Highlights of the System

- The servers running on the cloud can be accessed through the network.
- Users can host their website using the cloud.
- This system can be used to host any kind of application over a cloud.
- Simple access through the web GUIs.
- Simple uploading and downloading mechanism of data to/from the cloud.
- Cost saving in terms of hardware for the users.
- Simple tool to host servers on the available resources.
- Application Services integration.
- Sharing application within the private network.
- Snapshots and Backup APIs are available in case of failure.
- A platform provided to support a wide range of applications and various operating systems.
- Agility in terms of deploying new services to quickly respond to client demand.
- Valid customizability in terms of managing and allocating the resources.
- Reduce the server load and network congestion by launching load balancer system.

Conclusions

The development of the system described in this article was to help colleges, schools, and small organizations implement this to support their IT infrastructure. At the same time, it provides a platform for students to learn about this very new and emerging technology of cloud. As the system is a fully developed one and fascinating functionalities are available, it could be practised for the business plan after performing some adjustments and enhancements on demand.

Cloud is a limitless technology with a lot of possibilities. Everyday, something new is being added in this sea of technology. Internet of Things (IOT), Analytics, Hadoop, and other emerging technologies can be incorporated easily to this cloud to enhance the user experience and implement this system for the business purpose. Many large companies are already using OpenStack to provide services to their users.

Scope for Future Work

While this system was developed with College/School in mind, it can also be used for small-scale organizations. Since Cloud is an emerging technology, a lot of enhancements can be done to this system. Here are a few:

- A simple mobile application can be developed for the administrator or users to manage the cloud resources, which will work in Android, Windows, or Mac environments.
- This system can further be developed and implemented to work for individual departments as well as for the whole institutional IT infrastructure.
- Real-time monitoring of each department (in terms of the need for the servers and storage space) and students (in terms of the work or task assigned to them) can be achieved.
- This system can be tweaked for business establishments.
- SAN and NAS infrastructure can be attached to provide more space.
- A user interface can be created for the platform using PaaS which users can efficiently run their servers.
- The web interface can be modified and platforms supporting multiple languages (JAVA, C++, PYTHON, C#, etc.) can be provided, enabling users to develop applications or just test their programs.

Finally, this system can guide the college or department in establishing their own cloud platform, and thereby help students learn this technology. Utilizing this technology, the college will also be able to purchase and provide access of costly software to the students free-of-cost on the cloud. Students who can't complete their tasks in a lab session can access the platform anytime from anywhere and complete their lab exercises.

Bibliography

- [1] OpenStack manual, Feb. 2015.
- [2] S. A. Baset. Open source cloud technologies. In Proceedings of the Third ACM Symposium on Cloud Computing, SoCC '12, pages 28:1–28:2, New York, NY, USA, 2012. ACM.
- [3] M. Bist, M. Wariya, and A. Agarwal. Comparing delta, OpenStack and Xen cloud platforms: A survey on open source IaaS. In Advance Computing Conference (IACC), 2013 IEEE 3rd International, pages 96–100. IEEE, 2013.
- [4] Z. Y. Duan and Y. Z. Cao. The implementation of cloud storage system based on OpenStack swift. *Applied Mechanics and Materials*, 644:2981–2984, 2014.
- [5] L. Girish and H. Guruprasad. Building private cloud using OpenStack. *International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)*, 3(3), 2014.
- [6] M. Goossens, F. Mittelbach, and A. Samarin. *OpenStack Beginner's Guide*. CSS-CORP, India, 2012.
- [7] P. R. Gupta, S. Taneja, and A. Datt. Using heat and ceilometer for providing autoscaling in OpenStack.
- [8] J.-M. Kim, H.-Y. Jeong, I. Cho, S. M. Kang, and J. H. Park. A secure smart-work service model based OpenStack for cloud computing. *Cluster Computing*, 17(3):691–702, 2014.
- [9] R. Kumar and B. B. Parashar. Dynamic resource allocation and management using OpenStack. *Nova*, 1:21, 2010.
- [10] I. Kureshi, C. Pulley, J. Brennan, V. Holmes, S. Bonner, and Y. James. Advancing research infrastructure using OpenStack. *International Journal of Advanced Computer Science and Applications*, 3(4):64–70, 2013.
- [11] I. M. Llorente and R. S. Montero. *Opennebula*.
- [12] D. Milojevic, I. M. Llorente, and R. S. Montero. *Opennebula: A cloud management tool*. *IEEE Internet Computing*, 15(2):11–14, 2011.
- [13] J. Murty. *Programming Amazon Web Services*. O'Reilly, first edition, 2008.
- [14] D. Nurmi, R. Wolski, C. Grzegorzczak, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov. The eucalyptus open-source cloud-computing system. In Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, CCGRID '09, pages 124–131, Washington, DC, USA, 2009. IEEE Computer Society.
- [15] O. Sefraoui, M. Aissaoui, and M. Eleuldj. OpenStack: toward an open-source solution for cloud computing. *International Journal of Computer Applications*, 55(3):38–42, 2012.

- [16] B. Soudan, S. Edalatpanah, D. V. Sharma, M. S. A. Mahmoud, A. El-Moursy, A. A. A. Wahab, M. F. M. Salleh, G. Sadashivappa, K. Chaturvedi, H.-f. S. Lee, et al. International journal site. Journal Publication, 4(1), 2015.
- [17] W. Xia. Discussion on the OpenStack room management under the network environment [j].Sci-Tech Information Development & Economy, 25:032, 2007.
- [18] S. Yadav. Comparative study on open source software for cloud computing platform: Eucalyptus, OpenStack and open nebula. International Journal Of Engineering And Science, 3(10):51–54, 2013.

Glossary

Block storage- Typically a type of storage where data are stored in Volume. Here, Cinder is a block storage device.

Bridge network - Type of network topologies where we can connect one LAN network with another LAN network that uses the same protocol.

Compute Node - The workhorse of a cloud system and the place where users' applications will run.

Dashboard - A GUI-based API from where admins/users can analyse their server.

Disk File System - File System which manages data on block storage device, Cinder.

Ephemeral Disk - The physical disk which is attached to any instances.

Ext3 - Journal file system commonly used by Linux kernel.

FAT32 - File System commonly used in earlier versions of Windows OS.

QEMU - An open source machine emulator and virtualizer.

GIT - A distributed revision control system with an emphasis on speed, data integrity, and support for distributed, non-linear workflows.

Glance - Glance service provides different kinds of images of different OS.

Hypervisor - Computer software that creates and run the virtual machines. It is also known as Virtual Machine Monitor.

Internet Protocol - The set of techniques employed for transmitting data over the Internet.

Iptables - user-space application program that allows a system administrator to configure the tables provided by the Linux kernel firewall and the chains and rules it stores.

Key-Pair - A set of Private Key and Public Key in a cryptographic algorithm.

Masquerade - A masquerade attack is an attack that uses a fake identity, such as a network identity, to gain unauthorized access to personal computer information through legitimate access identification.

NAT - Network address translator is used to remapping of IP addresses.

NTFS - File System widely used in recent versions of Windows OS.

Object Storage- Approaches to handling discrete unit of data storage.

Ping - A command used to check the connectivity of server.

Platform as a Service (PaaS) - A type of cloud where clients have the platform to deploy and test their own applications.

Port Forwarding - Port forwarding or port mapping is an application of network address translation (NAT).

Private Key - In cryptography, a private or secret key is an encryption/decryption key known only to the party or parties that exchange secret messages.

Public Key - In cryptography, a public key is a value provided by some designated authority as an encryption key that, combined with a private key, redirects a communication request from one address and port number combination to another while the packets are traversing a network gateway, such as a router or firewall.

REST - Representational State Transfer, a software architecture style for creating scalable web services.

Root Disk - The disk which holds the main setup program.

Router - A device that forwards data packets along networks.

SSH - Secure Shell (SSH) is a cryptographic network protocol for secure data communication, remote command-line login, remote command execution, and other secure network services between two networked computers.

Subnet - Logical division of IP addresses. Subnet Mask - A 32-bit number that masks an IP address, and divides the IP address into a network address and host address. Subnet Mask is made by setting network bits to all "1"s and setting host bits to all "0"s.

Tenant- The owner of the particular project.

Transmission Control Protocol (TCP) - A connection-oriented protocol that provides reliable, full-duplex byte streams to its users. In essence, TCP sockets are an example of Stream sockets.

VCPUs- Number of physical CPU assigned to Virtual Machine.

VLAN - In computer networking, a single Layer 2 network may be partitioned to create multiple distinct broadcast domains, which are mutually isolated so that packets can only pass between them via one or more routers; such a domain is referred to as a virtual local area network, virtual LAN, or VLAN.

EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS PROVIDED “AS IS.” EMC CORPORATION MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.