Design and Implementation of a Cloud Based Virtual Lab

A research project submitted in partial fulfillment of the requirements for the Degree of Bachelor of Science (Engineering) in Computer Science and Engineering

**Submitted By**

Mirza Inkiad Ahmed Prangon

CE 13015

Tammum Islam

CE 13030

Session: 2012-2013

**Supervised By**

Professor Dr. Md. Motiur Rahman



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

MAWLANA BHASHANI SCIENCE AND TECHNOLOGY UNIVERSITY

SANTOSH, TANGAIL-1902

BANGLADESH

OCTOBER 2018

# DECLARATION

We hereby declare that the thesis is form student’s own work and effort under the supervision of Dr. Md. MotiurRahman, Professor, Department of Computer Science & Engineering (CSE), Mawlana Bhashani Science and Technology University, Santosh, Tangail-1902, Bangladesh. We also declare that the content in this thesis is original and has not been submitted previously for a degree at this or any other university. To the best of our knowledge and belief, this thesis contains no material previously published or written by any other person, except where due reference is made.

|  |  |
| --- | --- |
| Countersigned | Signature |
| --------------------------------------------  Dr. MD. Motiur Rahman  Professor  Department of Computer Science & Engineering  Mawlana Bhashani Science and Technology University  Santosh, Tangail-1902, Bangladesh  Supervisor | Mirza Inkiad Ahmed Prangon  CE13015  Tammum Islam  CE13030 |

# ACKNOWLEDGEMENTS

By the grace of Almighty, we would like to thank our supervisor Dr. Md. MotiurRahman, Professor, Department of Computer Science and Engineering for his valuable guidance and insight, encouragement, support and reliance throughout research.

We are grateful to all of my teachers of the Dept. of CSE, Mawlana Bhashani Science and Technology University.

We would like to thank all members of our department staff and we are also grateful to our friends for their help and support during completion of our research project.

Finally, we like to express a special thanks to our parents because of their hard contribution in financial support and great love.

# Abstract

Virtual machine is a logical machine that behaves exactly like a physical machine. Cloud system is a system that provides computational service remotely form another machine located in different physical location than user. This research project is to create a virtual lab environment for use in university computer lab. Students and teachers will be able to use virtual machines as personal computers. Environment is provided through private cloud system. Existing systems maintenance is hard and time consuming. Our project provides easy maintenance and provides data accessibility. Proper user role, data accessibility and easy maintenance provides an enjoyable computing experience.

# Contents

[DECLARATION ii](#_Toc529121707)

[ACKNOWLEDGEMENTS iii](#_Toc529121708)

[Abstract iv](#_Toc529121709)

[Contents v](#_Toc529121710)

[List of Figures vii](#_Toc529121711)

[Chapter 1 Introduction 1](#_Toc529121712)

[1.1 Project Introduction 1](#_Toc529121713)

[1.2 Project Purpose 1](#_Toc529121714)

[1.3 Project Scope 1](#_Toc529121715)

[1.4 Project Objective/Goal 1](#_Toc529121716)

[Chapter 2 Cloud computing 3](#_Toc529121717)

[2.1 Cloud Computing 3](#_Toc529121718)

[2.2 Virtualization 3](#_Toc529121719)

[2.3 Cloud Computing Models 5](#_Toc529121720)

[2.3.1 Infrastructure as a Service (IaaS) 5](#_Toc529121721)

[2.3.2 Platform as a Service (PaaS) 5](#_Toc529121722)

[2.3.3 Software as a Service (SaaS) 5](#_Toc529121723)

[2.3.4 Mobile Backend as a Service (MBaaS) 6](#_Toc529121724)

[2.3.5 Function as a Service (FaaS) 6](#_Toc529121725)

[2.3.6 Serverless Computing 6](#_Toc529121726)

[2.4 Private Cloud 7](#_Toc529121727)

[2.5 Public Cloud 7](#_Toc529121728)

[2.6 Hybrid Cloud 8](#_Toc529121729)

[2.7 Other 8](#_Toc529121730)

[2.7.1 Community Cloud 8](#_Toc529121731)

[2.7.2 Distributed Cloud 8](#_Toc529121732)

[2.7.3 Multicloud 8](#_Toc529121733)

[2.7.4 Big Data Cloud 8](#_Toc529121734)

[2.7.5 HPC Cloud 9](#_Toc529121735)

[Chapter 3 Existing System 10](#_Toc529121736)

[3.1 Commercial Existing Services 10](#_Toc529121737)

[3.1.1 Amazon Web Services 10](#_Toc529121738)

[3.1.2 Microsoft Azure 11](#_Toc529121739)

[3.1.3 Google Cloud 12](#_Toc529121740)

[Chapter 4 Proposed model 14](#_Toc529121741)

[4.1 Project Perspective 14](#_Toc529121742)

[4.2 Project Architecture 15](#_Toc529121743)

[4.3 Storage Architecture 16](#_Toc529121744)

[4.4 Use Case Diagram 17](#_Toc529121745)

[4.5 Diagram of virtual lab environment 19](#_Toc529121746)

[4.6 Data Flow Diagram 20](#_Toc529121747)

[4.7 Entity Relationship Diagram 21](#_Toc529121748)

[Chapter 5 Project Implementation 22](#_Toc529121749)

[5.1 Project Requirements 22](#_Toc529121750)

[5.1.1 Hardware Requirement 22](#_Toc529121751)

[5.1.2 Software Requirement 22](#_Toc529121752)

[5.2 Deployment Tool Used 22](#_Toc529121753)

[5.2.1 Openstack 22](#_Toc529121754)

[5.2.2 Packstack 23](#_Toc529121755)

[5.3 Project Features and Functionality 24](#_Toc529121756)

[5.3.1 Image Creation 24](#_Toc529121757)

[5.3.2 Launching Instances 24](#_Toc529121758)

[5.3.3 Accessing Instances 24](#_Toc529121759)

[5.3.4 Accessing Instances Through Secure Shell (SSH) 24](#_Toc529121760)

[5.3.5 Creating Snapshot of Instances 25](#_Toc529121761)

[5.4 Project Screenshots 25](#_Toc529121762)

[5.4.1 Accessing the Openstack login page 25](#_Toc529121763)

[5.4.2 Creating Users 27](#_Toc529121764)

[5.4.3 Image Uploading 30](#_Toc529121765)

[5.4.4 Launching Instances 31](#_Toc529121766)

[5.4.5 Launching GUI Based Operating System 32](#_Toc529121767)

[5.4.6 Launching Command Based Operating System 33](#_Toc529121768)

[5.4.7 Using Putty for Remote Access 34](#_Toc529121769)

[Chapter 6 Conclusion 35](#_Toc529121770)

[Chapter 7 References 36](#_Toc529121771)

# List of Figures

[Figure 2‑1 Simple Architecture of Cloud 4](#_Toc529130695)

[Figure 2‑2 Cloud Computing Service Model 6](#_Toc529130696)

[Figure 3‑1 Amazon Web Services 11](file:///C:\Users\WhiteRidge\Desktop\Book2.docx#_Toc529130697)

[Figure 3‑2 Microsoft Azure Services 12](file:///C:\Users\WhiteRidge\Desktop\Book2.docx#_Toc529130698)

[Figure 3‑3 Google Cloud Services 13](file:///C:\Users\WhiteRidge\Desktop\Book2.docx#_Toc529130699)

[Figure 4‑1 Proposed Model Architecture 15](file:///C:\Users\WhiteRidge\Desktop\Book2.docx#_Toc529130700)

[Figure 4‑2 Storage Architecture 16](#_Toc529130701)

[Figure 4‑3- Use Case Diagram 17](#_Toc529130702)

[Figure 4‑4- Virtual lab environment system 19](#_Toc529130703)

[Figure 4‑5 Data Flow Diagram of the Proposed Model 20](#_Toc529130704)

[Figure 4‑6 Entity Relationship Diagram 21](#_Toc529130705)

[Figure 5‑1 Image Creation 24](#_Toc529130706)

[Figure 5‑2 Launching Instances 25](#_Toc529130707)

[Figure 5‑3 Accessing Instance through Web Interface and SSH 25](#_Toc529130708)

[Figure 5‑4 Creating Snapshots of Instances 25](#_Toc529130709)

[Figure 5‑5 Openstack Login Page 26](#_Toc529130710)

[Figure 5‑6 Creating Users 27](#_Toc529130711)

[Figure 5‑7 Configuring the Network 28](#_Toc529130712)

[Figure 19 Network Topology 29](file:///C:\Users\WhiteRidge\Desktop\Book2.docx#_Toc529130713)

[Figure 5‑9 Image Uploading 30](#_Toc529130714)

[Figure 5‑10 Determining Flavors for the image to launch 31](#_Toc529130715)

[Figure 5‑11 Launching GUI based Operating System 32](#_Toc529130716)

[Figure 5‑12 Accessing Ubuntu Operating System 33](#_Toc529130717)

[Figure 5‑13 Using Putty for Remote Login 34](#_Toc529130718)

# Introduction

## Project Introduction

We propose to move our whole lab environment to a Cloud System. We have built a cloud system which will cater to our need. Our project is Designing and implementing a cloud based virtual lab environment. We have designed a proper cloud system for the lab need of university. In this cloud, the main machines are virtual instances. Virtual machines do not need any physical maintenance, is easy to create and can be deleted in an instant when not needed. All the hardware maintenance is concentrated at a single physical location for easy maintenance and upgrade.

## Project Purpose

The purpose of the project is to mitigate the existing problems in the lab environment. Also to make the system easier to maintain and use. To ease moving to cloud system we must make the interface as easy to use as possible. Required software will be easy to add in a machine. As a result, students will able to use their time in actual use cases like coding instead of troubleshooting software problems. The whole system is in virtual environment, so spinning up a new machine with needed requirements will take only minutes.

## Project Scope

For on premise use we need private cloud system. The project is primarily deploying a private cloud and designing required system. Project scope is selecting private cloud infrastructure, collect physical server which meets minimum hardware requirement, deploy cloud, make proper virtual machine images. This project provides –

* Virtual lab environment for lab.
* Cloud system to run virtual machines.
* Configured images to start virtual machines with.
* Allocating machines to individual students.
* Various images providing different OS to boot from.
* BASH scripting to deploy the private cloud.

## Project Objective/Goal

Main goal of the project is to reduce time waste in managing machines and installing required software in computers.

* Make it easy to deploy multiple virtual machines with same configuration.
* Have total power over the computational ability of each machines.
* Make maintenance of hardware and software painless.
* Remove software installation redundancy in many physical machines.
* Have the ability to allocate individual virtual machines to each individual student enabling option to retain student data.
* Give student enough amount of persistent storage so that they can keep their work on server to access at later date, time or class.

# Cloud computing

## Cloud Computing

Cloud computing refers to applications and services that run on distributed network using virtualized resources and accessed by com-mon Internet protocols and networking standards. It is distinguished the notion that resources are virtual and limitless and that details of the physical systems on which software runs are abstracted from the user. In an effort to better describe cloud computing, a number of cloud types have been defined.[[1]](#endnote-1)

Cloud computing represents a real paradigm shift in the way in which systems are deployed. The massive scale of cloud computing systems was enabled by the popularization of the Internet and the growth of some large service companies.[[2]](#endnote-2) Cloud computing makes the long-held dream of utility computing possible with a pay-as-you-go, infinitely scalable, universally available system. With cloud computing, you can start very small and become big very fast. That’s why cloud computing is revolutionary, even if the technology it is built on is evolutionary. Not all applications benefit from deployment in the cloud. Issues with latency, transaction control, and in particular security and regulatory compliance are of particular concern.

Cloud computing takes the technology, services, and applications that are similar to those on the Internet and turns them into a self-service utility. The use of the word “cloud” makes reference to the two essential concepts:

1. Abstraction: Cloud computing abstracts the details of system implementation from users and developers. Applications run on physical systems that aren’t specified, data is stored in locations that are unknown, administration of systems is outsourced to others, and access by users is ubiquitous.
2. Virtualization: Cloud computing virtualizes systems by pooling and sharing resources. Systems and storage can be provisioned as needed from a centralized infrastructure, costs are assessed on a metered basis, multi-tenancy is enabled, and resources are scalable with agility.

## Virtualization

Virtualization in computing often refers to the abstraction of some physical component into a logical object. By virtualizing an object, you can obtain some greater measure of utility from the resource the object provides. For example, virtual LANs (local area networks), or VLANs, provide greater network performance and improved manageability by being separated from the physical hardware. Likewise, storage area networks (SANs) provide greater flexibility, improved availability, and more efficient use of storage resources by abstracting the physical devices into logical objects that can be quickly and easily manipulated.[[3]](#endnote-3) Our focus, however, will be on the virtualization of entire computers. We have to be aware that virtualization is different from virtual reality—the technology that, through the use of sophisticated visual projection and sensory feedback, can give a person the experience of actually being in that created environment. But at the fundamental level, this is exactly what computer virtualization is all about: it is how a computer application experiences its created environment.

The first mainstream virtualization was done on IBM mainframes in the 1960s, but Gerald J. Popek and Robert P. Goldberg codifid the framework that describes the requirements for a computer system to support virtualization. Their 1974 article “Formal Requirements for Virtualizable Third Generation Architectures” describes the roles and properties of virtual machines and virtual machine monitors that we still use today.[[4]](#endnote-4)

By their definition, a virtual machine (VM) can virtualize all of the hardware resources, including processors, memory, storage, and network connectivity.[[5]](#endnote-5) A virtual machine monitor (VMM), which today is commonly called a hypervisor, is the software that provides the environment in which the VMs operate. Virtualization exhibits three properties in order to correctly satisfy their definition:

**Fidelity** The environment it creates for the VM is essentially identical to the original (hardware) physical machine.

**Isolation or Safety** The VMM must have complete control of the system resources.

**Performance** There should be little or no difference in performance between the VM and a physical equivalent. Because most VMMs have the first two properties, VMMs that also meet the final criterion are considered efficient VMMs.

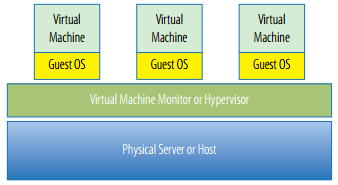


Figure 2‑1 Simple Architecture of Cloud

## Cloud Computing Models

Cloud computing has many different models depending on implementation and scale.[[6]](#endnote-6)

### Infrastructure as a Service (IaaS)

Infrastructure- and Hardware-as-a-Service (IaaS/HaaS) solutions are the most popular and developed market segment of cloud computing.[[7]](#endnote-7) They deliver customizable infrastructure on demand. The available options within the IaaS offering umbrella range from single servers to entire infrastructures, including network devices, load balancers, and database and Web servers. The main technology used to deliver and implement these solutions is hardware virtualization: one or more virtual machines opportunely configured and interconnected define the distributed system on top of which applications are installed and deployed.[[8]](#endnote-8)

### Platform as a Service (PaaS)

Platform-as-a-Service (PaaS) solutions provide a development and deployment platform for running applications in the cloud.[[9]](#endnote-9) They constitute the middleware on top of which applications are built. Application management is the core functionality of the middleware. PaaS implementations provide applications with a runtime environment and do not expose any service for managing the underlying infrastructure.[[10]](#endnote-10) They automate the process of deploying applications to the infrastructure, configuring application components, provisioning and configuring supporting technologies such as load balancer s and databases, and managing system change based on policies set by the user. Developers design their systems in terms of applications and are not concerned with hardware (physical or virtual), operating systems, and other low-level services.

### Software as a Service (SaaS)

Software-as- a-Service (SaaS) is a software delivery mode that provides access to applications

through the Internet as a Web-based service. It provides a means to free users from complex hardware and software management by offloading such tasks to third parties, which build applications accessible to multiple users through a Web browser. In this scenario, customers neither need install anything on their premises nor have to pay consider able up-front costs to purchase the software and the required licenses.[[11]](#endnote-11) They simply access the application web site, enter their credentials and billing details, and can instantly use the application, which, in most of the cases, can be further customized for their needs. On the provider side, the specific details and features of each customer’s application are maintained in the infrastructure and made available on demand.[[12]](#endnote-12)

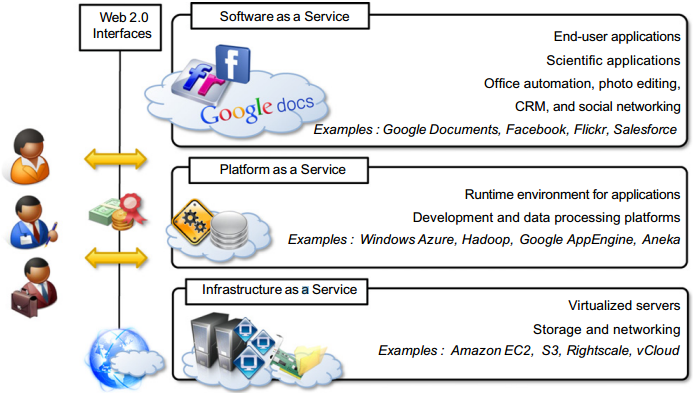


Figure 2‑2 Cloud Computing Service Model

### Mobile Backend as a Service (MBaaS)

Mobile backend as a service (MBaaS), also known as "backend as a service" (BaaS), is a model for providing web app and mobile app developers with a way to link their applications to backend cloud storage and APIs exposed by back end applications while also providing features such as user management, push notifications, and integration with social networking services. These services are provided via the use of custom software development kits (SDKs) and application programming interfaces (APIs). [[13]](#endnote-13)

### Function as a Service (FaaS)

Function as a service (FaaS) is a category of cloud computing services that provides a platform allowing customers to develop, run, and manage application functionalities without the complexity of building and maintaining the infrastructure typically associated with developing and launching an app. Building an application following this model is one way of achieving a "server less" architecture, and is typically used when building micro services applications. FaaS is an extremely recent development in cloud computing, first made available to the world by hook.io in October 2014, followed by AWS Lambda, Google Cloud Functions, Microsoft Azure Functions, IBM/Apache's Open Whisk (open source) in 2016 and Oracle Cloud Fn (open source) in 2017 which are available for public use.[[14]](#endnote-14) FaaS capabilities also exist in private platforms, as demonstrated by Uber's Schema less triggers.

### Serverless Computing

Serverless computing is a cloud-computing execution model in which the cloud provider acts as the server, dynamically managing the allocation of machine resources. [[15]](#endnote-15)Pricing is based on the actual amount of resources consumed by an application, rather than on pre-purchased units of capacity. It is a form of utility computing. The name "serverless computing" is used because the server management and capacity planning decisions are completely hidden from the developer or operator. Serverless code can be used in conjunction with code deployed in traditional styles, such as microservices. Alternatively, applications can be written to be purely serverless and use no provisioned servers at all.[[16]](#endnote-16)

## Private Cloud

Private cloud is cloud infrastructure operated solely for a single organization, whether managed internally or by a third-party, and hosted either internally or externally. Undertaking a private cloud project requires significant engagement to virtualize the business environment, and requires the organization to reevaluate decisions about existing resources.[[17]](#endnote-17) It can improve business, but every step in the project raises security issues that must be addressed to prevent serious vulnerabilities. Self-run data centers are generally capital intensive. They have a significant physical footprint, requiring allocations of space, hardware, and environmental controls. These assets have to be refreshed periodically, resulting in additional capital expenditures. They have attracted criticism because users still have to buy, build, and manage them and thus do not benefit from less hands-on management, essentially lacking the economic model that makes cloud computing such an intriguing concept.

## Public Cloud

A cloud is called a "public cloud" when the services are rendered over a network that is open for public use. Public cloud services may be free.[[18]](#endnote-18) Technically there may be little or no difference between public and private cloud architecture, however, security consideration may be substantially different for services (applications, storage, and other resources) that are made available by a service provider for a public audience and when communication is effected over a non-trusted network. Generally, public cloud service providers like Amazon Web Services (AWS), Oracle, Microsoft Azure and Google Cloud own and operate the infrastructure at their data center and access is generally via the Internet. AWS, Oracle, Microsoft, and Google also offer direct connect services called "AWS Direct Connect", "Oracle FastConnect", "Azure ExpressRoute", and "Cloud Interconnect" respectively, such connections require customers to purchase or lease a private connection to a peering point offered by the cloud provider.

## Hybrid Cloud

Hybrid cloud is a composition of two or more clouds (private, community or public) that remain distinct entities but are bound together, offering the benefits of multiple deployment models. Hybrid cloud can also mean the ability to connect collocation, managed and/or dedicated services with cloud resources. Gartner defines a hybrid cloud service as a cloud computing service that is composed of some combination of private, public and community cloud services, from different service providers.[[19]](#endnote-19) A hybrid cloud service crosses isolation and provider boundaries so that it can't be simply put in one category of private, public, or community cloud service. It allows one to extend either the capacity or the capability of a cloud service, by aggregation, integration or customization with another cloud service.

## Other

There are other kinds of cloud system aside from the discussed ones.

### Community Cloud

Community cloud shares infrastructure between several organizations from a specific community with common concerns (security, compliance, jurisdiction, etc.), whether managed internally or by a third-party, and either hosted internally or externally. The costs are spread over fewer users than a public cloud (but more than a private cloud), so only some of the cost savings potential of cloud computing are realized.

### Distributed Cloud

A cloud computing platform can be assembled from a distributed set of machines in different locations, connected to a single network or hub service. It is possible to distinguish between two types of distributed clouds: public-resource computing and volunteer cloud.

### Multicloud

Multicloud is the use of multiple cloud computing services in a single heterogeneous architecture to reduce reliance on single vendors, increase flexibility through choice, mitigate against disasters, etc. It differs from hybrid cloud in that it refers to multiple cloud services, rather than multiple deployment modes (public, private, legacy).

### Big Data Cloud

The issues of transferring large amounts of data to the cloud as well as data security once the data is in the cloud initially hampered adoption of cloud for big data, but now that much data originates in the cloud and with the advent of bare-metal servers, the cloud has become a solution for use cases including business analytics and geospatial analysis.

### HPC Cloud

HPC cloud refers to the use of cloud computing services and infrastructure to execute high-performance computing (HPC) applications. These applications consume considerable amount of computing power and memory and are traditionally executed on clusters of computers. Various vendors offer servers that can support the execution of these applications. In HPC cloud, the deployment model allows all HPC resources to be inside the cloud provider infrastructure or different portions of HPC resources to be shared between cloud provider and client on-premise infrastructure.

# Existing System

## Commercial Existing Services

### Amazon Web Services

Amazon.com is one of the most important and heavily trafficked Websites in the world. It provides a vast selection of products using an infrastructure based on Web services. As Amazon.com has grown, it has dramatically grown its infrastructure to accommodate peak traffic times. Over time the company has made its network resources available to partners and affiliates, which also has improved its range of products.[[20]](#endnote-20)

Starting in 2006, Amazon.com made its Web service platform available to developers on a usage-basis model. Amazon.com has made it possible to create private virtual servers that you can run worldwide.[[21]](#endnote-21) These servers can be provisioned with almost any kind of application software you might envisage, and they tap into a range of support services that not only make distributed cloud computing applications possible, but make them robust. Some very large Web sites are running on Amazon.com’s infrastructure without their client audience being any the wiser.

Amazon Web Services is based on SOA standards, including HTTP, REST, and SOAP transfer protocols, open source and commercial operating systems, application servers, and browser-based access. Virtual private servers can provision virtual private clouds connected through virtual private networks providing for reasonable security and control by the system administrator. AWS has a great value proposition: You pay for what you use. While you may not save a great deal of money over time using AWS for enterprise class Web applications, you encounter very little barrier to entry in terms of getting your site or application up and running quickly and robustly. [[22]](#endnote-22)AWS has much to teach us about the future of cloud computing and how virtual infrastructure can be best leveraged as a business asset. The Amazon is the world’s largest river. Amazon.com is the world’s largest online retailer with net sales in $24.51 billion, according to their 2009 annual report. The company is a long way past selling books and records. While Amazon.com is not the earth’s biggest retailer, Amazon.com offers the largest number of retail product SKUs through a large ecosystem of partnerships. By any measure, Amazon.com is a huge business. To support this business, Amazon.com has built an enormous network of IT systems to support not only average, but peak customer demands. Amazon Web Services (AWS) takes what is essentially unused infrastructure capacity on Amazon.com’s network and turns it into a very profitable business.

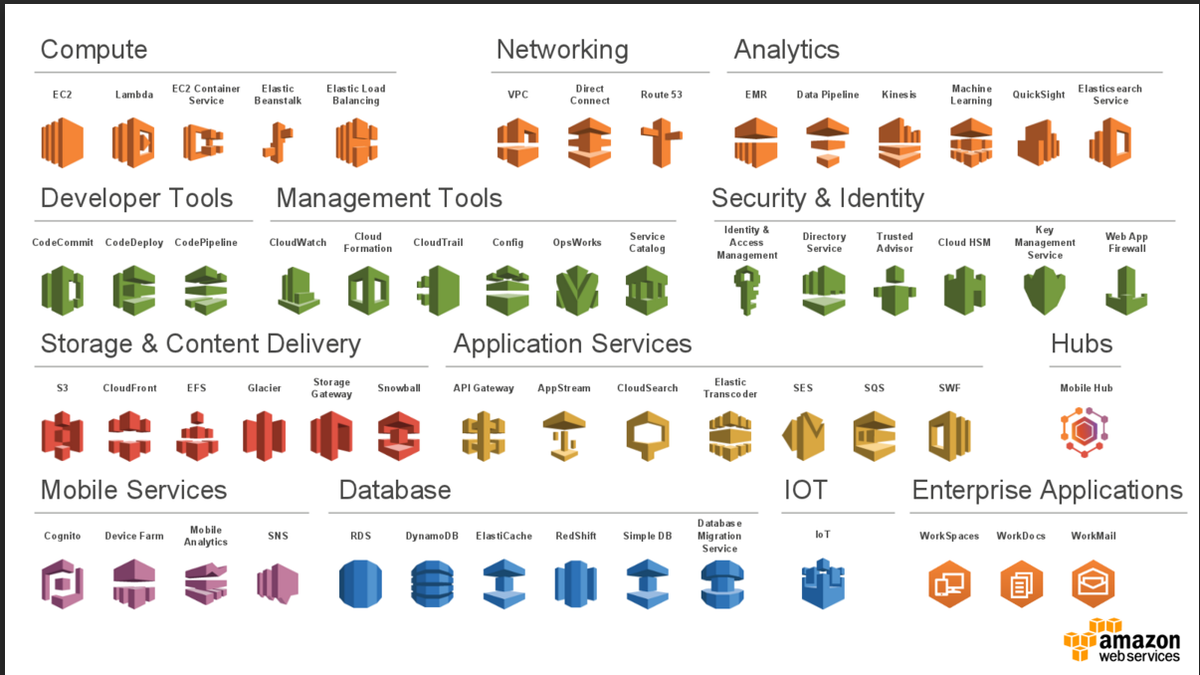


Figure ‑ Amazon Web Services

### Microsoft Azure

Microsoft has a very extensive cloud computing portfolio underactive development. Efforts to extend Microsoft products and third-party applications into the cloud are centered around adding more capabilities to existing Microsoft tools. Microsoft’s approach is to view cloud applications as software plus service. In this model, the cloud is another platform and applications can run locally and access cloud services or run entirely in the cloud and be accessed by browsers using standard Service Oriented Architecture (SOA) protocols. Microsoft calls their cloud operating system the Windows Azure Platform.[[23]](#endnote-23)

Azure is a combination of virtualized infrastructure to which the .NET Framework has been added as a set of .NET Services. The Windows Azure service itself is a hosted environment of virtual machines enabled by a fabric called Windows Azure App Fabric. You can host your application on Azure and provision it with storage, growing it as you need it. Windows Azure service is an Infrastructure as a Service offering. A number of services interoperate with Windows Azure, including SQL Azure (a version of SQL Server), SharePoint Services, Azure Dynamic CRM, and many of Windows Live Services comprising what is the Windows Azure Platform, which is a Platform as a Service cloud computing model. Eventually, many more services will be added, encompassing the whole range of Microsoft’s offerings.

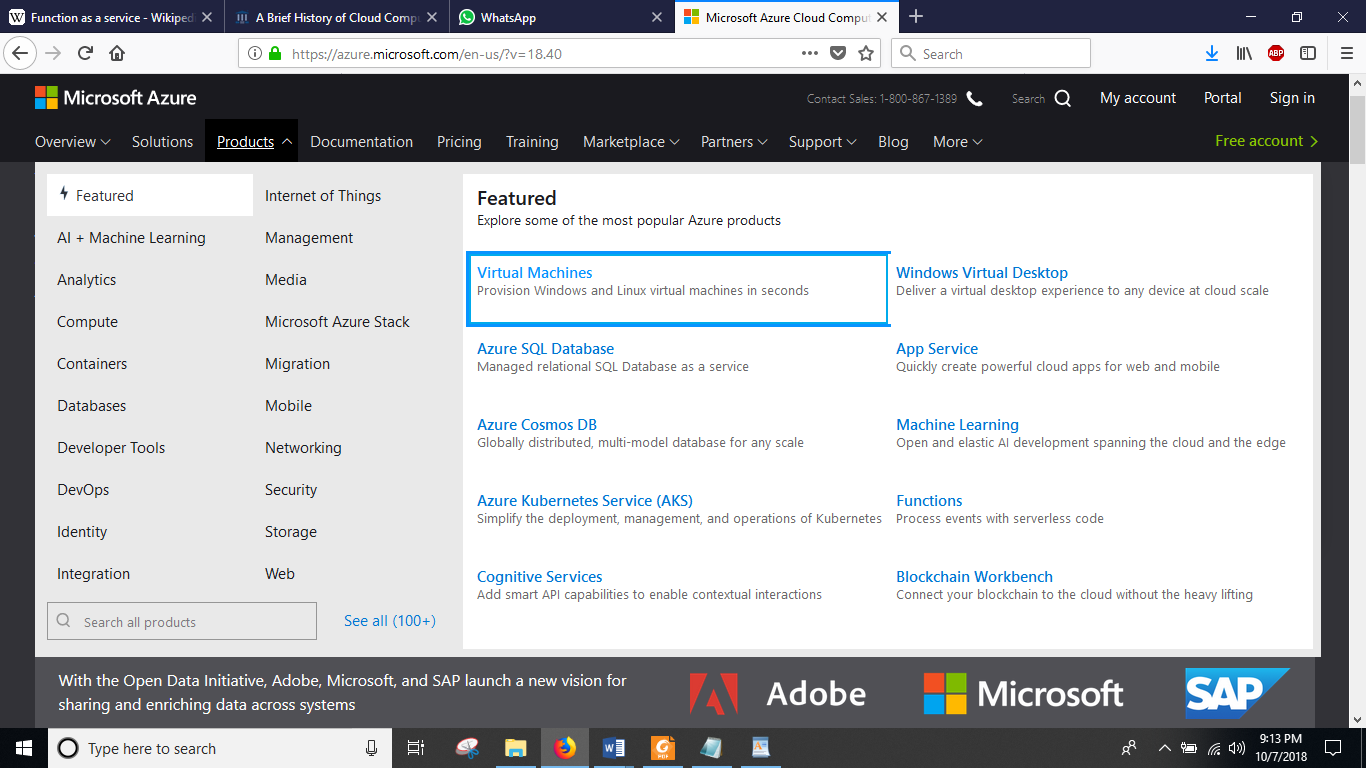


Figure ‑ Microsoft Azure Services

### Google Cloud

Google is the prototypical cloud computing services company, and it supports some of the largest Web sites and services in the world. At the center of Google’s core business is the company’s search technology. Google uses automated technology to index the Web.[[24]](#endnote-24) It makes its search service available to users as a standard search engine and to developers as a collection of special search tools limited to various areas of content. The application of Google’s searches to content aggregation has led to enormous societal changes and to a growing trend of disintermediation.

The most important commercial part of Google’s activities is its targeting advertising business: AdWords and AdSense. Google has developed a range of services including Google Analytics that supports its targeted advertising business. Google applications are cloud-based applications. The range of application types offered by Google spans a variety of types: productivity applications, mobile applications, media delivery, social interactions, and many more. Google has begun to commercialize some of these applications as cloud-based enterprise application suites that are being widely adopted. Google has a very large program for developers that spans its entire range of applications and services.[[25]](#endnote-25) Among the services highlighted are Google’s AJAX APIs, the Google Web Toolkit, and in particular Google’s relatively new Google Apps Engine hosting service. Using Google App Engine, we can create Web applications in Java and Python that can be deployed on Google’s infrastructure and scaled to a large size.

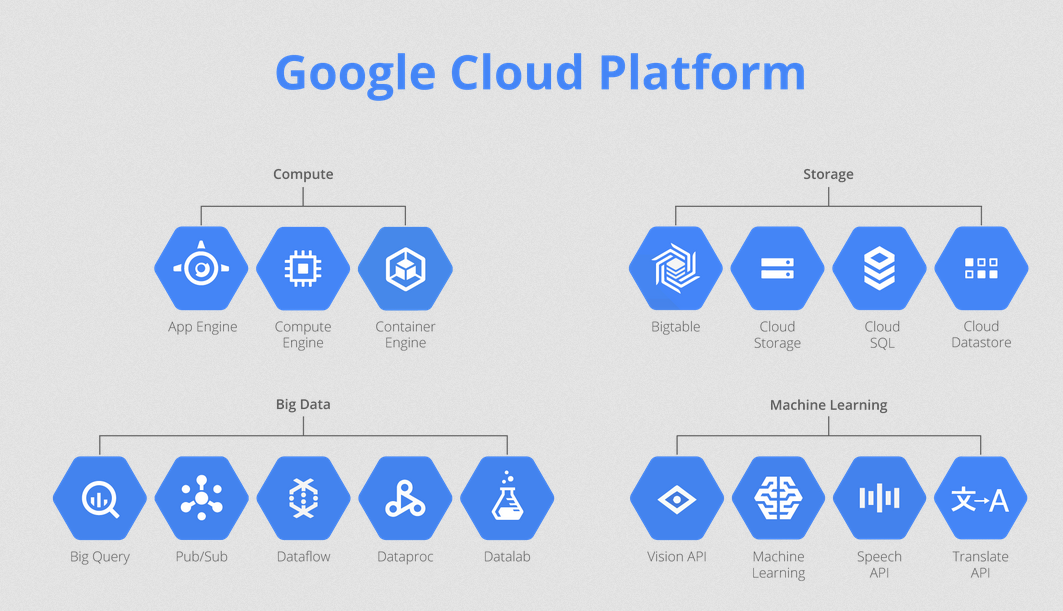


Figure ‑ Google Cloud Services

# Proposed model

## Project Perspective

Cloud computing is getting popular day by day for its advantages. Our project uses the core features of the cloud computing and it will remove hardware as well as software dependency in our lab environment.

We will implement a private cloud in our lab to use it as alternative to existing lab. Private cloud system will provide virtual machines to the students to use as personal computer. This will allow granular control over specific access rights such as storage, network easily. Creating new machines for new students or deleting existing ones will be only several mouse clicks.

On the following pages we describe the architecture of our proposed private cloud model to run virtual machines.

## Project Architecture

The internal architecture of the private cloud consists of openstack services namely

* Nova: Computer service. Provides computing service to instances running.
* Glace: Provides image service. Enables to upload new image, take snapshot of instance.
* Horizon: Web based GUI dashboard.
* Keystone: Provides identity service. This component keeps track of users and their permission.
* Neutron: Network service. Enable to create elaborate virtual networks for instances.
* Cinder: Block storage service. Provides instances block device such as partition for storage.

Virtual instances reside on the virtual network. Instances can also be given public IP from CIDR block if proper DHCP server is available.

A virtual network bridge connects the virtual network and physical network interface of the physical server. The bridge acts as a gateway for Neutron.

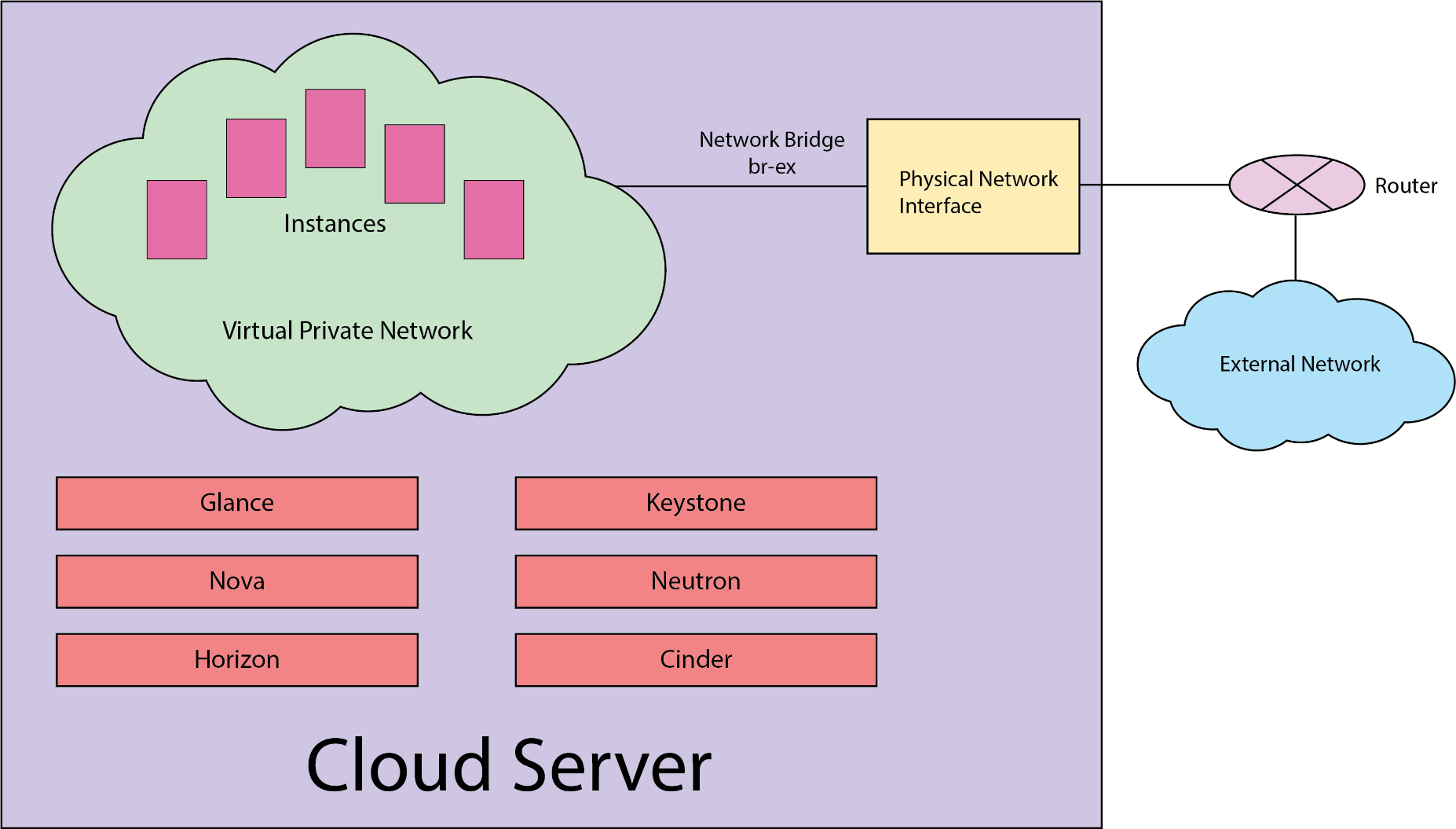


Figure ‑ Proposed Model Architecture

## Storage Architecture

Storage is proposed to use Logical Volume Manager (LVM). This enables to create a cohesive logical file spanning many physical devices or a single one. The size of the file dictates the available storage for use. The LVM creates logical partition or block device on the file for instances. This block storage can be attached to an instance and used as a physical storage.

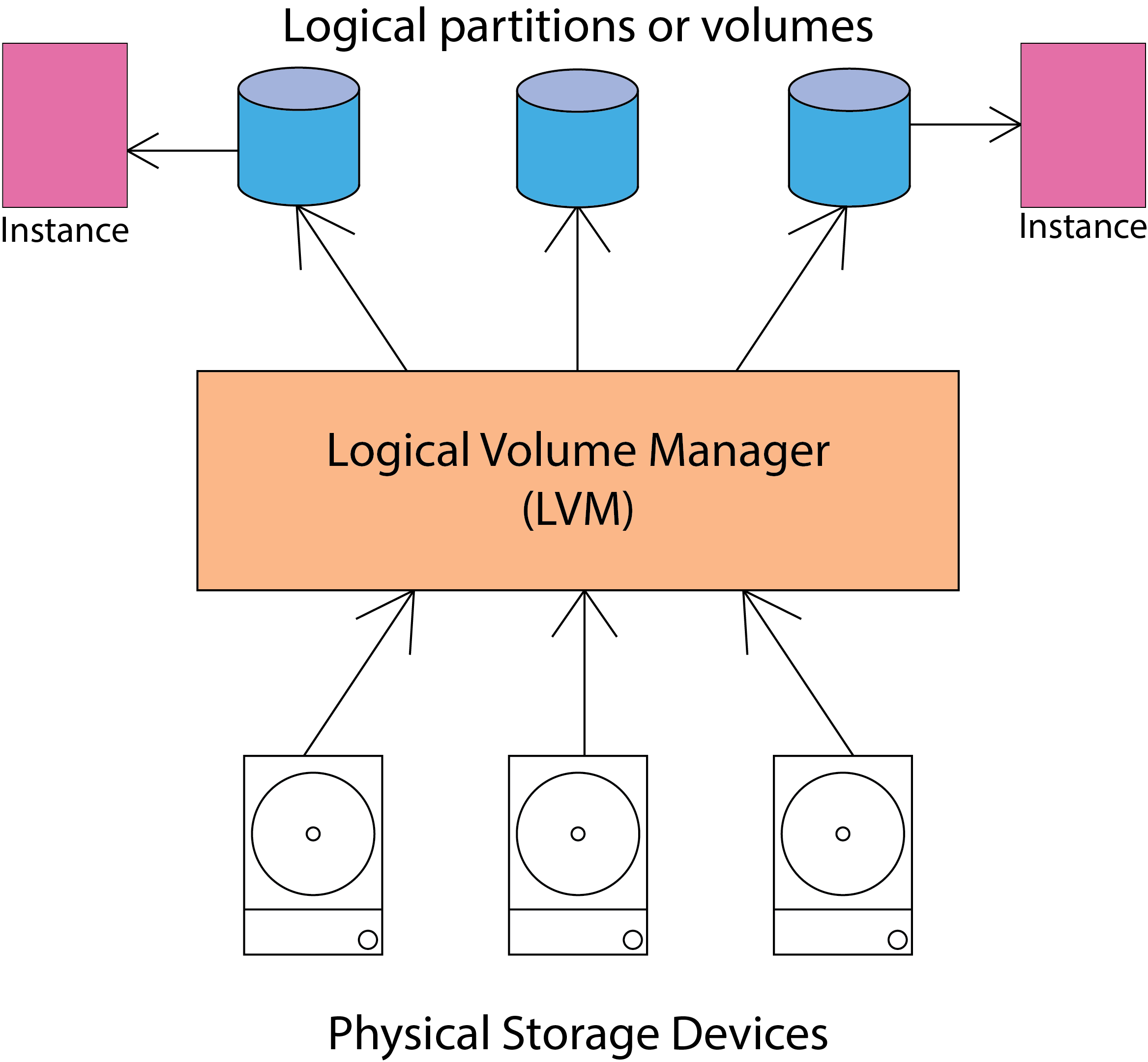


Figure 4‑2 Storage Architecture

## Use Case Diagram

A single use-case diagram captures a particular functionality of a system. Use-case diagrams are consisting of actors, use cases and their relationships. The diagram is used to model the system/subsystem of an application. The purpose of use case diagram is to capture the dynamic aspect of a system. Use-case diagrams are considered for high level requirement analysis of a system. Use case diagrams are drawn to capture the functional requirements of a system.

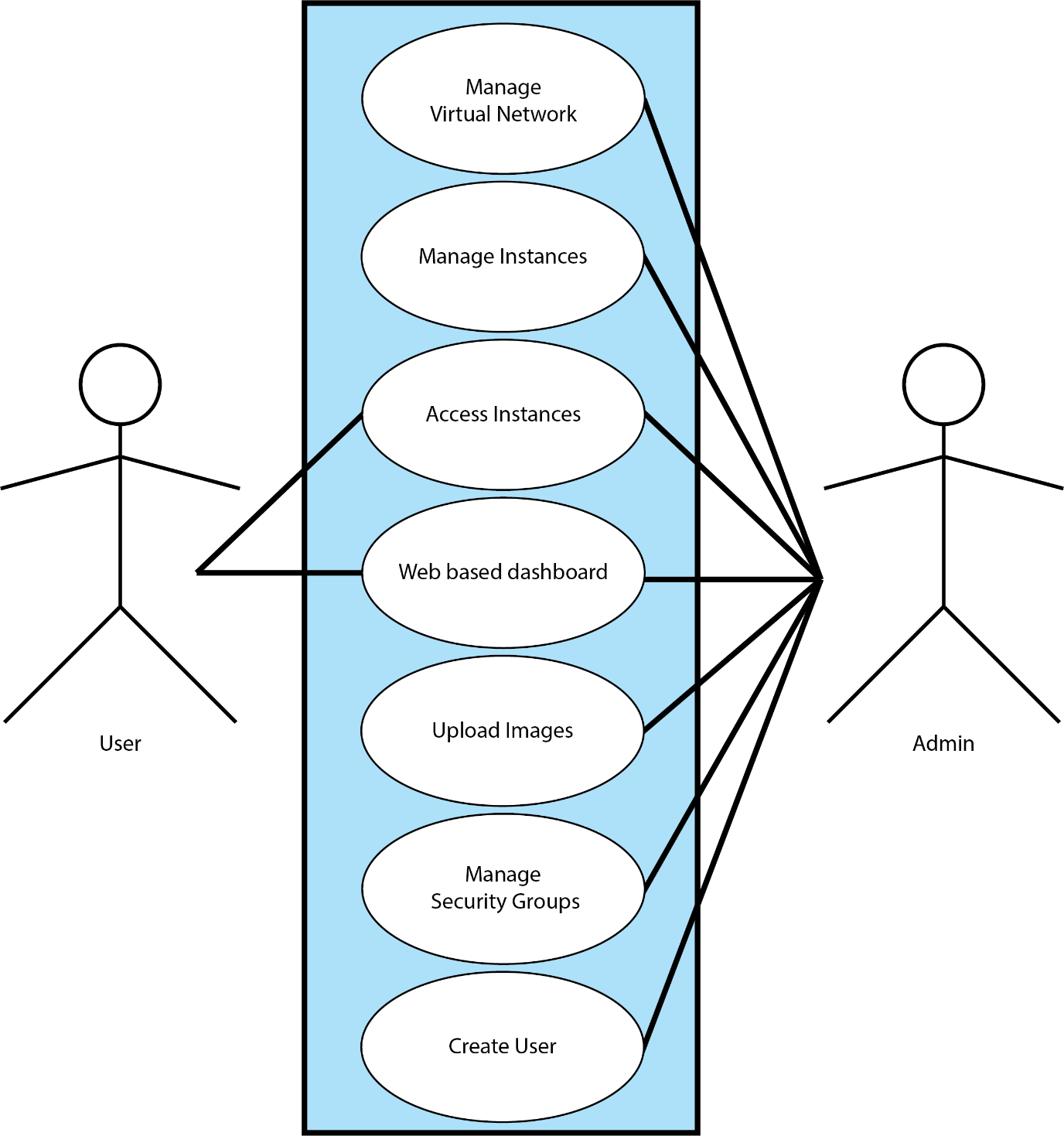


Figure 4‑3- Use Case Diagram

Here Admin and User are considered as actor. Admin can –

* Access web based dashboard
* Manage Instances
* Access instances
* Upload and manage images
* Create and delete users

Users can –

* Access web based dashboard interface
* Access instances/virtual machines allocated for that user.

## Diagram of virtual lab environment

Virtual lab environment consists of

* Cloud Server
* Clients
* Network in which server and clients reside.

Any client in the same network as cloud server can access the system through Horizon, web based interface. Cloud can be accessed from external network if proper port forwarding is implemented.

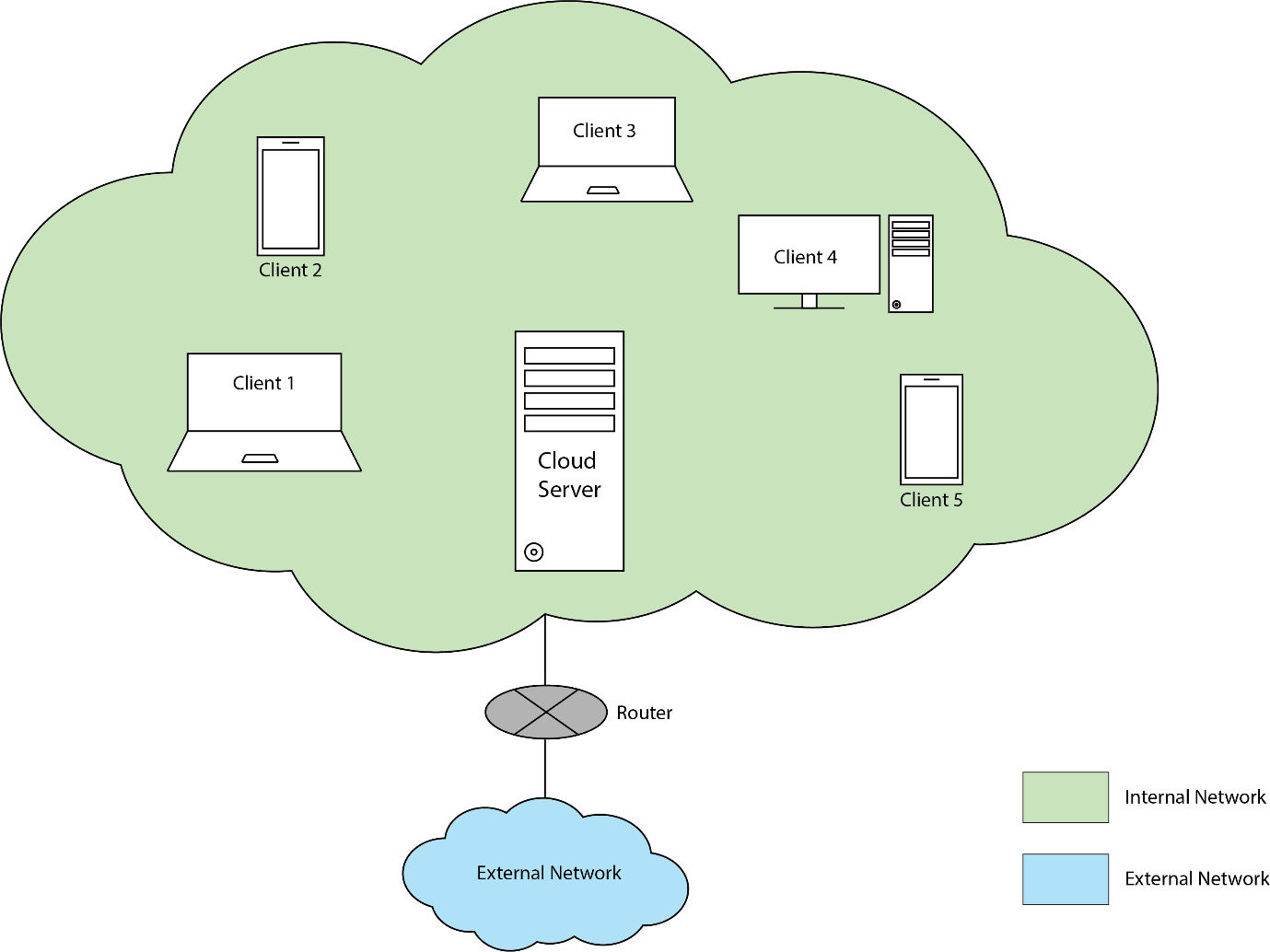


Figure 4‑4- Virtual lab environment system

## Data Flow Diagram

In the data flow diagram actions can be easily visualized on entities.

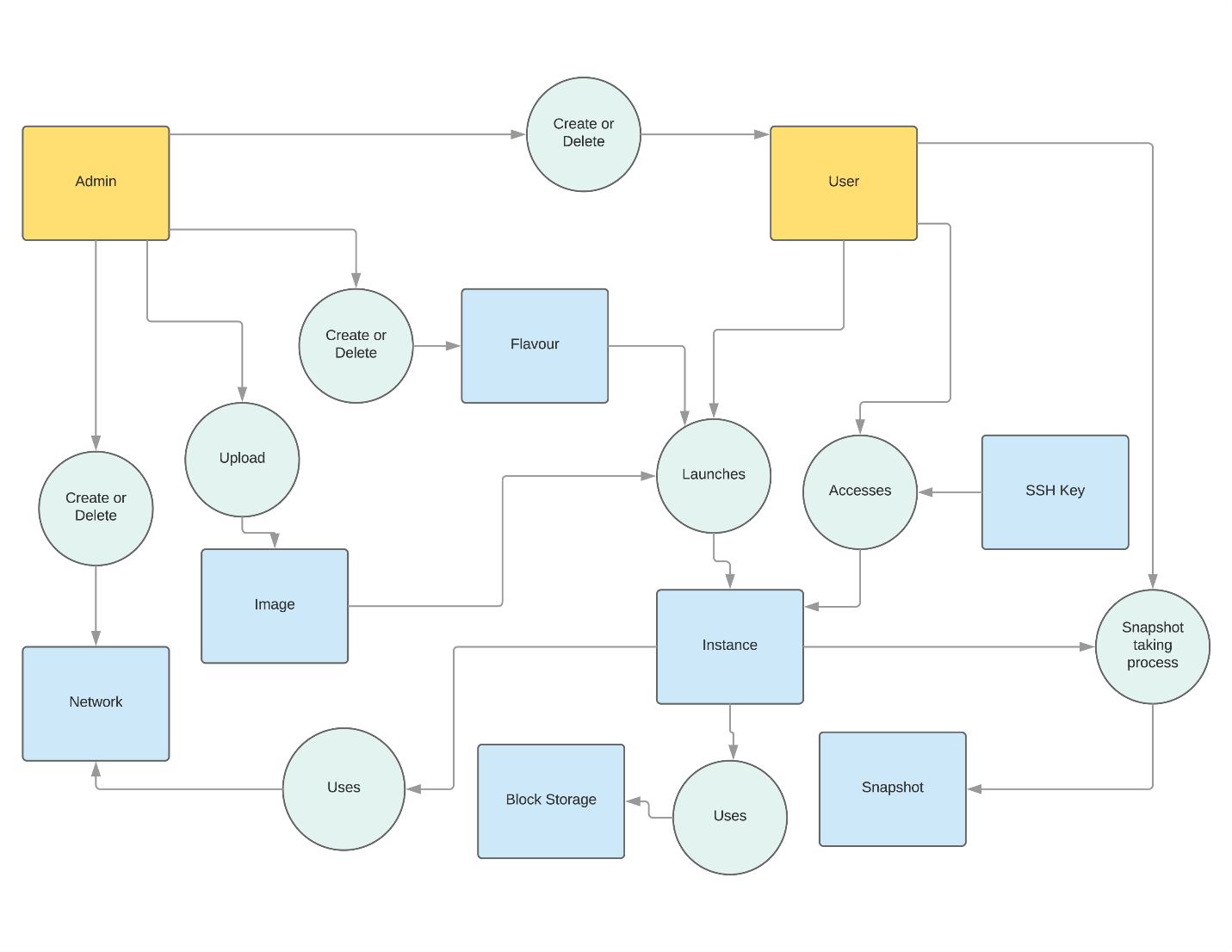


Figure 4‑5 Data Flow Diagram of the Proposed Model

## Entity Relationship Diagram

Entity Relationship Diagram visualizes relationship between entities.

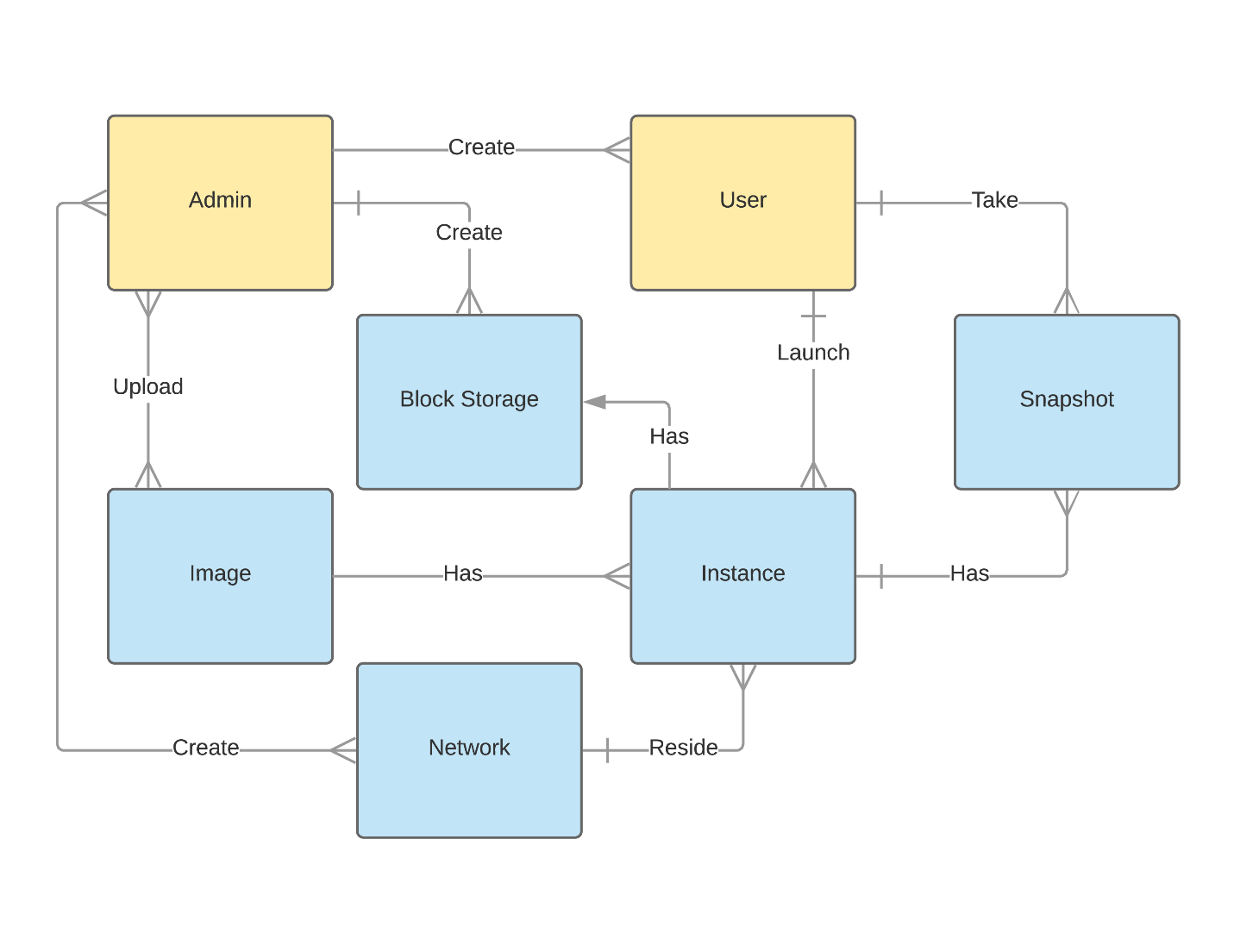


Figure 4‑6 Entity Relationship Diagram

# Project Implementation

## Project Requirements

We have proposed a virtual lab environment which will be established using Packstack RDO project. Project requires minimum hardware and software.

### Hardware Requirement

* CPU: 4 physical cores or higher with minimum 3GHz
* Memory: Minimum 16GB
* HDD: As needed for virtual machines and storing images
* Keyboard, monitor, mouse

### Software Requirement

* Operating System: CentOS, a linux based operating system
* Scripting language: BASH
* Environment: Command Line Interface
* Any Internet Browser
* Image creation tool: KVM with Qemu

## Deployment Tool Used

### Openstack

Openstack is one of the leading projects for Cloud based Computing which provides Infrastructure as a Service(IaaS). Openstack can be used to run private cloud system instead of paid third party services like Amazon EC2, Microsoft Azure and Google Cloud.[[26]](#endnote-26) In this document we compare and explore deployment projects for Openstack Cloud Computing System by which openstack can be installed and deployed on premise. We have provided enough information and checkpoints that one can reach decision on which project is best for their situation regarding their goal and need.

Many projects are available to deploy Openstack. Here is a non-exhaustive list which are discussed in this document –

1. Devstack
2. Conjure-Up Openstack Spell
3. Packstack
4. Kolla
5. Openstack Ansible
6. TripleO
7. TripleO quickstart

These deployment projects vary vastly in their deployment process. Although all of them uses linux as distribution, they are focused for different linux distribution.[[27]](#endnote-27) From the list, Devstack, Kolla, Openstack Ansible can be deployed in Ubuntu, Debian, Fedora, CentOS, RHEL and OpenSuse linux. Conjure-up is available only for Ubuntu based distributions. Whereas Packstack, TripleO and TripleO quickstart is aimed towards CentOS and RHEL distribution.

We will now discuss about Packstack from the list briefly. As it is the deployment tool used in this research project.

### Packstack

Packstack is a utility that uses Puppet modules to deploy various parts of OpenStack on multiple pre-installed servers over SSH automatically. Only CentOS, Red Hat Enterprise Linux (RHEL) and compatible derivatives of both are supported. The project is maintained by RDO community. RDO is a community of people using and deploying OpenStack on CentOS, Fedora, and Red Hat Enterprise Linux.

Packstack deploys a full-fledged Opnestack environment. Packstack is configurable, can be used to deploy openstack in multiple nodes. One controller node and other nodes will be used as compute node. It is also robust. With administratibe knowledge editing a configure file called answer file we can configure deploying Opesntack to a great degree. Installing and deploying via packstack is easy for AIO. Only installing in a single machine does not require much administrative knowledge. But configuring the environment for specific service like external network does require better acquaintance with linux. Deployed environment by packstack is a whole environment and can be used as one. Only thing missing in packsack is adding additional nodes other than compute node.

Packstack Pros

1. Deploys full operational Opesntack environment.
2. Configurable modules and services for deployment.
3. Deployment can be multiple node based.
4. Have plenty of documentation and have an active community.

Packstack Cons

1. Only CentOS and Red Hat Entiprise Linux (RHEL) is supported.
2. Additional only compute nodes can be added.
3. Requires linux acquaintance for configuring services.

For the aforementioned reasons, Packstack deployment tool has been chosen.

## Deployment Procedure

Deployment procedure is long and tedious, including assembling hardware, installing OS, configuring operating system, BASH scripting, configuring openstack etc.[[28]](#endnote-28) Simple overview of deployment procedure is discussed following.

1. Assembling actual physical server hardware comes first.
2. Installing OS is next. As discussed, CentOS 7 was chosen. Separate partition for root, /home, /var, /boot and /temp is created for easy storage expandability and security.
3. Configuring operating system includes installing required packages like QEMU, LVM, openSSH, Vim, lxd, netutils, git etc.
4. Required software are configured accordingly. Like creating a network bridge adapter.
5. BASH scripting used for easy configuration of OS and packstack.
6. Packstack configuration files changed as needed.
7. Deployment process starts after all configuration. Uninterrupted innternet connection is must in this step.
8. After Packstack has been deployed, necessary users, virtual networks are created; images are uploaded.

This is the simple overview of deployment process.

## Project Features and Functionality

### Image Creation

For image creation first we need to acquire the required Operating System(OS) ISO file. From the ISO file we have to install it on the kernel virtual machine (KVM). Then we need to make necessary changes in that operating system like installing specific packages according to users need. Also it is required to install cloud init software for early initialization of cloud instances. After preparing the operating system we need to clean up MAC address and host information from that OS. Then we need to create a qcow2 image of that machine.

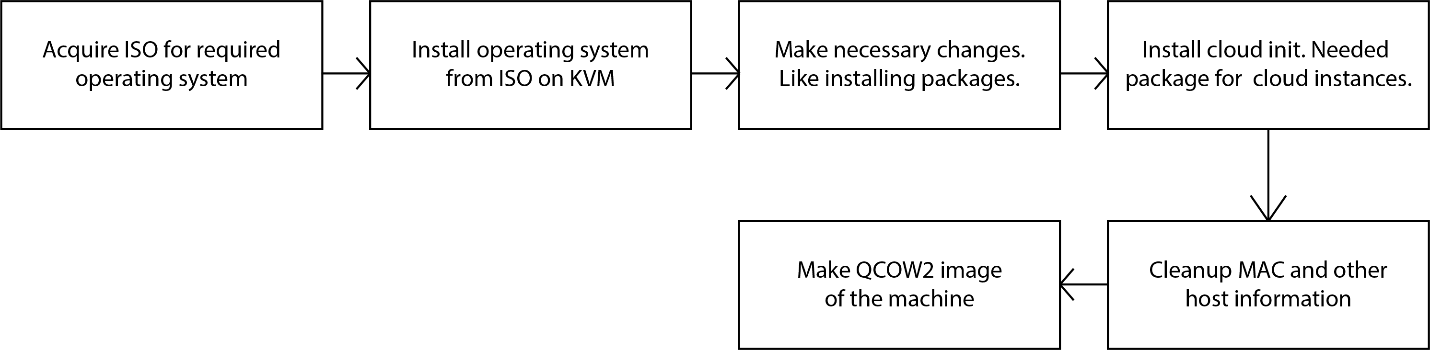


Figure 5‑1 Image Creation

### Launching Instances

After creating images and upload them to the cloud we can create instances form the images. First user need to select desired images from available images. Then user have to determine appropriate machine flavor and it will make sure of the resources available to instances. After that user need to select network which will be used by the instances for connecting to other instances under the same network. After launching instances, they can be accessed if running properly.

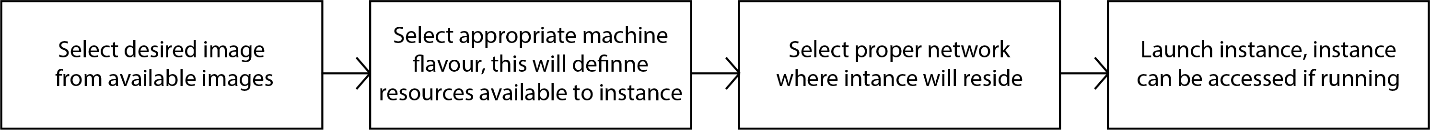


Figure 5‑2 Launching Instances

### Accessing Instances

Properly running instances can be accessed through the dashboard. User need to login to the web interface. Then selecting desired instances, one can use the instance by selecting the “show console” tab.

### Accessing Instances Through Secure Shell (SSH)

If user wants to access the instance through the SSH then a SSH key pair is needed for the instance. By writing the command “ssh –i <keyfile> <Instance IP>” in the command line, the instance can be accessed through that CLI interface.

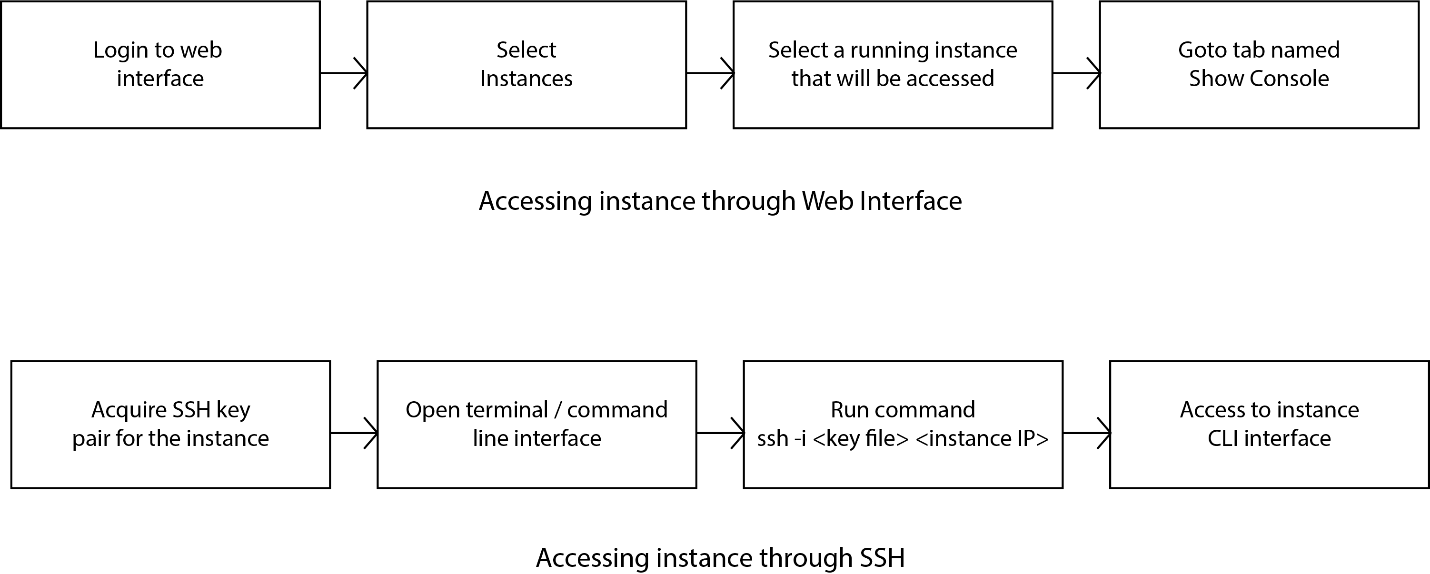


Figure 5‑3 Accessing Instance through Web Interface and SSH

### Creating Snapshot of Instances

User can create snapshots of the running instances. Before taking snapshot of the instances user can remove or add packages according to his/her requirement. Then one or multiple snapshots can be taken.

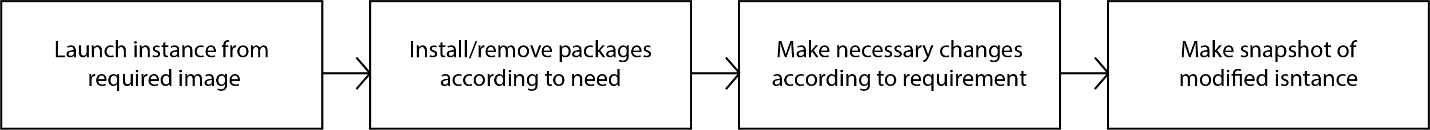


Figure 5‑4 Creating Snapshots of Instances

## Project Screenshots

### Accessing the Openstack login page

User will access login page and with user id and password the dashboard will appear. The dashboard can be found for each of the floating IP of the virtual machines. The dashboard is also called horizon.

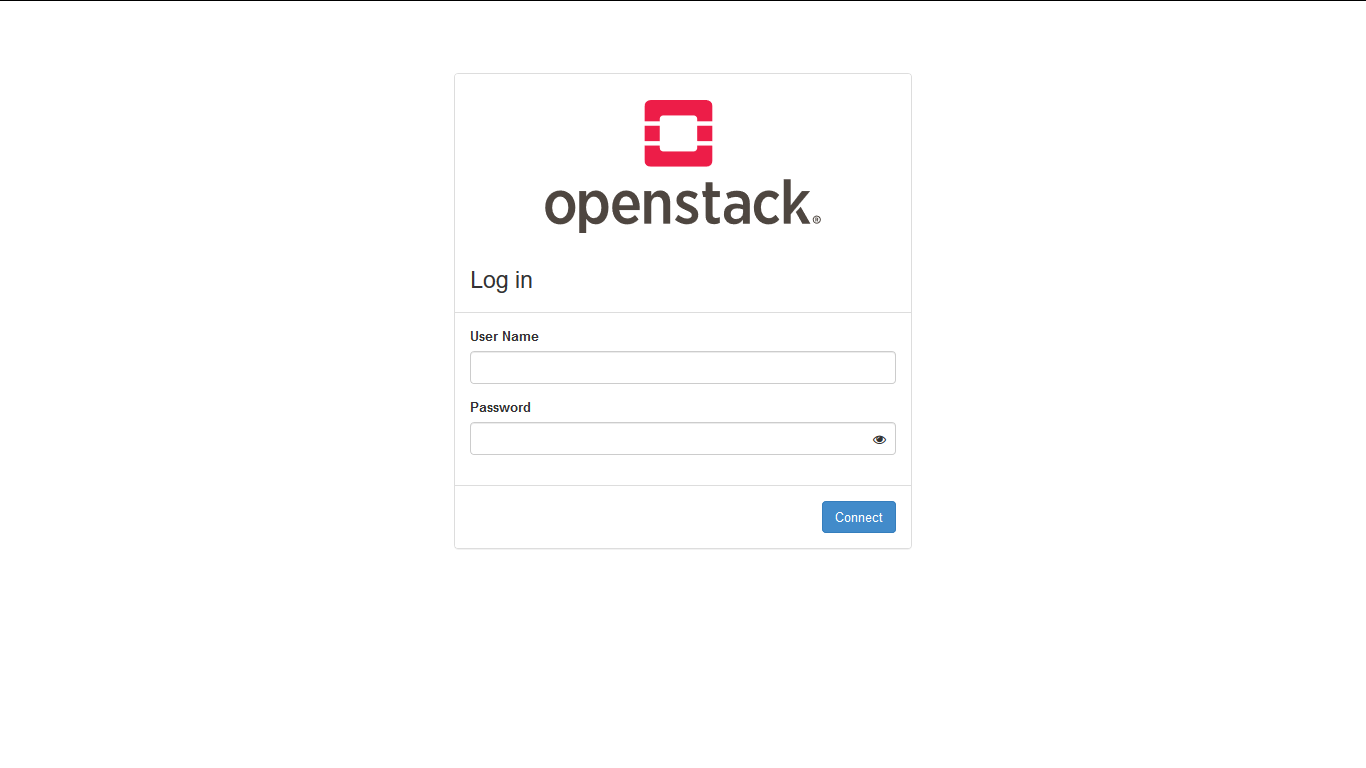


Figure 5‑5 Openstack Login Page

### Creating Users

Admin can create multiple users. Each user can be assigned required role. Each user will then launch various instances and work with them.

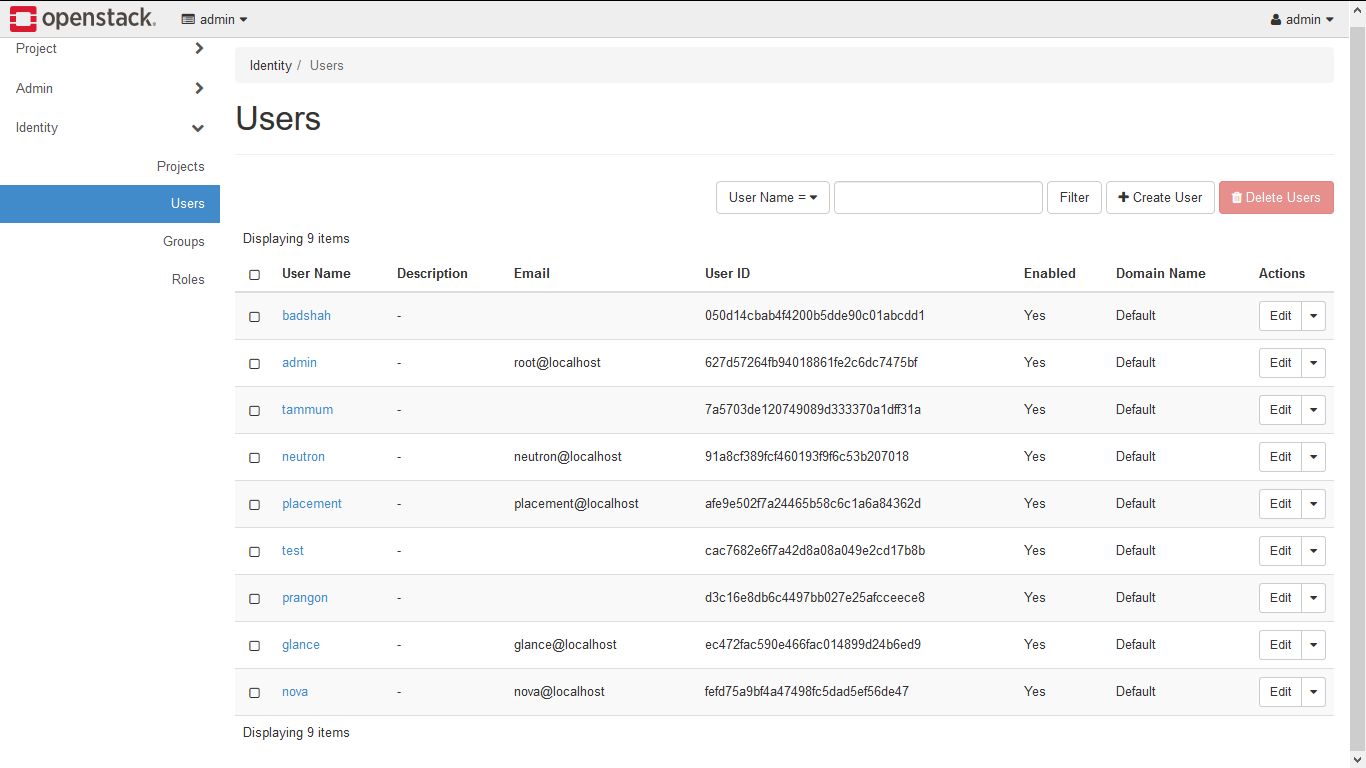


Figure 5‑6 Creating Users

### Configuring Network

Admin will configure the network for the virtual machines. There are only one external network and multiple private network. The virtual machines will be connected to the private network. Figure 10 shows an external network and a private network. A router joins the two network and there are two virtual machines connected to the private network.

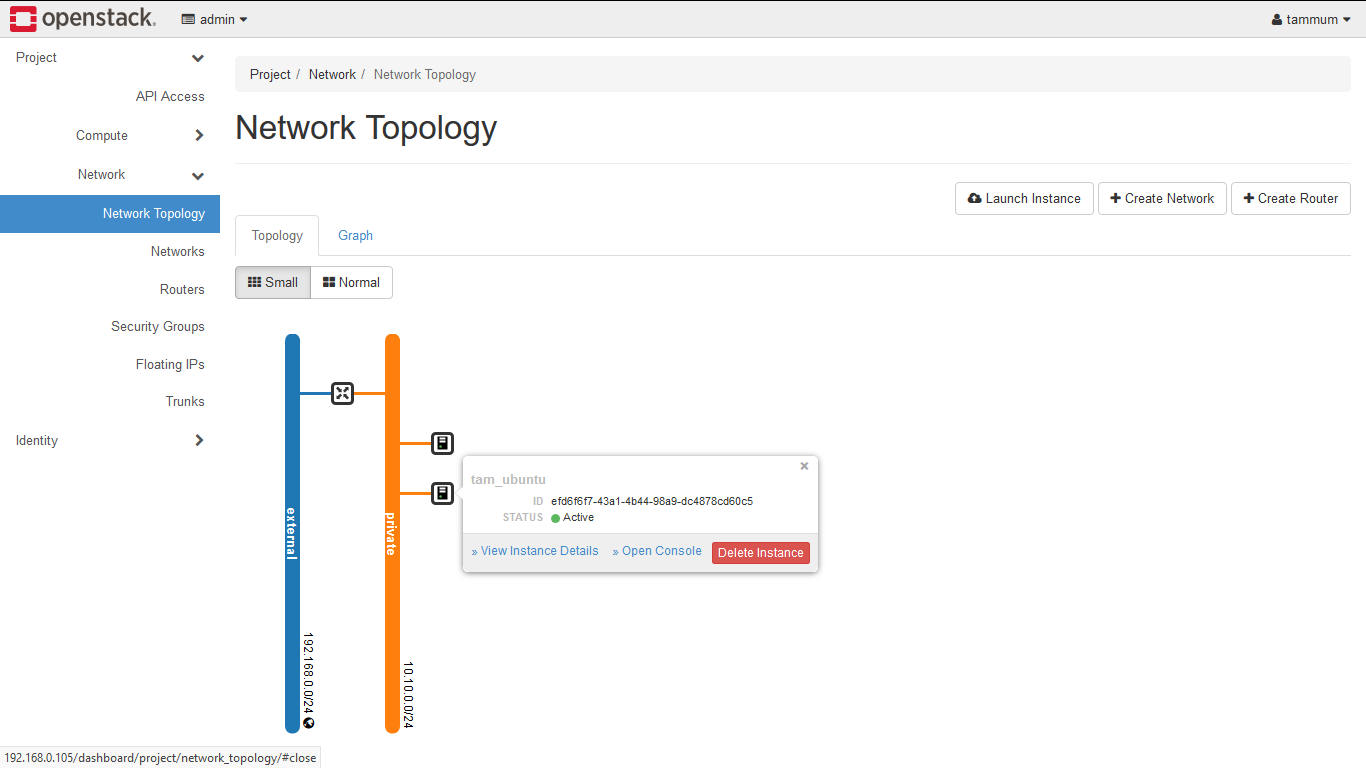


Figure 5‑7 Configuring the Network

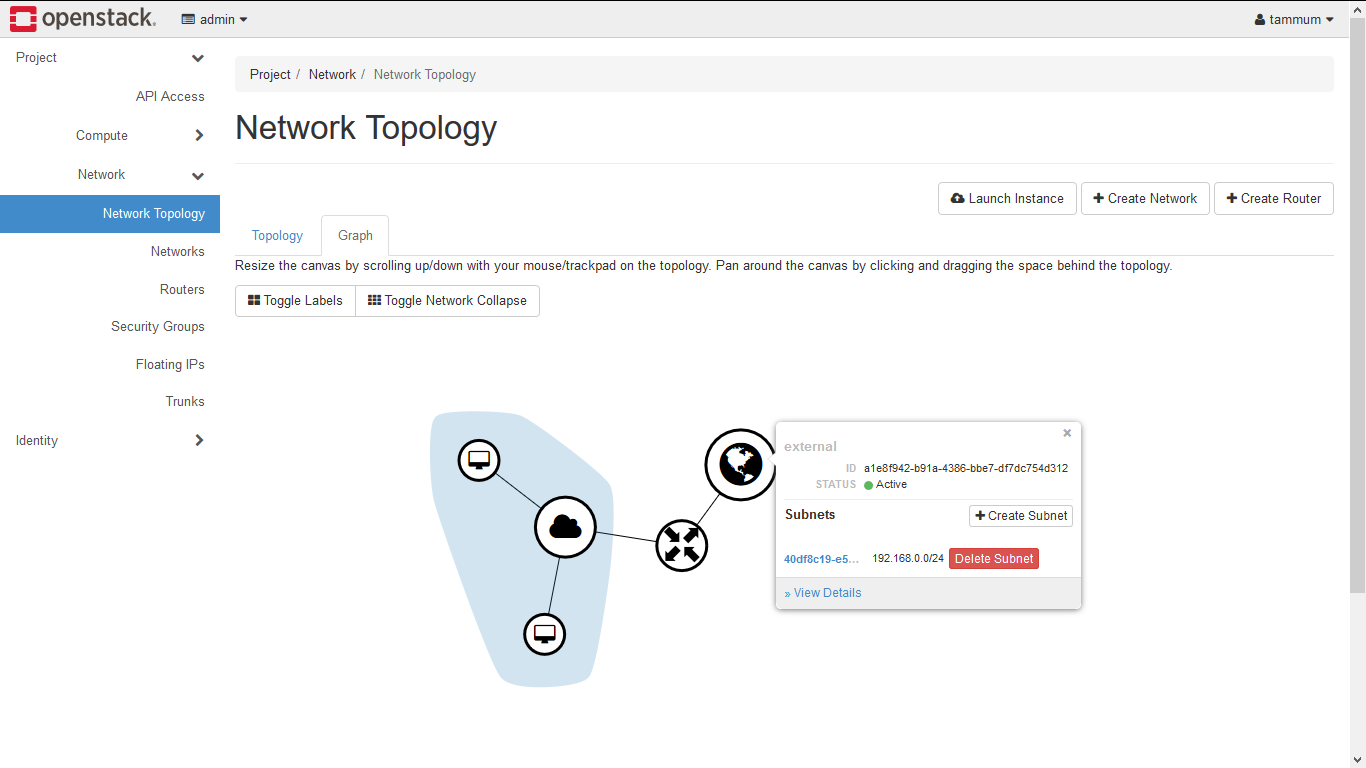
After configuring the network, the network topology can be visualized in the “Network Topology”. An example topology can be seen in the figure.

Figure Network Topology

### Image Uploading

Each user can create multiple images. The images can be of any operating system.

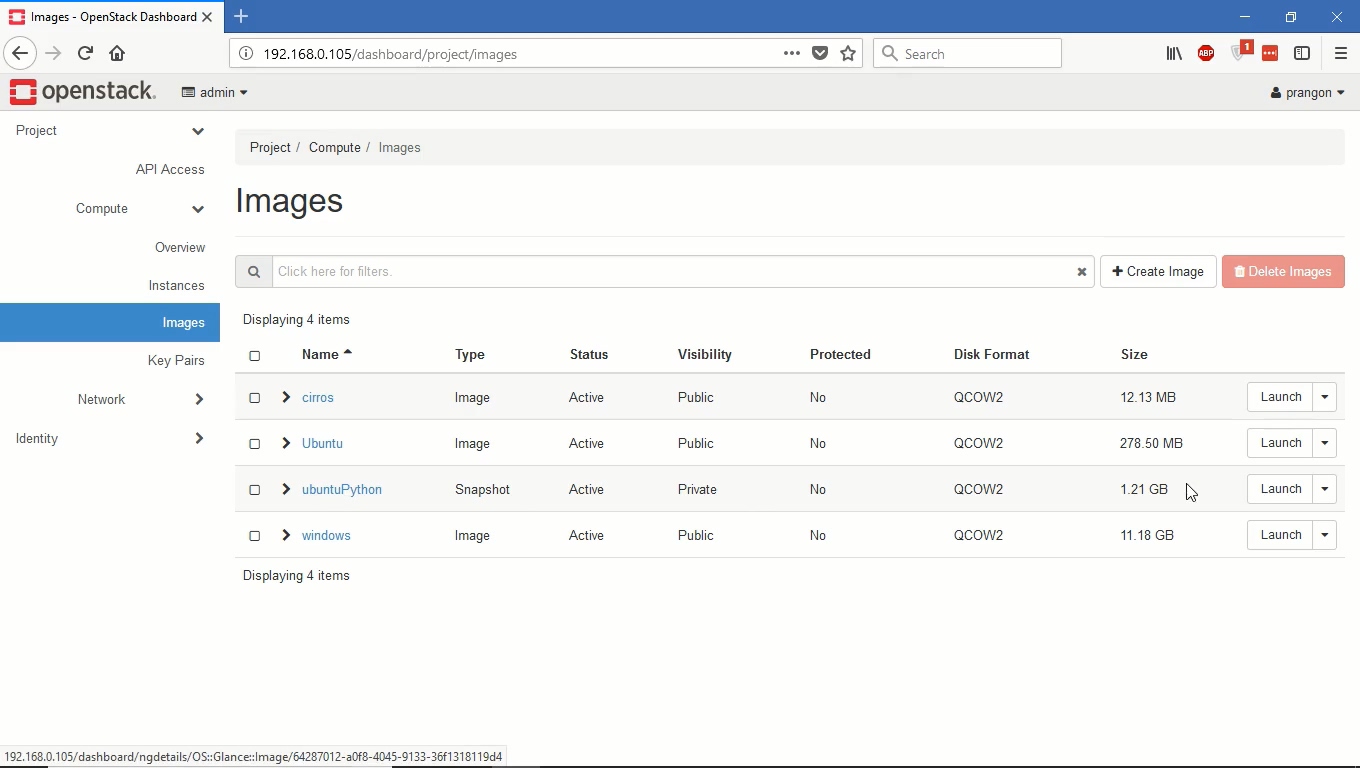


Figure 5‑9 Image Uploading

### Launching Instances

To launch any instance first user has to define the private network and the required flavor.

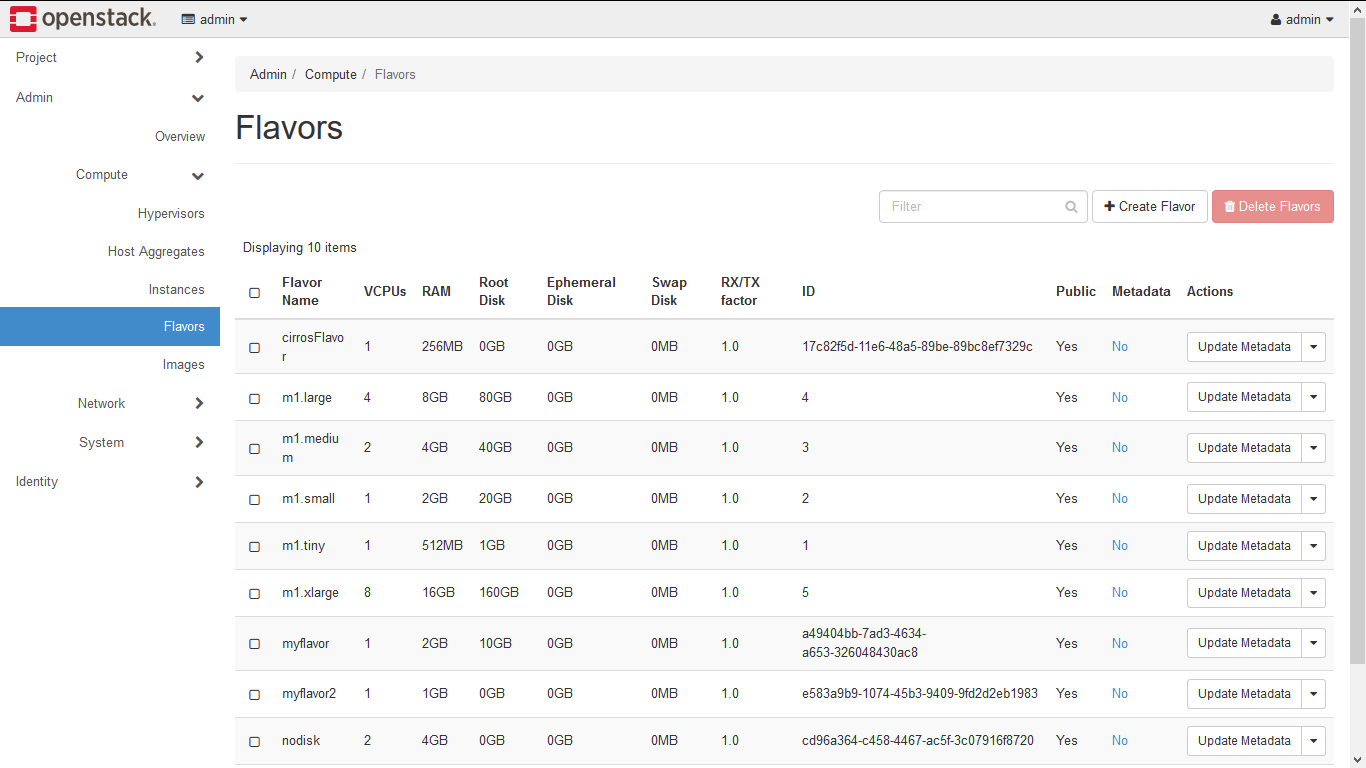


Figure 5‑10 Determining Flavors for the image to launch

### Launching GUI Based Operating System

Any image can be launched with required flavor. Microsoft window can be launched also using a large flavor.

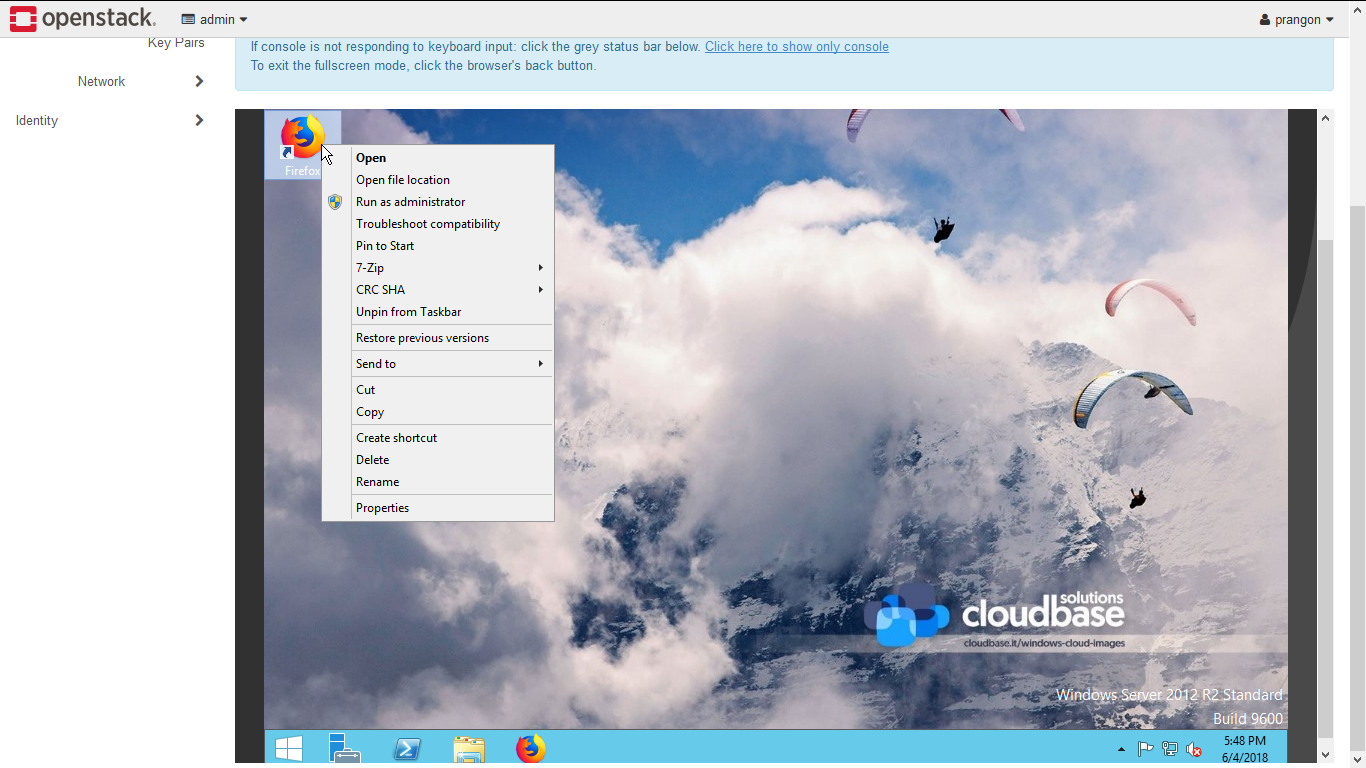


Figure 5‑11 Launching GUI based Operating System

### Launching Command Based Operating System

For launching Ubuntu command based operating system user need small flavor and Ubuntu command based image.

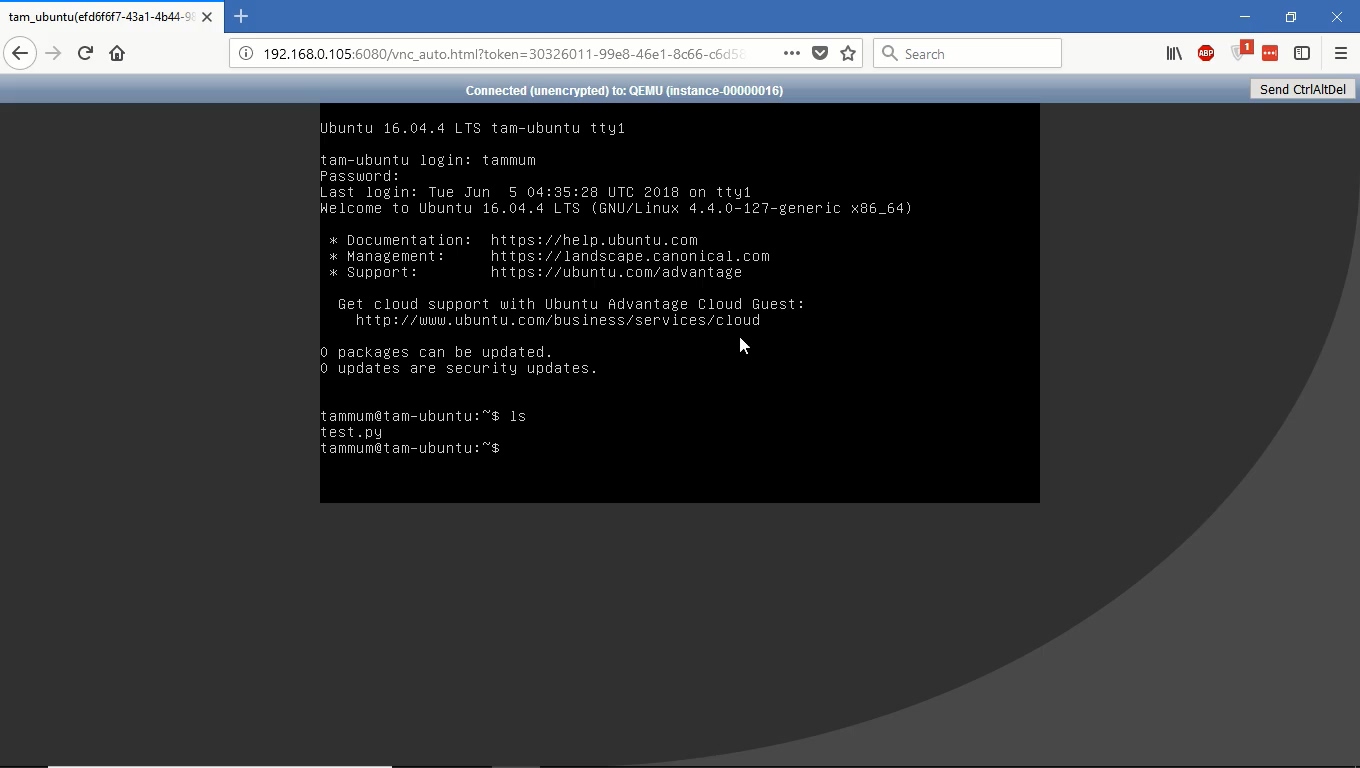


Figure 5‑12 Accessing Ubuntu Operating System

### Using Putty for Remote Access

For remotely accessing the virtual machines we need to have putty software installed. By using putty, we can remotely login to any virtual machines under the same network.

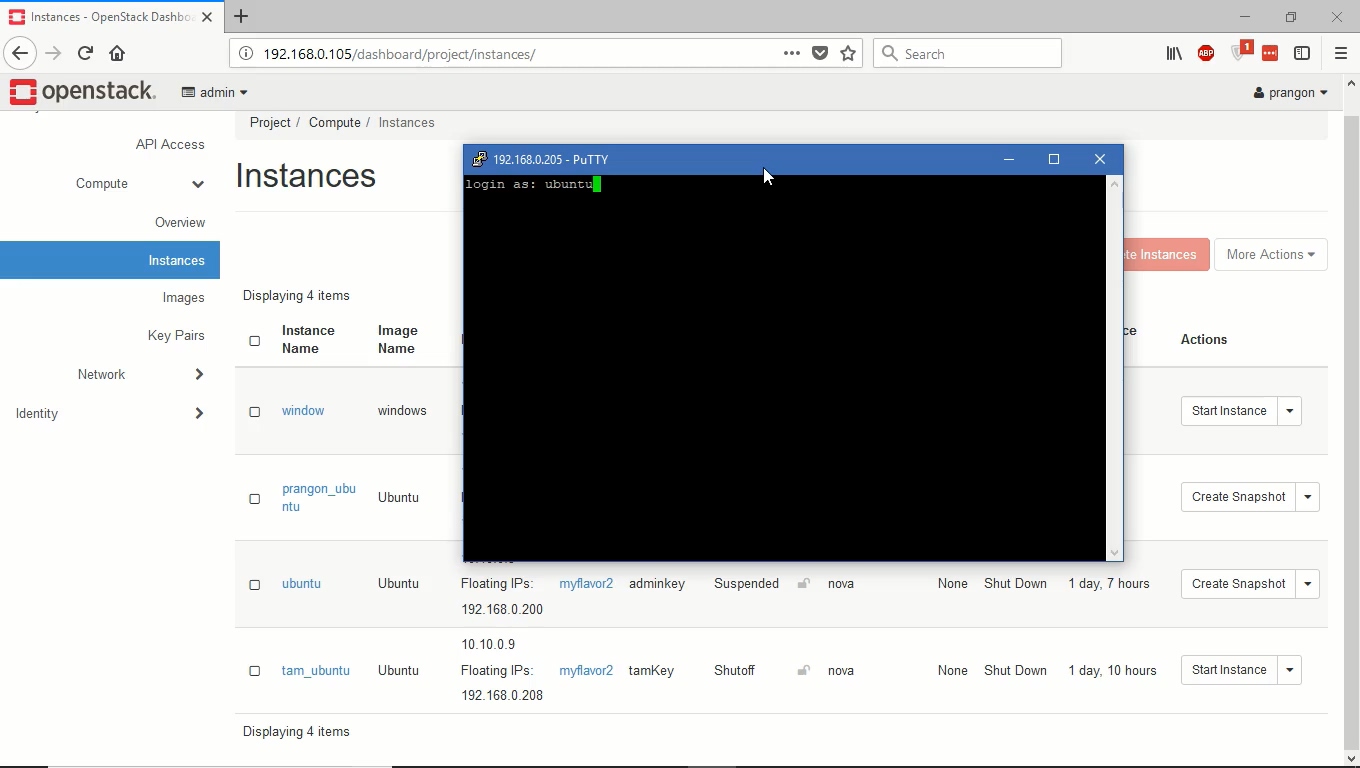


Figure 5‑13 Using Putty for Remote Login

# Conclusion

This project Designing and Implementation of Cloud Based Virtual Lab Environment provides a robust lab environment hosted on private cloud. This system is easy to maintain, requires little man power to run compared to traditional physical machine based environment. Configuring and running a new machine with different OS is a single click away. This system can ensure proper software is installed right when machine is first created. Persistent storage for individual students can also be enabled, which student can access anywhere from the same network.

Still There are limitations. As our test system has only 16GB of ram and a 4 core CPU, only 10 maximum number of command line based instances or virtual machines can be run concurrently from this particular setup. To serve more user, we need more memory and CPU power accordingly. If proper hardware is available, more power system which can serve more user can easily be deployed. Or underlying system can be made distributed so that availability and scalability becomes a feature.

This system minimizes the hassle of configuring a system and installing software, instead students and teachers alike can concentrate on research and development. We believe this cloud based virtual system is a proper next generation alternative to the traditional physical machine based lab environment.

# References

1. M. Klems, A. Lenk, J. Nimis, T. Sandholm and S. Tai. “What’s Inside the Cloud? An Architectural Map of the Cloud Landscape.” IEEE Xplore, pp 23-31, Jun. 2009. [↑](#endnote-ref-1)
2. Rochwerger, B.; Breitgand, D.; Levy, E.; Galis, A.; Nagin, K.; Llorente, I. M.; Montero, R.; Wolfsthal, Y.; Elmroth, E.; Caceres, J.; Ben-Yehuda, M.; Emmerich, W.; Galan, F. "The Reservoir model and architecture for open federated cloud computing". IBM Journal of Research and Development. [↑](#endnote-ref-2)
3. Diaz, Javier et al. " Abstract Image Management and Universal Image Registration for Cloud and HPC Infrastructures ", IEEE 5th International Conference on Cloud Computing (CLOUD), 2012. [↑](#endnote-ref-3)
4. HAMDAQA, Mohammad (2012). Cloud Computing Uncovered: A Research Landscape. Elsevier Press. ISBN 0-12-396535-7. [↑](#endnote-ref-4)
5. He, Sijin; Guo, L.; Guo, Y.; Ghanem, M. "Improving Resource Utilisation in the Cloud Environment Using Multivariate Probabilistic Models". 2012 2012 IEEE 5th International Conference on Cloud Computing, doi:10.1109/CLOUD.2012.66. ISBN 978-1-4673-2892-0. [↑](#endnote-ref-5)
6. Kostantos, Konstantinos, et al. "OPEN-source IaaS fit for purpose: a comparison between OpenNebula and OpenStack." International Journal of Electronic Business Management 11.3 (2013). [↑](#endnote-ref-6)
7. Marston, Sean; Li, Zhi; Bandyopadhyay, Subhajyoti; Zhang, Juheng; Ghalsasi, Anand (2011-04-01). "Cloud computing – The business perspective". Decision Support Systems. 51 (1). doi:10.1016/j.dss.2010.12.006 [↑](#endnote-ref-7)
8. Qevani, Elton, et al. "What can OpenStack adopt from a Ganeti-based open-source IaaS?." Cloud Computing (CLOUD), 2014 IEEE 7th International Conference on. IEEE, 2014. [↑](#endnote-ref-8)
9. Von Laszewski, Gregor, et al. "Comparison of multiple cloud frameworks.", IEEE 5th International Conference on Cloud Computing (CLOUD), 2012. [↑](#endnote-ref-9)
10. Mao, Ming; M. Humphrey (2012). "A Performance Study on the VM Startup Time in the Cloud". Proceedings of 2012 IEEE 5th International Conference on Cloud Computing (Cloud2012). doi:10.1109/CLOUD.2012.103. ISBN 978-1-4673-2892-0. [↑](#endnote-ref-10)
11. Wei-Tek Tsai, Bingnan Xiao, Ray Paul, Qian Huang, and Yinong Chen, "Global Software Enterprise: A New Software Constructing Architecture," in International Conference on E-Commerce, 2006 [↑](#endnote-ref-11)
12. Mark Chang, Jackson He, W.T. Tsai, Bingnan Xiao, and Yinong Chen, "UCSOA: User-Centric Service-Oriented Architecture," in IEEE International Conference on e-Business Engineering, 2006. [↑](#endnote-ref-12)
13. Yun Yang, Lie Wu and Wenping Hu, "Security architecture and key technologies for power cloud computing," Proceedings 2011 International Conference on Transportation, Mechanical, and Electrical Engineering (TMEE), Changchun, 2011, pp. 1717-1720. doi: 10.1109/TMEE.2011.6199543. [↑](#endnote-ref-13)
14. A Platform Computing Whitepaper. “Enterprise Cloud Computing: Transforming IT.” Platform Computing, 2010. [↑](#endnote-ref-14)
15. M. Jensen, J. Schwenk, N. Gruschka and L. L. Iacono, "On Technical Security Issues in Cloud Computing." in PROC IEEE ICCC, Bangalore 2009. [↑](#endnote-ref-15)
16. N. Leavitt. “Is Cloud Computing Really Ready for Prime Time?” Computer, vol. 42, 2009. [↑](#endnote-ref-16)
17. Khoa Huynh; Stefan Hajnoczi (2010). "KVM/QEMU Storage Stack Performance Discussion". ibm.com. Linux Plumbers Conference. Retrieved January 3, 2015. [↑](#endnote-ref-17)
18. US patent 5129088, Auslander, et al., "Data Processing Method to Create Virtual Disks from Non-Contiguous Groups of Logically Contiguous Addressable Blocks of Direct Access Storage Device", issued 1992-7-7 (fundamental patent). [↑](#endnote-ref-18)
19. Netto, M.; Calheiros, R.; Rodrigues, E.; Cunha, R.; Buyya, R. (2018). "HPC Cloud for Scientific and Business Applications: Taxonomy, Vision, and Research Challenges". ACM Computing Surveys. 51 (1): 8:1–8:29. doi:10.1145/3150224. [↑](#endnote-ref-19)
20. Yang, Choawei (2016). "Big Data and cloud computing: innovation opportunities and challenges". International Journal of Digital Earth (1). doi:10.1080/17538947.2016.1239771. [↑](#endnote-ref-20)
21. Vincenzo D. Cunsolo, Salvatore Distefano, Antonio Puliafito, Marco Scarpa: Volunteer Computing and Desktop Cloud: The Cloud@Home Paradigm. IEEE International Symposium on Network Computing and Applications, NCA 2009. [↑](#endnote-ref-21)
22. Sbarski, Peter (2017-05-04). Serverless Architectures on AWS: With examples using AWS Lambda (1st ed.). Manning Publications. ISBN 9781617293825. [↑](#endnote-ref-22)
23. Boniface, M.; et al. (2010). Platform-as-a-Service Architecture for Real-Time Quality of Service Management in Clouds. 5th International Conference on Internet and Web Applications and Services (ICIW). Barcelona, Spain: IEEE. doi:10.1109/ICIW.2010.91 [↑](#endnote-ref-23)
24. Amies, Alex; Sluiman, Harm; Tong, Qiang Guo; Liu, Guo Ning (July 2012). "Infrastructure as a Service Cloud Concepts". Developing and Hosting Applications on the Cloud. IBM Press. ISBN 978-0-13-306684-5. [↑](#endnote-ref-24)
25. Duan, Yucong; Fu, Guohua; Zhou, Nianjun; Sun, Xiaobing; Narendra, Nanjangud; Hu, Bo. "Everything as a Service (XaaS) on the Cloud: Origins, Current and Future Trends". IEEE. [↑](#endnote-ref-25)
26. Peter Mell; Timothy Grance (September 2011). The NIST Definition of Cloud Computing (Technical report). National Institute of Standards and Technology: U.S. Department of Commerce. doi:10.6028/NIST.SP.800-145. [↑](#endnote-ref-26)
27. He, Sijin; L. Guo; Y. Guo; C. Wu; M. Ghanem; R. Han. "Elastic Application Container: A Lightweight Approach for Cloud Resource Provisioning". 2012 IEEE 26th International Conference on Advanced Information Networking and Applications (AINA). doi:10.1109/AINA.2012.74. ISBN 978-1-4673-07147. [↑](#endnote-ref-27)
28. “Cloud computing – The business perspective". Decision Support Systems. 51 (1). doi:10.1016/j.dss.2010.12.006 [↑](#endnote-ref-28)