KLE Society's
KLE Technological University, Hubballi.

Company project on

A Minor Project Report

On

Access Control Policies Enforcement in OpenStack Cloud

submitted in partial fulfillment of the requirement for the degree of

Bachelor of Engineering

In

School of Computer Science and Engineering

Submitted By

SEEMA G AGER	01FE19BCS097
SHREYAS S KORTI	01FE19BCS106
VIRUPAKSHA B M	01FE20BCS425
KARTHIK KULKARNI	01FE20BCS426

Under the guidance of
Ms. HEENAKOUSAR

SCHOOL OF COMPUTER SCIENCE & ENGINEERING

HUBBALLI – 580 031

Academic year 2021-22

KLE Society's
KLE Technological University, Hubballi.

2020 - 2021

SCHOOL OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that Minor Project entitled “Access Control Policies Enforcement in OpenStack Cloud” is a bonafied work carried out by the student team Seema G Ager USN:01fe19bcs097, Shreyas S Korti USN:01fe19bcs106, Virupaksha B M USN:01fe20bcs425, Karthik Kulkarni USN:01fe20bcs426, in partial fulfillment of completion of Sixth semester B. E. in Computer Science and Engineering during the year 2020-2021. The project report has been approved as it satisfies the academic requirement with respect to the project work prescribed for the above said program.

Guide	Head, SoCSE
Ms. Heenakousar	Dr. Meena S. M

External Viva -Voce:

Name of the Examiners	Signature with date
1.	
2.	

Acknowledgement

We would like to thank our faculty and management for their professional guidance towards the completion of the project work. We take this opportunity to thank Dr. Ashok Shettar, Vice-Chancellor, Dr. N.H Ayachit, Registrar, Dr. P.G Tewari, and Dr. Narayan D G, Dean Academics, KLE Technological University, Hubballi, for their vision and support.

We also take this opportunity to thank Dr. Meena S. M, Professor and Head, SoCSE for having provided us direction and facilitated for enhancement of skills and academic growth.

We thank our guide Heenakousar, Assistant Professor, SoCSE for the constant guidance during interaction and reviews.

We extend our acknowledgement to the reviewers for critical suggestions and inputs. We also thank Project Co-Ordinator Mr. Uday N.Kulkarni, Mr. Guruprasad Konnurmath and all the reviewers for their support during the reviews and course of completion.

We express gratitude to our beloved parents for constant encouragement and support.

Seema G Ager - 01FE19BCS097
Shreyas S Korti - 01FE19BCS106
Virupaksha B M - 01FE20BCS425
Karthik Kulkarni - 01FE20BCS426

ABSTRACT

Cloud platforms are important and current topics of research interest and OpenStack is one of the most well-known and most used of them. As more companies start to use it to create and manage their virtual machines, virtual subnets, and Virtual networks, protection, and firewall usage get an increasing number of attention since OpenStack has a Firewall as a Service (FWaaS) module which allows packet filtering on virtual routers that join the subnets. The Firewall-as-a-Service (FWaaS) plug-in adds perimeter firewall control to OpenStack Networking (neutron). FWaaS makes use of iptables to apply firewall policy to all virtual routers inside a mission and supports one firewall policy and logical firewall instance consistent with the task. FWaaS operates at the fringe by means of filtering site visitors on the OpenStack Networking (neutron) router. This distinguishes it from security corporations, which perform at the example degree. We propose and put in force a firewalling services for network gadgets with the use of OpenStack. it may be utilized to provide firewall capabilities and it helps a rich set of packet filtering competencies, from the hyperlink layer as much as the software layer. The provider is lightweight, however, suggests that it may prevent efficaciously threats from outside of the networks with a low stage of useful resource intake.

Keywords : *Firewall as a Service, Virtual Router, Virtual Machine, Cloud Platform, OpenStack.*

CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

The cloud is a network-based environment where computations and resources are shared. Cloud computing is actually an Internet-based technology that tries to hide complexity from clients. Cloud computing refers to both applications that are delivered as services over the Internet and the hardware and software within the datacenters from which those services are provided. Virtualization technologies are used by cloud providers in conjunction with self-service capabilities to supply computing resources via network infrastructure. Cloud environments host several types of virtual machines on the same physical server as infrastructure.

The risks associated with network security have become increasingly severe with the development of science and technology networks. This is why firewall technology plays such an important role in network security. In recent years, firewall technology has evolved through several stages such as packet filtering, application proxy generation, stage detection generation, and so on. The management and deployment of multiple physical firewalls will increase the value of maintaining network security gadgets and increase the workload of network security equipment managers. With virtual firewall (FWaaS) technology, these problems are solved by partitioning a physical firewall into several logically independent virtual firewalls, each with an independent supervisor, device assets, user authentication, and security rules.

OpenStack is an open-source software program that manages a big pool of cloud resources. Compute, storage, and network are the most important resources OpenStack offer to its tenants. Those resources are configured and controlled through OpenStack REST full API, through the command-line interface, or the use of the OpenStack dashboard. OpenStack architecture includes a control node, one or more compute nodes and a network node. Optionally it may encompass one or extra storage nodes.The three-node architecture of OpenStack consists of one control node, one compute node and one network node. In our case, the most interesting part is networking. Virtual machines can be organized into virtual subnets, which can form virtual networks. Between networks, routers act as gateways, connecting one to another. while routers are useful as gateways, on every certainty one of them a single packet filtering firewall instance can be run. This provider is referred to as Firewall as a service (FWaaS). OpenStack can use various firewall software through drivers, and they all may be controlled by the identical API or the dashboard. For this, OpenStack FWaaS have a generalized structure for firewall rules, rulesets, and many others., from which the drivers generate configuration documents for the different firewalls. The generalized statistics can be gathered from OpenStack via the Networking API.

The valuable ideas with OpenStack firewalls are the notions of a firewall coverage and a firewall rule. A coverage is an ordered collection of regulations. A rule specifies a group of attributes (which include port ranges, protocol, and IP addresses) that constitute in shape criteria and an action to take (allow or deny) on matched traffic. A policy can be made public, so it could be shared throughout initiatives. Here we have used openstack cloud service to simulate a network in such a way it should be managed and easily they can mdified either removing and adding instances and it may be adding multiple routers so that when ever the communication happens in network it should be end-to-end encryption within this network only their is lot of hidden sensitive information their is a chance of getting leakage to over come this problem we have impemented firewall so that attacking chances are less when we are connected to ISP(Internet service provider) which is not safe so we have implemented firewall to overule this all problem. Here we are blocking all the instances through firewall via router by adding the policies.

1.1 Motivation

OpenStack network API may be used to achieve generalized information about the network topology, and the firewall rulesets, irrespective of the implementation of the firewall capability. This type of study was now not been completed before, and it offers a usual answer for many exclusive firewalls, which have drivers in OpenStack.

In this situation, the clients are system admins, whose mission is to maintain the network, and the firewall rulesets in it. while a large range of virtual networks and hosts are controlled in OpenStack, many regulations are needed in packet filtering, which isn't smooth to understand. The users are in need of equipment that assists them in the interpretation of the topology and the policies, perceive security leaks in the network, and find conflicts within the rulesets.

1.2 Problem Statement

Access control policies enforcement in OpenStack cloud

Approach:

In USA most of the companies get hacked due to the single layer security policy which is not safe by this all sensitive information get leaked by this they have to face lots of problem, to overcome this problem they have to implement multiple firewall by this security will be more and it will be difficult to hack the system.

1.3 Applications

- Firewall as a Service (FWaaS) of OpenStack secures the tenant's network.
- Virtual firewalls can secure public cloud services from providers such as Google Cloud Platform (GCP), Amazon Web Services (AWS), and Microsoft Azure
- Virtual firewalls can help secure virtual branch offices as well as software-defined networks and software-defined wide-area networks – SDN and SD-WAN, respectively.

1.4 Objectives and Scope of the project

A firewall is a network security device that monitors and filters incoming and outgoing network traffic based on an organization's previously established security policies. The objective and scope is

1.4.1 Objectives

- To Install OpenStack in the devstack environment.
- To implement a security policy.
- Only authorized traffic, as defined by the local security policy, will be allowed to pass
- To manage network traffic
- To protect the system from malware and hackers
- A virtual firewall prevents an unauthorized user from accessing and transmitting data and prevents an organization's employees from transforming sensitive data or documents.
- All traffic from inside to outside and vice versa must pass through the fire-wall. This is achieved by physically blocking all access to the local network except via the firewall.

1.4.2 Scope of the project

Our goal is to support all the functionalities in the dev stack environment FWaaS v2 specification. Given FWaaS v2 is still under development, a few features are not ready, this spec only covers the basic firewall functionalities, including map the Neutron FWaaS v2 data model to Dragon flow controller and apply the firewall rules to router interfaces. Address group, service group and applying firewall rules to VM ports, SFC ports are not implemented in Neutron FWaaS v2 yet, so they will not be covered in this specification.

Chapter 2

Related Work

2.1 Types of Protocols

2.1.1 Application Layer

The application layer is present at the top of the OSI model. It is the layer through which users interact. It provides services to the user.

Application Layer Protocols

- TELNET : Telnet stands for the TELetype NETwork. It helps in terminal emulation. It allows Telnet clients to access the resources of the Telnet server. It is used for managing files on the internet. It is used for the initial setup of devices like switches. The telnet command is a command that uses the Telnet protocol to communicate with a remote device or system. Port number of telnet is 23.
- FTP : FTP stands for file transfer protocol. It is the protocol that actually lets us transfer files. It can facilitate this between any two machines using it. But FTP is not just a protocol but it is also a program.FTP promotes sharing of files via remote computers with reliable and efficient data transfer. The Port number for FTP is 20 for data and 21 for control.
- TFTP : The Trivial File Transfer Protocol (TFTP) is the stripped-down, stock version of FTP, but it’s the protocol of choice if you know exactly what you want and where to find it. It’s a technology for transferring files between network devices and is a simplified version of FTP. The Port number for TFTP is 69.
- NFS : It stands for a network file system. It allows remote hosts to mount file systems over a network and interact with those file systems as though they are mounted locally. This enables system administrators to consolidate resources onto centralized servers on the network. The Port number for NFS is 2049.
- SMTP : It stands for Simple Mail Transfer Protocol. It is a part of the TCP/IP protocol. Using a process called “store and forward,” SMTP moves your email on and across networks. It works closely with something called the Mail Transfer Agent (MTA) to

send your communication to the right computer and email inbox. The Port number for SMTP is 25.

- LPD : It stands for Line Printer Daemon. It is designed for printer sharing. It is the part that receives and processes the request. A "daemon" is a server or agent. The Port number for LPD is 515.
- SNMP : It stands for Simple Network Management Protocol. It gathers data by polling the devices on the network from a management station at fixed or random intervals, requiring them to disclose certain information. It is a way that servers can share information about their current state, and also a channel through which an administrate can modify pre-defined values. The Port number of SNMP is 161(TCP) and 162(UDP).
- DNS : It stands for Domain Name System. Every time you use a domain name, therefore, a DNS service must translate the name into the corresponding IP address. For example, the domain name www.abc.com might translate to 198.105.232.4. The Port number for DNS is 53.
- DHCP : It stands for Dynamic Host Configuration Protocol (DHCP). It gives IP addresses to hosts. There is a lot of information a DHCP server can provide to a host when the host is registering for an IP address with the DHCP server. Port number for DHCP is 67, 68.

2.1.2 Transport Layer

Transport Layer is the second layer of the TCP/IP model. It is an end-to-end layer used to deliver messages to a host. It is termed as an end-to-end layer because it provides a point-to-point connection rather than hop-to- hop, between the source host and destination host to deliver the services reliably. The unit of data encapsulation in the Transport Layer is a segment.

The standard protocols used by Transport Layer to enhance its functionalities are TCP, UDP(User Datagram Protocol), DCCP(Datagram Congestion Control Protocol), etc.

Transport Layer Protocols

- UDP : UDP stands for User Datagram Protocol. UDP is a simple protocol and it provides non sequenced transport functionality. UDP is a connection less protocol. This type of protocol is used when reliability and security are less important than speed and size. UDP is an end-to-end transport level protocol that adds transport-level addresses, checksum error control, and length information to the data from the upper layer. The packet produced by the UDP protocol is known as a user datagram.

- TCP : TCP stands for Transmission Control Protocol. It provides full transport layer services to applications. It is a connection-oriented protocol means the connection established between both the ends of the transmission. For creating the connection, TCP generates a virtual circuit between sender and receiver for the duration of a transmission.

2.1.3 Network Layer

Network layer protocols exist in every host or router. The router examines the header fields of all the IP packets that pass through it. Networking software is used to attach the header to each data packet sent as well as to read it to determine how the packet is handled at the receiving end.

Transport Layer Protocols

- IP : IP stands for "Internet Protocol," which is the set of rules governing the format of data sent via the internet or local network. In essence, IP addresses are the identifier that allows information to be sent between devices on a network: they contain location information and make devices accessible for communication.

- ARP : Address Resolution Protocol (ARP) is a protocol or procedure that connects an ever-changing Internet Protocol (IP) address to a fixed physical machine address, also known as a media access control (MAC) address, in a local-area network (LAN).

- ICMP : The Internet Control Message Protocol (ICMP) is a protocol that devices within a network use to communicate problems with data transmission. In this ICMP definition, one of the primary ways in which ICMP is used is to determine if data is getting to its destination and at the right time.

2.2 Literature Survey

In [1], They present a firewall policy algebra for constructing and reasoning over anomaly-free policies. The algebra allows a policy specifier to compose policies in such a way that the result of the composition upholds the access requirements of each policy involved, and allows one to reason as to whether some policy is a safe (secure) replacement for another policy. The proposed algebra is used to reason about OpenStack cloud deployments. OpenStack is an open-source cloud operating system. It provides Infrastructure-as-a-Service (IaaS) through a collection of interrelated projects/services. These services are used to manage pools of computing, storage, and networking resources throughout a data center. This paper focuses on the OpenStack Neutron networking service.

In [2], The work automates the deployment of OpenStack Firewall as a service, Neutron with users selected network scenarios, technologies, and other basic services of OpenStack with GUI. This system supports a multi-node cluster cloud environment. The behavior of the firewall is tested on different Neutron scenarios. Still, there is a need for a system that will troubleshoot error that occurs during deployment, based on error patterns. And also, they're in need to automate other OpenStack services like Load Balancer as a Service, etc.

In [3], A brief summary of virtual firewalls and security mechanisms implemented in a different cloud environment is given. The basic design of the virtual firewall which includes cloud infrastructure and the virtual firewall as well as the deployment of virtual machines and implementation process is outlined. The virtual firewall will be deployed in the VMM, and the filtering procedure will be based on packet attributes and host information. In the next study, we will increase the filtering procedures, and protocols and evaluate the performance of the firewall.

In [4], A FWaaS system based on the OpenStack cloud resource platform is designed to provide users with a secure, controllable, automatic scheduling, on-demand, and instant service firewall system. The new model of detecting the efficiency of FWaaS effectively verifies its optimal deployment scheme and provides the most convenient and efficient service for the FWaaS vendors and users. However, the FWaaS designed in the experiment corresponds to the service project one-to-one, and its resources can't be better utilized and allocated. The following research enables users of the same project to share an FWaaS resource, and its specific operation still needs to be further studied.

In [5], Another firewall analysis tool is presented. It focuses on user-end scenarios and decomposes the Cisco IOS firewall's configuration into policies. Margrave maps both policies and queries into first-order logic formulas. To answer a query, its formula is conjoined with all references policy formulas, and then a solution is computed. Kodkod is used to solve the first-order formulas using SAT solving.

In [6], A VFW system based on OpenStack cloud resource platform is designed to provide users with a secure, controllable, automatic scheduling, on-demand and instant service firewall system. The new model of detecting the efficiency of VFW effectively verifies its optimal deployment scheme and provides the most convenient and efficient service for the VFW vendors and users. However, the VFW designed in experiment corresponds to the service project in one to one, and its resources can't be better utilized and allocated. The following research enables users of the same project to share a VFW resource, and its specific operation still needs to be further studied.

In [7], The major recommendations from In general, a firewall should fit into a current network's layout. However, an organization might change its network architecture at the same time as it deploys a firewall as part of an overall security upgrade. Different common network architectures lead to very different choices for where to place a firewall, so an organization should assess which architecture works best for its security goals. If an edge firewall has a DMZ, consider which outward-facing services should be run from the DMZ and which should remain on the inside network. Do not rely on NATs to provide the benefits of firewalls. In some environments, putting one firewall behind another may lead to a desired security goal, but in general such multiple layers of firewalls can be troublesome.

In[8], Network Function Virtualization will completely reshape the approach of telco operators to provide existing as well as novel network services, taking advantage of the increased flexibility and reduced deployment costs of the cloud computing paradigm. In this work, the problem of evaluating complexity and performance, in terms of sustainable packet rate, of virtual networking in cloud computing infrastructures dedicated to NFV deployment was addressed. An OpenStack-based cloud platform was considered and deeply analyzed to fully understand the architecture of its virtual network infrastructure. To this end, an ad hoc visual tool was also developed that graphically plots the different functional blocks (and related interconnections) put in place by Neutron, the OpenStack networking service. The main outcome of this work is that an open-source cloud computing platform such as OpenStack can be effectively adopted to deploy NFV in network edge data centers replacing legacy telco central offices. However, this solution poses some limitations to the network performance which are not simply related to the hosting hardware maximum capacity but also to the virtual network architecture implemented by OpenStack.

Chapter 3

REQUIREMENT ANALYSIS

The FWaaS system of cloud service platforms based on OpenStack is analysed with functional requirements and non-functional requirements, and the functional modules of the FWaaS are divided.

3.1 Functional Requirements

1. Firewall is used to enable the applications visibility and has the access control authority for only admin.
2. Firewall must decrypt outbound SSL.
3. Firewall must have multiple layers in between ISP(Internet service provider).
4. Firewall must identify control proxies,remote access and encrypted tunnel applications.
5. Firewall must able to deal with unknown traffic through policies and so. ....etc.
6. Firewall must enable the same application visibility and control for admin as for on-premise users.

3.2 Non Functional Requirements

1. system should be under surveillance 24*7
2. Sytem should be easy maintain and update the Software.
3. System should be able to block unwanted malicious threats
4. System performance will more than 90 percent and less than 95 percent.
5. It should block unwanted message as spam detection warning.
6. Always system as to give warning while accessing unsecured website.

3.3 Software Requirements

Cloud Service OpenStack

OpenStack is a set of software tools for building and managing cloud computing platforms for public and private clouds. Backed by some of the biggest companies in software development and hosting, as well as thousands of individual community members, many think that OpenStack is the future of cloud computing. OpenStack is managed by the OpenStack Foundation, a non-profit that oversees both development and community-building around the project. The cloud is all about providing computing for end-users in a remote environment, where the actual software runs as a service on reliable and scalable servers rather than on each end-user's computer. Cloud computing can refer to a lot of different things, but typically the industry talks about running different items "as a service"—software, platforms, and infrastructure. OpenStack falls into the latter category and is considered Infrastructure as a Service (IaaS). Providing infrastructure means that OpenStack makes it easy for users to quickly add new instances, upon which other cloud components can run. Typically, the infrastructure then runs a "platform" upon which a developer can create software applications that are delivered to the end-users.

Chapter 4

SYSTEM DESIGN

4.1 Architecture Design

FWaaS v1

The original FWaaS implementation, v1, provides protection for routers. When a firewall is applied to a router, all internal ports are protected. The following diagram depicts FWaaS v1 protection. It illustrates the flow of ingress and egress traffic for the VM2 instance:

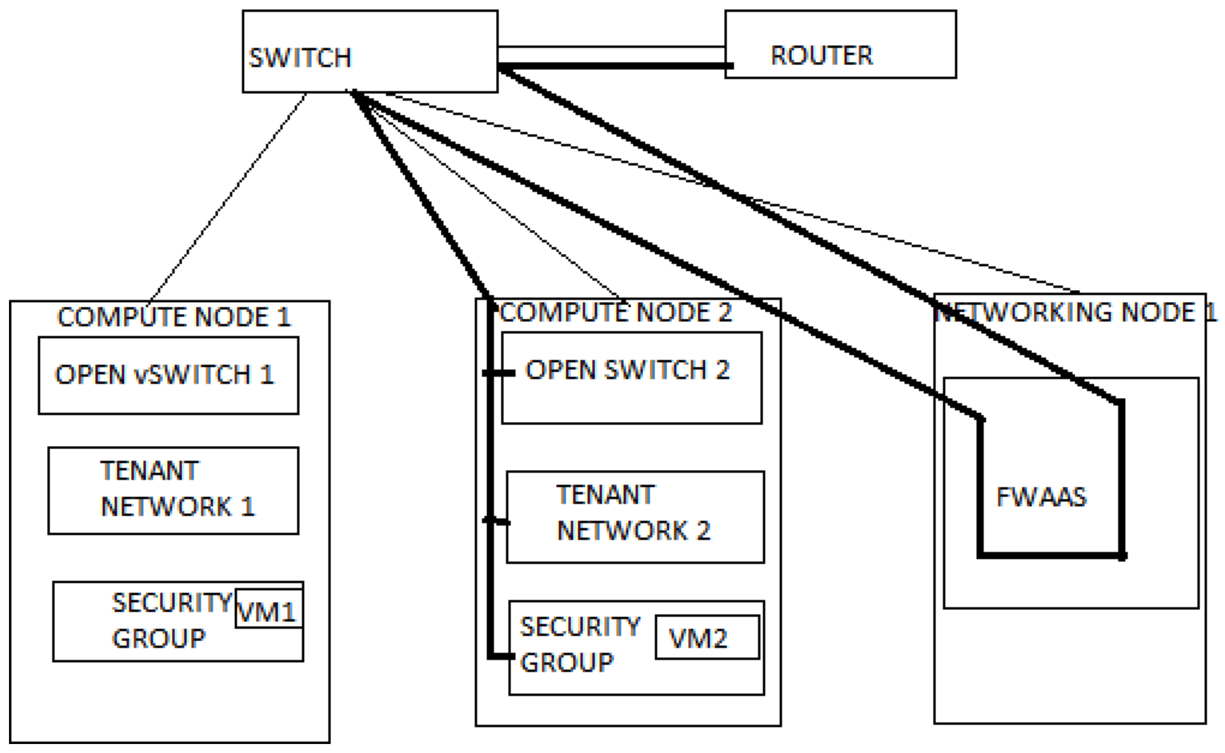


Figure 4.1: FWaaS v1

4.2 User Interface Design

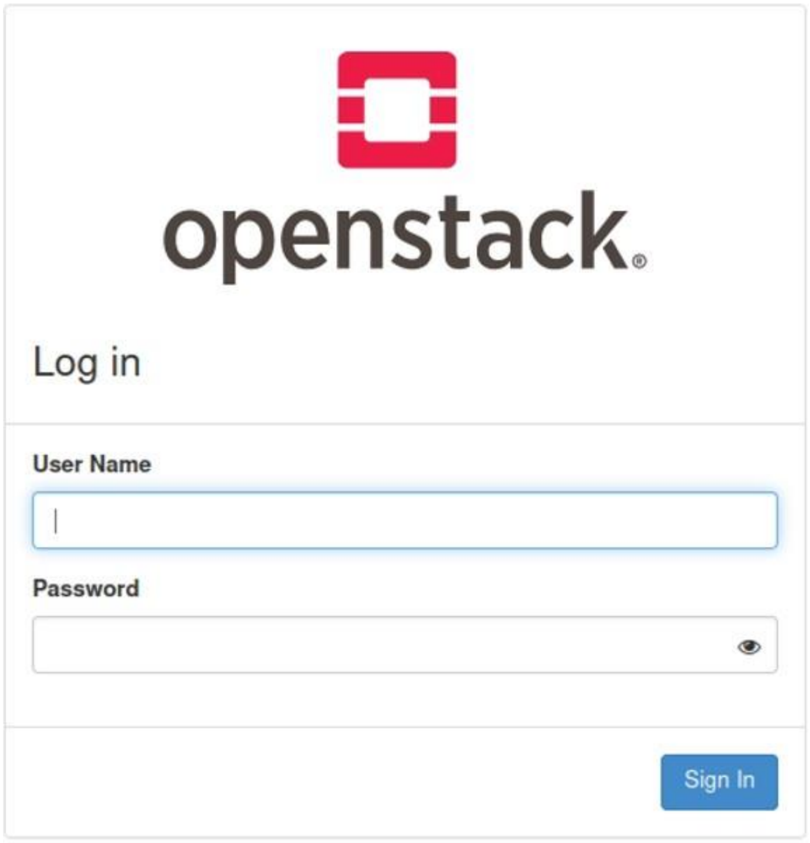
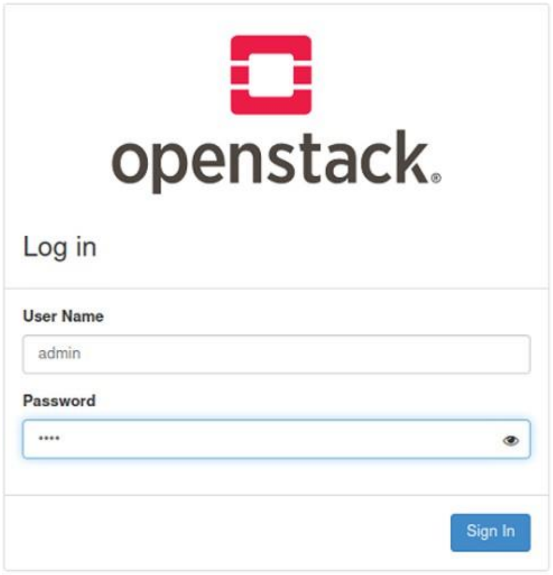


Figure 4.2: OpenStack Login Page

When we open the Openstack cloud operating system then we will see this OpenStack login page. Here we are going to login to the system.

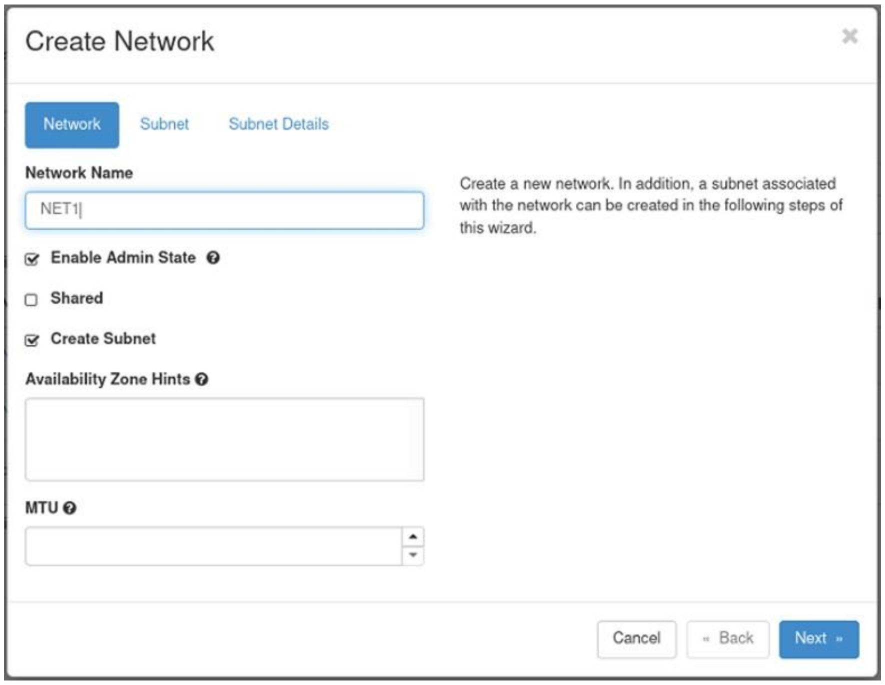


User Name: admin

Password: ••••

Figure 4.3: OpenStack Login

When we enter the User name as admin and Password as lab6and click on sign in button then we will be logged in.



Network Name: NET1

☑ Enable Admin State

☐ Shared

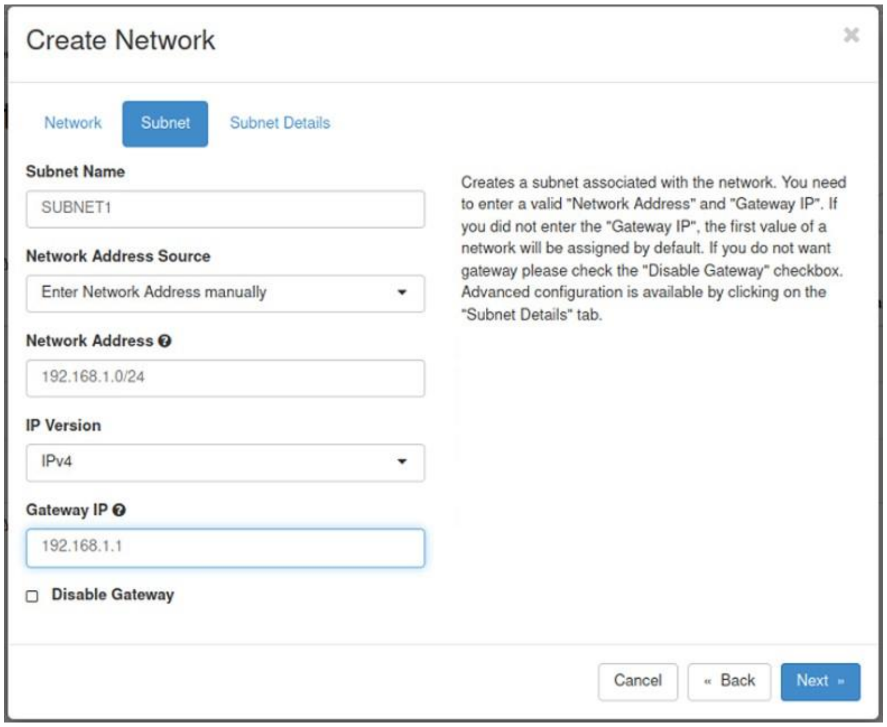
☑ Create Subnet

Availability Zone Hints

MTU

Create a new network. In addition, a subnet associated with the network can be created in the following steps of this wizard.

Figure 4.4: Naming Network



Create Network

Subnet Name: SUBNET1

Network Address Source: Enter Network Address manually

Network Address: 192.168.1.0/24

IP Version: IPv4

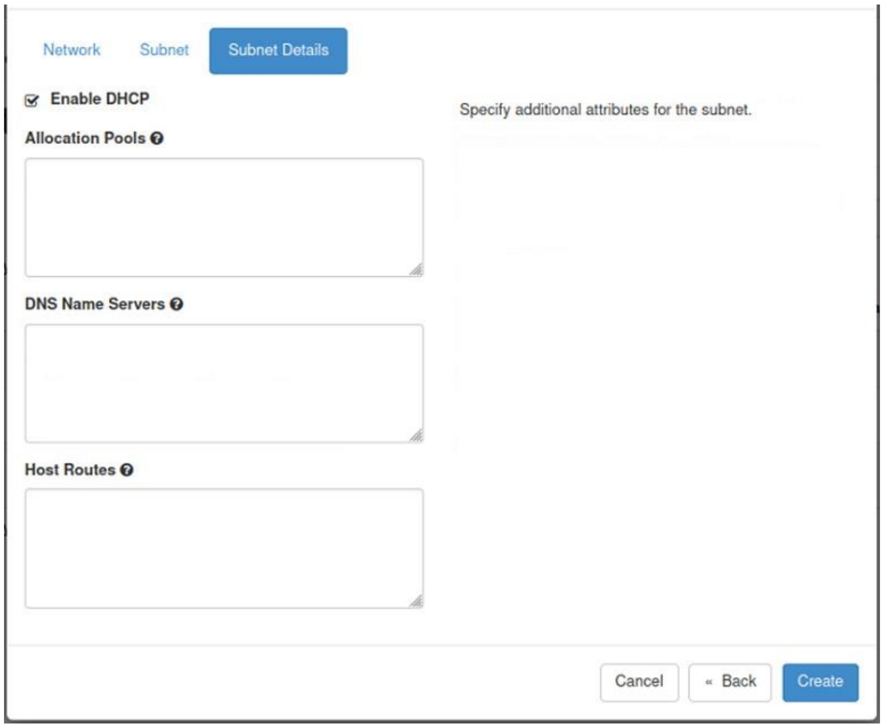
Gateway IP: 192.168.1.1

☐ Disable Gateway

Creates a subnet associated with the network. You need to enter a valid "Network Address" and "Gateway IP". If you did not enter the "Gateway IP", the first value of a network will be assigned by default. If you do not want gateway please check the "Disable Gateway" checkbox. Advanced configuration is available by clicking on the "Subnet Details" tab.

Figure 4.5: Create Network

Here we are going to create a network with an ip of 192.168.1.0/24 and named as SUBNET1. The IP version is selected as IPv4 and the gateway IP is 192.168.1.1.



☑ Enable DHCP

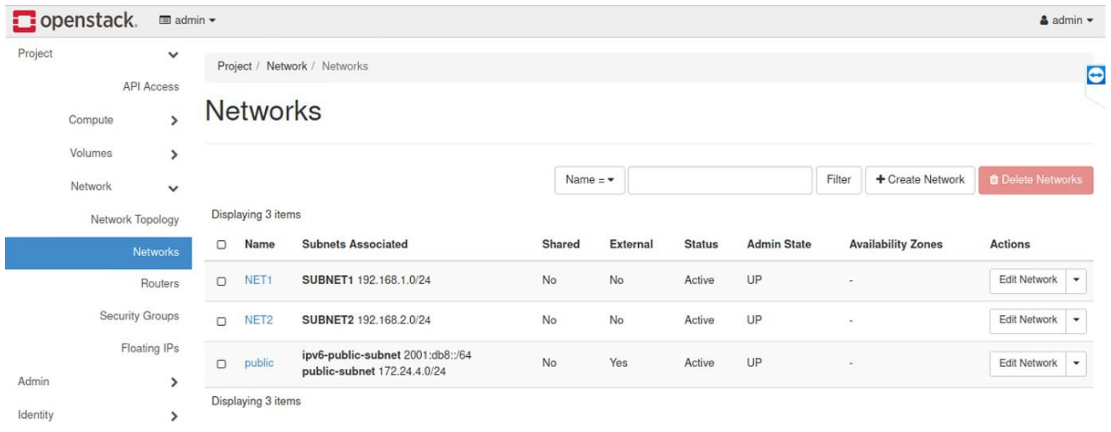
Allocation Pools

DNS Name Servers

Host Routes

Specify additional attributes for the subnet.

Figure 4.6: Enabling DHCP



Networks

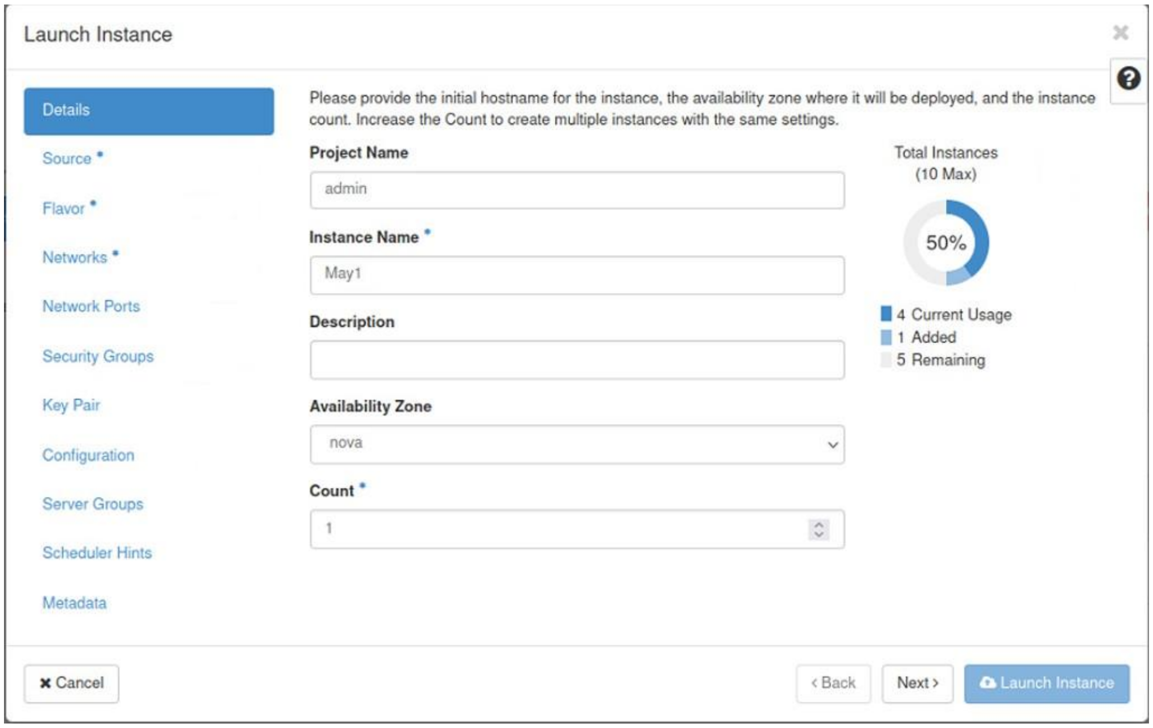
Displaying 3 items

	Name	Subnets Associated	Shared	External	Status	Admin State	Availability Zones	Actions
☐	NET1	**SUBNET1** 192.168.1.0/24	No	No	Active	UP	-	Edit Network
☐	NET2	**SUBNET2** 192.168.2.0/24	No	No	Active	UP	-	Edit Network
☐	public	**ipv6-public-subnet** 2001:db8::/64 **public-subnet** 172.24.4.0/24	No	Yes	Active	UP	-	Edit Network

Displaying 3 items

Figure 4.7: Networks

Here we have created two networks, NET1 and NET2 and the one public network.



Please provide the initial hostname for the instance, the availability zone where it will be deployed, and the instance count. Increase the Count to create multiple instances with the same settings.

Project Name: admin

Instance Name *: May1

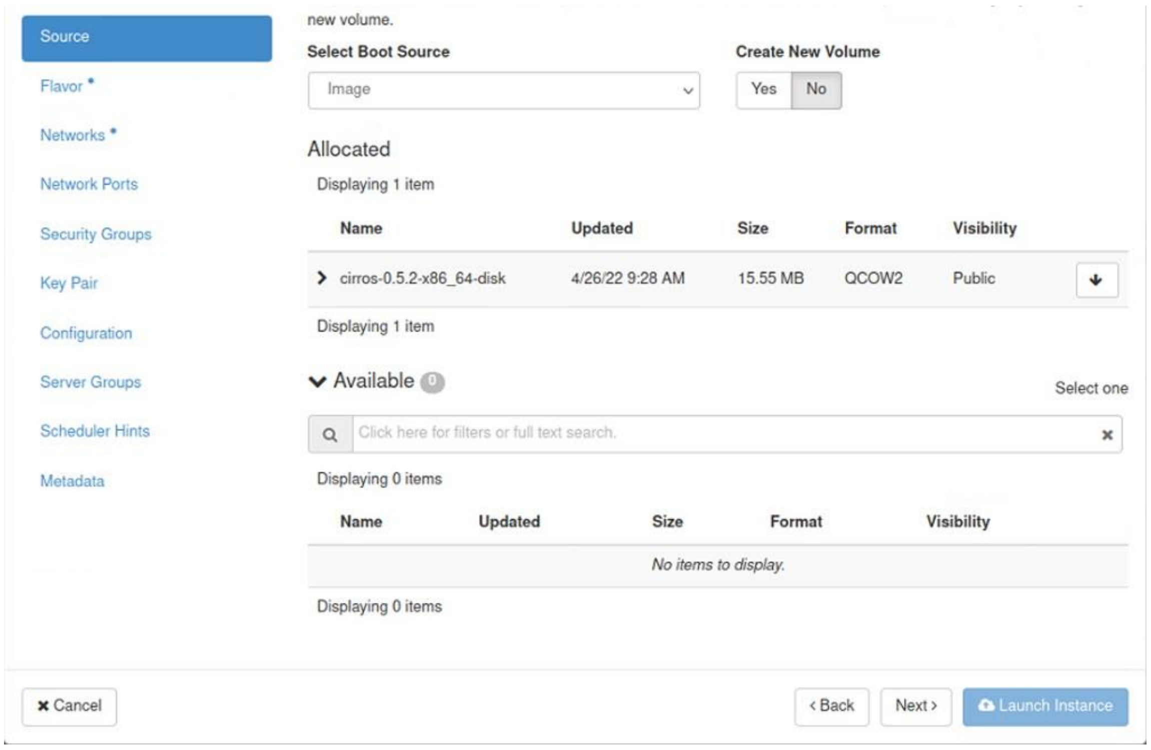
Description:

Availability Zone: nova

Count *: 1

4 Current Usage
1 Added
5 Remaining

Figure 4.8



new volume.

Select Boot Source: Image

Create New Volume: Yes [✓ No]

Allocated

Displaying 1 item

Name	Updated	Size	Format	Visibility
cirros-0.5.2-x86_64-disk	4/26/22 9:28 AM	15.55 MB	QCOW2	Public

Displaying 1 item

Available 0

Select one

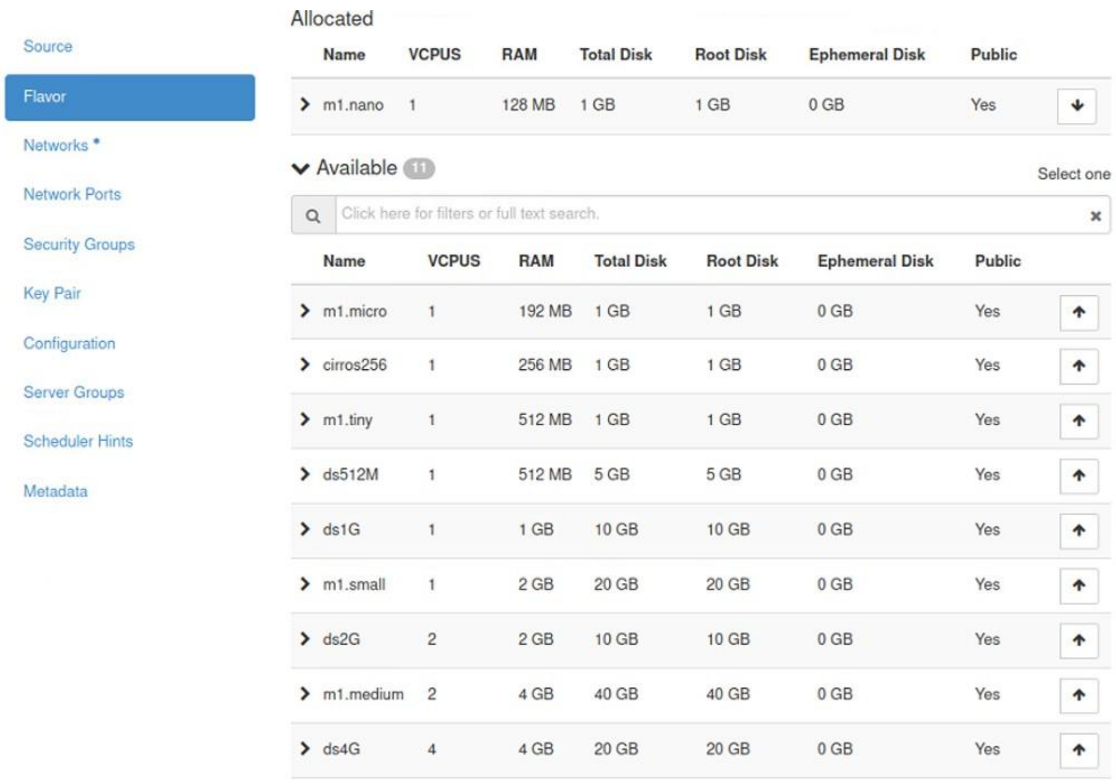
Click here for filters or full text search.

Displaying 0 items

Name	Updated	Size	Format	Visibility
No items to display.				

Displaying 0 items

Figure 4.9



Allocated

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
m1.nano	1	128 MB	1 GB	1 GB	0 GB	Yes

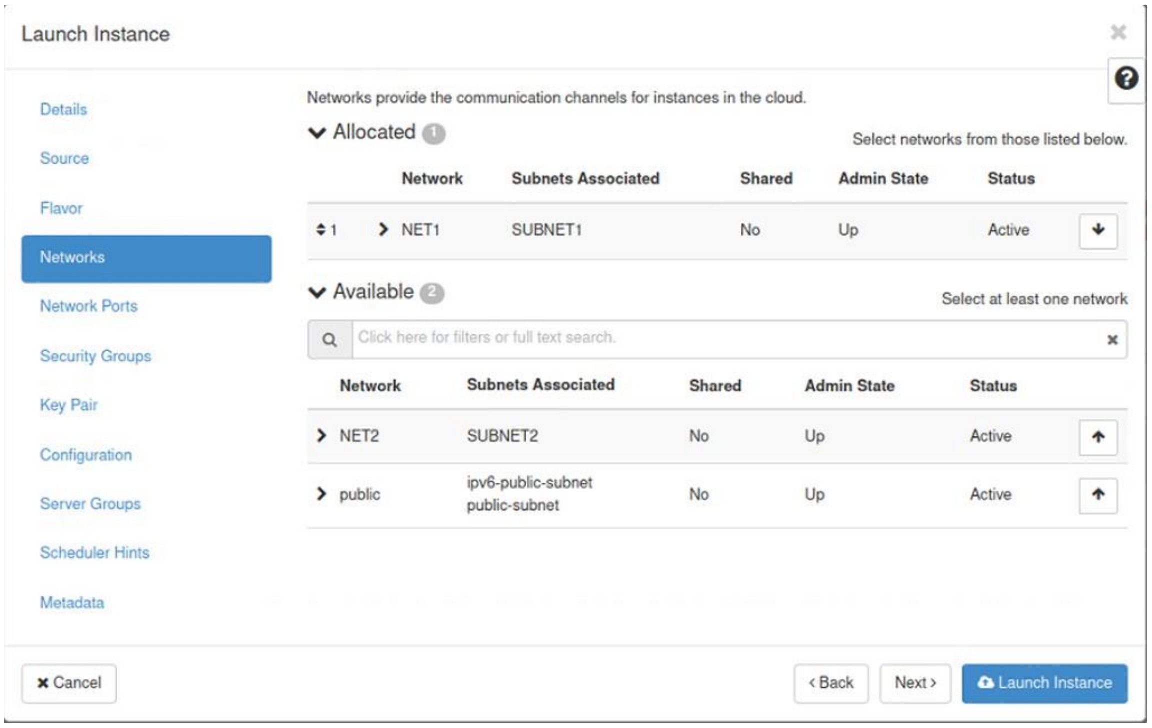
Available 11

Select one

Click here for filters or full text search.

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
m1.micro	1	192 MB	1 GB	1 GB	0 GB	Yes
cirros256	1	256 MB	1 GB	1 GB	0 GB	Yes
m1.tiny	1	512 MB	1 GB	1 GB	0 GB	Yes
ds512M	1	512 MB	5 GB	5 GB	0 GB	Yes
ds1G	1	1 GB	10 GB	10 GB	0 GB	Yes
m1.small	1	2 GB	20 GB	20 GB	0 GB	Yes
ds2G	2	2 GB	10 GB	10 GB	0 GB	Yes
m1.medium	2	4 GB	40 GB	40 GB	0 GB	Yes
ds4G	4	4 GB	20 GB	20 GB	0 GB	Yes

Figure 4.10



Networks provide the communication channels for instances in the cloud.

Allocated 1

Select networks from those listed below.

	Network	Subnets Associated	Shared	Admin State	Status
1	NET1	SUBNET1	No	Up	Active

Available 2

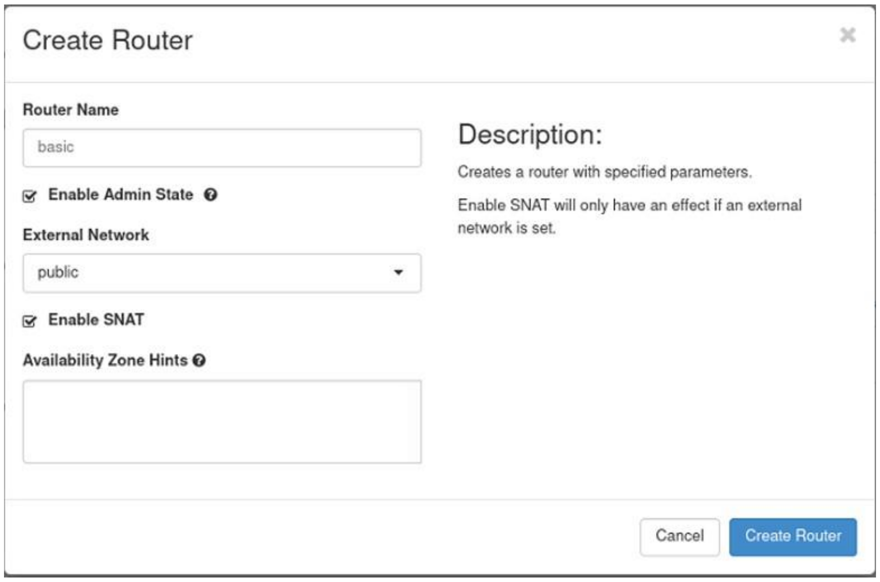
Select at least one network

Click here for filters or full text search.

Network	Subnets Associated	Shared	Admin State	Status
NET2	SUBNET2	No	Up	Active
public	ipv6-public-subnet public-subnet	No	Up	Active

Figure 4.11

Here we are going to create instances for NET1 and NET2. When we click on the launch instance will be created. Here we have created four instances.Every 2 instance is belonging to network1 and network2.



Router Name: basic

Enable Admin State ✓

External Network: public

Enable SNAT ✓

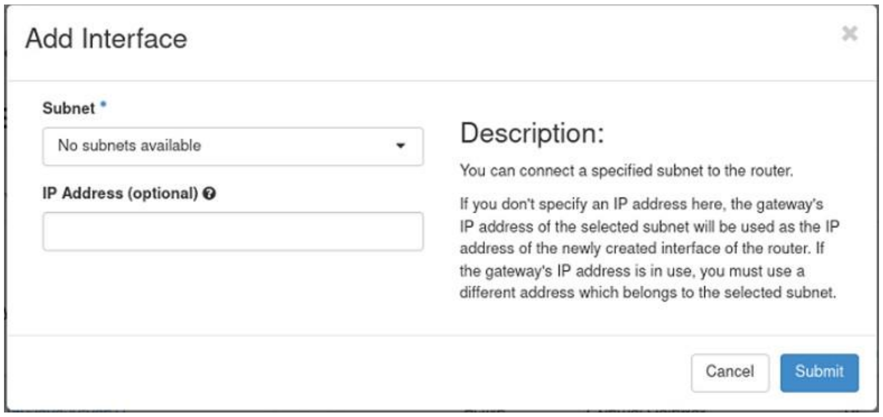
Availability Zone Hints

Description:

Creates a router with specified parameters.

Enable SNAT will only have an effect if an external network is set.

Figure 4.12: Create Router



Add Interface

Subnet *

No subnets available

IP Address (optional)

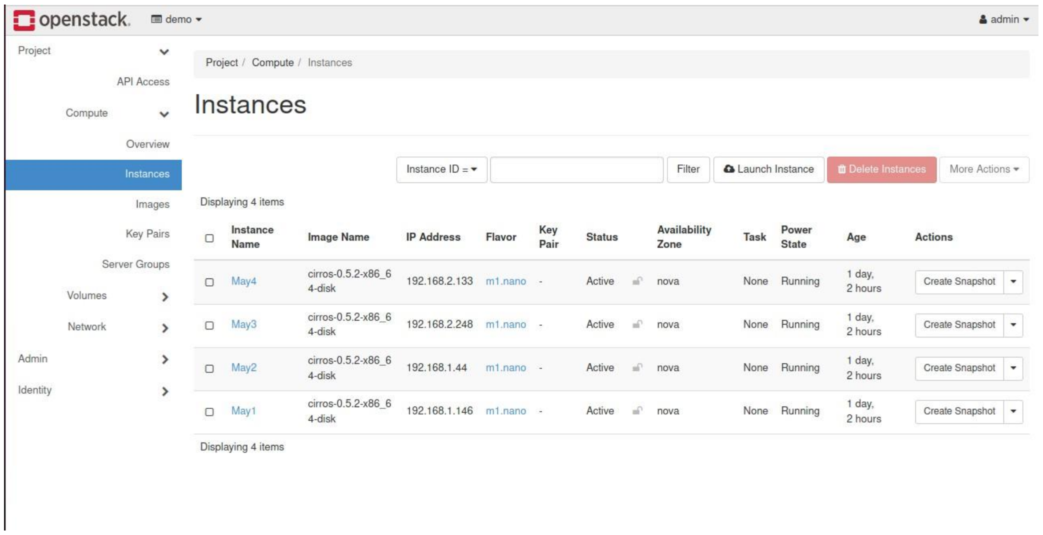
Description:

You can connect a specified subnet to the router.

If you don't specify an IP address here, the gateway's IP address of the selected subnet will be used as the IP address of the newly created interface of the router. If the gateway's IP address is in use, you must use a different address which belongs to the selected subnet.

Figure 4.13: Add Interface

Here we are going to create a router. while creating a router the sub-type we have choose public. then add policy so that it can connect to every network.



Instances

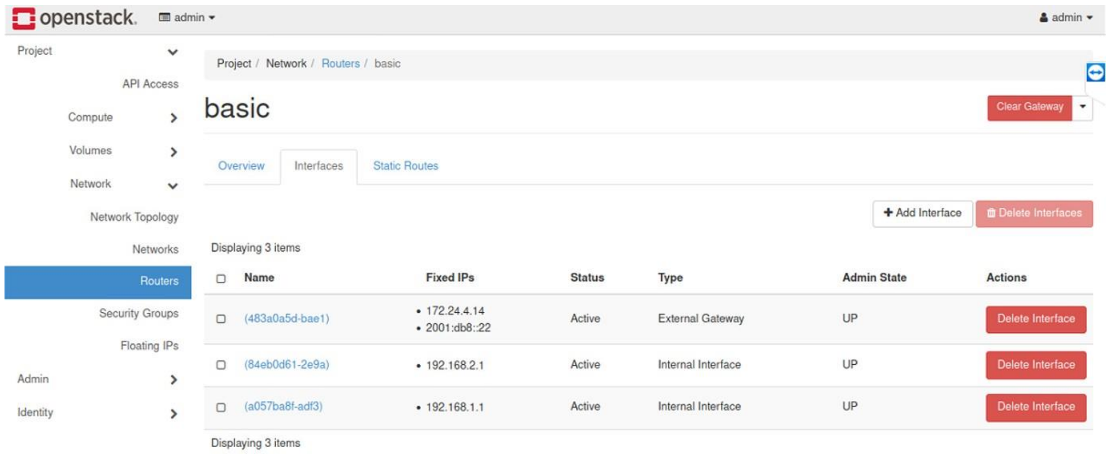
Displaying 4 items

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
May4	cirros-0.5.2-x86_64-disk	192.168.2.133	m1.nano	-	Active	nova	None	Running	1 day, 2 hours	Create Snapshot
May3	cirros-0.5.2-x86_64-disk	192.168.2.248	m1.nano	-	Active	nova	None	Running	1 day, 2 hours	Create Snapshot
May2	cirros-0.5.2-x86_64-disk	192.168.1.44	m1.nano	-	Active	nova	None	Running	1 day, 2 hours	Create Snapshot
May1	cirros-0.5.2-x86_64-disk	192.168.1.146	m1.nano	-	Active	nova	None	Running	1 day, 2 hours	Create Snapshot

Displaying 4 items

Figure 4.14: Instances

Here we can see all the instances that we have been created. Here May1, May2, May3 and May4 are the four instances which have been created. This page will be seen in Instances tab. This instances page contains Instances name, Image name, IP Address, Flavor, Kay pair, Status, Availability zone, Task, Power state, Age and Actions. If you want to create another instances, click on launch instances and then the same procedure will be continued which is explained earlier. You can delete the created instances also by clicking on delete instances button.

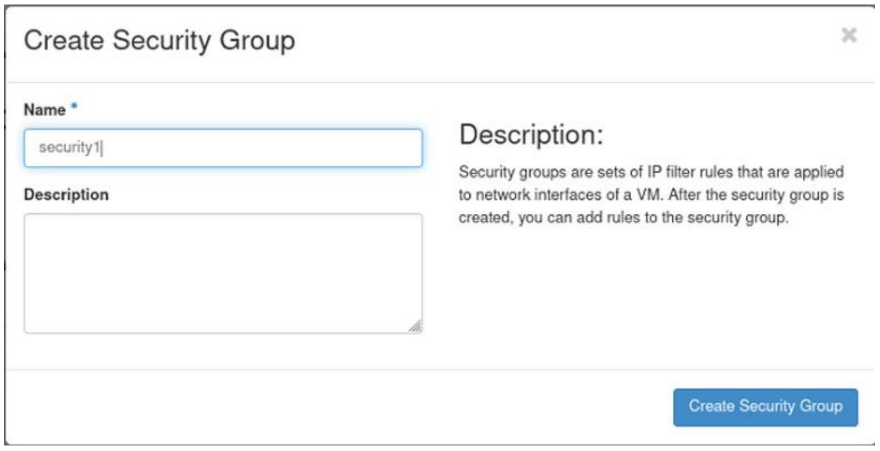


Displaying 3 items

Name	Fixed IPs	Status	Type	Admin State	Actions
(483a0a5d-bae1)	• 172.24.4.14 • 2001:db8::22	Active	External Gateway	UP	Delete Interface
(84eb0d61-2e9a)	• 192.168.2.1	Active	Internal Interface	UP	Delete Interface
(a057ba8f-adf3)	• 192.168.1.1	Active	Internal Interface	UP	Delete Interface

Displaying 3 items

Figure 4.15: Basic



Name *

security1

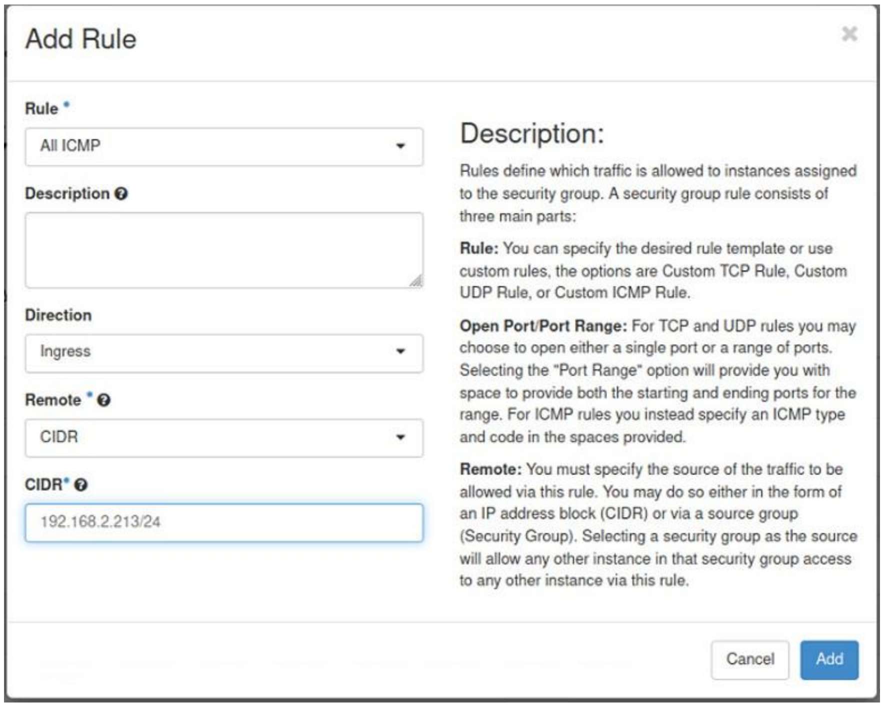
Description

Description:

Security groups are sets of IP filter rules that are applied to network interfaces of a VM. After the security group is created, you can add rules to the security group.

Figure 4.16: Create Security Group

Here we are going to create a security policy. First we have to put the name for security group. Then by clicking on create security Group, the security group name will be created.



Add Rule

Rule *
All ICMP

Description

Direction
Ingress

Remote *
CIDR

CIDR*
192.168.2.213/24

Description:

Rules define which traffic is allowed to instances assigned to the security group. A security group rule consists of three main parts:

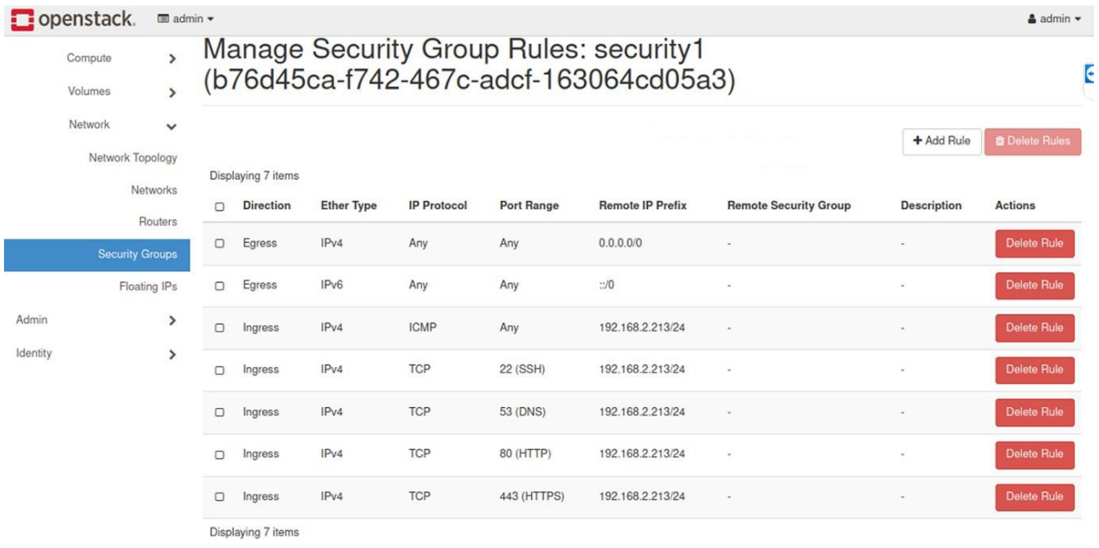
Rule: You can specify the desired rule template or use custom rules, the options are Custom TCP Rule, Custom UDP Rule, or Custom ICMP Rule.

Open Port/Port Range: For TCP and UDP rules you may choose to open either a single port or a range of ports. Selecting the "Port Range" option will provide you with space to provide both the starting and ending ports for the range. For ICMP rules you instead specify an ICMP type and code in the spaces provided.

Remote: You must specify the source of the traffic to be allowed via this rule. You may do so either in the form of an IP address block (CIDR) or via a source group (Security Group). Selecting a security group as the source will allow any other instance in that security group access to any other instance via this rule.

Cancel Add

Figure 4.17: Add Rule



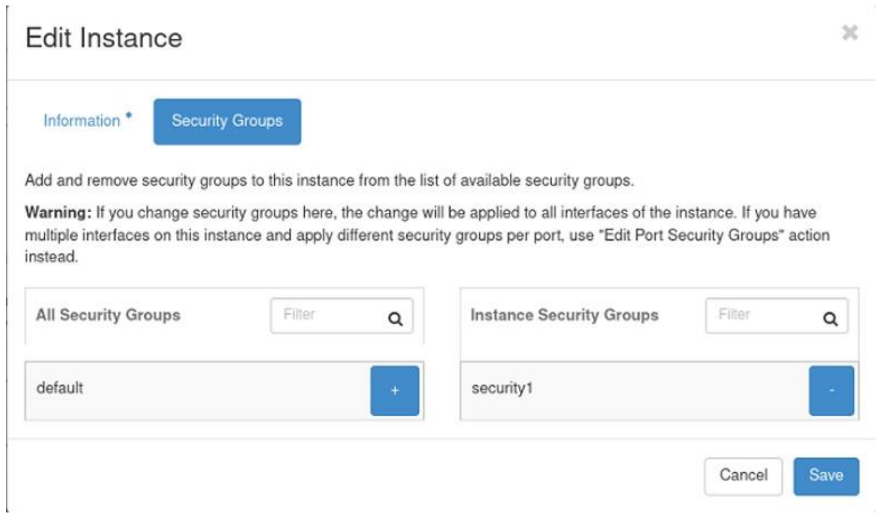
Manage Security Group Rules: security1 (b76d45ca-f742-467c-adcf-163064cd05a3)

Displaying 7 items

Direction	Ether Type	IP Protocol	Port Range	Remote IP Prefix	Remote Security Group	Description	Actions
Egress	IPv4	Any	Any	0.0.0.0/0	-	-	Delete Rule
Egress	IPv6	Any	Any	::/0	-	-	Delete Rule
Ingress	IPv4	ICMP	Any	192.168.2.213/24	-	-	Delete Rule
Ingress	IPv4	TCP	22 (SSH)	192.168.2.213/24	-	-	Delete Rule
Ingress	IPv4	TCP	53 (DNS)	192.168.2.213/24	-	-	Delete Rule
Ingress	IPv4	TCP	80 (HTTP)	192.168.2.213/24	-	-	Delete Rule
Ingress	IPv4	TCP	443 (HTTPS)	192.168.2.213/24	-	-	Delete Rule

Displaying 7 items

Figure 4.18: Manage Security Group Rules


Edit Instance

Add and remove security groups to this instance from the list of available security groups.

Warning: If you change security groups here, the change will be applied to all interfaces of the instance. If you have multiple interfaces on this instance and apply different security groups per port, use "Edit Port Security Groups" action instead.

All Security Groups	Instance Security Groups
default	security1

Figure 4.19: Edit Instance

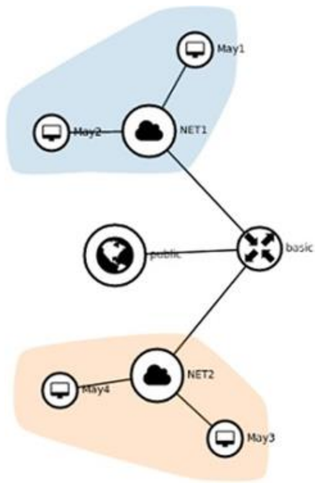
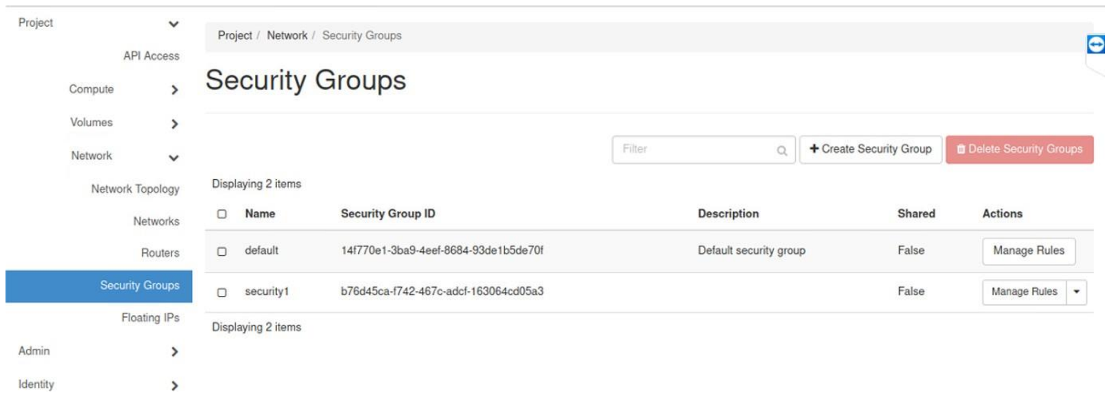


Figure 4.20: Network Graph



Security Groups

Displaying 2 items

Name	Security Group ID	Description	Shared	Actions
default	14f770e1-3ba9-4eef-8684-93de1b5de70f	Default security group	False	Manage Rules
security1	b76d45ca-f742-467c-adcf-163064cd05a3		False	Manage Rules

Displaying 2 items

Figure 4.21: Security Group

Once the security policy is created, then we have to click on manage rules. In that click on add rules. once you click on add rules you can able to see so many policies. once you apply any policiex like http,icmp....etc. once policies are created. Then go to router click on router in that go to static route their you see current instance ip address and net hop ip address. Then go to instances in that click on down arrow their you see the option of edit security group click on that then add the security policy by removing default policy. once all policies are added then we are good to go. we can block any instances through router through security policies.

Chapter 5

RESULTS AND DISCUSSIONS

5.1 Network Topology

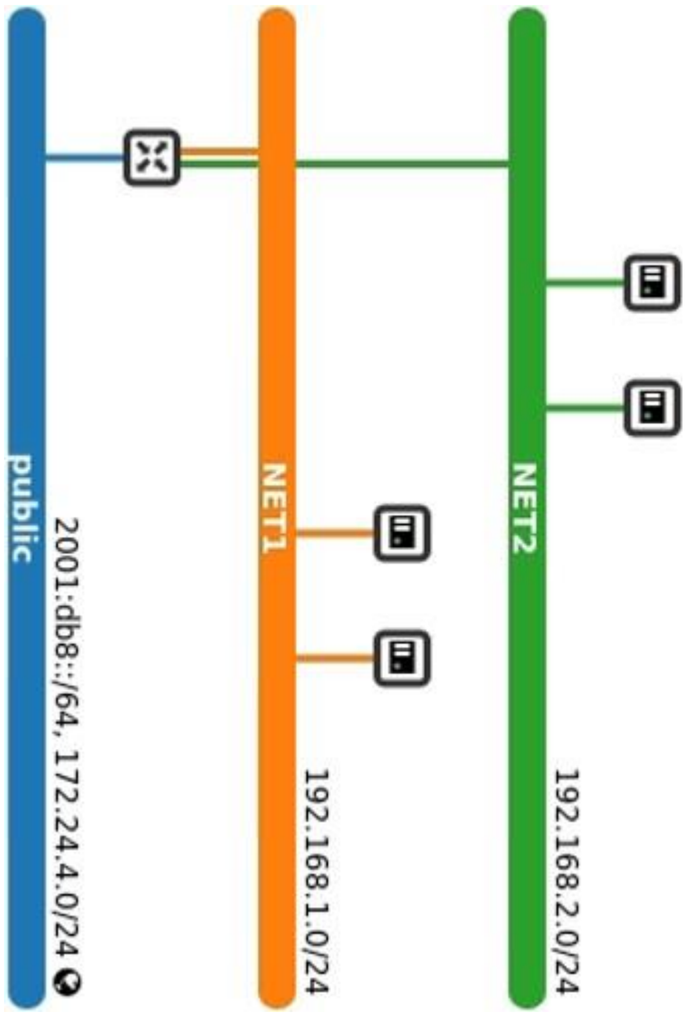


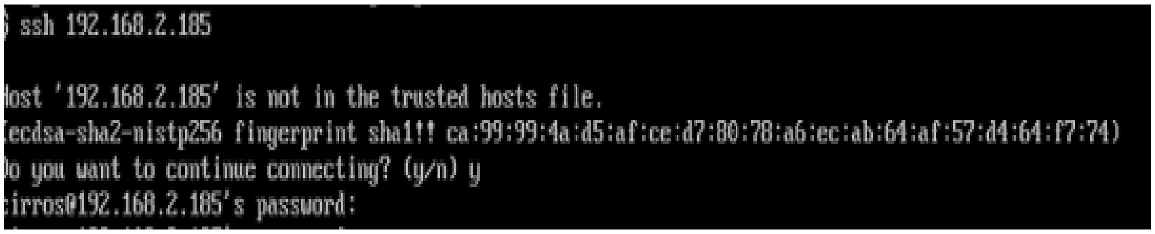
Figure 5.1: Network Topology

Above figure shows topology of our network which has two private networks and one public network. The public network IP address is 172.24.4.0/24 and private network NET1 IP address is 192.168.1.0/24 and another private network NET2 IP address is 192.168.2.0/24. NET1 and NET2 has two instances each.

5.2 Types of Protocols

5.2.1 Network layer:-

SSH:-



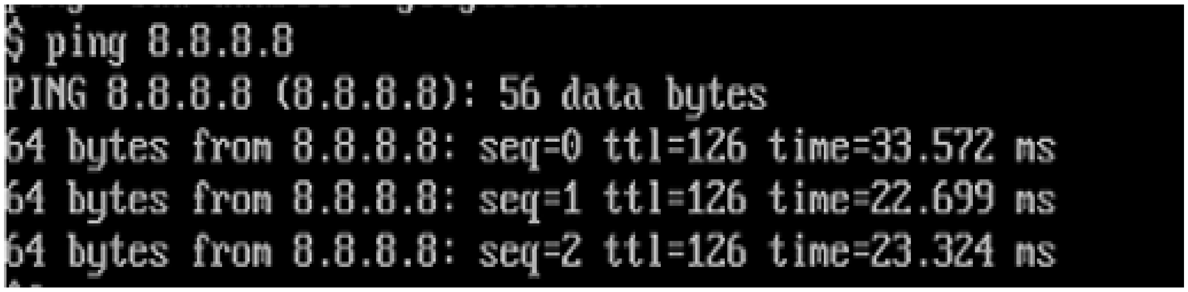
```
$ ssh 192.168.2.185

Host '192.168.2.185' is not in the trusted hosts file.
(ecdsa-sha2-nistp256 fingerprint sha1!! ca:99:99:4a:d5:af:ce:d7:80:78:a6:ec:ab:64:af:57:d4:64:f7:74)
Do you want to continue connecting? (y/n) y
cirros@192.168.2.185's password:
```

1. Basically ssh is a security shell once the policies are added to the instance and once we enter command ssh@ip address then it will give warning message that if you want to continue to access that instance press ‘Y’ else ‘ N’.

5.2.2 Transport layer:-

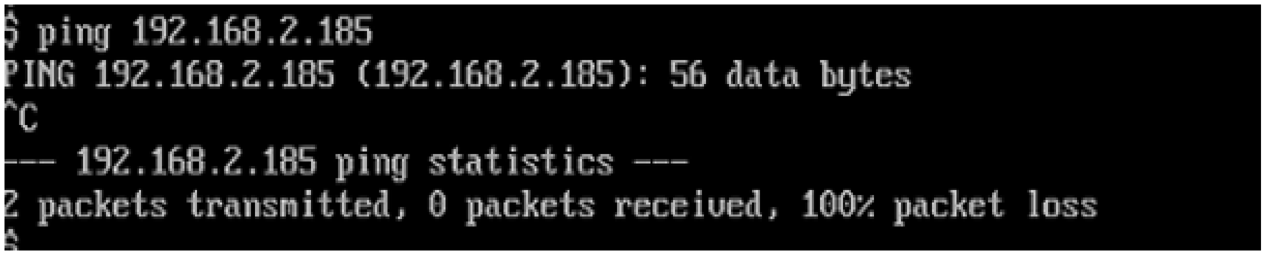
ICMP:-



```
$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: seq=0 ttl=126 time=33.572 ms
64 bytes from 8.8.8.8: seq=1 ttl=126 time=22.699 ms
64 bytes from 8.8.8.8: seq=2 ttl=126 time=23.324 ms
```

ICMP (Internet control message protocol) it is used to ping the instances with in the connected network and the once the policies re added tothe instances via router then it will not allowthe access.

5.2.3 Application layer:-



```
$ ping 192.168.2.185
PING 192.168.2.185 (192.168.2.185): 56 data bytes
^C
--- 192.168.2.185 ping statistics ---
2 packets transmitted, 0 packets received, 100% packet loss
$
```

Dns:-

Basically Dns stand for (Domain name server) when ever the user want to access google.com then it will search that at ip address in that database and it will fullfill the user query what ever he has typed in search engine through dns server he can attain result. Here also we have added policies to the instances.

Chapter 6

CONCLUSION

OpenStack is one of the most popular and widely used cloud platforms today. It can set up and manage virtual machines, which can be organized into huge virtual networks, connected by virtual routers, which can run simple packet filtering firewall instances. It is hard for system administrators to understand and maintain firewall rules in enormous networks. That is where our tool comes in.

REFERENCES

5.2.1 Mayer, A., Wool, A., Ziskind, E.: Offline firewall analysis. Int. J. Inf. Secur. 5(3), 125–144 (2006)

5.2.2 Nelson, T., Barratt, C., Dougherty, D.J., Fisler, K., Krishnamurthi, S.: The margrave tool for firewall analysis. In: LISA (2010)

5.2.3 Liu, A.X., Gouda, M.G.: Firewall policy queries. IEEE Trans. Parallel Distrib. Syst.
20(6), 766–777 (2009)

5.2.4 Stallings, W.: Handbook of Computer-Communications Standards; Vol 1: The Open Systems Interconnection (OSI) Model and OSI-Related Standards. Macmillan Publishing Co. Inc., New York (1987)

5.2.5 OpenStack Open Source Cloud Computing Software, 2 February 2016. Devstack - an OpenStack Community Production, 19 January 2016. Bartal, Y., Mayer, A., Nissim, K., Wool, A.: Firmato: a novel firewall management toolkit. ACM Trans. Comput. Syst. (TOCS) 22(4), 381–420 (2004) Torlak, E., Jackson, D.: Kodkod: a relational model finder. In: Grumberg, O., Huth, M. (eds.) TACAS 2007. LNCS, vol. 4424, pp. 632–647. Springer, Heidelberg (2007)

5.2.6 A. Applebaum, Z M. Li, and K. Levitt, "Firewall configuration: An application of multiagent metalevel argumentation," Argument and Computation, vol. 7, no. 2-3, pp. 201-221, 2016.

5.2.7 D.A Tirtharaj, "Intrusion detection using neural networks trained with evolutionary algorithms," Soft Computing, vol. 21, no. 10, pp.2687-2700, 2017.

5.2.8 C. Liu, J. Yang, and J. Q. Wu, "Web intrusion detection system combined with fea- ture analysis and SVM optimization," EURASIP Journal on Wireless Communications and Networking, pp.743-752, 2020.

5.2.9 B. Lv, Q. L. Gu, and Y. C. Zhi, "A Comparative Study on Cloud Platform Model of Chinese and American Maker Space," International Journal of Economics, vol. 7, no. 6, pp.187-196, 2019.

5.2.10 H. X. Jin, Y. Y. Fu, and G. L. Yang, “An intelligent scheduling algorithm for resource management of cloud platform”, Multimedia Tools and Applications: An International Jour- nal, vol. 79, no. 5, pp.5335-5353, 2020.

5.2.11 N. N. Sun, Y. Li, and L. H. Ma, “Research on cloud computing in the resource sharing system of university library services”, Evolutionary Intelligence, vol. 12, no. 3, pp. 377-384, 2019.

5.2.12 K. X. Liang, J. Y. Yu, and X. Wu, “Air-Conditioning Resource Management and Control Method based on Cloud Platform for Wind Power Consumption,” Processes, vol. 7, no. 7, pp. 467, 2019.

5.2.13 H. C. Toumi, F. Z. Fagroud, and A. Zakouni, “Implementing Hy-IDS, Mobiles Agents and Virtual Firewall to Enhance the Security in IaaS Cloud,” Procedia Computer Science,vol. 160, pp. 819-824, 2019.

5.2.14 S.I. Kuribayashi, “Allocation of Virtual Firewall Functions in NFV-Based Networks with Minimum Network Cost,” International Journal of Computer Networks and Communications,vol. 11, no. 2, pp.53-67, 2019.

[16] A. Sari, “Turkish national cyber-firewall to mitigate countrywide cyber-attacks,” Com- puters and Electrical Engineering, vol. 73, pp. 128-144, 2019.

[17] N. Huin, A, Tomassilli, and F. Giroire, “Energy-Efficient Service Function Chain Pro- visioning,” Electronic Notes in Discrete Mathematics, vol. 64, pp. 265-274, 2018.

[18] D. Naghmeh, and S. Saeed, “Learning-based dynamic scalable load-balanced firewall as a service in network function-virtualized cloud computing environments,” The Journal of Supercomputing, vol. 74, no. 7, pp. 3329-3358, 2018.

[19] S. K. Panda, and P. K. Jana, “Load balanced task scheduling for cloud computing: a probabilistic approach,” Knowledge and Information Systems, vol. 61, no. 3, pp. 1607-1631, 2019.

[20] X. Sui, D. Liu, and L. Li, “Virtual machine scheduling strategy based on machine learning algorithms for load balancing,” EURASIP Journal on Wireless Communications and Networking, pp.1-16, 2019.

[21] Y. Z. Li, S. Gao, and J. Pan, “Research and Application of Template Engine for Web Back-end Based on MyBatis-Plus,” Procedia Computer Science, vol. 166, pp. 206-212, 2020.

[22] C. Artho, Q. Gros, and G. Rousset, “Model-Based API Testing of Apache ZooKeeper,” international conference on software testing verification and validation, pp. 288-298, 2017.

[23] A. Khan, and A. Mathelier, “JASPAR RESTful API: accessing JASPAR data from any programming language,” Bioinformatics, vol. 34, no. 9, pp. 1612-1614, 2018.

[24] S. Kumari, and Deepak, “REST based API,” International Journal of Trend in Scientific Research and Development, vol. 1, no. 4, pp. 571-575, 2017.

[25] N. Niknejad, W. Ismail, and I. Ghani, “Understanding Service-Oriented Architecture (SOA): A systematic literature review and directions for further investigation,” Information Systems, vol. 91, pp. 101491, 2020.

[26] Y. L. Oladimejia, “SOADIWA: A Service-oriented Architecture for Data Interoperability in Web Applications,” Asian Journal of Research in Computer Science, vol. 4, no. 3, pp. 1-15, 2019.

[27] A. U. Khan, and S. Bagchi, “Software architecture and algorithm for reliable RPC for geo-distributed mobile computing systems,” Future Generation Computer Systems, vol. 86, pp. 185-198., 2018.

[28] M. Kekely, L. Kekely, and J. Kořenek, “General memory efficient packet matching FPGA architecture for future high-speed networks,” Microprocessors and Microsystems, vol. 73, pp. 102950, 2020.

[29] S. Prabakaran, and R. Ramar, “Stateful firewall-enabled software-defined network with distributed controllers: A network performance study,” International Journal of Communication Systems, vol. 32, no. 17, pp. 4237, 2019.