Network and Security Challenges in Cloud Computing Infrastructure as a Service Model

Anahita Abdollahifar

Department of Electrical Engineering and Computer

University of Stavanger, Norway

Spring 2013

Abstract

The aim of the project is improving available approaches in Cloud Computing security. Because there is some limitations for understanding there is not enough information that how available Clouds are working and security approaches are engaged. The mentioned data is limited by commercial vendors. So, an open source Cloud was implemented and security concerns were studied.
OpenStack and DevStack were chosen as an open source Clouds. Different modules of platform and their internal communication were taken in consideration. Because the main focus of the project is on the network component, a variety of network background structures were studied. Different security drawbacks were comprehended and listed as threats in network part. But the key concern is security between virtual machine of different customers.
There is lots of approaches for preventing and detecting security threats. In current observation detection is provided by intrusion detection system. The detective solution is employed for multiple compromising. But in this observation with usage of intrusion detection system, abnormal behaviors among the virtual machines of two customers were identified.

Acknowledgements

I would like to thank my supervisor, Dr.Chunming Rong, and my advisory committee, who gave me the directions and ideas during working on this dissertation. I also want to give my sincere thanks to my advisor, Aryan Taheri Monfared, who helped and encouraged me in all steps of this project. My parents receive my deepest gratitude and love for their dedication and the many years of support during my life that provided the foundation for this work.

Contents

1 Introduction **1**
- 1.1 Cloud Computing OverView . . . 1
  - 1.1.1 Service Model . . . 2
  - 1.1.2 Deployment Model . . . 2
- 1.2 Contribution and Report Structure . . . 3
  - 1.2.1 OpenStack and Available Network Structures . . . 3
  - 1.2.2 Installing OpenStack and DevStack . . . 4
  - 1.2.3 Available Network and Security Issues . . . 4
  - 1.2.4 Intrusion Detection System and Snort . . . 4
  - 1.2.5 Monitoring . . . 4

2 Design and Implementation **5**
- 2.0.6 Communication between Components . . . 5
- 2.1 OpenStack . . . 6
- 2.2 OpenStack Networking . . . 7
- 2.3 DevStack . . . 12
  - 2.3.1 Spawning Projects . . . 14
  - 2.3.2 Namespace networking in Linux . . . 17

3 Security Challenges and Solution Issues **18**
- 3.1 Available Threats . . . 18
- 3.2 Anomaly Detection . . . 20
  - 3.2.1 Input Data Type . . . 20
  - 3.2.2 Data Labels . . . 20
  - 3.2.3 Output . . . 20
- 3.3 Intrusion Detection and Traditional Approaches . . . 21
  - 3.3.1 Signature . . . 22
  - 3.3.2 Snort . . . 22
- 3.4 Issues of IDS . . . 23
- 3.5 What and Where Should be Monitored . . . 25
  - 3.5.1 Intrusion Detection on Cloud Computing . . . 29
  - 3.5.2 Role of Virtualization . . . 30

4 Discussion **31**
- 4.1 Conclusion . . . 33

A OpenStack Installation **36**
- A.1 Controller node . . . 36

A.1.1 Identity Service (Keystone) . . . 36
A.1.2 Image Service (Glance) . . . 38
A.1.3 Quantum (Network): . . . 40
A.1.4 Nova: . . . 41
A.1.5 Cinder (Volume): . . . 43
A.1.6 Dashboard (Horizon): . . . 45
A.2 Network Node . . . 45
A.2.1 Preparing Network machine . . . 45
A.2.2 OpenVSwitch . . . 46
A.2.3 Quantum . . . 46
A.3 Compute Node . . . 47
A.3.1 Preparing the Compute machine . . . 47
A.3.2 KVM . . . 47
A.3.3 OpenVSwitch . . . 48
A.3.4 Quantum . . . 48
A.3.5 Nova . . . 49

B

Snort Installation **52**
B.1 Snort configuration . . . 53
B.2 Installing BASE (Basic Analysis and Security Engine) . . . 55

List of Figures

1 Openstack outline [12]. 6
2 Single Flat Network [2]. 8
3 Multiple Flat Network [2]. 9
4 Mixed Flat and Private Network [2]. 10
5 Provider Router with Private Networks [2]. 11
6 Per-tenant Routers with Private Networks [2]. 12
7 Quantum map of DevStack [7]. 14
8 A web interface GUI which shows network map. 16
9 Overview of OpenStack dashboard with two tenants and three VMs. 17
10 Possible threats among different modules [13]. 18
11 SYN flood attack 26
12 Smurf flooding attack. 27
13 UDP flood procedure attack. 28
14 Attaching Snort to DevStack 30
15 Performance of victim during DDoS attack 31
16 An overview of TCP alerts on Barnyard 32
17 An overview of TCP alerts on BASE 33

1 Introduction

Cloud Computing has been launched seriously in recent years. Nowadays, it is considered as a vital mean with the same importance as electricity and gas utility. Some advantages of cloud computing are listed below:

- lower cost because of sharing resources computation.
- No upfront devices.
- On-demand service.

1.1 Cloud Computing OverView

We use the NIST definition of Cloud:

"Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models." [8].

The National Institute of Standards and Technology (NIST) counts multiple essential characteristics for Cloud Computing:

- *On-demand self service* When more computation abilities are needed (like network storage and server), it can be added without contributing human interaction. [8]
- *Broad network access* Access is possible over the internet and can be used by either thin or thick customer platforms (mobile, laptop, PDAs) [8].
- *Resource pooling* The provider sets up computing resources such as network bandwidth, virtual machines, storage, and memory regardless of location knowledge for clients.(Maybe in higher level can point out country or state) [8].
- *Rapid elasticity* Cloud provides elasticity by scaling in or scaling out resources. In customers view, resources are not limited and can be bought in any quantity at any time [8].

- *Measured service* The transparency report and control are provided for both service provider and customers over utilization of resources [8].

1.1.1 Service Model

There are three fundamental service models for Cloud:

Cloud Software as a Service (SaaS). In this model subscribers use programs where are running over infrastructure of Cloud via thin client interface (e.g. web browser). Users are not supposed to configure underlying infrastructure (network, servers, and operating systems) with limited user-specific application configuration settings [8].

Cloud Platform as a Service (PaaS). The services are provided in this model, are deployed onto the cloud infrastructure consumer-created or applications created using programming languages and tools which are supported by the providers. Customers are not responsible for managing the underlying cloud infrastructure such as network, virtual machine or operating system [8].

Cloud Infrastructure as a Service (IaaS). Services provisioned here are some fundamental computing resources (e.g., storage, networks)which customers can run arbitrary software on their infrastructures even operating system. Clients does not have access to control underlying infrastructure, maybe constrained manage of select networking components (host firewall) [8].

1.1.2 Deployment Model

Four deployment models are defined for cloud computing:

- The first kind of Cloud runs for one organization. It is controlled by the third party or organization in premise. This model is known as *Private Cloud.*

- The second kind of Cloud is shared among multiple organizations. It is able to share common concerns such as security issues and is managed by third party or an organization in premise or off premise. This system is referred as *Community Cloud.*

- The third kind of Cloud is available for large amount of companies or public usage, also owned by provider which sells Cloud services. This is the definition of *Public Cloud.*

- The fourth kind of cloud is a combination of two or more cloud types. Remaining entities are combined together according to standard that allow application portability. This is called *Hybrid Cloud.*

Performance of some applications which run on the cloud depends on connecting different clouds and connecting users to cloud. When these applications move to cloud, three factors become important: Capacity (more data send), Quality (low delay), and Availability. Separation between service provider and infrastructure provider results in providing new services online and scale services as demand. Service provider pays for accessing to resources on-demand with little time to change the capacity, also expenses will be reduced. As a result, Infrastructure provider makes large infrastructure and just pay for workload of customers. [13].

1.2 Contribution and Report Structure

The contribution has multiple aspects:

- OpenStack and Available Network Structures.
- Installing OpenStack and DevStack.
- Available Network and Security Issues.
- Intrusion Detection System and Snort.
- Monitoring.

1.2.1 OpenStack and Available Network Structures

In the first phase, the definition of Cloud Computing was studied by reading standard definition of NIST. Different types of services are available IaaS, PaaS, and SaaS. Various kinds of Cloud deployment were considered e.g., public, private, community, and hybrid. OpenStack is consisted from multiple modules such as Compute, authentication service, web interface, and

network. Network component was the most important part which was focused on. Afterwards, a network map has been chosen from five available structures. All definitions can be found in chapter one.

1.2.2 Installing OpenStack and DevStack

In the second phase, an open source platform of Cloud is required to be installed. So, OpenStack was selected. Because of some limitations of the VirtualBox and number of public IP addresses, switched to DevStack. Actually DevStack is identical to OpenStack with small differences in scalability and stability. Afterwards, DevStack was chosen with Single Flat Network which was not suited network structure for my project scenario. Finally, the DevStack with complicated network map was chosen. Detail for implementing the appropriate platform is available in section two and appendix A

1.2.3 Available Network and Security Issues

In the third phase, A group of security challenges in different dimension were listed and studied. From available threats, security of VMs in cloud was selected.

1.2.4 Intrusion Detection System and Snort

In the fourth phase, IDS will be taken in consideration. The ecosystem of IDS and different modes were examined. Snort was chosen as our detector to identify variety of risks. The detection procedure works based on patterns. It compares incoming traffic flow with rules to detect threats. The contents of packets such as source/destination IP address, port number, name and type attack will be stated. Snort shows result on web interface GUI.

1.2.5 Monitoring

In the last phase, Snort was attached to DevStack's network for monitoring the flow. According the rules which were downloaded and written, all forbidden activities are captured and forwarded as alerts. A DDoS attack was emulated from VMs of attacker's project to a VM of victim project. Snort could identify intrusion and report it to administrator. More information is available in the last chapter.

2 Design and Implementation

In following chapter, our platform is installed and configures. But some knowledge about it is necessary to understand how it working. A Brief Overview of OpenStack Components are defined below:

- Compute (Nova):
  Compute is responsible for VM instances[1], e.g. providing resources, live migration. Compute supports different Hypervisors.

- Network (Quantum): It is an ability which user can manage some parts of project network and allocate static and floating IP addresses.

- Dashboard (horizon): It is a web GUI (Graphical User Interface) for controlling different modules of Cloud for service provider and customers, such as launching instances, status of components, and managing storage.

- Image (Glance): It is responsible for managing the images with various formats, such as keeping and snapshotting.

- Identity service (Keystone): It provides authentication and level of accessibility to different resources. Different users are assigned to different tenants[2] (project) with limited accessibilities (role).

2.0.6 Communication between Components

As shown in Fig.1 computer, controller and network nodes are linked to each other by three connections.

- Management network (connection #1): It is used for local communication, hence have to reachable from inside the OpenStack.

- Data network (connection #2): It connects network and compute nodes for data transferring inside the cloud infrastructure.

- API network (connection #3): It is accessible from the internet. It provides API accessibility.

[1]They are virtual machines which are run on Compute node. It is possible bundle multiple instances from one image.

[2]Tenant specifies the organization or unique project, for instance can request online services based on tenant id which are related to specific project.

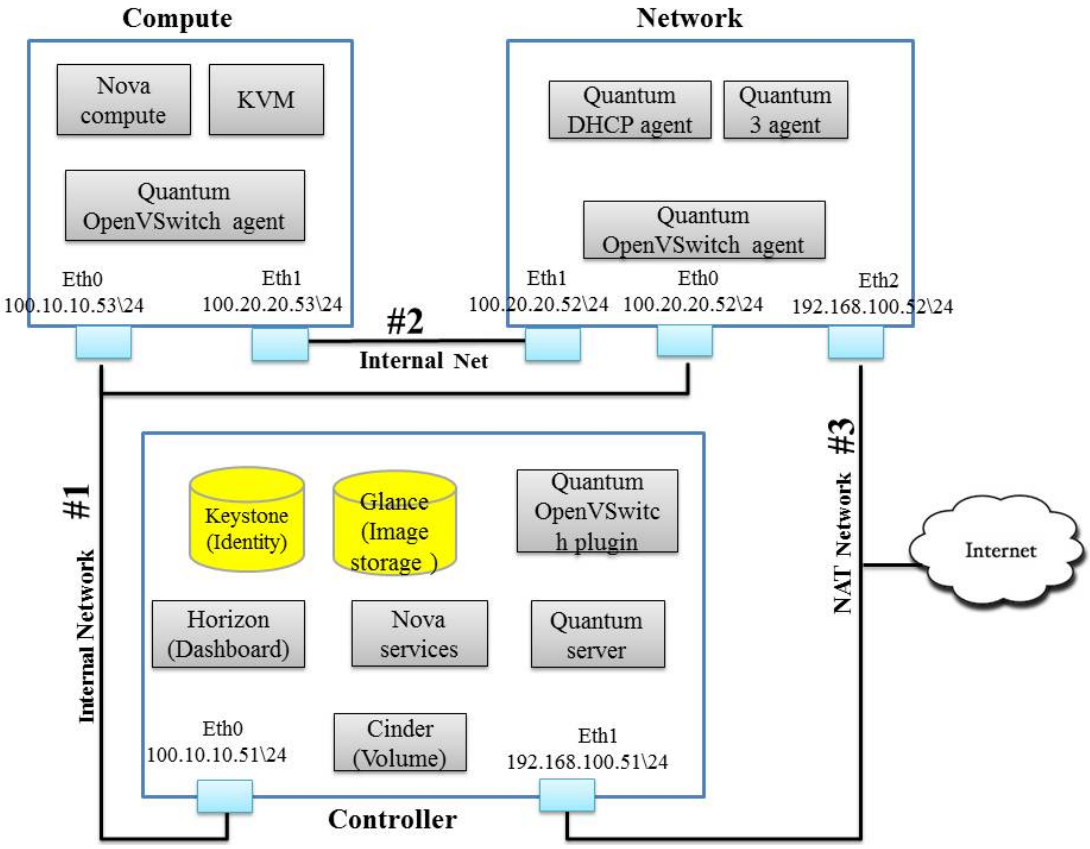

Figure 1: Openstack outline [12].

2.1 OpenStack

At the beginning, an infrastructure was needed for future analyses. The latest released version of OpenStack, Folsom, was installed on the machine. Figure 1 depicts OpenStack outline. As it can be seen the OpenStack consists of three components: Controller, Compute and Network nodes. As stated previously, the main part of the project is be done on network node (Quantum). For preparing the installation, it was decided to set up all components on a single physical machine. But in the middle of the process due to lack of hardware (Network Adaptor Card), virtualization was used. VirtualBox was used as a Hypervisor[3]. Three virtual machines are launched which named Controller, Compute and Network nodes. To see the detailed information of

[3]Hypervisor is the virtualization component that manages the guest OSs on a host and controls the flow of instructions between the guest OSs and the physical hardware.

installing Openstack, see appendix A. There were some problems in source guidance which is solved in the appendix A.

Three virtual machines were required to be connected by taking advantages of VirtualBox abilities:
Two Network Interface Controllers (NICs) for controller node, three NICs for network node and two NICs for computing were assigned. As shown in Fig. 1, network connection between eth0 of Controller, Compute and Network (connection #1) are configured as an internal network. The specification of internal network is local accessibility. As well, the internal network is set for eth1 among network and compute nodes (connection #2). Eth1 of controller and network nodes are configured as a NAT (Network Address Translator). The NAT was used due to access to internet (connection#3). There is a small change from the original model of Folsom. The static IP addresses are configured in the following file:
/etc/network/interfaces
VirtualBox can not support two VMs (in our case, network and controller nodes) in the same network address (same Local Area Network). Hence, two NATs (Network Address Translator) with different network addresses were allocated for network and controller nodes. The weak point of VirtualBox in this point causes invisibility among two NATs. Lack of multiple public IP addresses along with the mentioned problems were led the author to cease using of the model shown in Fig. 1 for implementation.

2.2 OpenStack Networking

Variety of terms are used in this area of study. Network, subnet, and port are the most frequently used ones. Network provides a virtual network. Subnet is available for IP version 4 and 6. Port can connect a device e.g.,NIC to virtual network. Each tenant has a virtual network with VLANs ability of Linux and IP table. OpenStack introduces a new term plugin which can handle API request by multiple technologies, such as tunneling, OpenFlow and Layer2 in layer3 [2]: OpenvSwitch, Linux Bridge, Cisco et al.
Network node could be installed on the same server which controller was installed. But the network node was separated. The core of networking part is quantum-server, which all agents need to communicate with it. There are three different kinds of agent: plugin agent, dhcp agent, and l3agent. Plugin agent is responsible for running on hypervisor of each machine and performing vswitch. DHCP agent executes DHCP in each tenant. The last one l3 agent is in charge of NAT as is demonstrated in Fig.1 [2].

There are different types of architectures for OpenStack networking:

- *Single Flat Network:*
  Single Flat Network is a shared network which is visible to all tenants. Each tenant has a static IP address where is assigned by the subnet that is fixed for mentioned network. The administrator can put one physical router, since VMs can be routed through external network [2]. Figure 2 shows the map of the Single Flat Network.

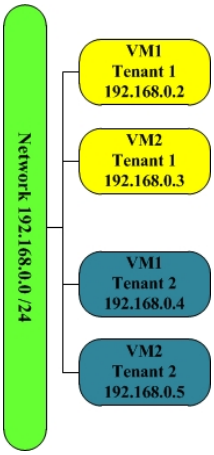

Figure 2: Single Flat Network [2].

- *Multiple Flat Network:*
  Multiple Flat Network (Figure 3) is similar to Single Flat Network, apart from tenants can see divers shared networks and can select which one are going to be used [2].

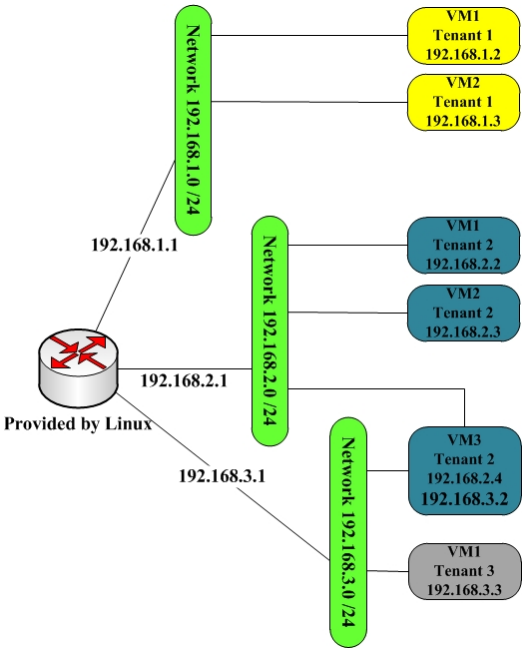

Figure 3: Multiple Flat Network [2].

- *Mixed Flat and Private Network:*
  This structure (shown in Figure 4) is combination of two above models. Tenants have ability to access shared network, while can create private network, which is available only to users of specified tenant. Thus, every tenant can have multiple NICs. In this case VMs can be used for routing and load balancing [2].

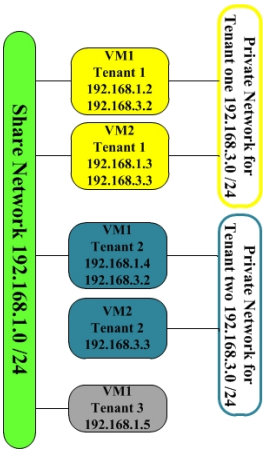

Figure 4: Mixed Flat and Private Network [2].

- *Provider Router with Private Networks:*
  In this network the model provides private network for each tenant. Tenant can just see its own network. The mentioned networks are connected to outside world by OpenStack router. Figure 5 depicts the pattern of Provider Router with Private Networks. This pattern allocates a floating IP[4] address (public IP) to a fixed IP[5] address of Virtual Machine. But the VM which has no public IP address still can make connection to outside. Administrator cannot make overlapping network address due to existence of one router [2].

[4]Floating IP is an address which is assigned dynamically to an instance for being reachable from the internet. Compute uses NAT to assign floating IP addresses. It is assigned by administrator, can be detached from one instance and allocated to another instance. Since floating IP is able to use different IP pools from different ISPs, in case of failure the connectivity is guaranteed [19].

[5]Fixed IP is assigned automatically to instances when it boots up, and if the instance is terminated the IP is not reusable.

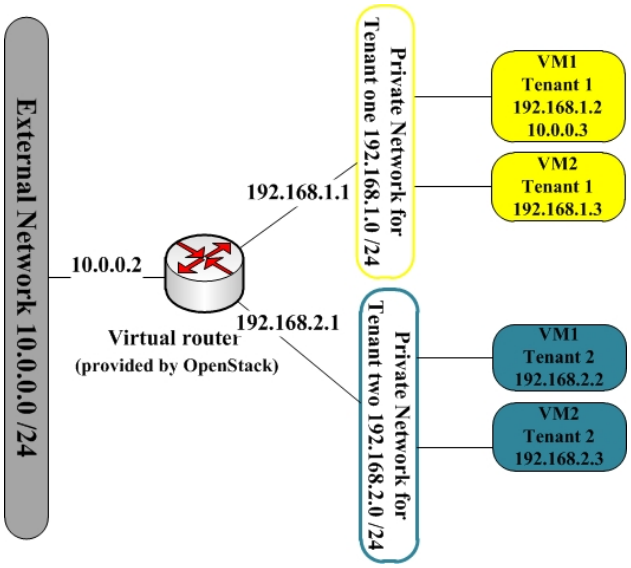

Figure 5: Provider Router with Private Networks [2].

- *Per-tenant Routers with Private Networks:*
  In this type of architecture tenant is assigned by, at least, one router. Because of different routers overlapping address can be used by tenants. Outside network is accessible whether by floating IP or SNAT. Per-tenant Routers with Private Networks provides multi layer network [2]. Figure 6 shows the network architecture.

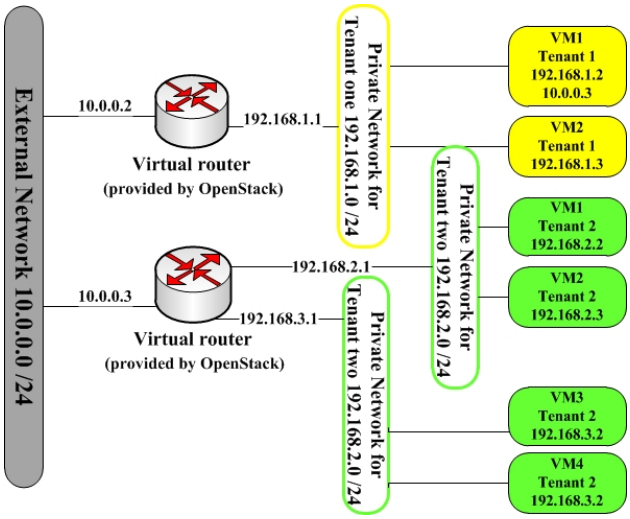

Figure 6: Per-tenant Routers with Private Networks [2].

2.3 DevStack

DevStack implements the OpenStack in small size on one machine and has equal concepts with OpenStack such as network part (different types of network architecture), Nova, and horizon. To start, DevStack was installed via localrc file (a file which sets the default values for installation such as IPs and passwords) on the system. The third structure (Provider Router with Private Networks) was selected for DevStack. The network map is depicted in Fig 5. Fig 8 demonstrates the structure of the implemented network in the DevStack. All the models defined earlier have basic security. The Cloud providers use additional tools for preventing and detecting abnormal behavior in order to make the Cloud safe [15].
Different methods of prevention (e.g, firewall) had the role of developing obstacles against generation of the incident. Network detection identifies some malicious activities which have been occurred or are occurring (intrusion detection system). In the third model, since the VMs in different projects are visible to each other, they are able to attack one another. The procedure of such attack will be addressed in next sections.

Several changes were made in default localrc, due to having one single public IP (152.94.0.159) which is shown in Fig. 7 as a gateway. Because of the limitation of available public IP addresses, VM instances will not be routed to outside the world. This IP is not in the same range of share and private network addresses.

As long as the purpose of the project is confined by monitoring, conducting security considerations and dealing with network challenges among the internal instances (e.g, VM1 and VM2 in Fig.7), no major issue is likely to happen without accessing to outside world.

Fig. 7 demonstrates the network map. Pinging the spawned instance was not possible. Attempts made to ping the external router gateway and another router gateway in the instance side (private network). None of which were successful. To solve the problem routing table of host was checked but no entry routes were found to be associated with external and private network. Furthermore, interfaces which included br-int and br-ex were checked. br-int and br-ex are OpenVSwitchs which are responsible for handling request to inside and outside world shown in Fig.7 respectively. Out of them, br-ex on the top structure was not possessing any IP address. A definition of OpenVswitch is stated in the following.

OpenVSwitch

OpenVSwitch is a virtualized switch which is located within Hypervisor. It acts as a layer two switch and provides some features such as RSPAN, simple ACL (Access Control List) and VLAN (Virtual Local Area Network). Another usage is attaching the virtual network to physical interface. It provides QoS (Quality of Service), mirrored port, NetFlow protocol and ties two interfaces together for better performance. It supports forwarding or dropping packets based on L2, L3 and L4 headers (OpenFlow).
It can provide each tenant with one subnet or private network (or more) and allocate VLAN tag to frames. But the numbers of VLANs are limited, so there is another option where is called GRE (Generic Routing Encapsulation) tunneling. The layer two data is wrapped through layer three packet and routed via best path which has been selected based on flow table (database of OpenFlow) [3].
Moving VMs have some limitations in layer two, although, IP configuration changing the same subnet is not required. Otherwise, one can opt to proceed to layer three. Thereby, tunneling can assist to transfer VMs in third layer with better capacity [4].

After restarting the machine, the routing table for DevStack was vanished. Therefore handful routes were added to routing table and startup network. At the first step, IP address is assigned to an interface of br-ex bridge in the interface file. The following network address was used as the external network which is demonstrated on Fig. 5 :
ifconfig br-ex 172.31.246.129 netmask 255.255.255.128 up

Then, IP addresses of router gateways was checked to insure their correctness. Then a route was defined in the host:
route add -net 172.31.246.128 255.255.255.128 gw 172.31.246.129 br-ex:
External gateway of router was accessible (nova side).
Add private network route to iptable:
ip route add 172.24.17.0/24 dev br-ex
172.24.17.1 instance side of router gateway. The second IP is allocated to DHCP.

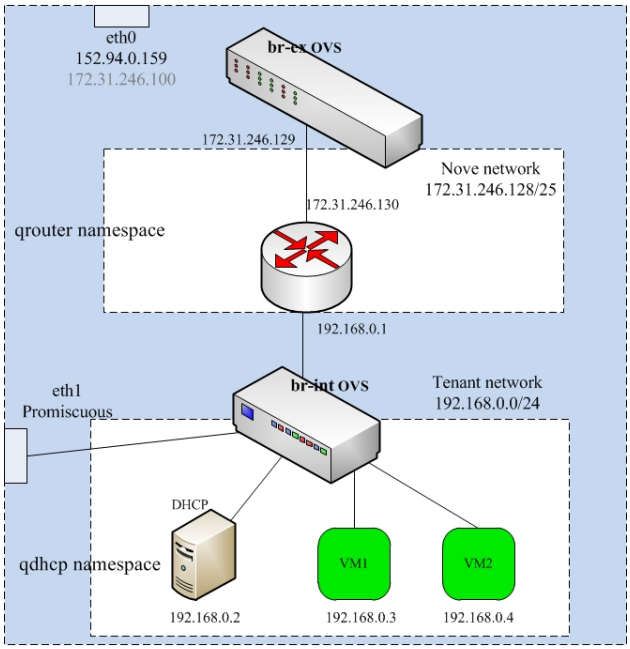

Figure 7: Quantum map of DevStack [7].

When DevStack was run, demo project has been created and got ready for bundling virtual machine instance. For the other projects, this process is performed manually. Different network types were defined before in OpenStack Network section. The chosen structure of network (Provider Router with Private Networks) is shown in Figure 5.

2.3.1 Spawning Projects

At first the terminal in DevStack is opened and a project was created and the name, project_one, was assigned:

```
# keystone tenant-create --name project1
```

Then, a user is made which is associated with project_one tenant id:

```
# keystone user-create --name=user_one --pass=user_one --tenant-
    id 08855548297e4c04b855ed880ece9808
```

At the next step, the member role to the user is assigned:

```
# keystone user-role-add --tenant-id 08855548297
    e4c04b855ed880ece9808  --user-id 3
    ff24b8cc7914d1c8f307cc47cbedcd1 --role-id 3
    b0a084fc5ec44ac9277f51478c25c00
```

A network and subnet for the project is needed to be created:

```
# quantum net-create --tenant-id 08855548297
    e4c04b855ed880ece9808 net_proj_one --provider:network_type
    vlan --provider:physical_network physnet1 --provider:
    segmentation_id 1024

# quantum subnet-create --tenant-id 08855548297
    e4c04b855ed880ece9808  net_proj_one 192.168.1.0/24
```

Due to the type of the chosen model, the subnet is assigned to an interface of available router (if Per-tenant Routers with Private Networks model had been selected, another router had to be created at this level:)

```
# quantum router-create --tenant-id 08855548297
    e4c04b855ed880ece9808 router_proj_one

# quantum router-interface-add 9b90a45b-78b7-4cc1-95e6-1
    cefc5170208 0d6008fa-0f3f-4f89-bf33-d35201794c99
```

After creating the tenant, service provider is supposed to upload images[6]. In next step, instances are launched by owner of the tenant whether via dashboard or command line. For running instances, handful factors should be assigned: flavour[7],security key (e.g. RSA key which is injecting to instance while it is coming up. The key will be used for SSH login.) and the private network which was built previously.

After creating the tenant, service provider is supposed to upload images[8]. In the next step, instances are launched by owner of the tenant whether via dashboard or command line. For running instances, a handful of factors

[6]They are disk images which provide virtual machine file system template and glance module is responsible for controlling the storage. The size of images are small.

[7]The specifications of virtual hardware, such as RAM, number of CPU and hard disk.

[8]They are disk images which provide the virtual machine file system with template and glance module is responsible for controlling the storage. The size of images are small.

should be assigned: flavour[9], security key (e.g. RSA key which is injecting to instance while it is coming up. The key will be used for SSH login.) and the private network which was built previously.

Figure 9depicts spawning of two project with three instances. The following picture was shown on Horizon (web interface of DevStack). As shown in Figure 8 the VMs of one project are not visible to another project on Horizon.

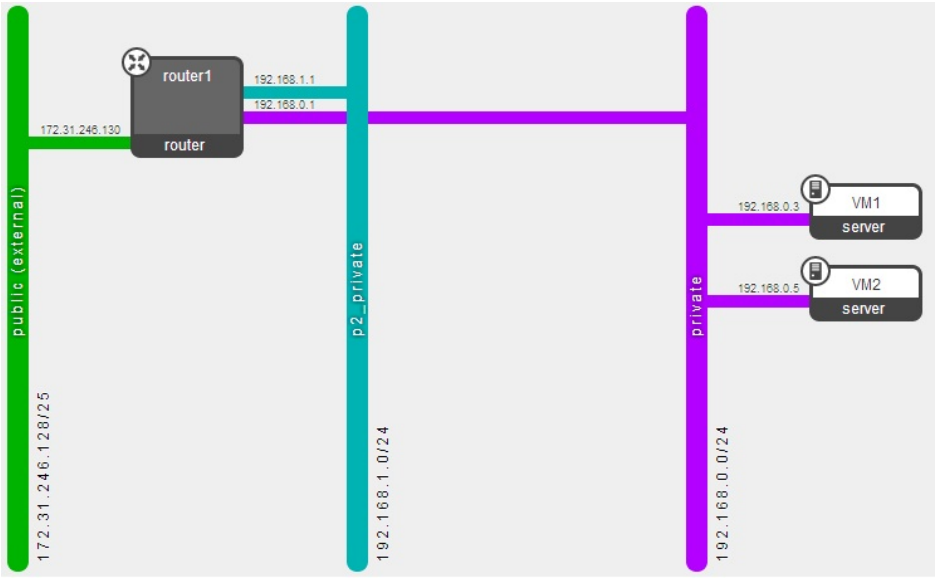

Figure 8: A web interface GUI which shows network map.

[9]The specifications of virtual hardware, such as RAM, number of CPU and hard disk.

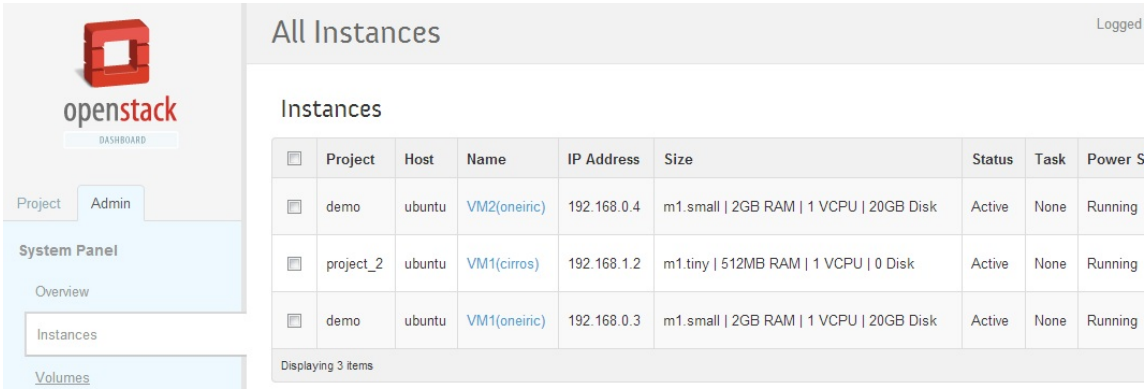

Figure 9: Overview of OpenStack dashboard with two tenants and three VMs.

2.3.2 Namespace networking in Linux

The namespace concept provide private area among shared components such as ports, sockets, and interfaces in the Kernel. When one namespace is launched only one loopback interface is visible and then the rest of network is created by operator. The generated namespace interfaces e.g., vlan, veth, or tap, are movable, while physical interfaces are fixed. Another advantage of namespace is that if more than one project is launched, overlapping IP addresses can be used, since they are not visible to each other.

There are two namespaces which can be showed by ip net lists. Each tenant is allocated by one namespace which vlan concept is provided with namespace. All ports associated to a specific tenant in the br-int are labeled by tag: 1. Each namespace can be attached to one openvswitch. Each VM in the Network is allotted by one bridge. The mentioned bridge is connected to br-int by veth pair which acts similar to a tunnel. It forwards the packet to another side (br-int).

If it is opted to use l3-agent and dhcp-agent run in the same machine it is mandatory to enable namespace in the quantum.conf file in the quantum.

The configuration of the route was edited /etc/network/interface:

```
ip route add 172.31.246.128/25 dev br-ex

auto br-ex
iface br-ex inet static
address 172.31.246.129
netmask 255.255.255.128
```

```
up route add -net 192.168.0.0/24 gw 172.31.246.130 dev br-ex
```

3 Security Challenges and Solution Issues

There are different concerns about security of cloud such as: authentication, integrity (users retrieve data which stored before precisely) and confidentiality (data appeared to someone who is not acceptable), auditing, and abusing available components and services. Virtual machine monitoring can be referred to the Hypervisor which is a layer for managing physical resources and allocating them to different VMs, providing isolation between VMs and et al.
Using virtualization and multiple VMs on one physical machine might cause some threats: location and extraction. At first attacker embeds VM on same physical infrastructure which target is located. Since VMs are using same resources such as CPU data cache, intruder can manage cross-VM side channel attack [13].

3.1 Available Threats

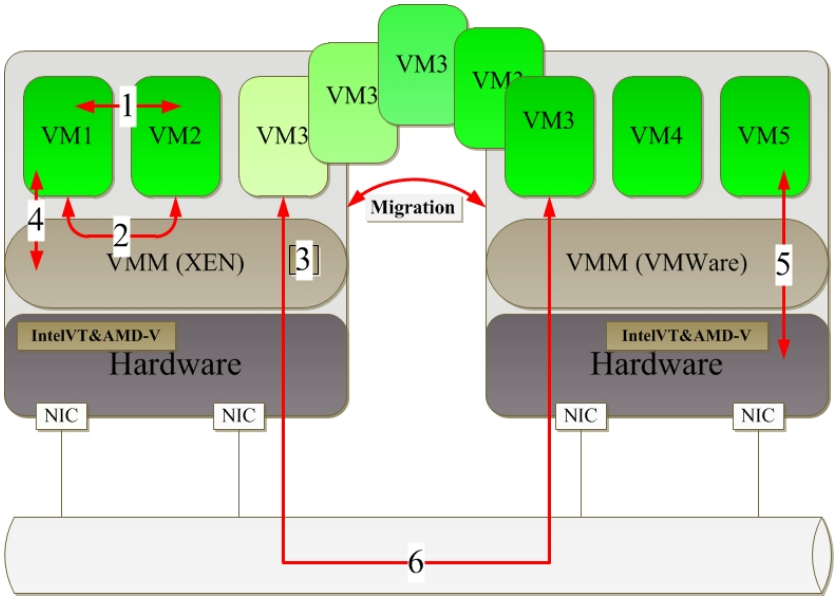

Figure 10: Possible threats among different modules [13].

1. See another VMs packet: For allocating physical resources among multiple VMs, for instance Network Interface Card (NIC), hypervisor uses different methods: NAT, Bridged and routed networking. In case of using bridge, it must be ensured that the packets which are sent for specific VM, cannot be read by another VM which is connected to the same bridge. Figure 10 shows the overview of the sniffing [13].

2. Access to another component resources: Each element is assigned some supplies which are not accessible by the other components. An example is the memory which is owned by a VM1. In this case VM2 is not supposed to access to VM1s memory [13]. Figure 10 part 2 illustrates the threat.

3. Untrusted hypervisor: It is necessary that the subscriber of VM provides its own security while the provider uses untrusted hypervisor for watching data of VMs. Encrypted method can be used for preventing threat. Number three demonstrates kind of attack in Figure 10 [13].

4. Virtual Machine threat: It is possible a VM tries to access the physical machine. For instance a user of Cloud injects code in software layer service and can access to other layers. In worst case, attacker can be able to penetrate through the physical machine [13]. Figure 10 gives a picture of the threat.

5. Direct communication: New technologies has made VM to be able sending requests to the processor directly, instead of hypervisor. There are some threats, e.g. VMBR and BluePill [13].10

6. For transferring VMs among variety physical machines, some attacks to the network part might be happened [13].10

7. With learning the method for assigning the IP addresses to different VMs, the possibility that a VM is located by attacker in the same infrastructure is high. The mentioned VM is able to comprehend for checking whether victim VM is in the same physical machine or not. The attacker might use side-channel attack (cache-based or network-based techniques) [16].

8. Adding the services on the cloud are on demand, the attackers can abuse this ability for Denial of Service (DoS) or spamming. For preventing mentioned attack can use the abnormal detection system for identifying fast fluxing [13].

3.2 Anomaly Detection

It is an abnormal behavior when the model does not conform the default. An example in real world is cyber intrusion and credit card fraud. There are some challenges for distinguishing criteria for region of normal and abnormal behaviors. Some issues are associated in IDS (Intrusion Detection System) such as: input data type, data labels, types of anomaly, output of anomaly detection, and considering anomaly technics detections [10].

3.2.1 Input Data Type

For recognizing the anomaly detection a collection of data is taken which each consists of different attributes e.g., binary, continuous, categorical. These data consist whether one attribute (univariate) or multiple attributes (multivariate).

3.2.2 Data Labels

Label is assigned to data to identify whether anomalous or normal instance. The anomalous behavior is usually different in real world and there is not precise definition.
Three divers labeling can use:

- supervised anomaly detection: In this mode we have both normal and anomalous tag.
- Semi-supervised: In this case we only use the normal behavior and use the mentioned model for diagnosing anomalous class.
- Unsupervised anomaly: There is not any special label. An assumption is made that anomalous class is far from normal activities.

Diverse types of anomaly: For point anomaly we have two collections of instances but we have one instance away from these two groups. For contextual example, we have one graph which normal behavior is gentle increasing but in the middle one dramatic decreasing is observed. It is not normal in comparison with the rest of graph. Collective anomaly is an abnormal behavior of associated data not the individual one.

3.2.3 Output

First data is examined and labeled anomaly or normal then, anomaly will be assigned score which later can be classified.

3.3 Intrusion Detection and Traditional Approaches

Since Intrusion Detection is a mean for detecting an abnormal behavior from normal pattern. It should handle a large amount of data in-line. Unsupervised and semi-supervised anomaly detection is accepted.
An example in real world, can be a house with security camera. when a burglary is happened, you can check the log of camera for identifying who was and what has he done. Intrusion detection are divided into couple of areas: Host-based and Network based. Identifying intrusion detection can be in diverse levels e.g., packet level traces or cisco net-flows data. The problem is that the nature of attacks can be varied which they are designed to skip intrusion detection solutions.

They use IDS for finding vulnerabilities of system. According rules, logs are made and forwarded to core element [14]. There are diverse types of NIDS such as Snort, Bro, and Suricata for implementing IDS, monitoring, incoming, and outgoing flow network [14]. Network based attacks which are designed from the outside of the network for accessing to data (inside the network) without authorization for disrupting the network. Generally, NIDS (Network Intrusion Detection System) is set in a switch with mirror port, or network tap. For monitoring the network, sensors are fixed in boundary area of network. Moreover, the contents of packets are captured for identifying abnormal behavior [14].
HIDS (Host based Intrusion Detection) is implemented on a host and related to operating system. By gathering logs, calling system etc. performance of system is estimated. Generally, software agents are used. There are two ways statistical and signatures. First one makes boundaries by looking at flow traffic of network by using ports and devices. If the traffic crosses the boundary, alarm is sent. The second one compares the receiving flow with available signatures. But, it cannot detect the novel threat or attacks [14].
There are some rules for detecting anomalous behavior. IDS can adjust setting of rules according to activities of network. There are three types of alarms:

- True positive:when attack is recognized and send right alarm.
- False positive:when attack has not been occurred and alarm was sent.
- False negative:when system could not identify intrusion [14].

3.3.1 Signature

It is a set of rules (signatures) which IDS can find attacks based on matching signatures. Attacks can be investigated in different parts of packets like network header (IP) or transport layer header (TCP or UDP) or application layer header or payload [18]. The system which IDS is installed on it, referred to sensor.
The place that set sensor depends on network map and which aspect we are looking for, if the network which is monitored, has just one internal point from the internet (router), one sensor is located behind the router, for examining. If domestic threats are searched, for each network segment can embed one sensor. But keep in mind with increasing the number of sensors, amount of work will be increased [18].
Packet decoder gets the packets from different interface (e.g. Ethernet, PPP) and prepares them for next step preprocessor. Preprocessor prepares the packets for forwarding to detection engine. Detection engine employs the rules to packets, if the signature is matched, may alerts are reported otherwise packet is dropped [18].
Snort system has two interfaces: one is connected to mirror/SPAN port for receiving traffic one is connected to management core for forwarding alerts. The hand which is used for sniffing should be set to promiscuous mode (in layer two all ports receive flow but when the destination address of frame is not for receiver port, it drops frame. But when the interface is adjusted to promiscuous, it gathers entire traffic [18]). Moreover, IP is not allocated to sniffer interface of IDS.
There is some limitation for using rules, since these rules has been written for known attacks and even two current known attacks combined, IDS cannot identify. Another thing is IDS produces big amount of log and it is hard to recognize the anomaly behavior while system is under attacks

3.3.2 Snort

Snort is an open source IDS which was released 1998 by Martin Roesch.

Snort performs in three modes.It run as a packet sniffer, logs packets, and an IDS based on rules for network traffic [5]

- Sniffer mode: read the content of packets.Snort is working properly in sniffing mode which can be enabling via -v:

  ```
  ./snort -v -i eth0
  ```

  It shows in console all detail which have been sent to promiscuous eth0

Packet logger: It captures packets and stores them on disk with -l switch (second mode):

```
./snort -i eth0 -l /home/anna/files/log
```

Above command stores traffic as a log which is forwarded to eth0. Types of logs depend on output configuration of snort.conf where I set it as unified2.

- NIDS: monitoring network traffic based on defined rules, and executes some tasks consequently. For running intrusion detection mode, rules were downloaded from snort web page which half of them were empty. In some of them have been written some patterns but according to network addresses, network topology and what we are going to monitor can be altered.Records can be stored in database. For this purpose, I used the model which log is written to disk by snort in unified format. Afterwards, barnyard2 which is configured to read data from unified file from disk and write it in MySQL.For running snort in IDS mode, the following command is used:

```
# /usr/local/snort/bin/snort -c /usr/local/snort/etc/snort.
    conf -i eth0
```

For reading configuration -c flag is used and the default path for saving log is set in /var/log/snort directory. Then run barnyard2 for recording events in database.

```
# barnyard2 -c /usr/local/snort/etc/barnyard2.conf -d /var/
    log/snort -f snort.u2 -w
/var/log/snort/barnyard2.waldo
```

Switch -d states location of log file and -f name of log file. In my installation barnyard2 could not send log to MySQL but after adding rev: 1 to ICMP rules the issue was solved.

Inline mode: based on IPtable and rules packets are accepted or rejected (IPS) [7].

3.4 Issues of IDS

- Analysing data in the network: capturing packets need vast area for storing data, time which is spent for analysing, and the ability of application for encoding key and anomaly behavior [11].

How specific communication can be found among trillions of packets, Big Data can solve problems, while offering parallel computation, entire data can be stored (complete packet capturing) with tools such as PIG, Apache Hadoop, and NoSQL. They can be used for processing data in the cloud. Strengths of this approach are time saving, improving data [11].
We need tools for finding out how attacker exploited the network and how long stayed and which data has been accessed. These tools help to provide details for attacking. Capturing complete data communication in the network is addressed as network flight recorder or network tivo [11].
In addition of providing strong defense, need to provide strength technique detection. A threat for security is scalability, in some areas we have to sacrifice the accuracy against keeping data in system. Since mentioned system were not designed for such a huge information. For analysing data, they use copy of copy and keep it in central location, it is reason for missing fidelity.
Google published the first paper about MapReduce in 2004 and Bigtable in 2006. The main idea was storing numerous data on multiple nodes and computation is distributed.
Packetpig is open source tool which provides a platform for analysing security, whether on own hardware clustering or Amazon Elastic MapReduce (EMR) and Amazon Simple Storage Service (S3).
Those people who are not sophisticated in MapReduce can make queries about small amount of capturing data in a laptop, and afterwards launch system in Hadoop in multiple commodity.
So, log is stored by Snort and Packetpig identifies specific communication. The first thing for analysing data is using loaders. For getting info related to IP and tcp, PacketLoader() can be used. After using the last loader, then a file is extracted with the integer and char values. We can make group and gather all packets which have been forwarded for specific port like tcp.

- False alarms: sending alarm based on normal traffic. The reason is weak rules which forward wrong alarms.

- Drop packets: The large number of non attack packets are sent by attacker that causes IDS drops the packets. When the packets are out of the capability of IDS, packets will be dropped. Consequently, some harmful packets are not detective.

- Ability to identify new type of attacks: when the signature model is

used, it cannot identify the novel type of attacks. But different type of anomaly such as contextual or point can be used.

3.5 What and Where Should be Monitored

Because of different implementation Intrusion Detection System can be divided to VM-based, hardware-based, and software-based IDS. Along IDS has to detect the known attacks and malicious, it supposed to identify unknown and evasion ones. There are variant types of IDS solutions regardless of open source and commercial (e.g. F-Secure Linux Security, Samhain, and Snort). Even we can use different IDS method and use Intrusion Detection Message Exchange Format (IDMEF) between them which mentioned earlier.
Cloud provider need to monitor underlying infrastructure, even though an end user need monitor his project for avoiding ID [17].

System should be compatible for different environment like network or virtualized components. For collaboration between multiple sensors, central management is required. Alerts were produced by different sensors have to be synchronized, gathered in one place, and finally examined [17]. For managing IDS can take advantage of different VM monitoring paradigm and level of virtualization. For collaborating data of sensors can used fast and reliable network in IDMEF.
Since, the infrastructure and components of cloud are complicated and large, managing system seems quite hard. Each layer can be a target for accessing the attackers, for instance software layer can be attacked by injecting code. Intrusion can go through layers and use cloud for getting more victims. Even infrastructure and VMs which are running on top of host can be exploited by attacker and abused for distributed Denial of Service (DDoS). For preventing the invasions, the traditional IDS solutions can be implemented [17].

DDoS (Distributed Denial of Service): the attacker uses the other computers (zombie) for attacking to target machine for making it slow or unavailable. Eventually the victim uses all resources and cannot respond to other legitimate requests and failure will be occurred [5]. Type of attacks can be different, e.g. smurf attack, ping flood. An attacker can take advantage of spoofing, port scanning for executing DDoS.

- Syn flooding:
  For establishing a TCP connection between two machines, three phases are needed. First stage is a client sends SYN message to server for requesting session, then server answers with SYN/ACK message to the client (server keeps pending connection on its memory). The last

phase is that client sends ACK message to server side for establishing the connection [1].
For Syn attacking the attacker tries to send SYN message as much as he can (or use IP spoofing). After a while all memory of the server is filled by pending list. Even though timeout is available for pending connection but if sending SYN is faster than server memory will be disrupted [1]. hping3 -a spoof IP -i u1 -S -p 445 target IP u1: delay between sending packets is 1 microsecond. -a spoof hostname. -S means SYN timestamp. -p 445: is TCP port hping3 -a 4.5.4.1 -S 192.168.0.3 -p 445 -i u1

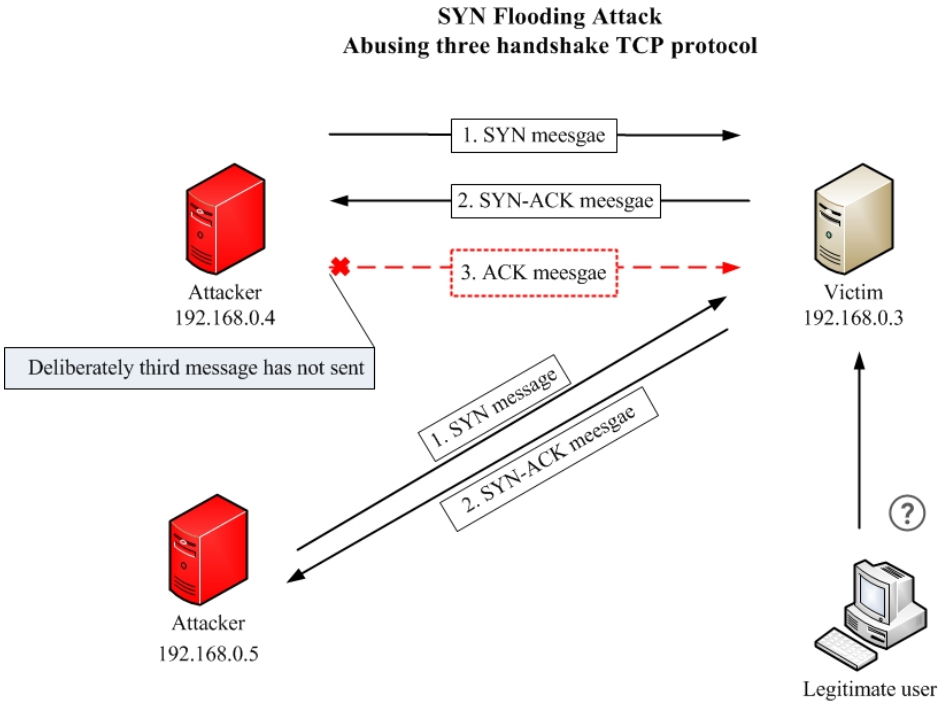

Figure 11: SYN flood attack

- Smurf attack:
  Another kind of attacks is sending packets for machines with spoof IP (this forge IP is the machine which is target for DDoS attack). Since the source IP address is forged to victim IP address, lots of replied packets are sent to victim. Consequently, victim is unable to handle entire packets due to bandwidth is saturated with heavy traffic [1]. Finally, victim is unable to answer legitimate traffic.

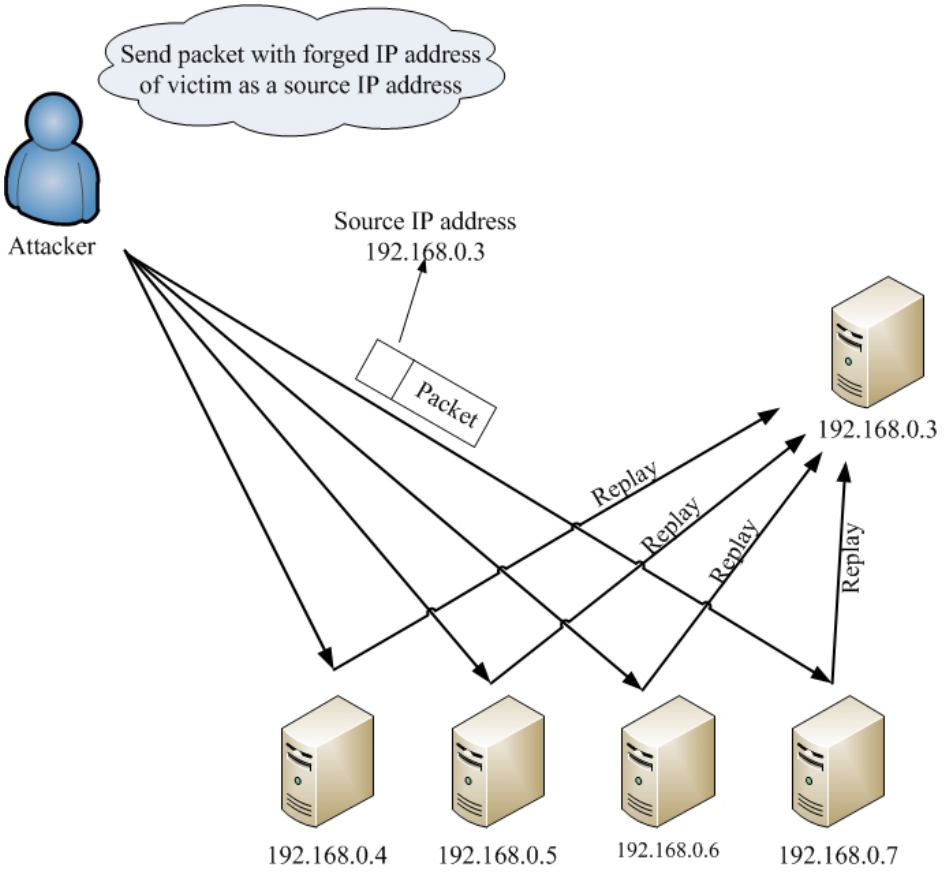

Figure 12: Smurf flooding attack.

- UDP flood:
  This attack sends large number of UDP (User Datagram Packets) to random ports of target. Receiver checks the ports for listening and sends responds to specific application which was mentioned by port number. It responds to sender with destination host unreachable messages. It consumes power and resources from target. Finally, victim is unable to respond legitimate flows.

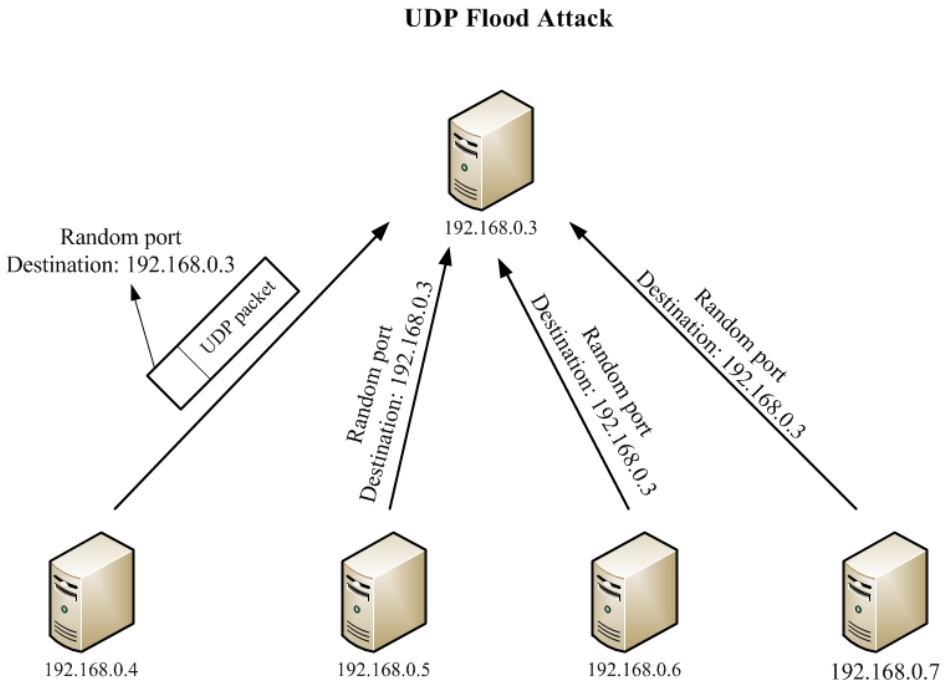

Figure 13: UDP flood procedure attack.

Each sensor forwards alert to IDS management. Big attack can be detected from processing the entire alerts which have been sent to management IDS. When service provider figures out attacks from users or on users can run automatic solution like: limit the resources which assigned to users or even power off the host for preventing more harm. As well users can consider the type of IDS and rulesets for each sensor according to the use management.

The best place for plugging sensors are the points which access to internet or provide services. Another point is a link to another organization in general network map. Also can put a sensor behind the firewall which monitors all open inbound port [6].

So, the appropriate place for sensor should be found. The purpose is monitoring abnormal activities among Cloud tenants and VMs. As Fig /refQuantum showed, two OpenVSwitchs are available in cloud architecture. Because of namespace and VLAN, two projects are not supposed to see each other by default. One assumption was mage which make a path (threat) between first and second projects.

Due to looking for a place to embed sensor in Cloud layers, as showed. SO, the best place for sensor is that For attaching the sensors, traffic of one or more ports are copied for forwarding to a specific port (mirror port) then

NIDS is connected to it. Keep in mind that do not send more than capacity of mirror port, otherwise the extra load will be lost [6].
IDS has two interfaces, one for reporting alerts which forwards warnings to management point and the other interface is using for monitoring. This interface is not configured with IP, so it is not visible to other members of network, it is called stealth interface [6].

3.5.1 Intrusion Detection on Cloud Computing

I had two choices for locating IDS sensor: first place snort in one VM instance inside the Cloud, another approach was a Virtual Machine in VirtualBox (beside DevStack). Second option was chosen and was pretty easier.
For preparing the IDS machine, an Ubuntu VM in VirtualBox was created. The network between IDS VM and DevStack VM has been chosen as an internal network type with name of promiscuous and enable promiscuous option (allow all) in advance part of eth0 of VirtualBox.
For keeping the IDS node secure, assign non routable IP to the interface which is connected to the mirrored port. It decreases probability of compromising the IDS [9].
Configuration was written in the /etc/network/interfaces for eth0 and restart the network:

auto eth0 iface eth0 inet manual up ifconfig eth0 0.0.0.0 up up ip link set eth0 promisc on down ip link set eth0 promisc off down ifconfig eth0 down

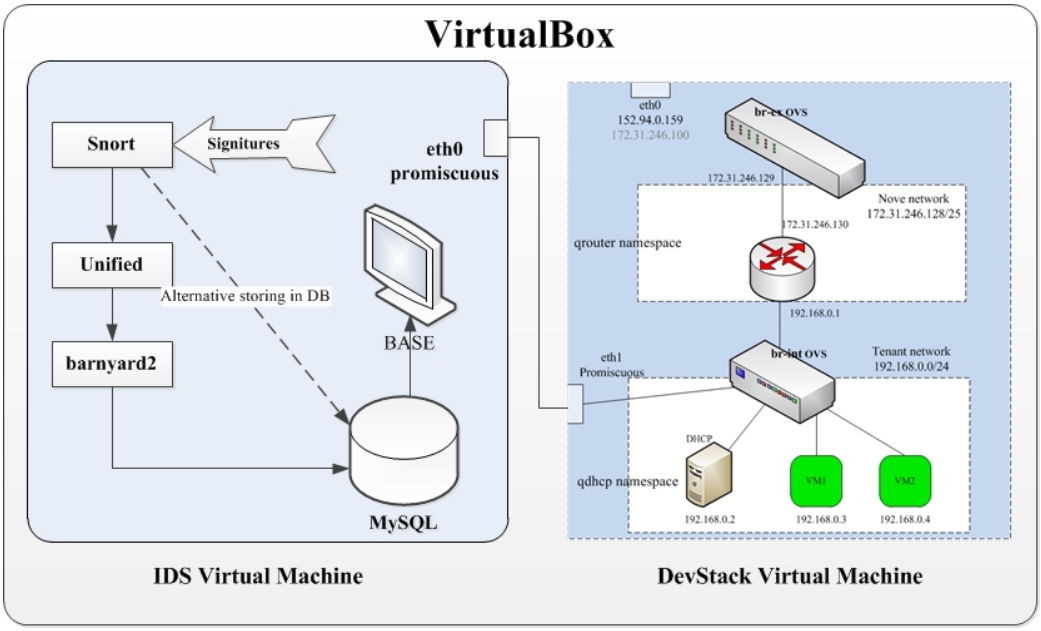

Figure 14: Attaching Snort to DevStack

Home network variable is set manually in Snort configuration file. The monitoring is occurred between two projects currently (project1 and project2). The IP address for Home network is configured 192.168.1.0\24.
A rule was written which detects ICMP packets are being sent to VM instances of home project from other projects.
By adding the following rule, can be figured out that Snort is working properly or not. After adding the signature to rules of snort,echo messages were being sent from the virtual machine in the first project with IP address 192.168.0.2 (it is accounted as an external object) to VM with address of 192.168.1.2. At the same time Snort is running in another machine. Snort started to store the logs on database and alerting. The Snort information log which is saved in MySQL database, will be showed on web interface BASE. BASE is easier way to understand the contents of alert.

```
alert icmp $External_Net any -> $Home_Net any (msg: "ICMP echo"
    ; itype:8 ; sid:2000000;rev:1;)
```

3.5.2 Role of Virtualization

The reason that is preferred to use virtualization are: First of all, it provides an isolated system to decrease effect of attacks. In case of failure or dangerous

can be recovered easily and fast. Even a VM can be paused or resume in case of ongoing intrusion. Furthermore, we can use cross-platform sensors on VMs based system. That is why virtualization is appropriate option [17].

4 Discussion

At the end of the project it was managed to create a scenario. Snort is ready for detecting abnormal behavior. An abnormal behavior is defined as DDoS attack at this level. SYN flood attack was implemented by two VM instances in one project against another VM at second tenant. For checking the performance of the victim, echo messages are being sent by the victim continuously.

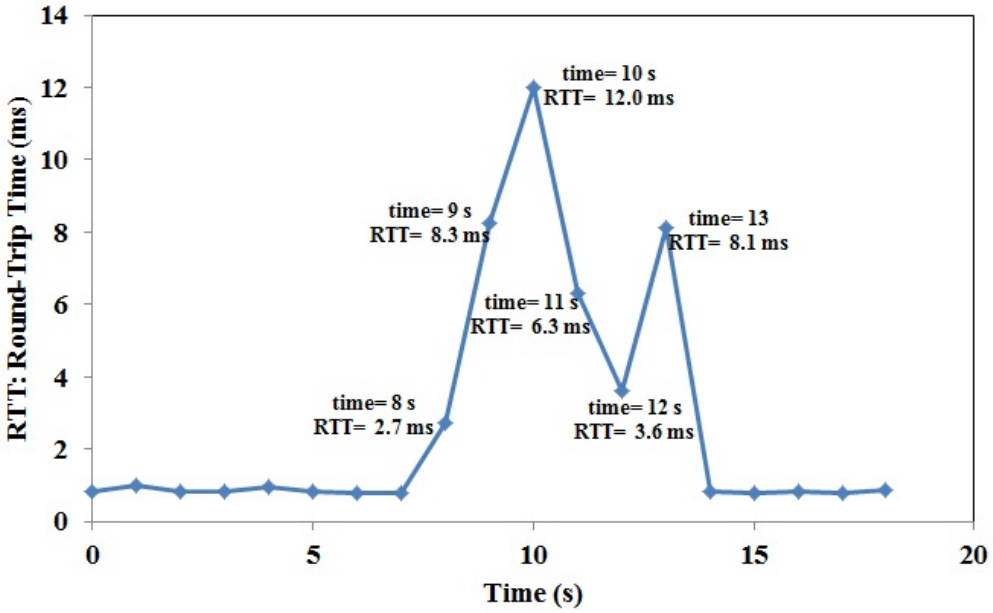

Figure 15: Performance of victim during DDoS attack

Graph in Fig. 15 demonstrates performance of a cloud's VM where is under attack. Before flooding is launched the Round-Trip Time[10] (RTT) has a linear trend approximately with the average rate of 0.833 ms. Once the SYN flood is launched by two VMs of another project in cloud, one dramatic

[10]It is the period of time which signal has been sent to destination and returned as an acknowledgment for availability.

increasing is occurred for RTT. It reaches the peak of 12 ms. Afterwards, the victim handles the flooding traffic and RTT is decreased. But if the large number of VMs are available an incident can be happened and VM even tenant goes down. Because of limitation of resources such as RAM and memory of the physical machine, at the best case three VMs were spawned.
The mentioned attack was captured by Snort and forwarded alerts to database. Before the alerts are transferred to database, stored on barnyard file which shows the alert real time on Fig. 16.Then, the detail of attack is shown on BASE web interface (Fig. 17) e.g., source/destination IP address, port number and type of attack. The way of showing on BASE is quite simple for understanding what is going on.

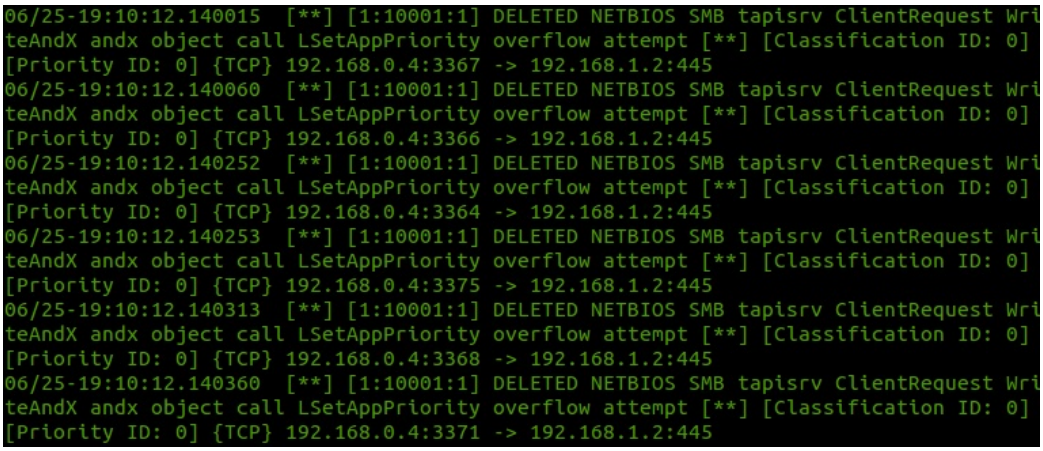

Figure 16: An overview of TCP alerts on Barnyard

< Timestamp >	< Source Address >	< Dest. Address >	< Layer 4 Proto >
2013-06-25 19:10:12	192.168.0.4:3359	192.168.1.2:445	TCP
2013-06-25 19:10:12	192.168.0.4:3328	192.168.1.2:445	TCP
2013-06-25 19:10:12	192.168.0.4:3342	192.168.1.2:445	TCP
2013-06-25 19:10:12	192.168.0.4:3361	192.168.1.2:445	TCP
2013-06-25 19:10:12	192.168.0.4:3345	192.168.1.2:445	TCP
2013-06-25 19:10:12	192.168.0.4:3343	192.168.1.2:445	TCP
2013-06-25 19:10:12	192.168.0.4:3353	192.168.1.2:445	TCP
2013-06-25 19:10:12	192.168.0.4:3344	192.168.1.2:445	TCP
2013-06-25 19:10:12	192.168.0.4:3354	192.168.1.2:445	TCP
2013-06-25 19:10:12	192.168.0.4:3336	192.168.1.2:445	TCP
2013-06-25 19:10:12	192.168.0.4:3355	192.168.1.2:445	TCP
2013-06-25 19:10:12	192.168.0.4:3362	192.168.1.2:445	TCP
2013-06-25 19:10:12	192.168.0.4:3349	192.168.1.2:445	TCP
2013-06-25 19:10:12	192.168.0.4:3352	192.168.1.2:445	TCP
2013-06-25 19:10:12	192.168.0.4:3356	192.168.1.2:445	TCP
2013-06-25 19:10:12	192.168.0.4:3339	192.168.1.2:445	TCP

Figure 17: An overview of TCP alerts on BASE

4.1 Conclusion

Cloud Computing is new utility such as electricity and water. Access to accurate structure of Cloud is restricted by commercial Cloud provider. Therefore, OpenStack as an open source of Cloud has been chosen and studied. The platform of Cloud was implemented in miniature model which is called DevStack. Multiple security and network challenges are counted as threats. Intruding from attacker's VM was studied. The victim can be located whether in same project or another one. Running DevStack in lab gave good overview of the Cloud components, especially network part. Some attacks were emulated on current structure. Moreover, solutions were studied and applied to current infrastructure for detecting abnormal behavior. During working on project two topics are valuable to work on it:
Implementing OpenStack platform in big scale and applied studied attacks and solution on it. It is close to realistic infrastructures which are being used. Another point is that writing a piece of code for transferring data from network part to IDS automatically. Information about number of customer project's VMs, subnet addresses are forwarded to IDS for detecting suspected traffic. The last not least, for making powerful IDS, using the last released signatures and altering them based on network map are vital.

References

[1] Advisory ca-1996-21 tcp syn flooding and ip spoofing attacks. *CERT: Community Emergency Response Team*, November 2000.

[2] Openstack networking administration guide. *OpenStack Foundation*, June 2013. <http://docs.openstack.org/trunk/openstack-network/admin/bk-quantum-admin-guide-trunk.pdf>.

[3] P Ben, P Justin, K Teemu, A Keith, C Martin, and S Scott. Extending networking into the virtualization layer. *HotNets-VIII, New York, NY, USA*, 2009.

[4] Feng Wang Bhumip Khasnabish Bin Liu, Baohua Lei. Requirements for mobility and interconnection of virtual machine and virtual network elements. *ZTE Corporation and China Telecom*, October 2012. http://tools.ietf.org/html/draft-khasnabish-vmmi-problems-02.

[5] Amit Kulkarni , Stephen Bush. Detecting distributed denial-of-service attacks using kolmogorov complexity metrics. *Journal of Network and Systems Management*, pages 69–80, March 2006.

[6] Christopher Gerg, Kerry J. Cox. Managing security with snort & ids tools. *O'Reilly Media*, August 2004. chapter 6.

[7] Will Dennis. Openstack quantum network implementation in linux. *Packet Pushers*, December 2012. http://packetpushers.net/openstack-quantum-network-implementation-in-linux/.

[8] Peter Mell, Timothy Grance. The nist definition of cloud computing. *NIST Special Publication 800-145*, 2012.

[9] David Gullet. Snort 2.9.3 and snort report 1.3.3 on ubuntu 12.4 lts installation guide. *Symmetrix Technology*, July 2012. http://www.symmetrixtech.com/.

[10] VARUN CHANDOLA,ARINDAM BANERJEE,VIPIN KUMAR. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 41, July 2009.

[11] David Turnbull, Gerald Kaszuba Michael Baker,. Finding needles in haystacks. http://media.blackhat.com/bh-eu-12/Baker/bh-eu-12-Baker-Needles_Haystacks-WP.pdf.

[12] Bilel Msekni. Openstack folsom install guide. 2012. <https://github.com/mseknibilel/OpenStack-Folsom-Install-guide/blob/master/OpenStack_Folsom_Install_Guide_WebVersion.rst>.

[13] P.Schoo,V.Fusenig,V.Souza,M.Melo,P.Murray,H.Debar,H.Medhioub,D.Zeghlache. Challenges for cloud networking security. *Mobile Networks and Management*, 68:pp 298–313, September 2011.

[14] Jonas Taft Rdfoss. Comparison of open source network intrusion detection systems. *University of Oslo Department of Informatics*, May 2011.

[15] Qianhui Liang, Bu Sung Lee. Ryan K L Ko,Peter Jagadpramana, Miranda Mowbray, Siani Pearson, Markus Kirchberg,. Trustcloud: A framework for accountability and trust in cloud computing. *IEEE World Congress on Services*, July 2011.

[16] T. Ristenpart, E. Tromer, H. Shacham, S. Savage. Hey, you, get off of my cloud! exploring information leakage in third-party compute clouds. *In ACM Conference on Computer and Communications Security*, 68:298–313, September 2009.

[17] University of Potsdam. Sebastian Roschke, Feng Cheng, Christoph Meinel, Hasso Plattner Institute,. Intrusion detection in the cloud. *Dependable, Autonomic and Secure Computing. DASC '09. Eighth IEEE International Conference on*, pages 729–734, December 2009.

[18] University of Potsdam. Sebastian Roschke, Feng Cheng, Christoph Meinel, Hasso Plattner Institute,. Intrusion detection systems with snort advanced ids techniques using snort, apache, mysql, php, and acid. *Dependable, Autonomic and Secure Computing. DASC '09. Eighth IEEE International Conference on*, pages 729–734, December 2009.

[19] Piotr Siwczak. Configuring floating ip addresses for networking in openstack public and private clouds. *Mirantis*. http://www.mirantis.com/blog/configuring-floating-ip-addresses-networking-openstack-public-private-clouds/.

Appendix A
OpenStack Installation

Components are established on each node independently, start with controller at first step. Requirement: Because Ubuntu 12.04 LTS has by default Essex version of OpenStack, we give path to ubuntu for using Folsom version by editing /etc/apt/source.list:
deb http://ubuntu-cloud.archive.canonical.com/ubuntu precise-updates/folsom main
Otherwise there will be some problems for installing various packages.
#apt-get install ubuntu-cloud-keyring

A.1 Controller node

A.1.1 Identity Service (Keystone)

Authentication has two main functions; first what users allowed to do, second which services are available and the location of the associated API endpoints. First part has three concepts: Users, Tenant and Roles. User is a name for authentication with some information like email and password. Tenant specifies the organization or unique project, for instance can request online services based on tenant id which are related to specific project. Role is what users are allowed to do. Following commands are samples where we used for filling up the keystones database. A database is established for keystone and assigned full permission to the user of keystone database.

```
mysql> CREATE DATABASE keystone;
mysql> GRANT ALL ON keystone.* TO 'keystoneUser'@'%' IDENTIFIED
    BY 'keystonePass';


Create tenant:
First of all, token is selected equal: 012345SECRET99TOKEN012345
#keystone --token 012345SECRET99TOKEN012345 --endpoint {http
    ://100.10.10.51:35357/v2.0 tenant-create --name openstackDemo
     --enabled true

Create user:
While we are making user, we mention it is associate with which
    tenant. Below command demonstrates adminUser which is
    associated with tenant openstackDemo.
System makes the openstackDemo tenant with associated id: 8
    d48c2fd96dd48cebb1585eeb7c2b1fd.
```

```
#keystone --token 012345SECRET99TOKEN012345 --endpoint http
    ://100.10.10.51:35357/v2.0 user-create --tenant_id 8
    d48c2fd96dd48cebb1585eeb7c2b1fd --name adminUser --pass
    adminUser --enabled true. System makes adminUser and returns
    the id value 4b5cb875505644dcbfc076254c70d60a.

Create role:
#keystone --token 012345SECRET99TOKEN012345 --endpoint http
    ://100.10.10.51:35357/v2.0 role-create --name admin

We make another tenant and call it service, for services which
    want to communicate with identity service then make user for
    per service; quantum, glance, nova, ec2, and swift. Then
    admin role is assigned for them.


Grant role to user in a tenant:
In this phase admin role is granted to the adminUser users in
    openstackDemo, even multiple roles can be assigned to one
    user.

#keystone --token 012345SECRET99TOKEN012345 --endpoint http
    ://100.10.10.51:35357/v2.0 user-role-add --user 4
    b5cb875505644dcbfc076254c70d60a --tenant 38
    e63a5e42084fb78371a193e243ac79 --role
    c919af5029764619a6f0cb136d2abd1a

Keystone is used for specifying the location of the API
    endpoints for OpenStack services. For this purpose, we use
    keystone service-create command for filling service table in
    keystone database which consists of the following information
    ; name of service, type of service and description. Next, use
     keystone endpoint-create command for filling endpoint table
    in keystone database command to show how different endpoints
    can connect to services. Different attributes should use like
     region (number which is used for our current cloud),
    service_id (the value that previous command  service
    creation  returned), publicurl (shows the address how can
    be connected outside the region), internalurl (for local
    communicating) and adminurl.

#keystone --token 012345SECRET99TOKEN012345 --endpoint http
    ://100.10.10.51:35357/v2.0/ service-create --name=keystone
     --type=identity

#keystone --token 012345SECRET99TOKEN012345 --endpoint http
    ://100.10.10.51:35357/v2.0/ endpoint-create --region
    RegionOne --service_id=15c11a23667e427e91bc31335b45f4bd --
    publicurl=http://192.168.100.51:5000/v2.0 --internalurl=http
```

```
    ://100.10.10.51:5000/v2.0 --adminurl=http
    ://100.10.10.51:35357/v2.0


#keystone --token 012345SECRET99TOKEN012345 --endpoint http
    ://100.10.10.51:35357/v2.0 user-create --tenant_id
    d80faa9657334317ad1ab5d9772443db --name cinder --pass cinder
    --enabled true

#keystone --token 012345SECRET99TOKEN012345 --endpoint http
    ://100.10.10.51:35357/v2.0 user-role-add --user-id 49
    f62f4f4bd14fb4993ca9c13df813b4 --tenant_id
    d80faa9657334317ad1ab5d9772443db --role-id
    c919af5029764619a6f0cb136d2abd1a

#keystone --token 012345SECRET99TOKEN012345 --endpoint http
    ://100.10.10.51:35357/v2.0/ service-create --name=cinder
    --type=volume --description="Cinder Service"


#keystone --token 012345SECRET99TOKEN012345 --endpoint http
    ://100.10.10.51:35357/v2.0/ endpoint-create --region
    CIPSIRegionOne --service_id=05063185c79d4b9d8e583d1549fa72ea
     --publicurl='http://192.168.100.51:8776/v1/%(tenant_id)s'
    --internalurl='http://100.10.10.51:8776/v1/%(tenant_id)s' --
    adminurl='http://100.10.10.51:8776/v1/%(tenant_id)s'

#keystone --token 012345SECRET99TOKEN012345 --endpoint http
    ://100.10.10.51:35357/v2.0/ service-create --name=quantum --
    type=network --description="OpenStack Networking service"

#keystone --token 012345SECRET99TOKEN012345 --endpoint http
    ://100.10.10.51:35357/v2.0/ endpoint-create --region
    RegionOne --service_id=7f5e6d6f792042ca861dcd7cf18ba301 --
    publicurl=http://192.168.100.51:9696/v2 --internalurl=http
    ://100.10.51:9696/v2 --adminurl=http://100.10.10.51:9696/v2
```

A.1.2 Image Service (Glance)

It is a component for keeping and retrieving of virtual machine images. Images are accessible whenever are needed [1].

```
First of all install glance package:
#apt-get install glance

For ensure that glance is installed correctly, can use below
    command.
```

```
#glance --version
```

Next, create a database for glance, then make a user for glance database with full permission:

```
mysql> CREATE DATABASE glance;
mysql> GRANT ALL ON glance.* TO 'glance'@'%' IDENTIFIED BY '
    glance';
```

For connecting to keystone database we have to update the default value.
Start with glance-api-paste.ini configuration:
Those part that use auth_ it associates to admin service in keystone. It queries keystone to ensure validity of authentication tokens. The admin authentication values are used for retrieving admin tokens which will be used as authorization of user tokens.

```
#gedit /etc/glance/glance-api-paste.ini
[filter:authtoken]
admin_tenant_name = service
admin_user = glance
admin_password = glance

[pipeline:glance-api]
pipeline = versionnegotiation authtoken auth-context apiv1app
```

Add the below command at the end of glance-api.conf

```
#gedit /etc/glance/glance-api.conf
[paste_deploy]
flavor = keystone
```

Add the below command at the end of /etc/glance/glance-registry.conf

```
#gedit /etc/glance/glance-registry.conf
[paste_deploy]
flavor = keystone
```

Update the values in glance-registry-paste.ini file and following pipeline for keystone:

```
#gedit Update /etc/glance/glance-registry-paste.ini
[filter:authtoken]
admin_tenant_name = service
admin_user = glance
admin_password = glance

[pipeline:glance-registry]
```

```
pipeline = authtoken auth-context context registryapp

Then for connecting the glance to keystone (internal connection)
     need to edit sql connection in glance-registry.conf and
    glance-scrubber.conf
sql_connection = mysql://glance:glance@100.10.10.51/glance

Restart the glance services to make changes :
#service glance-api restart
#service glance-registry restart

Verification of glance installation:
Make a directory and download an image.Then upload the image
    which is downloaded.
#mkdir images
#wget https://launchpad.net/cirros/trunk/0.3.0/+download/cirros
    -0.3.0-x86_64-disk.img
#glance image-create --name myFirstImage --is-public true --
    container-format bare --disk-format qcow2 < cirros-0.3.0-
    x86_64-disk.img

glance image-create --location http://uec-images.ubuntu.com/
    releases/12.04/release/ubuntu-12.04-server-cloudimg-amd64-
    disk1.img --is-public true --disk-format qcow2 --container-
    format bare --name "Ubuntu"

Below command list the images that you have uploaded:
(Check credentials where are used be identical with api file
    configuration.)
#glance image-list
```

A.1.3 Quantum (Network):

In this phase , install the quantum server package.

```
# apt-get install quantum-server quantum-plugin-openvswitch
    python-pyparsing

Make quantum database with full permission user:
# mysql -u root -p
mysql> CREATE DATABASE quantum;
mysql> GRANT ALL ON quantum.* TO 'quantumUser'@'%' IDENTIFIED BY
     'quantumPass';
mysql> quit;

Then configure quantum services:
```

```
Open the following script and edit it  /etc/quantum/plugins/
    openvswitch/ovs_quantum_plugin.ini
[DATABASE]
sql_connection = mysql://quantumUser:quantumPass@100.10.10.51/
    quantum
[OVS]
tenant_network_type=vlan
network_vlan_ranges = physnet1:1:4094

As well edit /etc/quantum/api-paste.ini:
[filter:authtoken]
paste.filter_factory = keystone.middleware.auth_token:
    filter_factory
auth_host = 100.10.10.51
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = quantum
admin_password = quantum

Finally restart the service:
# service quantum-server restart
```

A.1.4 Nova:

```
First of all install the nova package:
# apt-get install -y nova-api nova-cert novnc nova-consoleauth
    nova-scheduler nova-novncproxy

For installing nova package ubuntu asks to remove the glance-
    client for replacing by python-glance client. After
    installing python-glance client  the Glance did not work
    properly. Solution was change version and update the system.
    This bug is solved in new version.

Next, database of nova is made with full permission novaUser.
#mysql -u root -p
mysql>CREATE DATABASE nova;
mysql>GRANT ALL ON nova.* TO 'novaUser'@'%' IDENTIFIED BY '
    novaPass';
mysql>quit;


Revise the /etc/nova/api-paste.ini for connecting to keystone:

[filter:authtoken]
paste.filter_factory = keystone.middleware.auth_token:
    filter_factory
```

```
auth_host = 100.10.10.51
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = nova
admin_password = nova
signing_dirname = /tmp/keystone-signing-nova
```

Modify the /etc/nova/nova.conf file:

```
[DEFAULT]
logdir=/var/log/nova
state_path=/var/lib/nova
lock_path=/run/lock/nova
verbose=True
api_paste_config=/etc/nova/api-paste.ini
scheduler_driver=nova.scheduler.simple.SimpleScheduler
s3_host=100.10.10.51
ec2_host=100.10.10.51
ec2_dmz_host=100.10.10.51
rabbit_host=100.10.10.51
dmz_cidr=169.254.169.254/32
metadata_host=100.10.10.51
metadata_listen=0.0.0.0
sql_connection=mysql://novaUser:novaPass@100.10.10.51/nova
root_helper=sudo nova-rootwrap /etc/nova/rootwrap.conf

# Auth
auth_strategy=keystone
keystone_ec2_url=http://100.10.10.51:5000/v2.0/ec2tokens
# Imaging service
glance_api_servers=100.10.10.51:9292
image_service=nova.image.glance.GlanceImageService

# Vnc configuration
vnc_enabled=true
novncproxy_base_url=http://192.168.100.51:6080/vnc_auto.html
novncproxy_port=6080
vncserver_proxyclient_address=192.168.100.51
vncserver_listen=0.0.0.0

# Network settings
network_api_class=nova.network.quantumv2.api.API
quantum_url=http://100.10.10.51:9696
quantum_auth_strategy=keystone
quantum_admin_tenant_name=service
quantum_admin_username=quantum
quantum_admin_password=quantum
quantum_admin_auth_url=http://100.10.10.51:35357/v2.0
```

```
libvirt_vif_driver=nova.virt.libvirt.vif.
    LibvirtHybridOVSBridgeDriver
linuxnet_interface_driver=nova.network.linux_net.
    LinuxOVSInterfaceDriver
firewall_driver=nova.virt.libvirt.firewall.
    IptablesFirewallDriver

# Compute #
compute_driver=libvirt.LibvirtDriver

# Cinder #
volume_api_class=nova.volume.cinder.API
osapi_volume_listen_port=5900


Make tables for nova:
# nova-manage db sync

Restart all services of nova:
# cd /etc/init.d/; for i in $( ls nova-* ); do sudo service $i
    restart; done

check installation for nova services:
# nova-manage service list
```

A.1.5 Cinder (Volume):

```
Install the following packages:
# apt-get install cinder-api cinder-scheduler cinder-volume
    iscsitarget open-iscsi iscsitarget-dkms

Because there is a bug in /etc/tgt/targets.conf, we change the
    script:
# include /etc/tgt/conf.d/*.conf
include /etc/tgt/conf.d/cinder_tgt.conf

Configure and start the below services:
# sed -i 's/false/true/g' /etc/default/iscsitarget
# service iscsitarget start
# service open-iscsi start

Create cinder database and grant full permission to cinderUser:
# mysql -u root -p
mysql>CREATE DATABASE cinder;
mysql>GRANT ALL ON cinder.* TO 'cinderUser'@'%' IDENTIFIED BY '
    cinderPass';
mysql>quit;

Modify /etc/cinder/api-paste.ini of cinder:
```

```
[filter:authtoken]
paste.filter_factory = keystone.middleware.auth_token:
    filter_factory
service_protocol = http
service_host = 192.168.100.51
service_port = 5000
auth_host = 100.10.10.51
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = cinder
admin_password = cinder
```

```
Modify /etc/cinder/cinder.conf file:
[DEFAULT]
rootwrap_config=/etc/cinder/rootwrap.conf
sql_connection = mysql://cinderUser:cinderPass@100.10.10.51/
    cinder
api_paste_confg = /etc/cinder/api-paste.ini
iscsi_helper= tgtadm
volume_name_template = volume-%s
volume_group = cinder-volumes
verbose = True
auth_strategy = keystone
#osapi_volume_listen_port=5900
```

```
Synchronize database:
# cinder-manage db sync

Create a 2GB test loopfile in cinder:
(dd for copying and converting file, if: read from file, of:
    write in file )
# dd if=/dev/zero of=cinder-volumes bs=1 count=0 seek=2G


Then mount it to loop device (loop2):
# losetup /dev/loop2 cinder-volumes

Create physical volume then make volume group which call it
    cinder-volumes:
# pvcreate /dev/loop2
# vgcreate cinder-volumes /dev/loop2
```

```
Then fdisk the partition, type n for adding the partition.
    Enter p for making primary partition and specify
    number of partition as 1 . Change partition system s
    id by entering t , changed system type of partition 1 to
     8e by adding 8e and finally enter w for writing
    table to disk and exit.

The volume group that was made will be lost after restarting.
Restart the services:
# service cinder-api restart
# service cinder-scheduler restart
# service cinder-volume restart
```

A.1.6 Dashboard (Horizon):

```
Install following package:
# apt-get install openstack-dashboard memcached
Then
service apache2 restart; service memcached restart
Therefore dashboard is available at 192.168.100.51/horizon. You
    can use Usernames and Passwords which are made for identity
    service.
```

A.2 Network Node

A.2.1 Preparing Network machine

```
Update the machine:

#apt-get update
#apt-get upgrade
#apt-get dist-upgrade

Install ntp service:
#apt-get install ntp

Configure the NTP server to follow the controller node:
#sed -i 's/server ntp.ubuntu.com/server 100.10.10.51/g' /etc/ntp
    .conf
#service ntp restart
Install other services:

#apt-get install vlan bridge-utils

Enable IP_Forwarding:
#sed -i 's/#net.ipv4.ip_forward=1/net.ipv4.ip_forward=1/' /etc/
    sysctl.conf
To save you from rebooting, perform the following
sysctl net.ipv4.ip_forward=1
```

A.2.2 OpenVSwitch

```
br-eth1 is used for VM configuration, Create a new bridge named
    br-eth1 and add port eth1 to it:
#ovs-vsctl add-br br-eth1
#ovs-vsctl add-port br-eth1 eth1

Create bridge br-ex and assign eth2 port. It is used for
    accessing from the internet:
#ovs-vsctl add-br br-ex
#ovs-vsctl add-port br-ex eth2
```

A.2.3 Quantum

```
Install the Quantum openvswitch agent, l3 agent and dhcp agent:
#apt-get -y install quantum-plugin-openvswitch-agent quantum-
    dhcp-agent quantum-l3-agent

Edit /etc/quantum/api-paste.ini:
[filter:authtoken]
paste.filter_factory = keystone.middleware.auth_token:
    filter_factory
auth_host = 100.10.10.51
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = quantum
admin_password = service_pass

Edit the OVS plugin configuration file /etc/quantum/plugins/
    openvswitch/ovs_quantum_plugin.ini with:
-Under the database section
[DATABASE]
sql_connection = mysql://quantumUser:quantumPass@100.10.10.51/
    quantum

-Under the OVS section
[OVS]
tenant_network_type=vlan
network_vlan_ranges = physnet1:1:4094
bridge_mappings = physnet1:br-eth1

In addition, update the /etc/quantum/l3_agent.ini:
auth_url = http://100.10.10.51:35357/v2.0
auth_region = RegionOne
admin_tenant_name = service
admin_user = quantum
admin_password = service_pass
```

```
metadata_ip = 192.168.100.51
metadata_port = 8775

Make sure that your rabbitMQ IP in /etc/quantum/quantum.conf is
    set to the controller node:
rabbit_host = 100.10.10.51

Restart all the services:
service quantum-plugin-openvswitch-agent restart
service quantum-dhcp-agent restart
service quantum-l3-agent restart
```

A.3 Compute Node

A.3.1 Preparing the Compute machine

```
#apt-get update
#apt-get upgrade
#apt-get dist-upgrade

Install ntp service:
#apt-get install ntp

Configure the NTP server to follow the controller node:
#sed -i 's/server ntp.ubuntu.com/server 100.10.10.51/g' /etc/ntp
    .conf
#service ntp restart

Install other services:
#apt-get install vlan bridge-utils

Enable IP_Forwarding:
#sed -i 's/#net.ipv4.ip_forward=1/net.ipv4.ip_forward=1/' /etc/
    sysctl.conf

-To save you from rebooting, perform the following
sysctl net.ipv4.ip_forward=1
```

A.3.2 KVM

```
make sure that your hardware enables virtualization:
#apt-get install cpu-checker
kvm-ok

Normally you would get a good response. Now, move to install kvm
     and configure it:
#apt-get install -y kvm libvirt-bin pm-utils

Edit the cgroup_device_acl array in the /etc/libvirt/qemu.conf
    file to:
```

```
cgroup_device_acl = [
"/dev/null", "/dev/full", "/dev/zero",
"/dev/random", "/dev/urandom",
"/dev/ptmx", "/dev/kvm", "/dev/kqemu",
"/dev/rtc", "/dev/hpet","/dev/net/tun"
]

Delete default virtual bridge
#virsh net-destroy default
#virsh net-undefine default

Enable live migration by updating /etc/libvirt/libvirtd.conf
    file:
listen_tls = 0
listen_tcp = 1
auth_tcp = "none"

Edit libvirtd_opts variable in /etc/init/libvirt-bin.conf file:
env libvirtd_opts="-d -l"

Edit /etc/default/libvirt-bin file:
libvirtd_opts="-d -l"

Restart the libvirt service to load the new values:
#service libvirt-bin restart
```

A.3.3 OpenVSwitch

```
Install the openVSwitch:
#apt-get install -y openvswitch-switch openvswitch-datapath-dkms

Create the bridges:
#ovs-vsctl add-br br-int

br-int will be used for VM integration
br-eth1 will be used for VM configuration

#ovs-vsctl add-br br-eth1
#ovs-vsctl add-port br-eth1 eth1
```

A.3.4 Quantum

```
Install the Quantum openvswitch agent:
apt-get -y install quantum-plugin-openvswitch-agent

Edit the OVS plugin configuration file /etc/quantum/plugins/
    openvswitch/ovs_quantum_plugin.ini with:
-Under the database section
[DATABASE]
```

```
sql_connection = mysql://quantumUser:quantumPass@100.10.10.51/
    quantum

-Under the OVS section
[OVS]
tenant_network_type=vlan
network_vlan_ranges = physnet1:1:4094
bridge_mappings = physnet1:br-eth1
Make sure that your rabbitMQ IP in /etc/quantum/quantum.conf is
    set to the controller node:

rabbit_host = 100.10.10.51

Restart all the services:
service quantum-plugin-openvswitch-agent restart
```

A.3.5 Nova

```
Install nova's required components for the compute node:
apt-get install nova-compute-kvm

Now modify authtoken section in the /etc/nova/api-paste.ini file
     to this:

[filter:authtoken]
paste.filter_factory = keystone.middleware.auth_token:
    filter_factory
auth_host = 100.10.10.51
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = nova
admin_password = service_pass
signing_dirname = /tmp/keystone-signing-nova
Edit /etc/nova/nova-compute.conf file

[DEFAULT]
libvirt_type=kvm
libvirt_ovs_bridge=br-int
libvirt_vif_type=ethernet
libvirt_vif_driver=nova.virt.libvirt.vif.
    LibvirtHybridOVSBridgeDriver
libvirt_use_virtio_for_bridges=True

Modify the /etc/nova/nova.conf like this:
[DEFAULT]
logdir=/var/log/nova
state_path=/var/lib/nova
```

```
lock_path=/run/lock/nova
verbose=True
api_paste_config=/etc/nova/api-paste.ini
scheduler_driver=nova.scheduler.simple.SimpleScheduler
s3_host=100.10.10.51
ec2_host=100.10.10.51
ec2_dmz_host=100.10.10.51
rabbit_host=100.10.10.51
dmz_cidr=169.254.169.254/32
metadata_host=100.10.10.51
metadata_listen=0.0.0.0
sql_connection=mysql://novaUser:novaPass@100.10.10.51/nova
root_helper=sudo nova-rootwrap /etc/nova/rootwrap.conf

# Auth
use_deprecated_auth=false
auth_strategy=keystone
keystone_ec2_url=http://100.10.10.51:5000/v2.0/ec2tokens
# Imaging service
glance_api_servers=100.10.10.51:9292
image_service=nova.image.glance.GlanceImageService

# Vnc configuration
novnc_enabled=true
novncproxy_base_url=http://192.168.100.51:6080/vnc_auto.html
novncproxy_port=6080
vncserver_proxyclient_address=100.10.10.53
vncserver_listen=0.0.0.0

# Network settings
network_api_class=nova.network.quantumv2.api.API
quantum_url=http://100.10.10.51:9696
quantum_auth_strategy=keystone
quantum_admin_tenant_name=service
quantum_admin_username=quantum
quantum_admin_password=service_pass
quantum_admin_auth_url=http://100.10.10.51:35357/v2.0
libvirt_vif_driver=nova.virt.libvirt.vif.
    LibvirtHybridOVSBridgeDriver
linuxnet_interface_driver=nova.network.linux_net.
    LinuxOVSInterfaceDriver
firewall_driver=nova.virt.libvirt.firewall.
    IptablesFirewallDriver

# Compute #
compute_driver=libvirt.LibvirtDriver

# Cinder #
volume_api_class=nova.volume.cinder.API
```

```
osapi_volume_listen_port=5900

Restart nova-* services:
cd /etc/init.d/; for i in $( ls nova-* ); do sudo service $i
    restart; done

Check for the smiling faces on nova-* services to confirm your
    installation:
nova-manage service list
```

Appendix B
Snort Installation

```
First of all, install all following packages:
#sudo apt-get install mysql-server
#sudo apt-get install libmysqlclient-dev
#sudo apt-get install libpcap0.8-dev
#sudo apt-get install nbtscan
#sudo apt-get install flex
#sudo apt-get install make
#sudo apt-get install libpcap-ruby
#sudo apt-get install libpcre3-dev
#sudo apt-get install autoconf
#sudo apt-get install libtool
#sudo apt-get install bison

apache2
libapache2-mod-php5
php5-gd
php5-mysql
libphp-adodb
php-pear


Download and install Data Acquisition API:
#wget daq-2.0.0.tar.gz http://www.snort.org/downloads/2103
#cd daq-2.0.0
#sudo ./configure
#sudo make
#sudo make install


Download and install libdnet:
#wget http://libdnet.googlecode.com/files/libdnet-1.12.tgz
#sudo tar zxvf libdnet-1.12.tgz
#cd libdnet-1.12/
#sudo ./configure
#sudo make
#sudo make install
#sudo ln -s /usr/local/lib/libdnet.1.0.1 /usr/lib/libdnet.1


Download and install Snort report:
#sudo vi /var/www/snortreport-1.3.3/srconf.php
Set the password which has been chosen for SQL.
```

```
Download and install Snort:
#wget http://www.snort.org/downloads/2225 -O snort-2.9.4.5.tar.
    gz
#sudo tar zxvf snort-2.9.4.5.tar.gz
-Snort is installed in specified path /usr/local/snort
#sudo ./configure --prefix=/usr/local/snort --enable-sourcefire
#sudo make
#sudo make install
#sudo mkdir /var/log/snort
#sudo mkdir /var/snort
#sudo groupadd snort
#sudo useradd -g snort snort
#sudo chown snort:snort /var/log/snort


Download the last ruleset for Snort:
#wget https://www.dropbox.com/s/ewr9ttyq4d5yua2/snortrules-
    snapshot-2941.tar.gz
#tar zxvf snortrules-snapshot-2941.tar.gz -C /usr/local/snort
#sudo mkdir /usr/local/snort/lib/snort_dynamicrules


Be careful about 64 or 32 bit operating system (x86-64 is for 64
    bit)
#cp  /usr/local/snort/so_rules/precompiled/Ubuntu-10-4/x86
    -64/2.9.4.0/* /usr/local/snort/lib/snort_dynamicrules
#touch /usr/local/snort/rules/white_list.rules
#touch /usr/local/snort/rules/black_list.rules
#ldconfig
```

B.1 Snort configuration

```
#sudo vi /usr/local/snort/etc/snort.conf

-Edit the paths in snort.conf:
var WHITE_LIST_PATH /usr/local/snort/rules
var BLACK_LIST_PATH /usr/local/snort/rules

dynamicpreprocessor directory /usr/local/snort/lib/
    snort_dynamicpreprocessor/
dynamicengine /usr/local/snort/lib/snort_dynamicengine/
    libsf_engine.so
dynamicdetection directory /usr/local/snort/lib/
    snort_dynamicrules

Output from unified2 for Barnyard:
Add the line below instead of (output unified2...)
output unified2: filename snort.u2, limit 128
```

```
Download and install Barnyard2
It reduces the load detect engine and improve the performance of
    IDS.

#wget https://github.com/firnsy/barnyard2/archive/master.tar.gz
#tar zxvf master.tar.gz
#cd barnyard2*
#autoreconf -fvi -I ./m4


During installation ./configure I received below error
ERROR: unable to find mysqlclient library (libmysqlclient.*)

Then run the following command for finding location of
    libmysqlclient:
#locate libmysqlclient


Afterwards, path was fixed which system was looking for:
#./configure --with-mysql --with-mysql-libraries=/usr/lib/
    x86_64-linux-gnu/
#make
#make install
#cp /home/anna/files/barnyard2-master/etc/barnyard2.conf /usr/
    local/snort/etc
#mkdir /var/log/barnyard2
#chmod 666 /var/log/barnyard2
#touch /var/log/snort/barnyard2.waldo
#chown snort.snort /var/log/snort/barnyard2.waldo


Creating database for snort:
#echo "create database snort;" | mysql -u root -p
#mysql -u root -p -D snort < ./schemas/create_mysql
-Make a user for snort database:
#echo "grant create, insert, select, delete, update on snort.*
    to snort@localhost identified by '44990650' " | mysql -u root
    -p

Edit the barnyard2 configuration:
#sudo vi /usr/local/snort/etc/barnyard2.conf
config reference_file: /usr/local/snort/etc/reference.config
config classification_file: /usr/local/snort/etc/classification.
    config
config gen_file: /usr/local/snort/etc/gen-msg.map
config sid_file: /usr/local/snort/etc/sid-msg.map

Add the following lines:
```

```
config hostname: localhost
config interface: eth1
output database: log, mysql, user=snort password=44990650 dbname
    =snort host=localhost
```

B.2 Installing BASE (Basic Analysis and Security Engine)

```
First download the base-1.4.5 source code by below link:
#wget \ http://downloads.sourceforge.net/project/secureideas/
    BASE/base-1.4.5/base-1.4.5.tar.gz?r=http%3A%2F%2Fsourceforge.
    net%2Fprojects%2Fsecureideas%2Ffiles%2FBASE%2Fbase-1.4.5%2F&
    ts=1368009891&use_mirror=surfnet

Then extract the content.Download the database abstraction
    library for PHP
#wget \ http://downloads.sourceforge.net/project/adodb/adodb-
    php5-only/adodb-518-for-php5/adodb518a.tgz?r=http%3A%2F%2
    Fsourceforge.net%2Fprojects%2Fadodb%2Ffiles%2Fadodb-php5-only
    %2Fadodb-518-for-php5%2F&ts=1368013157&use_mirror=heanet
Then extract the content.
Because we install apache2 and BASE for running the base, we
    need to change some configuration.

Add two lines to /var/www/apache2/php.ini under the Dynamic
    extension section:
extension=mysql.so
extension=gd.so

Afterwards, restart the apache2 service:
#/etc/init.d/apache2 restart

For testing the system, create the vim /var/www/test.php file.
    Enter on browser http://152.94.0.235/test.php.
If the following page shows php information, everything is
    working properly.

Move adodb to /var/www directory
Make web directory and move the BASE to it (/var/www/web).
For preparing the BASE need to change the permission for base
    folder.
#chmod 757 /var/www/web/base-1.4.5

Because BASE is working with graphic, handful extensions are
    installed:
#pear install Image_Color
#pear install Image_Canvas-alpha
#pear install Image_Graph-alpha
```

```
For setting up the BASE, open the browser and type following
    address, then enter continue.
http://152.94.0.235/web/base-1.4.5/setup

There is five levels:
1. Specify the place of adodb5(/var/www/adodb5).
2. Database type = MySQL, Database name = snort, Database Host =
    localhost, Database username = root, Database Password =
    xxxxx
3. Enter username and password for base authentication.
4. Click on Create BASE.
5. The last step shows previous levels are done successful and
    save the page.

Finally, change the permission of the BASE folder.
#chmod 775 /var/www/web/base-1.4.5
```