

**VIVEKANAND EDUCATION SOCIETY'S INSTITUTE OF
TECHNOLOGY**

Department of Computer Engineering



Project Report on

SDN based Firewall

In partial fulfillment of the Fourth Year, Bachelor of Engineering (B.E.) Degree in Computer
Engineering at the University of Mumbai Academic Year 2017-2018

Submitted by

Yash Bajaj (D17 - B, Roll no - 05)

Yogesh Rohra (D17 - B, Roll no - 61)

Viren Wadhwani (D17 - B, Roll no - 72)

Project Mentor

Prof. Mrs. Arthi C I

(2017-18)

VIVEKANAND EDUCATION SOCIETY'S INSTITUTE OF TECHNOLOGY

Department of Computer Engineering



Certificate

This is to certify that *Yash Bajaj, Yogesh Rohra, Viren Wadhwani* of Fourth Year Computer Engineering studying under the University of Mumbai have satisfactorily completed the project on “*SDN based Firewall*” as a part of their coursework of PROJECT-II for Semester-VIII under the guidance of their mentor *Prof. Mrs. Arthi C I* in the year 2017-2018 .

This thesis/dissertation/project report entitled *SDN based Firewall* by *Yash Bajaj, Yogesh Rohra, Viren Wadhwani* is approved for the degree of *B.E.*

Programme Outcomes	Grade
PO1, PO2, PO3, PO4, PO5, PO6, PO7, PO8, PO9, PO10, PO11, PO12, PSO1, PSO2	

Date:

Project Guide: Prof. Mrs. Arthi C I

Project Report Approval

For

B. E (Computer Engineering)

This thesis/dissertation/project report entitled *SDN based Firewall* by *Yash Bajaj, Yogesh Rohra, Viren Wadhwani* is approved for the degree of *B.E.*

Internal Examiner

External Examiner

Head of the Department

Principal

Date:

Place:

Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)

(Signature)

Yash Bajaj (D17 - B, Roll no - 05)

Yogesh Rohra (D17 - B, Roll no – 61)

(Signature)

Viren Wadhwani (D17 - B, Roll no – 72)

Date:

ACKNOWLEDGEMENT

We are thankful to our college Vivekananda Education Society's Institute of Technology for considering our project and extending help at all stages needed during our work of collecting information regarding the project.

It gives us immense pleasure to express our deep and sincere gratitude to Assistant Professor **Mrs. Arthi C I** (Project Guide) for her kind help and valuable advice during the development of project synopsis and for her guidance and suggestions.

We are deeply indebted to Head of the Computer Department **Dr.(Mrs.) Nupur Giri** and our Principal **Dr. (Mrs.) J.M. Nair** , for giving us this valuable opportunity to do this project.

We express our hearty thanks to them for their assistance without which it would have been difficult in finishing this project synopsis and project review successfully.

We convey our deep sense of gratitude to all teaching and non-teaching staff for their constant encouragement, support and selfless help throughout the project work. It is great pleasure to acknowledge the help and suggestion, which we received from the Department of Computer Engineering.

We wish to express our profound thanks to all those who helped us in gathering information about the project. Our families too have provided moral support and encouragement at several times.

Computer Engineering Department
COURSE OUTCOMES FOR B.E PROJECT

Learners will be able to,

Course Outcome	Description of the Course Outcome
CO 1	Able to apply the relevant engineering concepts, knowledge and skills towards the project.
CO2	Able to identify, formulate and interpret the various relevant research papers and to determine the problem.
CO 3	Able to apply the engineering concepts towards designing solution for the problem.
CO 4	Able to interpret the data and datasets to be utilized.
CO 5	Able to create, select and apply appropriate technologies, techniques, resources and tools for the project.
CO 6	Able to apply ethical, professional policies and principles towards societal, environmental, safety and cultural benefit.
CO 7	Able to function effectively as an individual, and as a member of a team, allocating roles with clear lines of responsibility and accountability.
CO 8	Able to write effective reports, design documents and make effective presentations.
CO 9	Able to apply engineering and management principles to the project as a team member.
CO 10	Able to apply the project domain knowledge to sharpen one's Competency.
CO 11	Able to develop professional, presentational, balanced and structured approach towards project development.
CO 12	Able to adopt skills, languages, environment and platforms for creating innovative solutions for the project.

Abstract

In today's evolving technology, the need to adapt is the only necessity. The same is applicable when we are talking about the field of Network and Security i.e the SDN technology. SDN is the newest evolving topic in the field of Networking. There are many problems faced by the traditional networks. These are solved by SDN. The different advantages of SDN are:

- It is a way of making networks more manageable and more dynamic.
- It is very helpful in realizing the whole network beforehand thus giving a clear idea of the topologies required and the issues which can be solved even before setting the network up.
- Provides economic help as it has lower operating costs.
- SDN makes it easier to optimize commoditized hardware. Existing hardware can be repurposed using instructions from the SDN controller and less expensive hardware can be deployed to greater effect.
- The specific advantages of software defined networking will vary from network to network, but there are benefits from network abstraction and the agility it offers for network administration and automation

But as in a new technology comes up it brings in new threats and issues. Thus our aim is to build a firewall over this new technology to protect the integrity of it. In this project we focus on designing and developing a Firewall for our SDN controller which in turn performs the function of safe guarding the network from inside as well as the outside internet. To perform experiment, we have installed Mininet emulator for creating network topologies. In here, we present the implementation details of the firewall application.

Table of Contents

Chapter no.	Title	Page no.
1	Introduction	1-11
1.1	Introduction to the project	1
1.2	Motivation	3
1.3	Problem Definition	3
1.4	Relevance of the project	4
1.5	Methodology employed for development	5-11
2	Literature Survey	12-18
2.1	Papers	12
2.2	Patent Search	15
3	Requirement Gathering	19-20
3.1	Functional Requirements	19
3.2	Non-Functional Requirements	20
3.3	Hardware Requirements	20
3.4	Software Requirements	20
4	Proposed Design	21-31
4.1	Block Diagram of the system	21
4.2	System Design	23
4.3	Detailed Design	24
4.4	Project Scheduling & Tracking	28

5	Implementation Details	31
6	Testing	32-36
7	Result Analysis	36
8	Conclusion	37
9	Plan For the Entire Semester	38
	References	34
	Appendix	37
	CD	

List of figures

Figure 1	Comparison with Existing System
Figure 2	Firewall GUI basic screen
Figure 3	Tree Topology
Figure 4	Mesh Topology
Figure 5	Append Rule

Figure 6	Delete Rule
Figure 7	Insert Rule
Figure 8	Attack Prevention UI
Figure 9	SDN Physical and Virtual network Integration
Figure 10	SDN & Firewall
Figure 11	Block Diagram 1
Figure 12	Block Diagram 2
Figure 13	System Design
Figure 14	DFD Level 0
Figure 15	DFD level 1
Figure 16	DFD level 2
Figure 17	State Diagram
Figure 18	Project Scheduling
Figure 19	Gantt Chart 1
Figure 20	Gantt Chart 2
Figure 21	Gantt Chart 3
Figure 22	Gantt Chart 4
Figure 23	Gantt Chart 5
Figure 24-34	Testing 1-10
Figure 35,36	Project Review 1,2

List of tables

Table 1	Plagiarism report (1.1)
Table 2	Plagiarism report (1.2)
Table 3	Plagiarism report (1.3)
Table 4	Plagiarism report (2.1)
Table 5	Plagiarism report (2.2)
Table 6	Plagiarism report (2.3)

1. Introduction

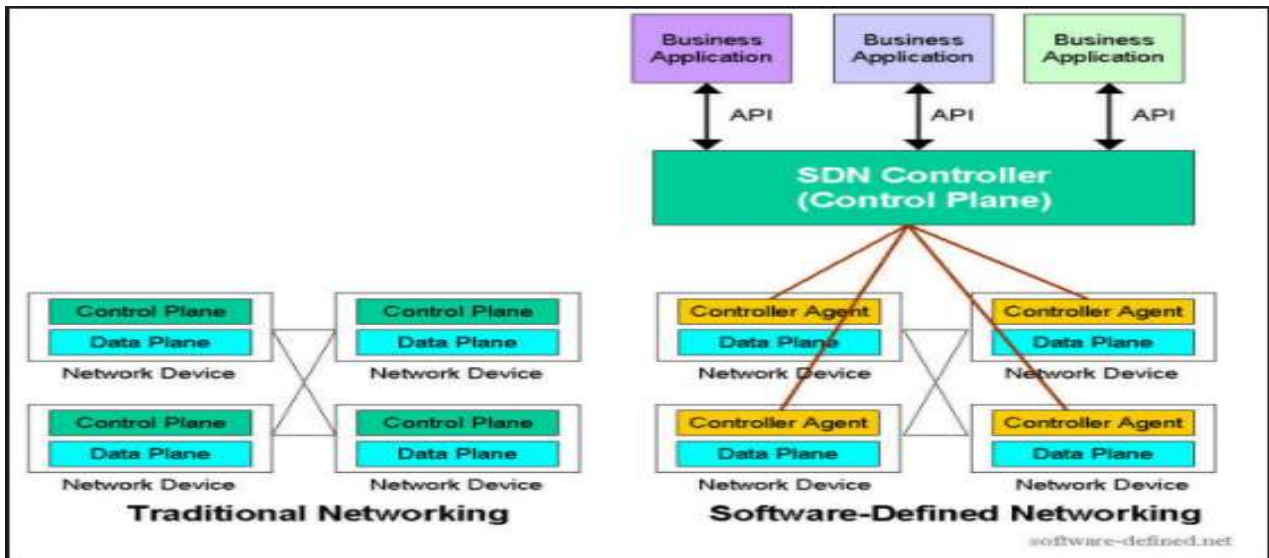
1.1 Introduction to the project:

The current internet network is well set and huge in terms of topologies and its connections all over the world. But it has its drawbacks too. The data and the control plane being together yet makes it difficult in finding optimality in terms of routing, maintaining security and much more. Also adding or removing even a single router makes us change the whole network and is tedious. In a view of finding solution the SDN technology was introduced in 2005 where in the control plane gets totally removed and what we get is a controller i.e. the brain of the network and a bunch of routers that are meant for handling data and routing them. The new brain so formed is called as the SDN controller in the world of SDN. A controller responsible for taking the routing, protocol decisions whenever a packet arrives at its door.

Our project aims at building a firewall over this controller that will in turn help the controller to take the decisions regarding the packets that arrive at the controller. The controller accompanied with the firewall will have the ability to accept, drop, or reject the incoming packets thus ensuring the safety of the system from malicious packets. The Firewall thus created will stand in between the controller and outside internet. Also it has the ability to handle the flow of packets shown by the controller thus providing an all-rounder functionality where the controller is the mega brain and has the record of every bit of data irrespective whether it flows in our out of network. Also the firewall which is coded in Python, provides a simplified GUI for the users to set up the firewall as per his/her convenience and also it will be able to save the system from any malicious attack by an intended attacker.

Thus the project aims at emulate a network topology. The implementation includes the respective emulation of the network along with the firewall.

On viewing the existing system and comparing it with the project, we find that the basic disadvantages are the more privileged uses of the SDN technology. As mentioned earlier, the existing networks are highly complex. The actual comparison between the two can be depicted by the figure.



The traditional and yet existing system consists of networking elements that possess both the control and the data plane in every element that is installed in the network. This puts a slight disadvantage as setting up a new element would require making changes in control plane of almost every element that exists in the network. This leads to decrease in optimality goal of the network. On the other hand the control plane from every element of the network is detached and the SDN controller is created and the elements left without the control pane are now termed as OvSwitches that are just used to regulate the flow data packets. This provides more configuration flexibility and accuracy accompanied with Data Flow optimization.

Same thing happens in the case of Firewalls in the traditional and SDN networks. The firewalls in the traditional networks have Limited functionalities where as in SDN more privileges are granted to the same firewall thus providing better security.

1.2 Motivation

Well Computer Science isn't just programming. In the real world, you'll often need to interact with customers, teammates, supervisors and others in a variety of ways. This can range from explaining how exactly to interact with different features, to more mundanely explaining concisely what work has been completed. Our project is "SDN based Firewall". Firewall inspects packets that attempt to cross a network boundary. Firewall rejects any illegal packets such as Incoming requests to open illegal TCP connections, Packets of other illegal types (e.g., UDP and ICMP), and IP datagrams with illegal IP addresses (or ports). Firewall provides security at the loss of flexibility and the cost of network administration. SDN helps us in separation of traffic monitoring and forwarding mechanisms. It is basically concerned to securities blending with networking. The motivation behind the project is our interest in securities and the urge to try out new evolving techniques that soon are going to emerge as the dominant technologies.

1.3 Problem Definition:

The current internet network is well set and huge in terms of topologies and its connections all over the world. But it has its drawbacks too. The data and the control plane being together yet makes it difficult in finding optimality in terms of routing, maintaining security and much more. Also adding or removing even a single router makes us change the whole network and is tedious. In a view of finding solution the SDN technology was introduced in 2005 where in the control plane gets totally removed and what we get is a controller i.e. the brain of the network and a bunch of routers that are meant for handling data and routing them. The new brain so formed is called as the SDN controller in the world of SDN. A controller responsible for taking the routing, protocol decisions whenever a packet arrives at its door.

Our project aims at building a firewall over this controller that will in turn help the controller to take the decisions regarding the packets that arrive at the controller. The controller accompanied with the firewall will have the ability to accept, drop, or reject the incoming packets thus ensuring the safety of the system from malicious packets. The Firewall thus created will stand in between the controller and outside internet. Also it has the ability to handle the flow of packets shown by the controller thus providing an all rounder functionality where the controller is the mega brain and has the record of every bit of data irrespective

whether it flows in our out of network. Also the firewall which is coded in Python, provides a simplified GUI for the users to set up the firewall as per his/her convenience and also it will be able to save the system from any malicious attack by an intended attacker.

1.4 Relevance of the Project

Traditional network has hardware and software which is directed across series of router or switches where switches has the logic of passing the packets and protocols. The control and data plane were connected together and hence if the data plane was congested then control planes functions were affected too and therefore the system would allow slow forwarding mechanisms and traffic monitoring. The concept of SDN in networking helps us solving the above mentioned problems. Software Defined Networking (SDN) helps us to separate the control and data plane thus giving us the power to make the network programmable and divert the data as per our requirement. Hence there will be a centralized control with a better view of the network. Since the switches need not be intelligent and behave like a dumb terminal, their cost would be less. Our project is based on incorporating a “Firewall” in SDN which will help us in achieving more benefits.

1.5 Methodology employed for development

The System proposed is a firewall that is based on the concept of SDN technology i.e. it will perform the functionalities of a firewall based on the fact that it has to safeguard the Controller as well as the network it is working on. It will not only keep a check on the outside network but also the emulated network in the Mininet.keeping this view in record, the Firewall will be implemented in following steps

1.4.1 Coding the Firewall in Python Language:

Since we will be working with mininet, it supports Python codes and hence the firewall to sit upon the Controller will be coded in Python. We will use tkinter library of Python for creating the GUI of our firewall. The GUI will have two major blocks one is the Status which will tell the current network connections of the controller and the network emulated and other will be Policy which will display the rules set up by the user. It will also have different toggles and options for insert update and delete of functionalities and help too.

The Screenshot below shows the GUI created for the firewall.

FIREWALL FOR SDN					
Firewall Edit Event Policy Help					
Status			Policy		
Source Add	Source Port	Destination Add	Destination Port	Protocol	State
192.168.43.71	55498	172.217.160.163	80	tcp	SYN_SENT
192.168.43.71	58082	54.69.184.117	443	tcp	SYN_SENT
192.168.43.71	58086	54.69.184.117	443	tcp	SYN_SENT
2405	60604	2404	80	tcp6	SYN_SENT
192.168.43.71	36402	192.168.43.1	53	udp	ESTABLISHED
192.168.43.71	45473	192.168.43.1	53	udp	ESTABLISHED
192.168.43.71	38524	192.168.43.1	53	udp	ESTABLISHED
192.168.43.71	47927	192.168.43.1	53	udp	ESTABLISHED
0.0.0.0	68	0.0.0.0	*	udp	

1.4.2 Installing Mininet and Creating the Network Topology.:

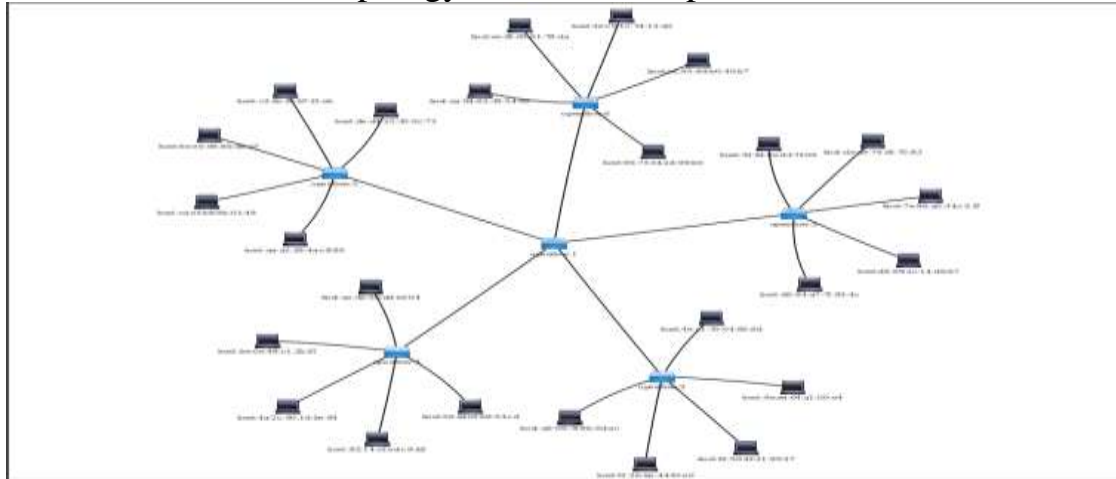
Our project will be emulated in Mininet which is a network emulator which creates a network of virtual hosts, switches, controllers, and links in other sense a fully functioning network as per convenience. The hosts thus created run standard Linux network software, and its switches support highly flexible custom routing i.e SDN.

The Advantages of Mininet are:

- It provides a simple and inexpensive network testbed for development and research.
- It allows multiple concurrent developers to work independently in the same network on the same topology.
- It allows testers to perform complex topology testing, without the need to wire up a physical network and analyze it for proper optimal results.
- Mininet can be used without programming that is out of the box, but also provides an extensible Python API for network creation and experimentation purposes.
- It helps in getting an easy way to get correct system behavior(and, to the extent supported by your hardware, performance) and to experiment with topologies which can be created or imagined.

These networks which will be created virtually resemble a real Linux kernel and network stack (including any kernel extensions which you may have available, as long as they are compatible with network namespaces.) Thus GUI developed will be tested on Mininet, for an SDN controller, with modified switch, or host, also can be moved to a real system with minimal changes, for real-world testing, performance evaluation, and deployment. Thus indicating that a design that works in Mininet can usually be moved directly to hardware. After installation setting up the network is simply work of commands to specify the type of topology and number of nodes.

One such Tree topology is show in the picture below.

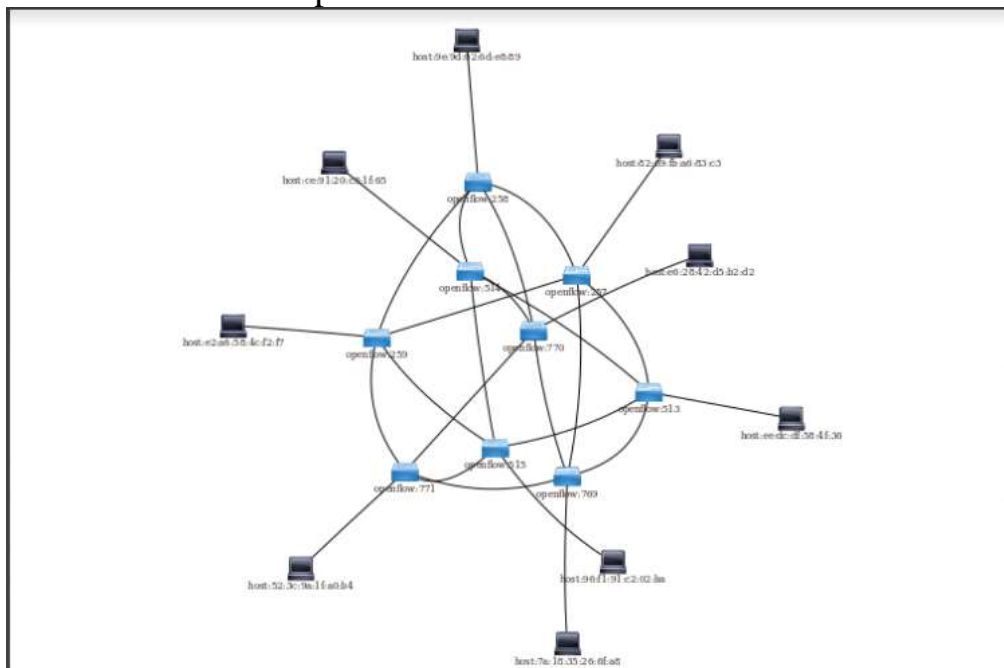


This Topology has the SDN controller at the center. Each switch including the controller has 5 connections. So in all it has 1 controller 5 switches and 25 nodes.

The command used for the same is

“sudo mn –topo tree,depth=2 fanout=5 –controller=remote”

Another example for the same is:

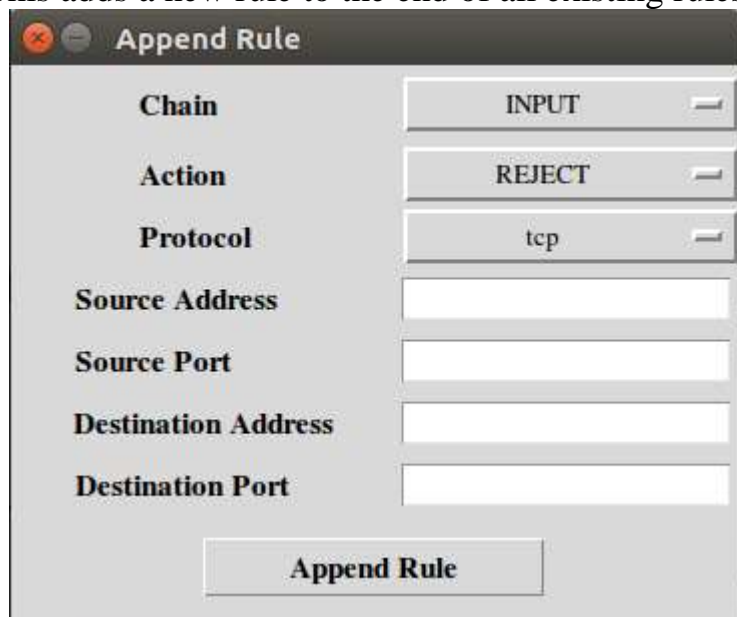


1.4.3 Setting the Firewall(Firewall Setup Phase).

This is obviously the essential step of the project as after the setting of the firewall the rules that guard the firewalls judgment are decided.

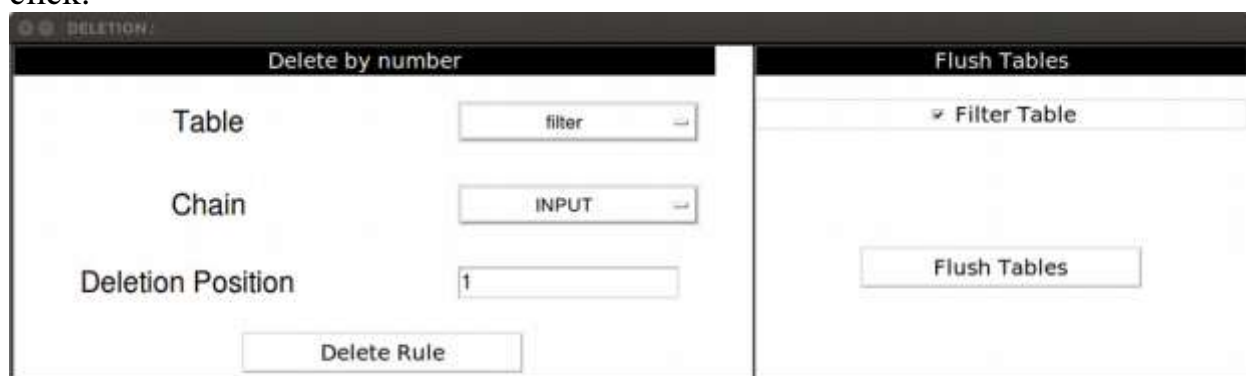
In this step the GUI created in step 1 is put to actual use. The user uses the GUI to insert proper rules in the iptables of the firewall and and checks the network statistics Like the traditional networks, here also the iptables consist of Input Output and Forward tables with following functionalities

a)Append:- This adds a new rule to the end of all existing rules.



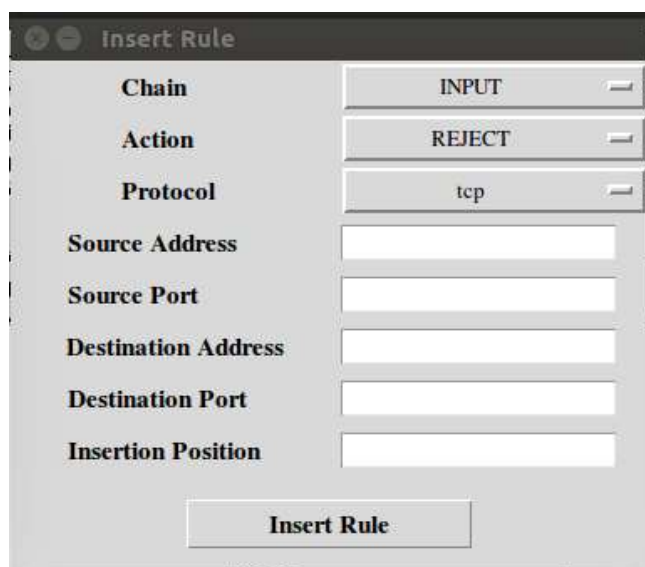
The 'Append Rule' dialog box is a window with a title bar containing a close button and a minus button. It has a light gray background. The dialog contains several labeled fields on the left and corresponding input controls on the right. The labels are 'Chain', 'Action', 'Protocol', 'Source Address', 'Source Port', 'Destination Address', and 'Destination Port'. The input controls are dropdown menus for 'Chain' (showing 'INPUT'), 'Action' (showing 'REJECT'), and 'Protocol' (showing 'tcp'), and text input boxes for 'Source Address', 'Source Port', 'Destination Address', and 'Destination Port'. At the bottom center is a button labeled 'Append Rule'.

b)Delete: this is used to delete a rule where as Flush totally clears all the rules in a single click.



The 'Delete Rule' dialog box is a window with a title bar containing a close button and a minus button. It has a light gray background. The dialog is split into two panes. The left pane is titled 'Delete by number' and contains labels 'Table', 'Chain', and 'Deletion Position'. The 'Table' label is next to a dropdown menu showing 'filter'. The 'Chain' label is next to a dropdown menu showing 'INPUT'. The 'Deletion Position' label is next to a text input box containing '1'. At the bottom of the left pane is a button labeled 'Delete Rule'. The right pane is titled 'Flush Tables' and contains a dropdown menu labeled 'Filter Table' with a checkmark icon. At the bottom of the right pane is a button labeled 'Flush Tables'.

c)Insert: to add a rule at the start .



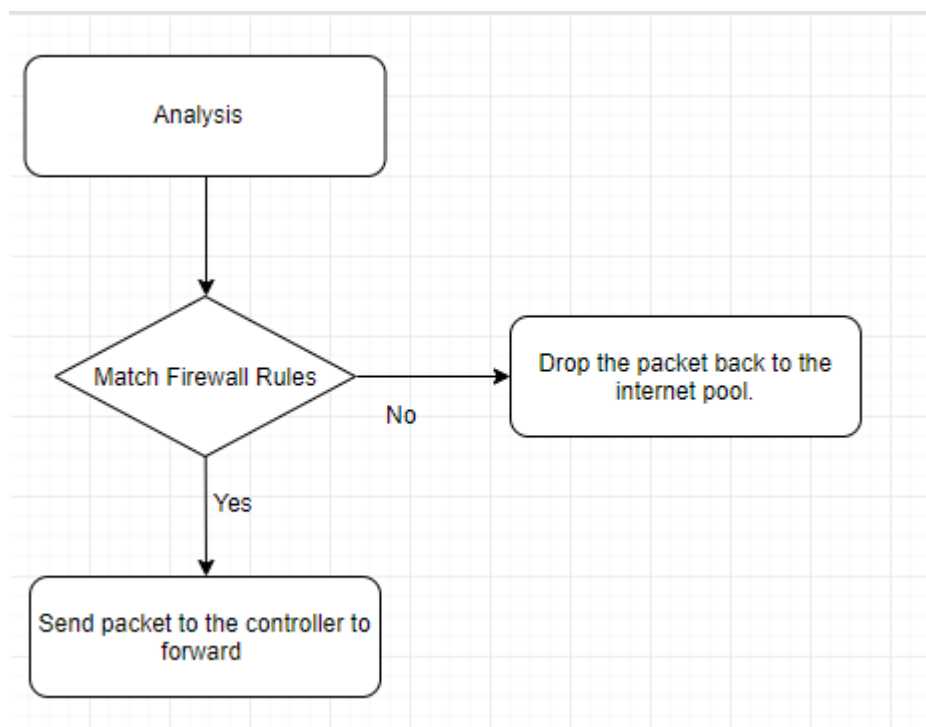
The 'Insert Rule' dialog box is a window with a title bar containing a close button and a minus button. It has a light gray background. The dialog contains several labeled fields on the left and corresponding input controls on the right. The labels are 'Chain', 'Action', 'Protocol', 'Source Address', 'Source Port', 'Destination Address', 'Destination Port', and 'Insertion Position'. The input controls are dropdown menus for 'Chain' (showing 'INPUT'), 'Action' (showing 'REJECT'), and 'Protocol' (showing 'tcp'), and text input boxes for 'Source Address', 'Source Port', 'Destination Address', 'Destination Port', and 'Insertion Position'. At the bottom center is a button labeled 'Insert Rule'.

Likewise it has three functions as:

- a)Accept: to allow the packets to pass in the network.
- b)Drop: to disallow any packet without even any acknowledgment
- c)Reject: to disallow any packet with an acknowledgment

1.4.4 Checking the FUNCTIONALITIES (Operational Phase)

In this step, the firewall that is deployed is put to function. Here in each and every functionality of the Firewall is exploited and checked upon to verify its efficiency. This Step can be depicted diagrammatically as below:



The different steps are:

Step1: Analysis

The packet that arrived at the firewall needs to be analyzed for various factors like the source address, destination address, the network subnet and the headers etc. Each and every aspect that needs to be checked for maintaining integrity of the data packet received and to be assured that the contents are error-less are performed in this very step. The end of this step marks a verified data packet and that packet itself has to face the firewall created by us. The analysis verifies the packet and if non suitable case found, the packet is thrown away into the internet pool and is not sent further for any verification. On the other hand the packet that packet analysis moves ahead to face the wall. Thus analysis proves as the door step verification or as the first line of verification that can be performed by the firewall.

Step 2: Match Firewall Rules:

The second step marks the second line of verification in the project. Here the packet that passed the analysis is put to face the Firewall itself. The firewall uses the rules that it sets up in the Setup Phase of the project and verifies the packet accordingly. It checks for contents of the packet and check for malicious data, bugs,worms that may or may not be present in the data packet. The rules that are set help the firewall in deciding whether the packet received is suitable for the host or not. The questions like “whether the packet is true or not?” and “whether the packet will cause harm or no harm at all?” are to be answered by the firewall and then as per the answers it decides the quality of the packet. This quality factor proves helpful in deciding the further fate of the data packet. If the firewall feels that the packet is suitable for the host then it is passed and forwarded to the respective OVSwitch by the SDN controller. On the other hand if it fails to pass, then it is straight away thrown back into the internet pool and left there to travel until its TTL comes to an end. There also may exist a possibility where in the packet keeps on re-arriving at the same firewall and getting rejected a multiple times by the same procedure.

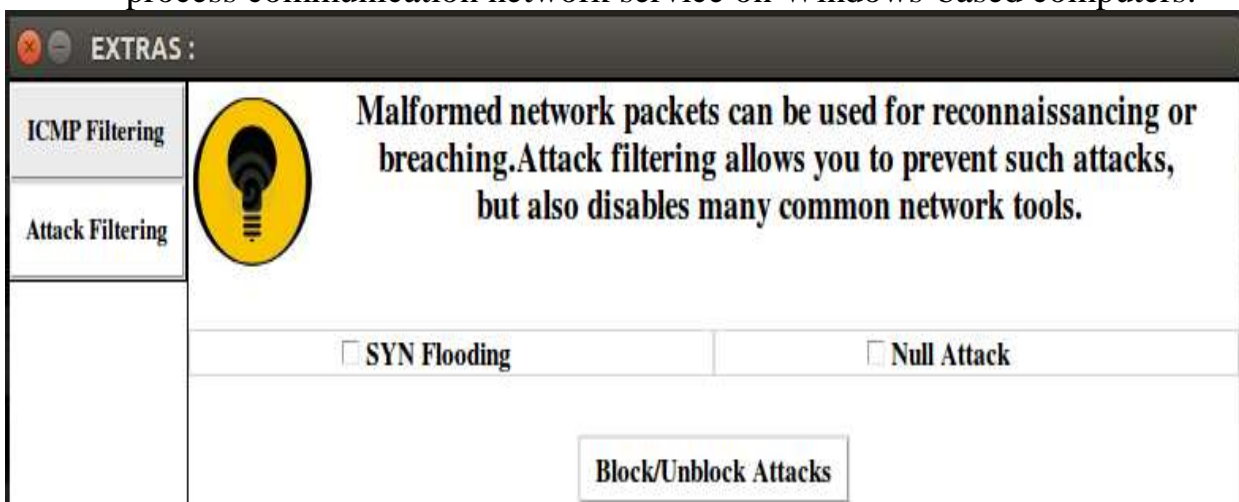
1.4.5 The Attack on the Firewall:

This is the final testing step in the project. Inhere on purpose a malicious attack will be performed on the firewall to check for its functionalities. The attack can be of any type and the firewall must be able to defend against it.

The attacks performed are:-

a)Syn Flooding:- In here a flow of continued SYN request is sent to the target so as to consume up the resources of the target.

b)Null attack: A null session is an anonymous connection to an inter-process communication network service on Windows-based computers.



2.Literature Survey

2.1 Papers:

2.1.1 Software Defined Network:-

Software-defined networking (SDN) technology is an approach to computer networking that allows network administrators to programmatically initialize, control, change, and manage network behavior dynamically via open interfaces and provide abstraction of lower-level functionality. SDN is meant to address the fact that the static architecture of traditional networks doesn't support the dynamic, scalable computing and storage needs of more modern computing environments such as data centers. This is done by decoupling or disassociating the system that makes decisions about where traffic is sent (the SDN controller, or control plane) from the underlying systems that forward traffic to the selected destination (the data plane). SDN was commonly associated with the OpenFlow protocol (for remote communication with network plane elements for the purpose of determining the path of network packets across network switches) since the latter's emergence in 2011. Since 2012, however, many companies have moved away from OpenFlow, and have embraced different techniques. Software-defined networking (SDN) is an architecture purporting to be dynamic, manageable, cost-effective, and adaptable, seeking to be suitable for the high-bandwidth, dynamic nature of today's applications. SDN architectures decouple network control and forwarding functions, enabling network control to become directly programmable and the underlying infrastructure to be abstracted from applications and network services. The OpenFlow protocol can be used in SDN technologies. The SDN architecture is:

1. **Directly programmable:** Network control is directly programmable because it is decoupled from forwarding functions.
2. **Agile:** Abstracting control from forwarding lets administrators dynamically adjust network-wide traffic flow to meet changing needs.
3. **Centrally managed:** Network intelligence is (logically) centralized in software-based SDN controllers that maintain a global view of the network, which appears to applications and policy engines as a single, logical switch.
4. **Programmatically configured:** SDN lets network managers configure, manage, secure, and optimize network resources very quickly via dynamic, automated SDN programs, which they can write themselves because the programs do not depend on proprietary software.

5. **Open standards-based and vendor-neutral:** When implemented through open standards, SDN simplifies network design and operation because instructions are provided by SDN controllers instead of multiple, vendor-specific devices and protocols.

Need for SDN:-

1. Changing traffic patterns-

Within the enterprise data center, traffic patterns have changed significantly. In contrast to client-server applications where the bulk of the communication occurs between one client and one server, today's applications access different databases and servers, creating a flurry of "east-west" machine-to-machine traffic before returning data to the end user device in the classic "north-south" traffic pattern. At the same time, users are changing network traffic patterns as they push for access to corporate content and applications from any type of device (including their own), connecting from anywhere, at any time. Finally, many enterprise data centers managers are contemplating a utility computing model, which might include a private cloud, public cloud, or some mix of both, resulting in additional traffic across the wide area network.

2 The "consumerization of IT"

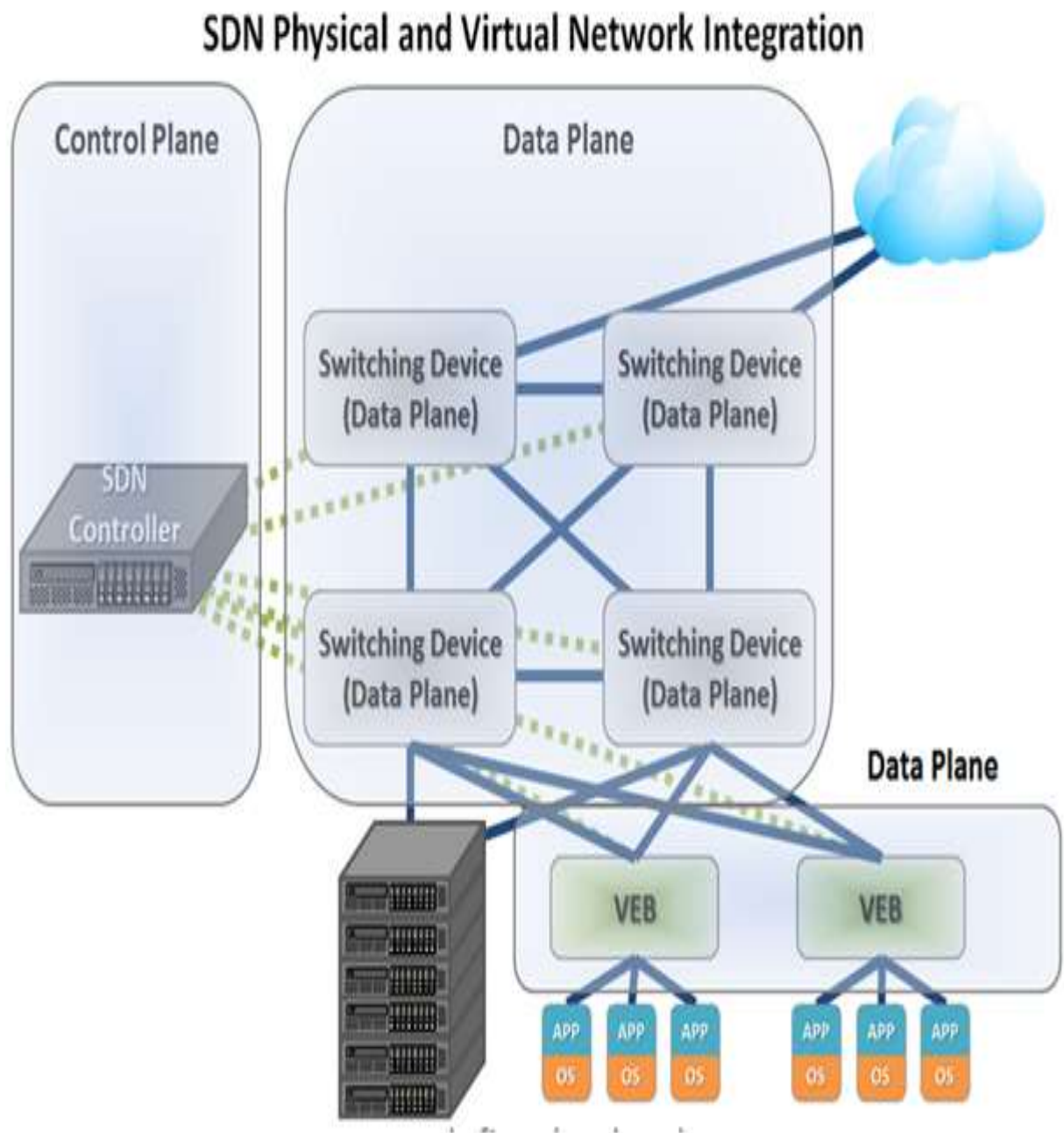
Users are increasingly employing mobile personal devices such as smartphones, tablets, and notebooks to access the corporate network. IT is under pressure to accommodate these personal devices in a fine-grained manner while protecting corporate data and intellectual property and meeting compliance mandates.

3 The rise of cloud services

Enterprises have enthusiastically embraced both public and private cloud services, resulting in unprecedented growth of these services. Enterprise business units now want the agility to access applications, infrastructure, and other IT resources on demand and à la carte. To add to the complexity, IT's planning for cloud services must be done in an environment of increased security, compliance, and auditing requirements, along with business reorganizations, consolidations, and mergers that can change assumptions overnight. Providing self-service provisioning, whether in a private or public cloud, requires elastic scaling of computing, storage, and network resources, ideally from a common viewpoint and with a common suite of tools.

4 “Big data” means more bandwidth

Handling today's "big data" or mega datasets requires massive parallel processing on thousands of servers, all of which need direct connections to each other. The rise of mega datasets is fueling a constant demand for additional network capacity in the data center. Operators of hyperscale data center networks face the daunting task of scaling the network to previously unimaginable size, maintaining any-to-any connectivity without going broke.



2.1.2 Firewall:-

A firewall is a network security device that monitors incoming and outgoing network traffic and decides whether to allow or block specific traffic based on a defined set of security rules. Firewalls have been a first line of defense in network security for over 25 years. They establish a barrier between secured and controlled internal networks that can be trusted and entrusted outside networks, such as the Internet. A firewall can be hardware, software, or both.

Types of firewalls:-

1.Proxy firewall:

An early type of firewall device, a proxy firewall serves as the gateway from one network to another for a specific application. Proxy servers can provide additional functionality such as content caching and security by preventing direct connections from outside the network. However, this also may impact throughput capabilities and the applications they can support.

2.Stateful inspection firewall:

Now thought of as a “traditional” firewall, a stateful inspection firewall allows or blocks traffic based on state, port, and protocol. It monitors all activity from the opening of a connection until it is closed. Filtering decisions are made based on both administrator-defined rules as well as context, which refers to using information from previous connections and packets belonging to the same connection.

3.Unified threat management (UTM) firewall:

A UTM device typically combines, in a loosely coupled way, the functions of a stateful inspection firewall with intrusion prevention and antivirus. It may also include additional services and often cloud management. UTMs focus on simplicity and ease of use.

4.Next-generation firewall (NGFW):

Firewalls have evolved beyond simple packet filtering and stateful inspection. Most companies are deploying next-generation firewalls to block modern threats such as advanced malware and application-layer attacks.

According to Gartner, Inc.’s definition, a next-generation firewall must include:

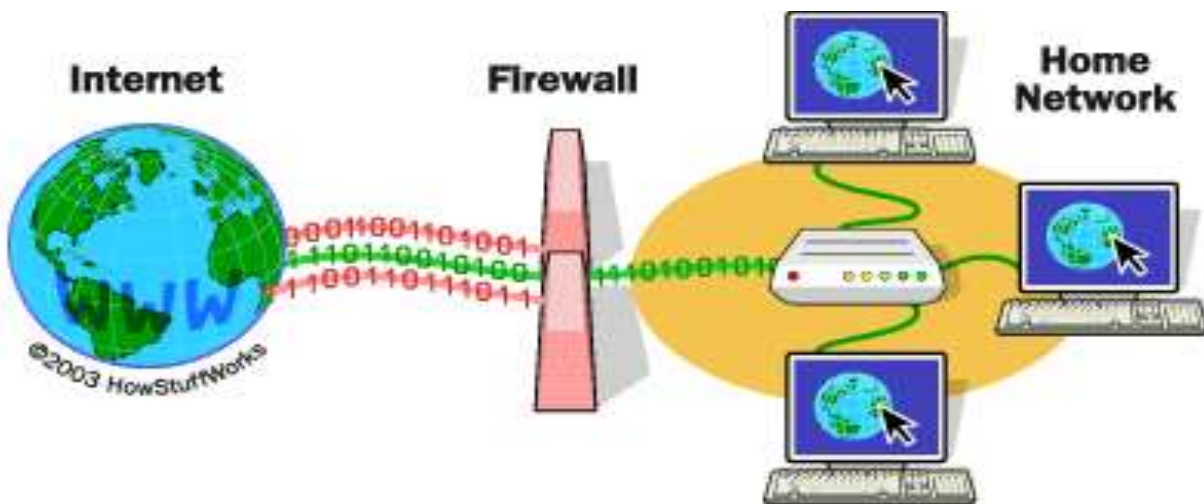
- Standard firewall capabilities like stateful inspection.
- Integrated intrusion prevention.
- Application awareness and control to see and block risky apps.
- Upgrade paths to include future information feeds.
- Techniques to address evolving security threats.

While these capabilities are increasingly becoming the standard for most companies, NGFWs can do more.

5. Threat-focused NGFW:

These firewalls include all the capabilities of a traditional NGFW and also provide advanced threat detection and remediation. With a threat-focused NGFW you can:

- Know which assets are most at risk with complete context awareness.
- Quickly react to attacks with intelligent security automation that sets policies and hardens your defenses dynamically.
- Better detect evasive or suspicious activity with network and endpoint event correlation.
- Greatly decrease the time from detection to cleanup with retrospective security that continuously monitors for suspicious activity and behavior even after initial inspection.
- Ease administration and reduce complexity with unified policies that protect across the entire attack continuum.



6. Packet Filtering:

Also known as static filtering, it is a firewall technique used to control network access by monitoring outgoing and incoming packets and allowing them to pass or halt based on the source-destination, protocols and ports. It is pretty effective against computers outside the LAN. It is considered standard and cost effective means of security

2.2 Patent Search:

The following is the list of IEEE papers that were downloaded from “www.ieeexplore.ieee.org”

1. Building a Firewall over Software-Defined Network Controller:

Many have recognized the need to restructure the current internet work into a much more dynamic networking environment. It is difficult for today's inflexible infrastructure to cope with the fast changing demands of the users. As a result, Software-Defined Network (SDN) was introduced around 2005

to transform today's network to have centralized management, rapid innovation, and programmability by decoupling the control and data planes. This study focuses on developing a firewall application that runs over an OpenFlow-based SDN controller to show that most of the firewall functionalities are able to be built on software, without the aid of a dedicated hardware. Among many OpenFlow controllers that already exist for the public, we have chosen POX written in Python for the experiment; and to create the SDN network topology, we have used VirtualBox and Mininet. In this study, we cover the implementation detail of our firewall application, as well as the experimentation result. The overall functioning of the project was understood from this paper. The paper incorporates the understanding of SDN followed by clear view of the Firewall that can be built on the OpenFlow controller.

2. An OpenFlow based Prototype of SDN-oriented Stateful Hardware Firewalls:

This paper describes an Open Flow-based prototype of a SDN-oriented stateful hardware firewall. The prototype of a SDN-oriented stateful hardware firewall includes an Open Flow-enabled switch and a firewall controller. The security rules are specified in the flow table in both the Open Flow-enabled switch and the firewall controller. The firewall controller is in charge of making control decisions on regulating the unidentified traffic flows. A communication channel is needed between a firewall controller and an Open Flow enabled switch. Through this channel, a switch sends to the controller with the information of unidentified flows, and the controller sends to the switch with the control decisions. Constraining this communication overhead is important to the applicability of the prototype because a high communication overhead could disturb the performance evaluation on the operation of a SDN-oriented stateful hardware firewall. This paper describes an OpenFlow-based prototype of a SDN-oriented stateful hardware firewall. The prototype of a SDN-oriented stateful hardware firewall includes an OpenFlow-enabled switch and a firewall controller. The security rules are specified in the flow table in both the OpenFlow-enabled switch and the firewall controller. The firewall controller is in charge of making control decisions on regulating the unidentified traffic flows.

3. Development of Distributed Firewall using SDN Technology:

Software defined networking (SDN) presents a new network architecture that separates the control logic of a network from its physical infrastructure. This allows for easy programmable networks without having to manually configure every network device individually. However, there are not much studies on security applications for SDN based networks. Hence, the goal of this work is to explore security possibilities by focusing on the development of a firewall prototype that maximizes the advantages of SDN. By building around the features of OpenFlow, an open SDN standard, a distributed flow-based firewall prototype was developed and tested on a simulated network through Mininet. The prototype was tested to show full functionality through ping tests in a distributed configuration without causing any delays in terms of latency. This paper made our understanding regarding SDN clear. Software defined networking (SDN) presents a new network architecture that separates the control logic of a network from its physical infrastructure. This allows for easy programmable networks without having to manually configure every network device individually. However, there are not much studies on security applications for SDN based networks.

Software defined networking (SDN) presents a new network architecture that separates the control logic of a network from its physical infrastructure. This allows for easy programmable networks without having to manually configure every network device individually. However, there are not much studies on security applications for SDN based networks. Hence, the goal of this work is to explore security possibilities by focusing on the development of a firewall prototype that maximizes the advantages of SDN. By building around the features of OpenFlow, an open SDN standard, a distributed flow-based firewall prototype was developed and tested on a simulated network through Mininet.

3. Requirement Gathering

3.1 Functional Requirements:

- The ability to detect malicious packets.

The firewall should recognise the packets coming from known malicious users and drop them purposely.

- The project should be able to drop the unnecessary packets and forward the needed ones.

Firewall should recognise the requested and intended packets for the users of the system and then further act according to this.

- The project should be able to withstand any security attack that may or may not be used against it by the attackers.

Firewall should withstand some basic attacks and shouldn't allow attacker to compromise its working.

- To reduce or eliminate the occurrence of unwanted network communications while allowing all legitimate communication to flow freely:-

Firewall should know the frequently used ip addresses and hence should allow legitimate users to communicate.

- The support of the programmability of network resources to mitigate network attacks:-

Since the firewall is built on programmable SDN technology, it should be able to incorporate the changes made and be able to cover the loop-holes after the changes are done in the network

- The support of an application interface allowing the management of access control policies in an autonomous and prompt manner:- Firewall should work in accordance with the application or graphical interface provided.

- The support of a resource-control interface for control of network resources to mitigate network attacks:- Firewall should be able to command the interface to work according to the design of firewall.

- The support of logically centralized control of network resources to mitigate network attacks.

The firewall built should be able to follow the ODL controller setup and act properly in respect to its interfaces and perform its functioning without causing any hindrances in the functioning of SDN.

3.2 Non-Functional Requirements:

- Performance should be optimum.

The firewalls functioning should be optimum with minimum wastage of resources and provide perfection in providing protection to the host

- The privileges to handle the firewall should be proper.

The personal for the inspection of the firewall should have knowledge of his work and should be able to properly allocate resources where-ever required

- The SDN n/w created should be tested properly.

This is important as the whole project is being built on the SDN technology. Hence any problems in the functioning of it will lead to errors in the further implementation of the project.

3.3 Hardware Requirements:

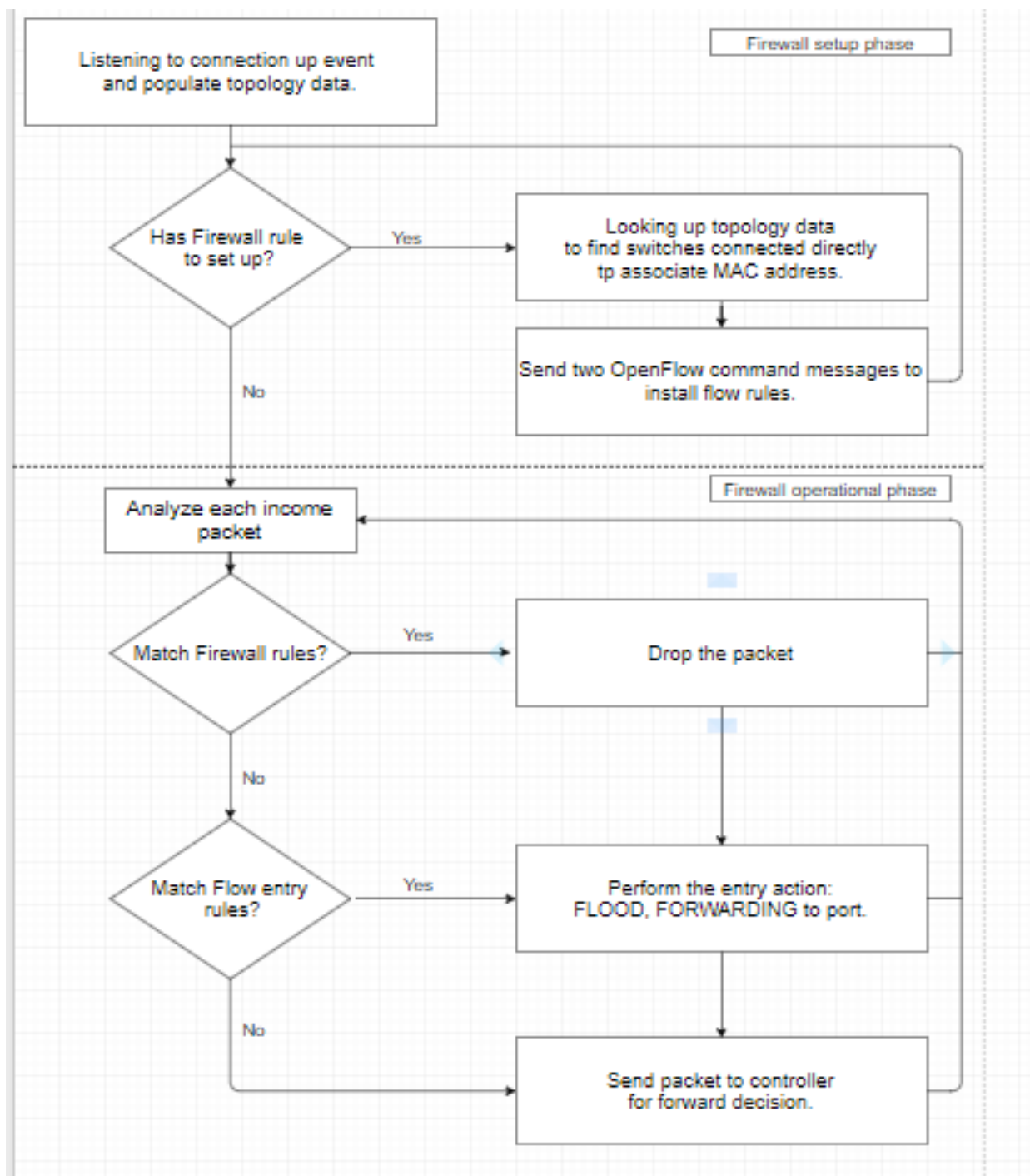
1. OS: Kali Linux/Windows/ Ubuntu 14.04
2. RAM: 10 GB
3. ROM: 500GB
4. System type: 64-bit Operating System with x64 based processor.

3.4 Software Requirements:

1. Mininet: Version 2.2.1
2. OpenDayLight based SDN Test Bed: Carbon
3. Kali Linux or Ubuntu 14.04 LTS (kernel 3.13)
4. Java 8.
5. VMware virtual Machine

4. Proposed Design

4.1 Block Diagram of the system



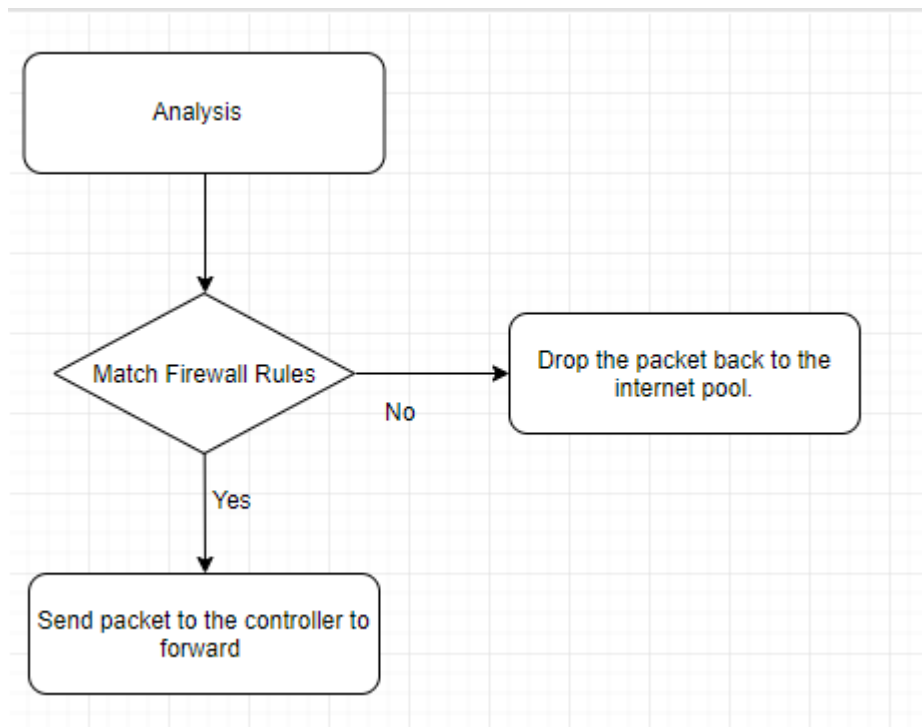
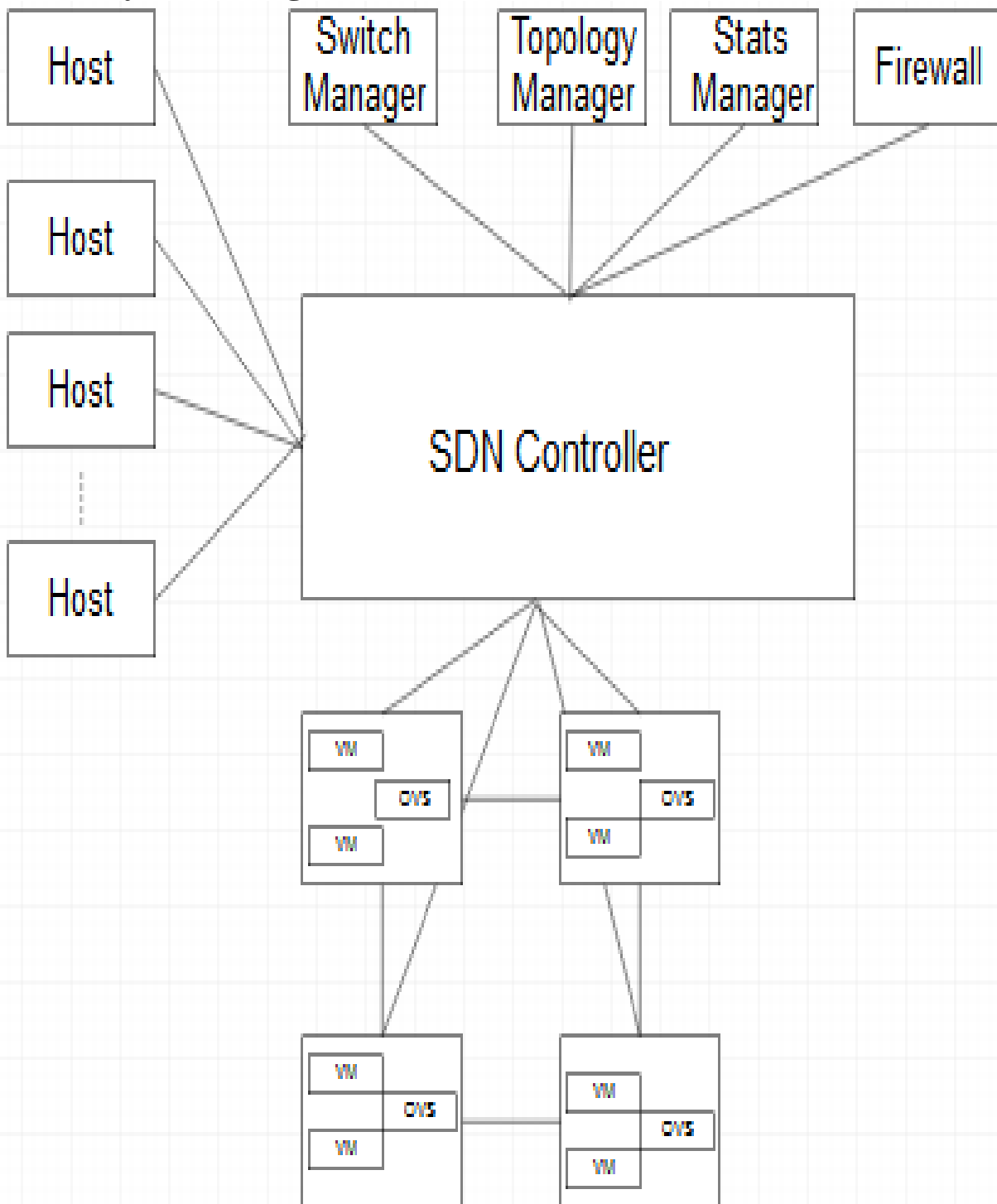


Fig 3: Block diagram

In the block diagram, client interacts with the system via a node the he is using. The functioning of the system is simple. Basically all the nodes are connected to SDN controller whereas the Firewall is placed on Controller itself. All the packets arriving are judged at the controller level and then passed ahead to the respective node

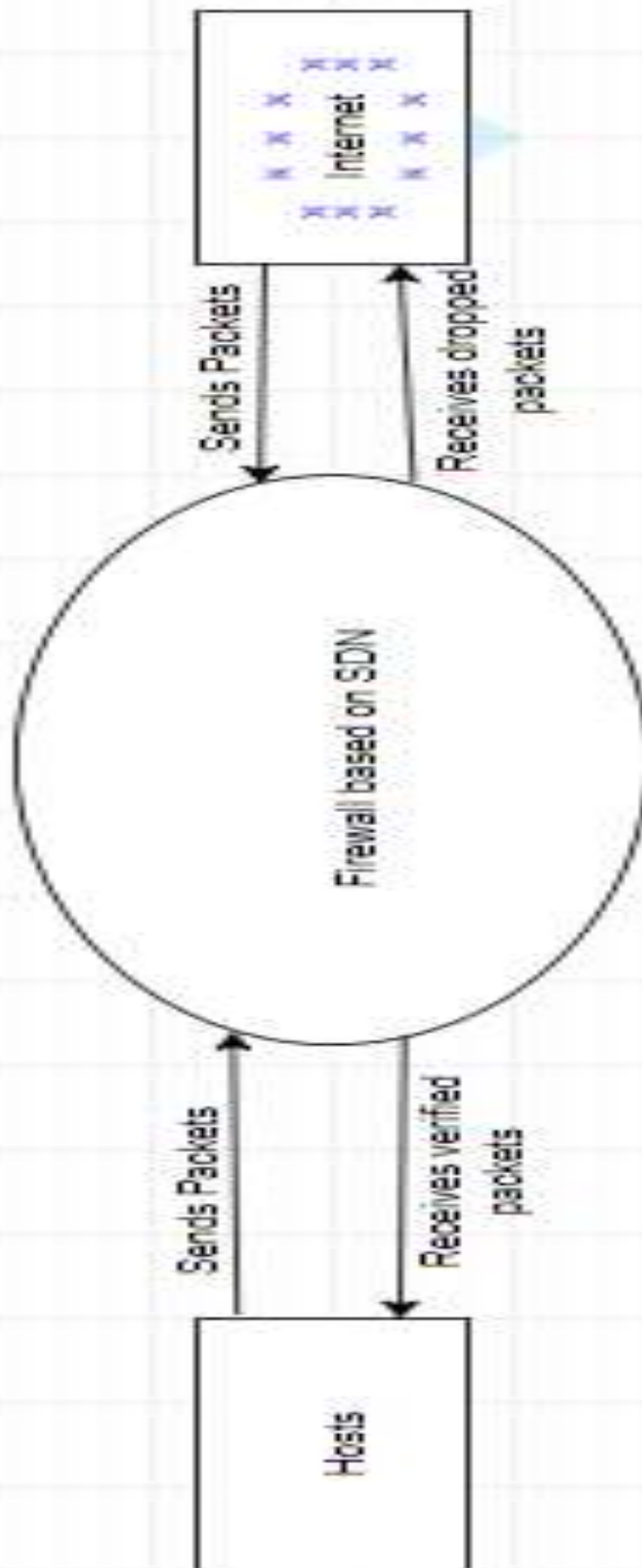
Proposed Design:

1. System Design:

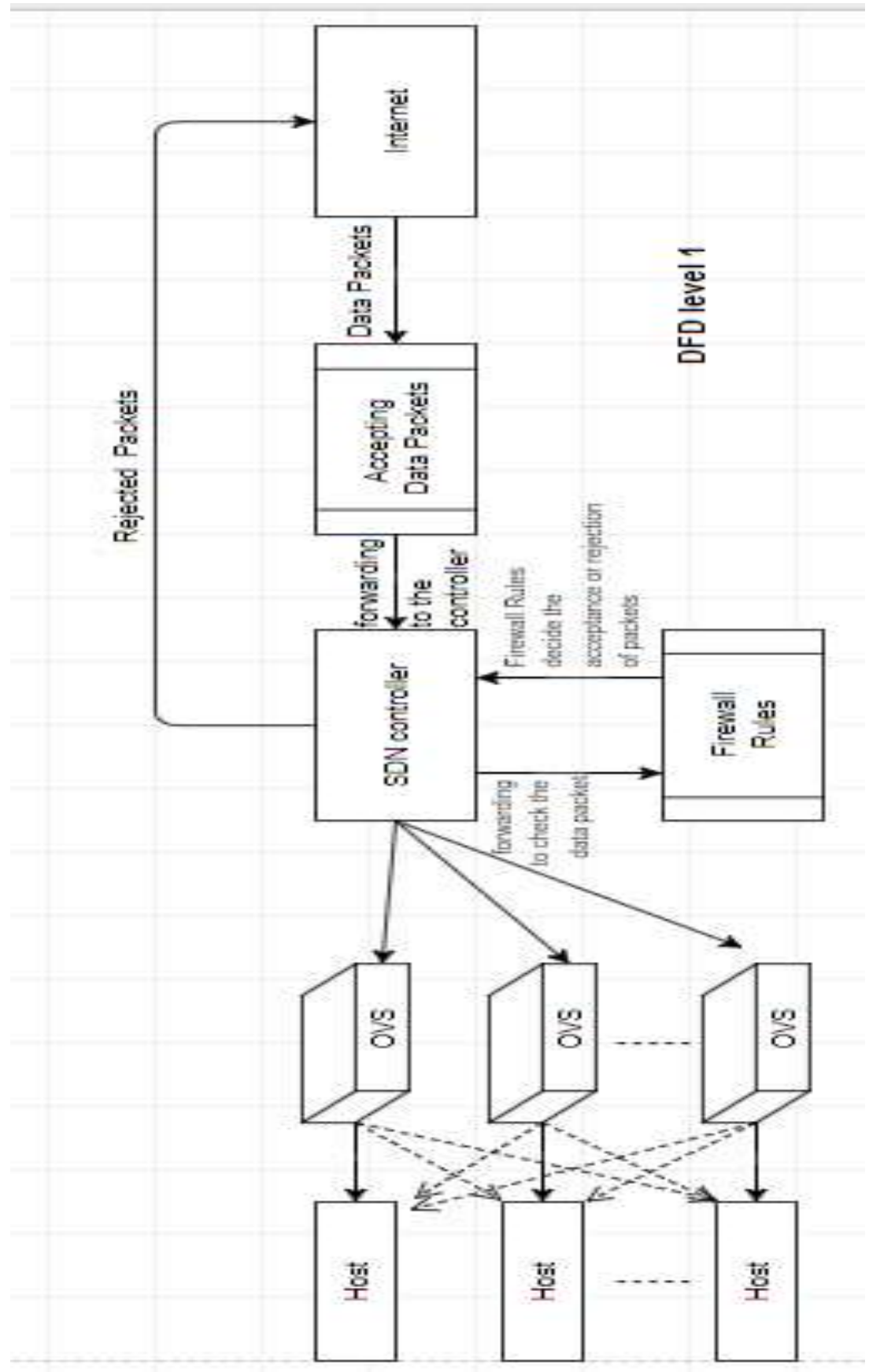


Detailed Design:
DFD:- Level 0:

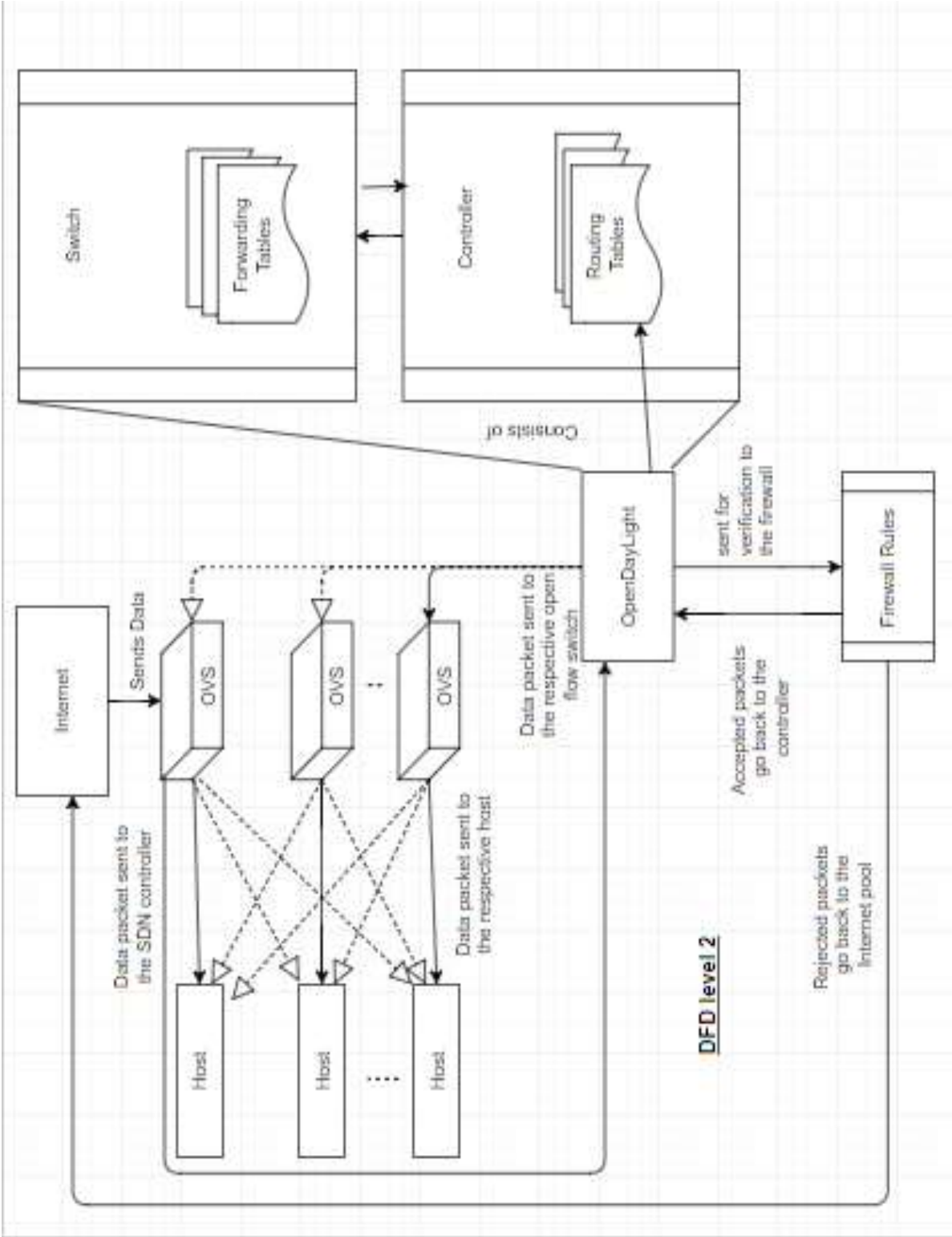
DFD level 0



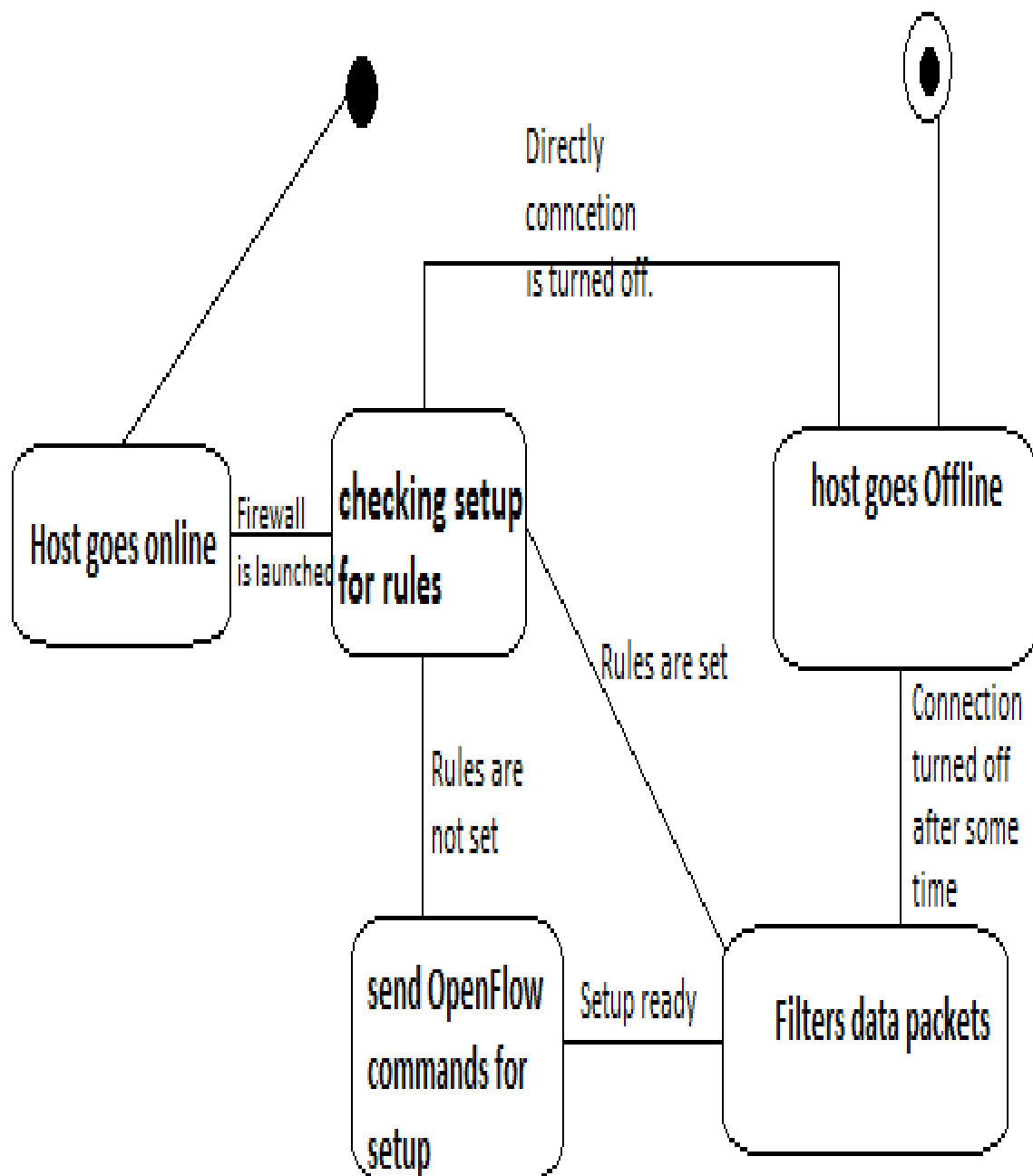
Level 1:



Level 2:



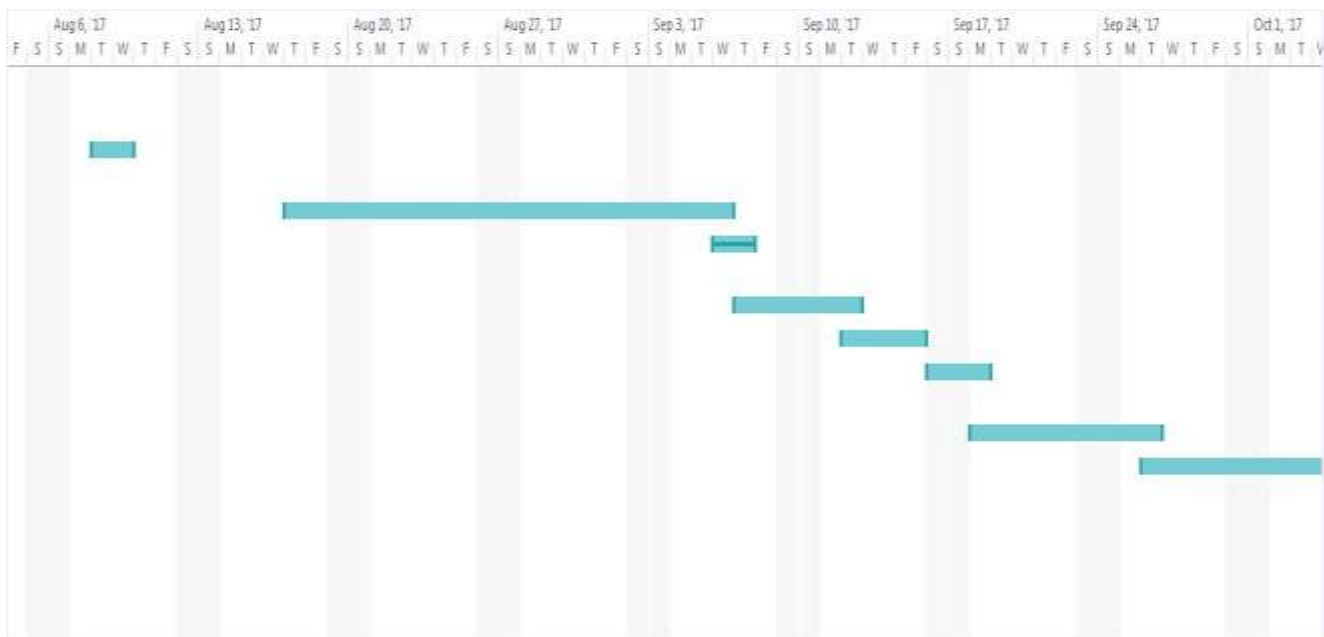
State Diagram:-



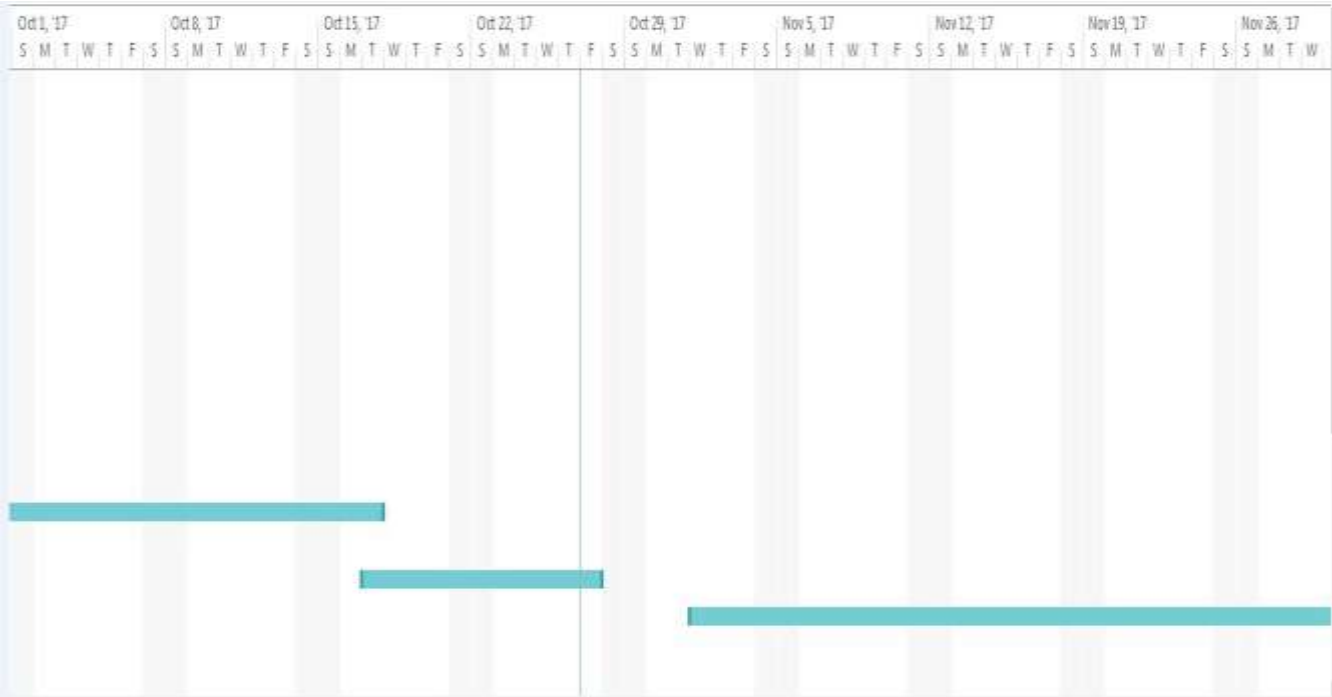
Scheduling:

	✖?								
✓	✖	Forming group	2 days	Tue 8/1/17	Wed 8/2/17				
	✖	Deciding the Domain	2 days	Tue 8/8/17	Wed 8/9/17				
	✖	Deciding the Topic	15 days	Thu 8/17/17	Wed 9/6/17				
✓	✖	Approval from the Mentor	2 days	Wed 9/6/17	Thu 9/7/17				
	✖	Finding IEEE papers	4 days	Thu 9/7/17	Tue 9/12/17				
	✖	Synopsis	4 days	Tue 9/12/17	Fri 9/15/17				
	✖	Corrections in Synopsis	2 days	Sat 9/16/17	Mon 9/18/17				
	✖	Presentation	7 days	Mon 9/18/17	Tue 9/26/17				
	✖	Implementation Phase 1	16 days	Tue 9/26/17	Tue 10/17/17				
	✖	Review Report	9 days	Tue 10/17/17	Fri 10/27/17				
	✖	Implementaion Phase 2	90 days	Wed 11/1/17	Tue 3/6/18				
	✖	Review Report 2	9 days	Tue 3/6/18	Fri 3/16/18				
	✖	Demonstration to the Mentor	30 days	Fri 3/16/18	Thu 4/26/18				
	✖	Demonstration to HOD	15 days	Thu 4/26/18	Wed 5/16/18				
	✖	Grading via	10 days	Sun 5/20/18	Thu 5/31/18				

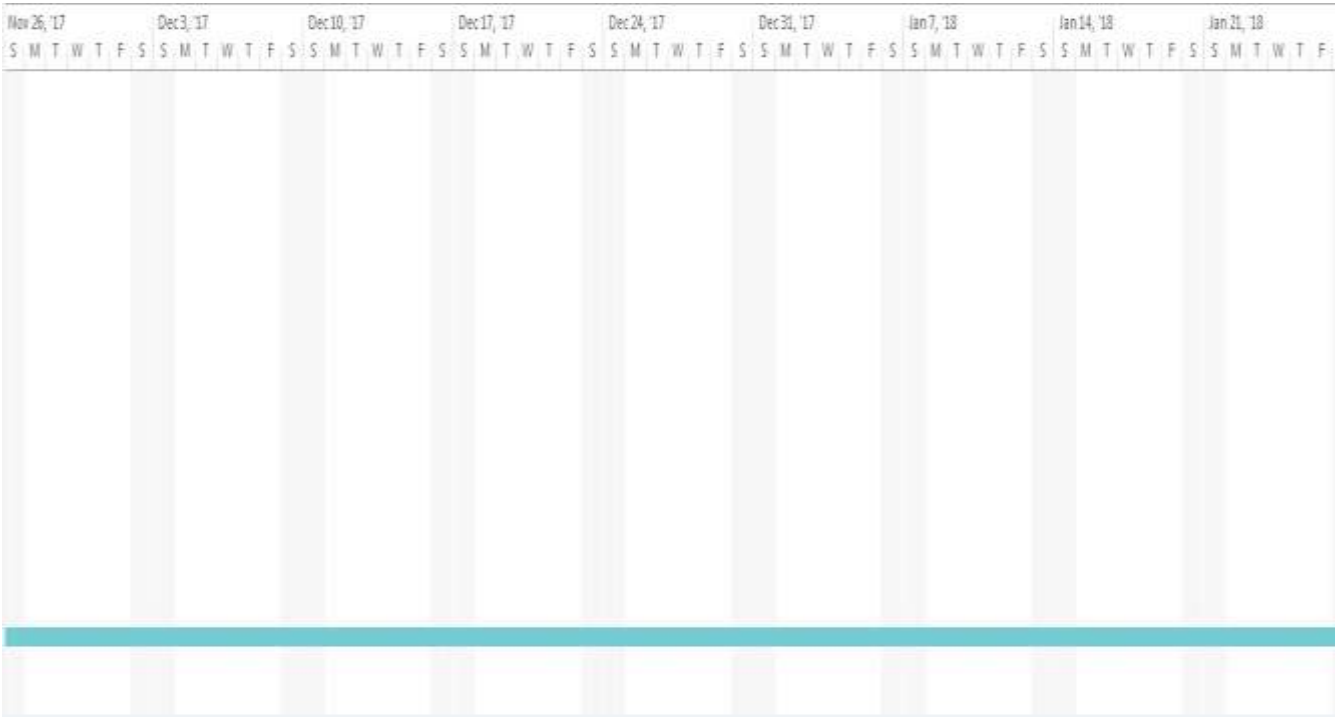
Gantt Chart 1



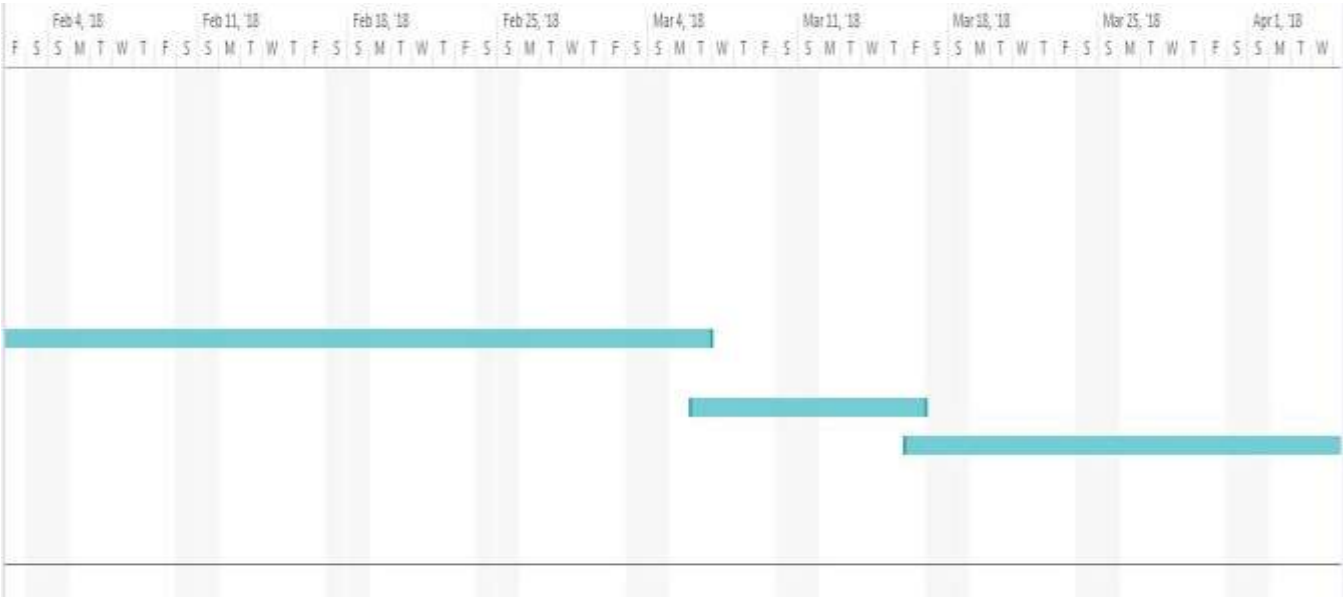
Gantt Chart 2



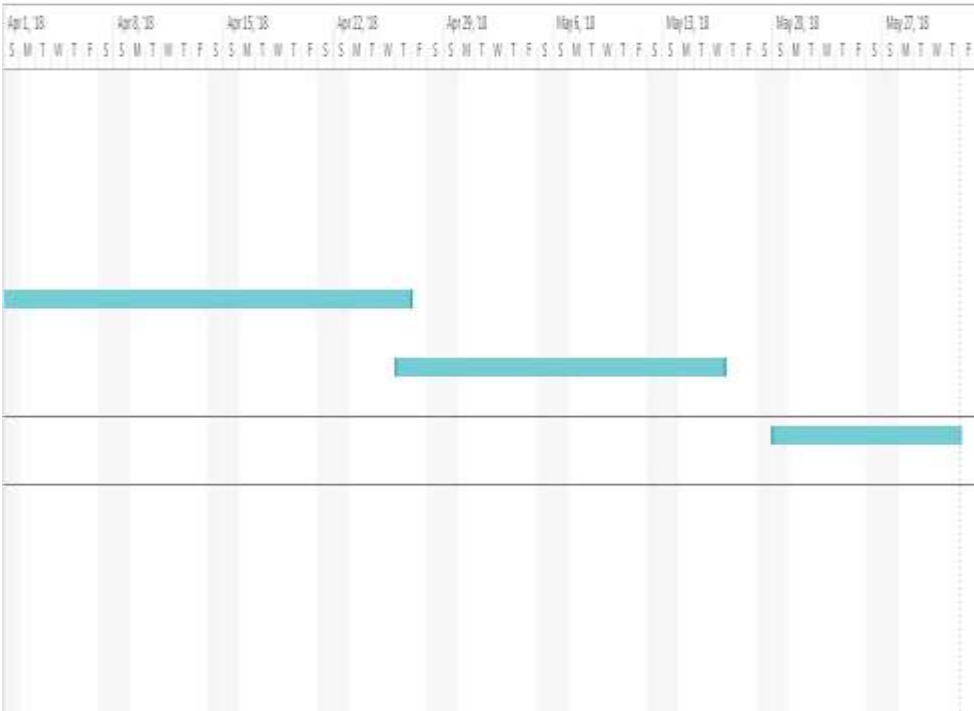
Gantt Chart 3



Gantt Chart 4



Gantt Chart 5



5. Implementation Steps:

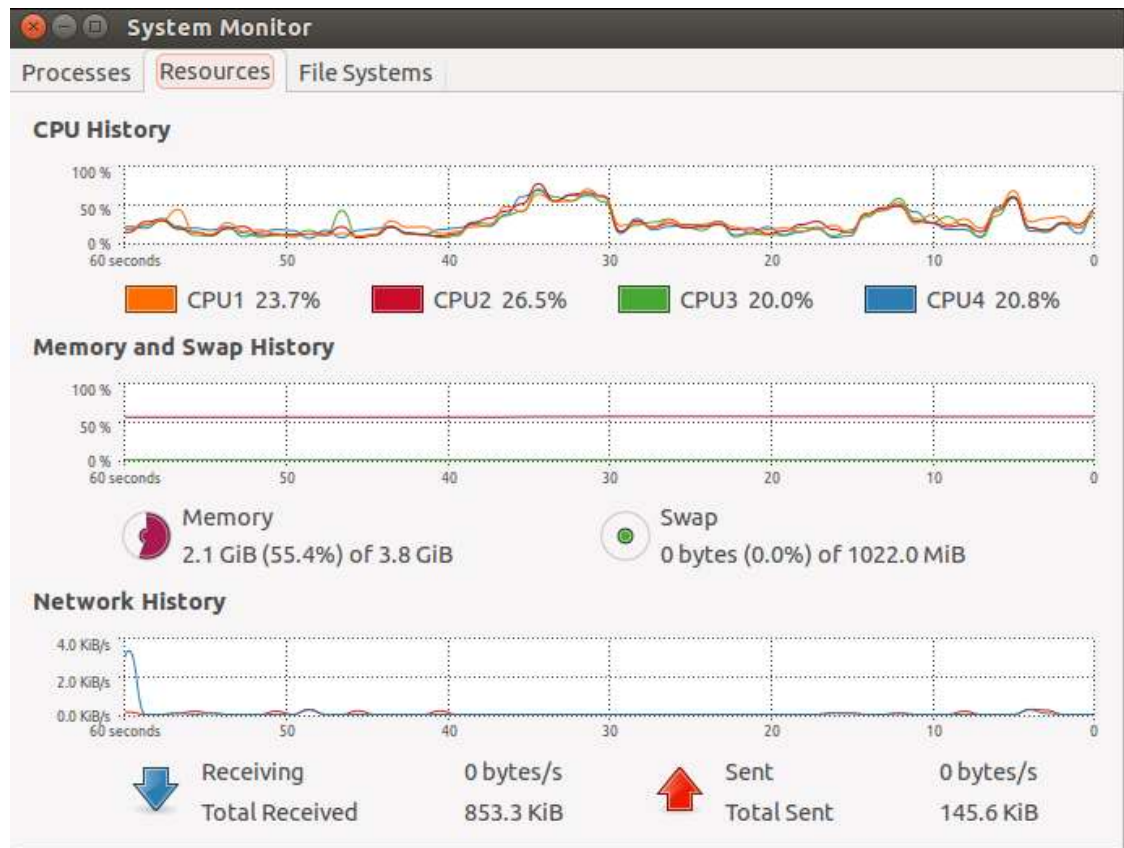
- Initialising the SDN controller by activating all the open V switch by using the command “service openvswitch-switch start”
- Create the Topology as :-“mn -topo=2,fanout=5 -controller=remote”
- This Command needs to be written inside Mini-net and use this Xterm to execute below commands: “xterm c0”
- Change to Firewall directory and execute :-“python firewall.pyw”

6. Testing:

After creation of Firewall, Syn Flood and Null Scan attacks were performed on the firewall and following results were obtained

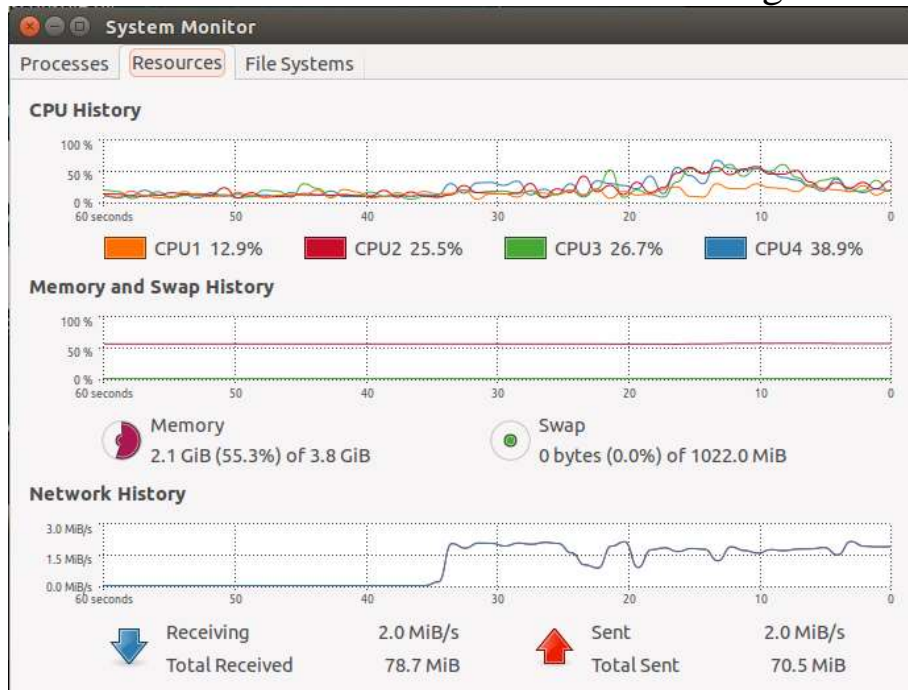
1:

Initial Network Statistics for Controller.



2:

Attack buffer of controller while attack being carried out.



3:

hping3 command used to flood the victim.

```
root@kali:~# hping3 -S --flood 192.168.142.128
HPING 192.168.142.128 (eth0 192.168.142.128): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
```

4:

23 lakh packets flooded to the target.

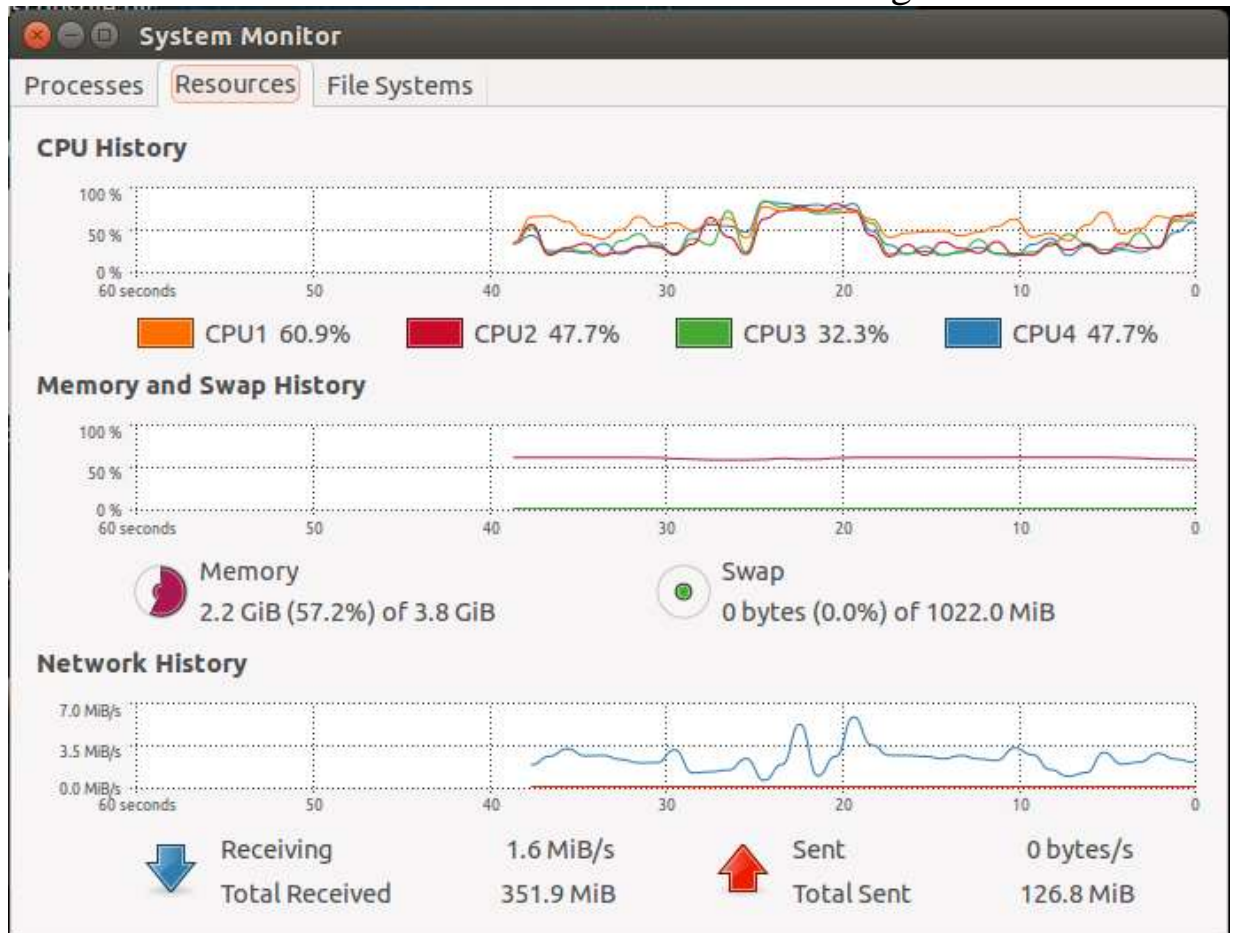
```
root@kali:~# hping3 -S --flood 192.168.142.128
HPING 192.168.142.128 (eth0 192.168.142.128): S
hping in flood mode, no replies will be shown
^C
--- 192.168.142.128 hping statistic ---
2300216 packets transmitted, 0 packets received,
```

5:

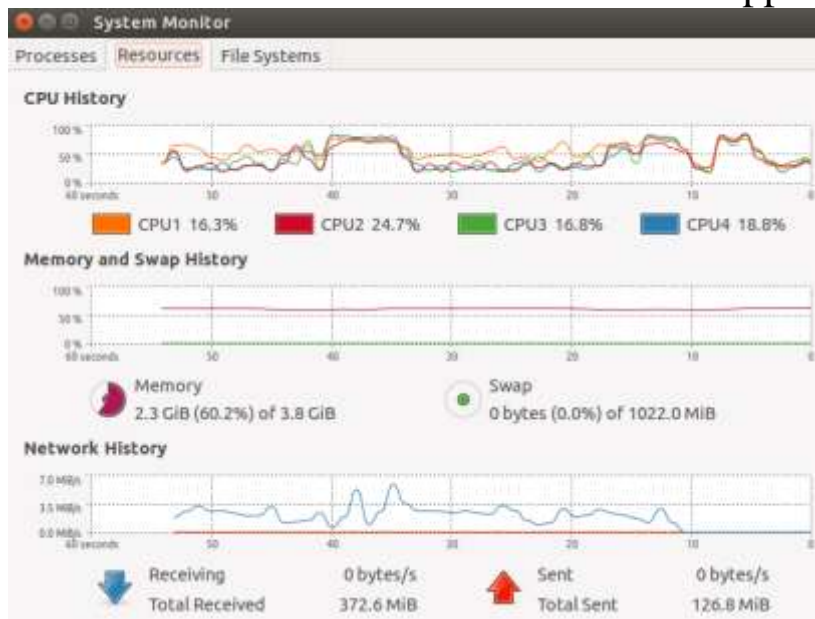
Firewall rule to mitigate Syn Flood.

INPUT					
Source Address	Destination Address	Protocol	Action	Miscellaneous	
anywhere	anywhere	tcp	DROP	NEW	

6: Network Statistics for controller after attack being blocked.



7: Network Statistics after attack has been stopped



8:

Null scan performed on controller prior to applying firewall rule

```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# nmap -sN 192.168.142.128  
  
Starting Nmap 7.40 ( https://nmap.org ) at 2018-04-23 21:04 UTC  
Nmap scan report for 192.168.142.128  
Host is up (0.000094s latency).  
Not shown: 995 closed ports  
PORT      STATE      SERVICE  
22/tcp    open|filtered ssh  
1099/tcp  open|filtered rmiregistry  
6666/tcp  open|filtered irc  
8080/tcp  open|filtered http-proxy  
8181/tcp  open|filtered intermapper  
MAC Address: 00:0C:29:63:27:E2 (VMware)  
  
Nmap done: 1 IP address (1 host up) scanned in 14.56 seconds  
root@kali:~#
```

9: Firewall rule to prevent Null Scan

INPUT				
Source Address	Destination Address	Protocol	Action	Miscellaneous
anywhere	anywhere	tcp	DROP	flags:FIN,SYN,RST,PSH,ACK,URG,NONE

10:

Null scan performed on controller after applying firewall rule

```
root@kali: ~  
File Edit View Search Terminal Help  
  
Starting Nmap 7.40 ( https://nmap.org ) at 2018-04-23 21:04 UTC  
Nmap scan report for 192.168.142.128  
Host is up (0.000094s latency).  
Not shown: 995 closed ports  
PORT      STATE      SERVICE  
22/tcp    open|filtered ssh  
1099/tcp  open|filtered rmiregistry  
6666/tcp  open|filtered irc  
8080/tcp  open|filtered http-proxy  
8181/tcp  open|filtered intermapper  
MAC Address: 00:0C:29:63:27:E2 (VMware)  
  
Nmap done: 1 IP address (1 host up) scanned in 14.56 seconds  
root@kali:~# nmap -sN 192.168.142.128  
  
Starting Nmap 7.40 ( https://nmap.org ) at 2018-04-23 21:06 UTC  
Nmap scan report for 192.168.142.128  
Host is up (0.00046s latency).  
All 1000 scanned ports on 192.168.142.128 are open|filtered  
MAC Address: 00:0C:29:63:27:E2 (VMware)  
  
Nmap done: 1 IP address (1 host up) scanned in 34.36 seconds  
root@kali:~#
```

7. Results and Discussions:

1. Test bed was designed for the first review of the project for a basic working of a simple network.
2. A complex network topology will be designed imitating a network similar to that of an enterprise or most likely our college.
3. Attempts to control OVSwitch using ODL's web application based controller.
4. Different approaches for building a firewall were studied.
5. Different approaches for breaching the firewall were studied.
6. Exploiting design vulnerabilities in firewall to breach it.
7. Patching vulnerability in order to avoid compromisation of the firewall and hence the network and users.

8. Conclusions:

Though Firewall has its own limitations, it's a healthy and safe procedure for the betterment of networking in various countries round the globe. Not only our security is increased, but its various problems are solved due to it thus resulting in less preaching and disloyalty. It has a bright future and economic scope thus not only it will generate employment but also it ll be a generator of money as well. Irrespective of the fact that with new technology new threats are released but dealing with that threats and breaking it down to the very core thus ensuring integrity is the scope of this project

9. Plans of the Entire Semester:

1. The first task performed by us was gaining insight of SDN and understanding it to its entirety.
2. Different IEEE papers that were mentioned above were studied for understanding purposes.
3. Start in learning about the syntax and the concepts of Python language for our whole firewall will be built in that.
4. Designed the GUI for Firewall and Test bed for setting up SDN controller installed.
5. Simulation of the firewall built for checking for any errors that might be present accompanied with the testing of SDN technology.
6. Embedding the firewall upon the SDN controller and making the interface for them to work hand in hand
7. Exploring the vulnerabilities of fully developed firewall and develop an attack on purposely exploiting those vulnerabilities and improve the functioning of the same.
8. Performed Network Attacks on the System to check the functionalities after integration.
9. Generation of the final report.

10. References:

1. Michelle Suh, Sae Hyong Park, Byungjoon Lee, Sunhee Yang “Building Firewall over the Software-Defined Network Controller” SDN Research Section, ETRI (Electronics and Telecommunications Research Institute), Korea
2. Jake Collings, Jun Liu “An OpenFlow-based Prototype of SDN-Oriented Stateful Hardware Firewalls” 2014, IEEE 22nd International Conference on Network Protocols
3. Justin Gregory V. Pena and William Emmanuel Yu “Development of a Distributed Firewall Using Software Defined Networking Technology” Department of Information Systems and Computer Science Ateneo De Manila University Loyola Heights, Quezon City, Philippines
4. Thuy Vinh Tran, Heejune Ahn “A Network Topology-aware Selectively Distributed Firewall Control in SDN” Department of Electrical and Information Engineering Seoul National University of Science and Technology Seoul, Republic of Korea
5. <https://turbonomic.com/blog/on-technology/sdn-software-defined-networking-primer-and-why-we-need-sdn/>
6. <https://www.pressreader.com/india/opensource-for-you/20160610/282157880544330>
7. <https://en.wikipedia.org/wiki/SDN>
8. <https://www.slideshare.net/lz1dsb/why-sdn>
9. <http://opensourceforu.com/2016/07/implementing-a-software-defined-network-sdn-based-firewall/>
10. <http://sdnhub.org/resources/useful-mininet-setups/>
11. <https://www.techopedia.com/definition/4038/packet-filtering>
12. <http://securityworld.worldiswelcome.com/packet-filtering-firewall-an-introduction>
13. <https://www.sdxcentral.com/sdn/definitions/sdn-controllers/opendaylight-controller/>
14. http://www.necoma-project.eu/m/filer_public/18/30/1830b40c-a2e2-4a0b-b83e-af8b2d846e61/imt_zhang_comnet2015.pdf
15. <https://www.ietf.org/proceedings/91/slides/slides-91-i2nsf-0.pdf>

11. Project Reviews 1. Review 1:

Project Evaluation Sheet 2017 - 18
GROUP NO.: 42

Project: Friction based on concept of SH
 Members: John Bojaj, Yagan Ramesh, Vireen Madhusovan

Engineering Concepts & Knowledge	Interpretation of Problem & Analysis	Design / Prototype	Interpretation of Data & Dataset	Modern Tool Usage	Societal Benefit, Safety Consideration	Environmentally Friendly	Ethics	Team work	Presentation Skills	Applied Engg & Mgmt principles	Life-long learning	Professional Skills	Team work	Time value	Total Marks
(1)	(1)	(1)	(1)	(1)	(2)	(2)	(2)	(2)	(1)	(3)	(1)	(1)	(1)	(4)	(50)
3	3	2	1	4	2	2	2	1	1	1	1	2	2	2	27

No Connectivity to the, 6/04/18

Reviewed By: [Signature]
 Name & Signature: [Signature]
 Reviewer: [Signature]

2. Review 2:

Instructor/ Faculty:

Project Evaluation Sheet 2017 - 18

Class: IT17 A/144

Group No: 42

Title of Project: SDN Based Firewall

Group Members: Yash Raj, Yogesh Kumar, Nisha Madhavan

Review of Project Stage 1	Engineering Concepts & Theoretical	Interpretation of Problem & Analysis	Design / Prototype	Interpretation of Data & Dataset	Module / Tool / Usage	Software, Hardware, Safety, Consideration	Environment / Hardware	Finals	Team work	Presentation on Skills	Applied Engineering principles	Life-long learning	Problem Solving Skills	Innovative Approach	Total Marks
3	3	3	3	2	3	1	1	1	1	1	2	2	2	3	27

Conceptually wrong & functionality implemented.

Rishabh Joshi
Name & Signature (Reviewer)

Review of Project Stage 1	Engineering Concepts & Knowledge	Interpretation of Problem & Analysis	Design / Prototype	Interpretation of Data & Dataset	Module / Tool / Usage	Software, Hardware, Safety, Consideration	Environment / Hardware	Finals	Team work	Presentation on Skills	Applied Engineering principles	Life-long learning	Problem Solving Skills	Innovative Approach	Total Marks
3	3	3	3	2	3	1	1	1	1	1	2	2	2	3	24

Date: 15th March, 2018

Rishi Sharma
Name & Signature (Reviewer)

12. Appendix:

12.1 List of Figures

Figure Number	Heading
<u>1</u>	Block Diagram 1
<u>2</u>	Block Diagram 2
<u>3</u>	System Design
<u>4</u>	DFD Level 0
<u>5</u>	DFD Level 1
<u>6</u>	DFD Level 2
<u>7</u>	State Diagram
<u>8</u>	Project Scheduling
<u>9</u>	Gantt Chart 1
<u>10</u>	Gantt Chart 2
<u>11</u>	Gantt Chart 3
<u>12</u>	Gantt Chart 4
<u>13</u>	Gantt Chart 5

FIREWALL BASED ON THE CONCEPT OF SDN

Yash Bajaj., Yogesh Rohra., Viren Wadhwani and Arthi CI

Computer Department, Vivekanand Institute of Technology

ARTICLE INFO

Article History:

Received 7th November, 2017

Received in revised form 13th

December, 2017

Accepted 3rd January, 2018

Published online 28th February, 2018

Key words:

Open-Flow, SDN, Firewall, Mini-net, Open-Day-Light.

ABSTRACT

In today's digital world many people have utterly felt the need to restructure the current internet work into one which is much more like a dynamic networking environment. Software Defined Networking finds a solution to these problems. Software Defined Networking is an exciting yet beautiful technology that enables innovation and flexibility in designing and managing networks present currently. But on the other hand it also introduces new security threats and issues beforehand. Hence our aim is to build a firewall over this new technology to protect the integrity of it. So designing and developing Open-Flow based firewall application will form the basis of the project. The making and implementation shows that most of the firewall functionalities can be built using a software, without use of any hardware explicitly. We are using open source OpenDayLight Controller Carbon Version for our experiments. To perform experiment, we have installed Mininet emulator for creating network topologies. In this paper, we present the implementation details of the firewall application.

Copyright©2018 Yash Baja et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

INTRODUCTION

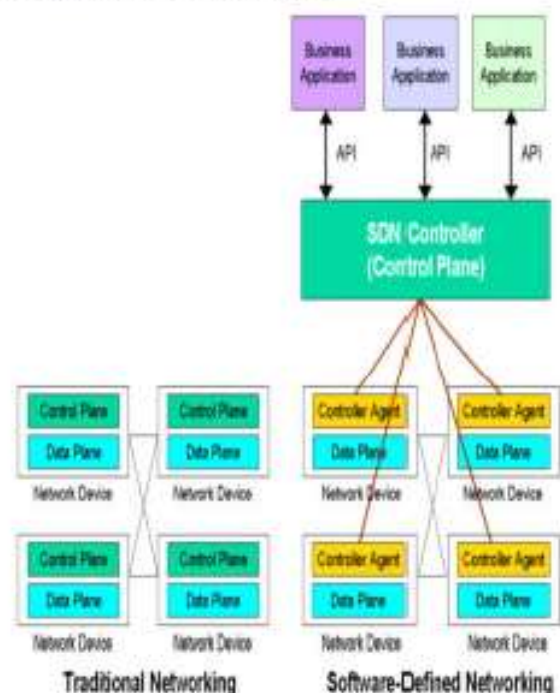
The current internet network is well set and huge in terms of topologies and its connections all over the world. But it has its drawbacks too. The data and the control plane being together yet makes it difficult in finding optimality in terms of routing, maintaining security and much more. Also adding or removing even a single router makes us change the whole network and is tedious. In a view of finding solution the SDN technology was introduced in 2005 where in the control plane gets totally removed and what we get is a controller i.e. the brain of the network and a bunch of routers that are meant for handling data and routing them. The new brain so formed is called as the OpenDayLight controller in the world of SDN. A controller responsible for taking the routing, protocol decisions whenever a packet arrives at its door. Our project aims at building a firewall over this controller that will in turn help the controller to take the decisions regarding the packets that arrive at the controller. The controller accompanied with the firewall will have the ability to accept, drop, or reject the incoming packets thus ensuring the safety of the system from malicious packets. Also the firewall which is created on POX platform, will be able to save the system from any malicious attack by an intended attacker. The remainder of this paper is organized as follows, Section 2 presents the comparison of this project with the existing system, followed by Section 3 which states the proposed system and the implementation of each block of the system, the Limitations are shown in section 4 and in the last section conclusions and references are presented.

*Corresponding author: Yash Bajaj

Computer Department Vivekanand Institute of Technology

Comparison With Existing System

On viewing the Existing systems and comparing it with the project we find that the basic disadvantages are the more privileged uses of SDN technology.



software-defined.pdf

As mentioned earlier the existing networks are highly complex. The actual comparison between the two can be depicted by the figure above. The traditional and yet existing system consists of networking elements that possess both the control and the data plane in every element that is installed in the network. This puts a slight disadvantage as setting up a new element would require making changes in control plane of almost every element that exists in the network. This leads to decrease in optimality goal of the network. SDN on the other hand takes the control plane from every element and creates the SDN controller and the elements left without the control plane are now termed as SDN switches that are just used to regulate the flow data packets. This provides more configuration flexibility and accuracy accompanied with Data Flow optimization. Same thing happens in the case of Firewalls in the traditional and SDN networks. The firewalls in the traditional networks have limited functionalities where as in SDN more privileges are granted to the same firewall thus providing better security.

The Proposed System

The System proposed is a firewall that is based on the concept of SDN technology i.e. it will perform the basic functionalities of a firewall based on the fact that the network that it is working in is SDN based and not the traditional one. Keeping this view in record, the Firewall will be implemented in following steps

1. Creating the Test Bed for OpenDayLight.
2. Configuring the OpenDayLight Controller
3. Setting up the Firewall (Firewall setup phase).
4. Checking the functionalities (Operation phase)
5. Learning about the vulnerabilities while implementing the firewall and performing attacks such as IP spoofing, Source routing.

Creating Test Bed for OpenDayLight

The Linux Foundation hosted a jewel OpenDayLight Project (ODL), which is an open source SDN project aimed at enhancing SDN by offering a community-led and industry-supported framework for the OpenDayLight Controller, which has been renamed the OpenDayLight Platform. Being an Open Source commodity even including the end users and customers, it provides a shared platform for those with SDN goals to work together to find new solutions. OpenDayLight includes support for the OpenFlow protocol, but can also support other open SDN standards. The first SDN standard, as made a consideration by OpenFlow, defines the open communications protocol that allows the SDN Controller to work with the forwarding plane and make changes to the network. This gives businesses the ability to better adapt to their changing needs, and have greater control over their networks. The OpenDayLight Controller can support a modular controller framework, but can provide support for other SDN standards and upcoming protocols and also is able to deploy in a variety of production network environments. Exposing the northbound APIs is one of the basic functionality of the OpenDayLight Controller (ODLC), which are then used by applications to collect information about the network, run algorithms to conduct analytics, and then use the same to create new rules throughout the network. The Controller is the actual point of communication of Firewall with the system because it is where the actual decision of packets will take place. The packets that arrive will either be dropped or

accepted by the Firewall and then forwarded by the controller to the specific Switch in the network.

Configuring the OpenDayLight Controller

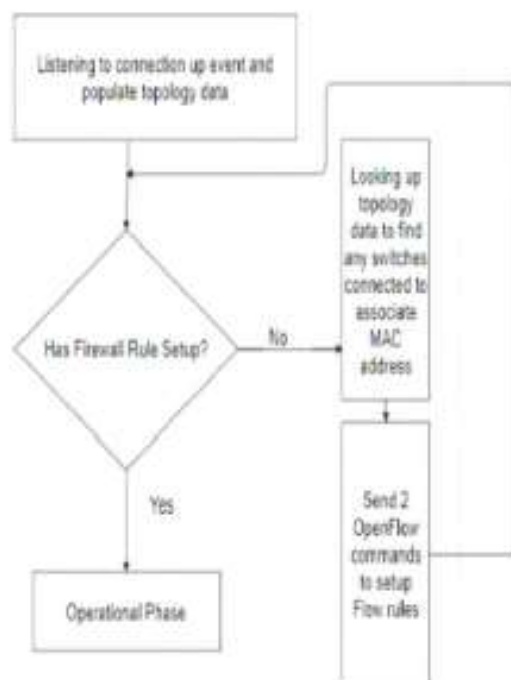
As mentioned above, the Controller is the actual point of communication of Firewall with the system because it is where the actual decision of packets will take place. Hence configuring the controller and setting up its functionality is one important step in this project. OpenDayLight is a project under Linux Foundation that is meant for the enhancement of SDN technology. Apart from this it also helps in Network Visualisation Formation. Starting from Hydrogen this controller has had 6 releases and the latest one i.e. Carbon will be used for setting up the SDN controller. OpenDayLight has a 3 layered architecture where in the 2nd layer is the OpenDayLight Controller consisting of two interfaces viz the northbound and the southbound interfaces. The northbound interface connects the ODL to the first layer consisting of northbound APIs i.e. the applications where actual data is generated and the appropriate processing on it is done. On the other hand the southbound interface connects the controller to the 3rd layer consisting of OpenVSwitches that allow the redirection or sending of data packets to addresses provided by the controller. So in a nutshell it can be said that the ODL works as the brain (control plane) of the SDN technology whereas the OpenVSwitches as the data plane. For a packet filtering Firewall, this happens to be easily of lot of help as the main aim of it is to filter data packets and allow the specified ones and ODL does the same. So the ODL will be setup on the topology created in the mininet emulator to show the actuality of SDN working and also the project. The topology setup there will have a flow of data packets in turn controlled by ODL and forwarded by the southbound interfaces of the same. It will comprise a firewall setup by ip-tables which will perform the work of packet filtering and verify the packets that arrive to the host.

For verifying the few of the possible things which could be checked are

1. Source IP address of the packet. This is necessary because IP spoofers might have changed the source IP address to reflect the origin of packet from somewhere else, rather than reflecting the original source.
2. Destination IP Address. The firewall rules should check for IP address rather than DNS names. This prevents abuse of DNS servers.
3. IP Protocol ID.
4. TCP/UDP port number.
5. ICMP message type.
6. Fragmentation flags.
7. IP Options settings.

Setting up the Firewall (Firewall setup Phase)

This is obviously the essential step of the project as after the setting of the firewall the rules that guard the firewalls judgment are decided. It is a continuous loop in the system that ends when all the rules are decided and we don't any more rules to be added. The functioning of this step can be shown with the help of picture below



It can be explained as follows

Step1: Listening to Connection up Event: After de-ployment or every time after start-up the firewall waits for connection up event i.e. it waits network to go inline. The main reason for this is for the firewall to be able to understand the network topology accompanied by association of mac to ip addresses. This is done to understand the network that the firewall is working in which in turn helps the admin to properly decide the rules needed to be setup for the firewall.

Step2: Checking for Rules: After understanding the net-work checking of rules is done. Here in checking is of two types one to check if there are any pre existing rules and if any understand them and accept or reject the rule as per requirement. Other is to decide the rule from the scratch i.e. no rule pre exists and each and every rule is decided. Irrespective the way be, proper checking is done and necessary action is taken whether or not the rules are all set. If all the required ones are set then the firewall moves to the operational phase else it moves on to use the understanding of the step1 and set up rules.

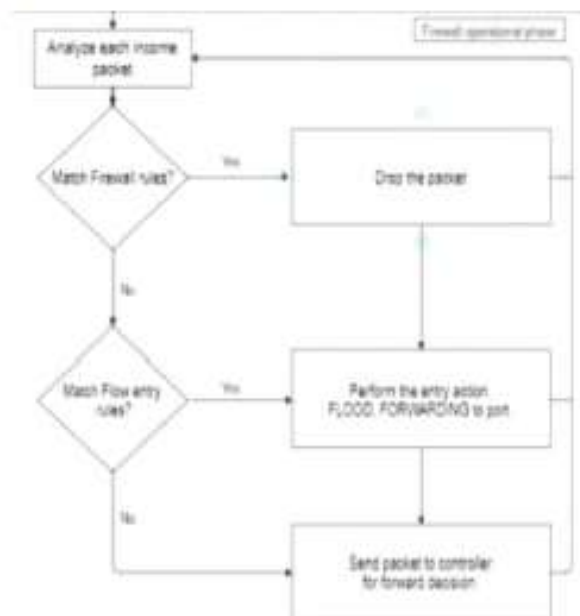
Step3: Finding switches connected to associate MAC addresses: In here the firewall now already knows that new rules needed to be set so it uses the understanding gained in step1 to use The firewall associates MAC addresses to the different OpenVSwitches that are specially designed for SDN technology in the view of distribution of packets. Now the job of these OpenV Switches is to remember the MAC addresses allocated to them and whenever a packet arrives at their depart forward it to that specific Mac Address itself thus ensuring redirection and network handling. Now in the present scenario,

the system starts allocating the MAC address To the particular switches and then switches to step 4

Step4: Sending OpenFlow Rules: This is indeed the main step. In here the firewall sends up the rules that are required and need to be followed by the Switches that are assigned and follow the controller. The rules set by the firewall over here are basic commands that contain the conditions for the data packets when they arrive at the respective switch and the switches follow it to perform the job of acceptance or rejection of the packet. This is the actual place where the firewall comes in the picture. The basic function of the firewall of setting up a wall of fire for maliciousness is performed here. With these steps being done, a continuous loop is repeated until all the rules are setup. Setting up the firewall brick by brick is done here and after the job being done it moves on the Operational Phase of the project.

Checking the Functionalities (Operational Phase)

In this step the firewall that is developed is put to function. Here in each and every functionality of the Firewall is exploited and checked upon to verify its efficiency. This step can be depicted diagrammatically as below.



Step1: Analysis: The packet that arrived at the firewall needs to be analyzed for various factors like the source address, destination address, the network subnet and the headers etc. Each and every aspect that needs to be checked for maintaining integrity of the data packet received and to be assured that the contents are error-less are performed in this very step. The end of this step marks a verified data packet and that packet itself has to face the firewall created by us. The analysis verifies the packet and if non suitable case found, the packet is thrown away into the internet pool and is not sent further for any verification. On the other hand the packet that packet analysis moves ahead to face the wall. Thus analysis proves as the door step verification or as the first line of verification that can be performed by the firewall.

Step 2: Match Firewall Rules: The second step marks the second line of verification in the project. Here the packet that

passed the analysis is put to face the Firewall itself. The firewall uses the rules that it sets up in the Setup Phase of the project and verifies the packet accordingly. It checks for contents of the packet and check for malicious data, bugs, worms that may or may not be present in the data packet. The rules that are set help the firewall in deciding whether the packet received is suitable for the host or not. The questions like whether the packet is true or not? and whether the packet will cause harm or no harm at all? are to be answered by the firewall and then as per the answers it decides the quality of the packet. This quality factor proves helpful in deciding the further fate of the data packet. If the firewall feels that the packet is suitable for the host then it is passed and forwarded to the respective OV Switch by the OpenDayLight controller. On the other hand if it fails to pass, then it is straight away thrown back into the internet pool and left there to travel until its TTL comes to an end. There also may exist a possibility where in the packet keeps on re-arriving at the same firewall and getting rejected a multiple times by the same procedure.

Step3: Matching Flow Entry Rules: After passing the firewall the packet reaches the OV Switch where its fate is further decided. The OVS on receiving the packets forwards it to the ODL where in the ODL checks within its tables whether for the destination IP mentioned does there even exists a mac in our system. If it does then it is re directed to the associated Ovs. If it doesn't exist then an entry is made for the following purposes. They are a) There exists an IP for which we had no MAC address b) FLOOD FORWARD to the respective port where it belonged to. Irrespective the scenario, the packet is forwarded to its proper destination by the OVS or left to roam in the Internet Pool

The Attack

This is merely a testing step in the project. Inhere on purpose an illegal attack will be performed on the firewall to check for its functionalities. The attack can be of any type and the firewall must be able to defend against it.

Attacks possible and will be tried are

- IP spoofing: attacker imitating as legitimate user and performing undesired actions
- Source Routing: using special routes for packets in order to appear coming from known entities
- Port Scan: to open up covert channels or establish using open services such as telnet
- Fragmenting: fragmenting packets to bypass the firewall

Limitations

The various limitations of this whole system are

1. Economically not feasible.
2. Properly network needed to be set on a larger scale
3. Constant maintenance required.
4. Proper tools and switches need to be designed specially for this.
5. Not possible to display it manually.

CONCLUSION

Though Firewall has its own limitations, its a healthy and safe procedure for the betterment of networking in various countries round the globe. Not only our security is increased, but its various problems are solved due to it thus resulting in less preaching and disloyalty. It has a bright future and economic scope thus not only it ll generate employment but also it ll be a generator of money as well. Irrespective of the fact that with new technology new threats are released but dealing with that threats and breaking it down to the very core thus ensuring integrity is the scope of this project

References

1. Michelle Suh, Sae Hyong Park, Byungjoon Lee, Sunhee Yang Building Firewall over the Software-Defined Network Controller SDN Research Section, ETRI (Elec-tronics and Telecommunications Research Institute), Korea
2. Jake Collings, Jun Liu An Open Flow-based Prototype of SDN-Oriented Stateful Hardware Firewalls 2014, IEEE 22nd International Conference on Network Protocols
3. Justin Gregory V. Pena and William Emmanuel Yu Development of a Distributed Firewall Using Soft-ware Defined Networking Technology Department of Information Systems and Computer Science Ateneo De Manila University Loyola Heights, Quezon City, Philippines
4. Thuy Vinh Tran, Heejune Ahn A Network Topology-aware Selectively Distributed Firewall Control in SDN Department of Electrical and Information Engineering Seoul National University of Science and Technology Seoul, Republic of Korea
5. <https://turbonomic.com/blog/on-technology/sdn-software-defined-networking-primer-and-why-we-need-sdn/>
6. <https://www.pressreader.com/india/opensource-for-you/20160610/282157880544330>
7. <https://en.wikipedia.org/wiki/SDN>
8. <https://www.slideshare.net/1z1dsb/why-sdn>
9. <http://opensourceforu.com/2016/07/implementing-a-software-defined-network-sdn-based-firewall/>
10. <http://sdnhub.org/resources/useful-mininet-setups/>
11. <https://www.techopedia.com/definition/4038/packet-filtering>
12. <http://securityworld.worldswelcome.com/packet-filtering-firewall-an-introduction>
13. <https://www.sdxcentral.com/sdn/definitions/sdn-controllers/.opendaylight-controller/>
14. <http://www.necoma-project.eu/m/filer-public/18/30/1830b40c-a2e2-4a0b-b83e-af8b2d846e61/imt-zhang-commnet2015.pdf>
15. <https://www.ietf.org/proceedings/91/slides/slides-91-i2nsf-0.pdf>

How to cite this article:

Yash Bajaj *et al* (2018) Firewall Based on the Concept of Sdn', *International Journal of Current Advanced Research*, 07(2), pp. 9960-9963. DOI: <http://dx.doi.org/10.24327/ijcar.2018.9963.1665>

Plagiarism Report

Check 1:

Plagiarism Scan Report

Summary	
Report Generated Date	01 Nov, 2017
Plagiarism Status	100% Unique
Total Words	995
Total Characters	6238
Any Ignore Url Used	

Check 2:

Plagiarism Scan Report

Summary	
Report Generated Date	01 Nov, 2017
Plagiarism Status	100% Unique
Total Words	982
Total Characters	5670
Any Ignore Url Used	

Check 3:

Plagiarism Scan Report	
Summary	
Report Genrated Date	01 Nov, 2017
Plagiarism Status	100% Unique
Total Words	821
Total Characters	4657
Any Ignore Url Used	

SDN based Firewall

Viren Wadhwani, Yash Bajaj, Yogesh Rohra

Computer Department

Vivekanand Institute Of Technology

viren.wadhwani@ves.ac.in

yash.bajaj@ves.ac.in

yogesh.rohra@ves.ac.in

Mentor:- Arthi C I

arthi.ci@ves.ac.in

ABSTRACT

In today's evolving technology, the need to adapt is the only necessity. The same is applicable when we are talking about the field of Network and Securities i.e the SDN technology. SDN is the newest evolving topic in the field of Networking. There are many problems faced by the traditional networks. These are solved by SDN. It is a way of making networks more manageable and more dynamic. It is very helpful in realizing the whole network beforehand thus giving a clear idea of the topologies required and the issues which can be solved even before setting the network up. But as in a new technology comes up it brings in new threats and issues. Thus our aim is to build a firewall over this new technology to protect the integrity of it. In this paper we

focus on designing and developing a Firewall for our SDN controller which in turn performs the function of safeguarding the network from inside as well as the outside internet. To perform experiment, we have installed Mininet emulator for creating network topologies. In this paper, we present the implementation details of the firewall application.

1. INTRODUCTION

The current internet network is well set and huge in terms of topologies and its connections all over the world. But it has its drawbacks too. The data and the control plane being together yet makes it difficult in finding optimality in terms of routing, maintaining security and much more. Also adding or removing even a single router makes us change the whole network and is tedious. In a view of finding solution the SDN technology was introduced in 2005 where in the control plane gets totally removed and what we get is a controller i.e. the brain of the network and a bunch of routers that are meant for handling data and routing them. The new brain so formed is called as the SDN controller in the world of SDN. A controller responsible for taking the routing, protocol decisions whenever a packet arrives at its door.

Our project aims at building a firewall over this controller that will in turn help the controller to take the decisions regarding the packets that

arrive at the controller. The controller accompanied with the firewall will have the ability to accept, drop, or reject the incoming packets thus ensuring the safety of the system from malicious packets. The Firewall thus created will stand in between the controller and outside internet. Also it has the ability to handle the flow of packets shown by the controller thus providing an all rounder functionality where the controller is the mega brain and has the record of every bit of data irrespective whether it flows in or out of network. Also the firewall which is coded in Python, provides a simplified GUI for the users to set up the firewall as per his/her convenience and also it will be able to save the system from any malicious attack by an intended attacker.

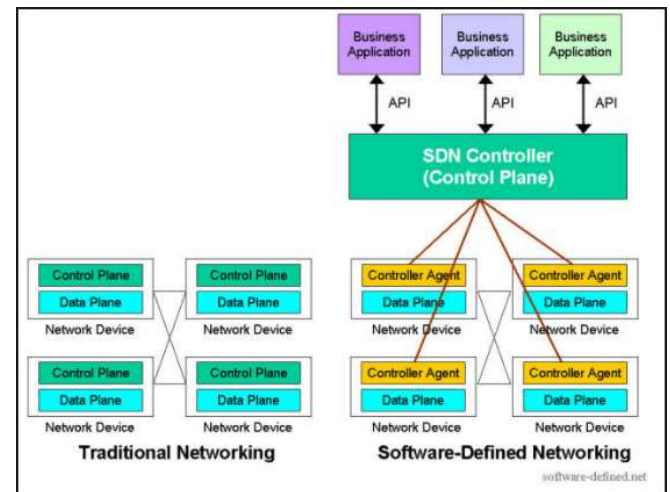
Thus the project aims at emulate a network topology. The implementation includes the respective emulation of the network along with the firewall.

The remainder of this paper is organized as follows, Section 2 presents the comparison of this project with the existing system, followed by Section 3 which states the proposed system and the implementation of each block of the system, the Conclusion is shown in section 4 and in the last section the references are presented.

2.Comparison with Existing System

On viewing the existing system and comparing it with the project, we find

that the basic disadvantages are the more privileged uses of the SDN technology.



As mentioned earlier, the existing networks are highly complex. The actual comparison between the two can be depicted by the figure above.

The traditional and yet existing system consists of networking elements that possess both the control and the data plane in every element that is installed in the network. This puts a slight disadvantage as setting up a new element would require making changes in the control plane of almost every element that exists in the network. This leads to a decrease in the optimality goal of the network.

On the other hand, the control plane from every element of the network is detached, and the SDN controller is created, leaving the elements without the control plane now termed as *OvSwitches* that are just used to regulate the flow of data packets. This provides more configuration flexibility and accuracy accompanied with Data Flow optimization.

Same thing happens in the case of Firewalls in the traditional and SDN networks. The firewalls in the traditional networks have Limited functionalities where as in SDN more privileges are granted to the same firewall thus providing better security.

3.The Proposed System.

The System proposed is a firewall that is based on the concept of SDN technology i.e. it will perform the functionalities of a firewall based on the fact that it has to safeguard the Controller as well as the network it is working on. It will not only keep a check on the outside network but also the emulated network in the Mininet. Keeping this view in record, the Firewall will be implemented in following steps

- 1)Coding the firewall in Python Language.
- 2)Installing Mininet and Creating the Network Topology.
- 3)Setting up the Firewall (Firewall setup phase).
- 4)Checking the functionalities(Operation phase)
- 5)Attack on the Firewall.

3.1 Coding the firewall in python Language.

Since we will be working with mininet, it supports Python codes and

hence the firewall to sit upon the Controller will be coded in Python.

We will use tkinter library of Python for creating the GUI of our firewall.

The GUI will have two major blocks one is the Status which will tell the current network connections of the controller and the network emulated and other will be Policy which will display the rules set up by the user. It will also have different toggles and options for insert update and delete of functionalities and help too.The Screenshot below shows the GUI created for the firewall.

Status				Policy	
Source Add	Source Port	Destination Add	Destination Port	Protocol	State
192.168.43.71	35498	172.217.100.143	80	tcp	SYN_SENT
192.168.43.71	36902	54.69.109.117	443	tcp	SYN_SENT
192.168.43.71	36906	54.69.109.117	443	tcp	SYN_SENT
192.168.43.71	60699	2484	80	tcp	SYN_SENT
192.168.43.71	36402	192.168.63.1	33	udp	ESTABLISHED
192.168.43.71	45473	192.168.63.1	33	udp	ESTABLISHED
192.168.43.71	38544	192.168.63.1	33	udp	ESTABLISHED
192.168.43.71	47927	192.168.63.1	33	udp	ESTABLISHED
0.0.0.0	0	0.0.0.0	0	udp	ESTABLISHED

3.2 Installing Mininet and Creating the Network Topology

Our project will be emulated in Mininet which is a network emulator which creates a network of virtual hosts, switches, controllers, and links in other sense a fully functioning network as per convenience. The hosts thus created run standard Linux network software, and its switches support highly flexible custom routing i.e SDN.

The Advantages of Mininet are:

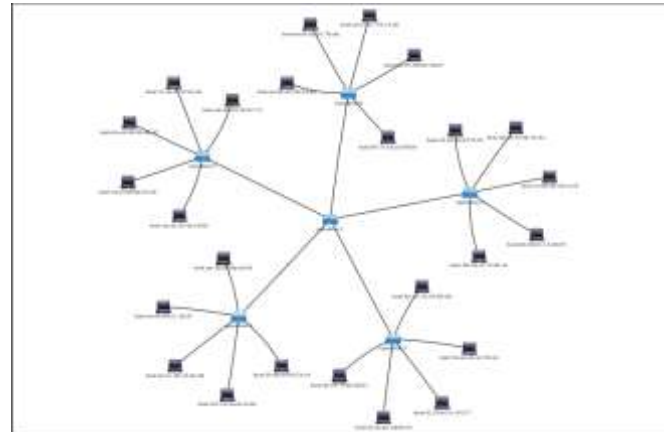
- It provides a simple and inexpensive network testbed for development and research

- It allows multiple concurrent developers to work independently in the same network on the same topology.
- **It allows testers to perform complex topology testing**, without the need to wire up a physical network and analyze it for proper optimal results.
- **Mininet can be used without programming that is out of the box**, but also provides an extensible **Python API** for network creation and experimentation purposes.
- It helps in getting an easy way to get correct system *behavior*(and, to the extent supported by your hardware, performance) and to experiment with topologies. Which can be created or imagined.

These networks which will be created virtually resemble a real Linux kernel and network stack (including any kernel extensions which you may have available, as long as they are compatible with network namespaces.)

Thus GUI developed will be tested on Mininet, for an SDN controller, with modified switch, or host, also *can be moved to a real system with minimal changes*, for real-world testing, performance evaluation, and deployment. Thus indicating that a design that works in Mininet can usually be moved directly to hardware.

After installation setting up the network is simply work of commands to specify the type of topology and number of nodes. One such Tree topology is show in the picture below.

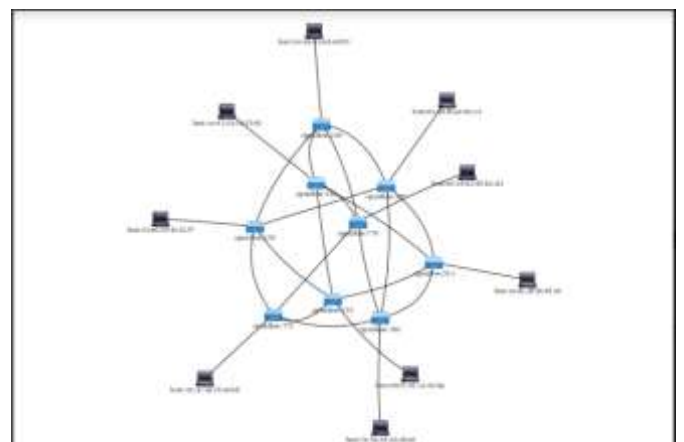


This Topology has the SDN controller at the center. Each switch including the controller has 5 connections. So in all it has 1 controller 5 switches and 25 nodes.

The command used for the same is

```
“sudo mn -topo tree,depth=2
fanout=5”
```

Another example for the same is:



3.3 Setting the Firewall(Firewall Setup Phase).

This is obviously the essential step of the project as after the setting of the firewall the rules that guard the firewalls judgment are decided.

In this step the GUI created in step1 is put to actual use. The user uses the GUI to insert proper rules in the iptables of the firewall and checks the network statistics

Like the traditional networks, here also the iptables consist of Input Output and Forward tables with following functionalities

a)Append:- This adds a new rule to the end of all existing rules.

b)Delete: this is used to delete a rule.

Where as Flush totally clears all the rules in a single click

c)Insert: to add a rule at the start .

Like wise it has three functions as

a)Accept: to allow the packets to pass in the network.

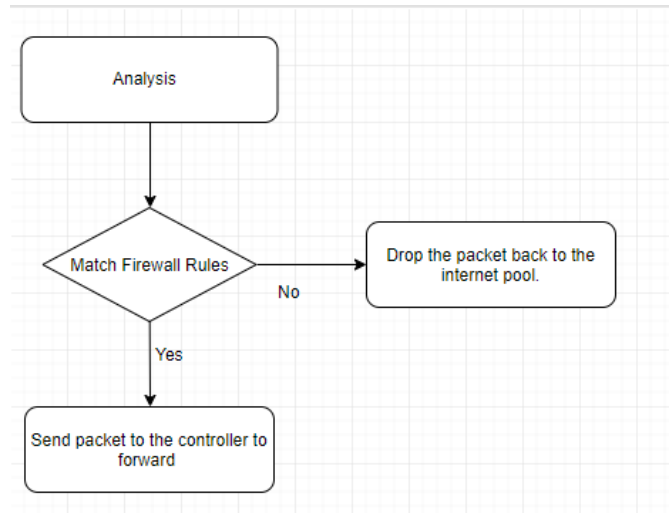
b)Drop: to disallow any packet without even any acknowledgment

c)Reject: to disallow any packet with an acknowledgment

3.4 CHECKING THE FUNCTIONALITIES (OPERATIONAL PHASE)

In this step, the firewall that is deployed is put to function. Here in

each and every functionality of the Firewall is exploited and checked upon to verify its efficiency. This Step can be depicted diagrammatically as below:



The different steps are:

Step1: Analysis

The packet that arrived at the firewall needs to be analyzed for various factors like the source address, destination address, the network subnet and the headers etc. Each and every aspect that needs to be checked for maintaining integrity of the data packet received and to be assured that the contents are error-less are performed in this very step. The end of this step marks a verified data packet and that packet itself has to face the firewall created by us. The analysis verifies the packet and if non suitable case found, the packet is thrown away into the internet pool and is not sent further for any verification. On the other hand the packet that packet analysis moves ahead to face the wall. Thus analysis proves as the

door step verification or as the first line of verification that can be performed by the firewall.

Step 2: Match Firewall Rules:

The second step marks the second line of verification in the project. Here the packet that passed the analysis is put to face the Firewall itself.

The firewall uses the rules that it sets up in the Setup Phase of the project and verifies the packet accordingly. It checks for contents of the packet and check for malicious data, bugs, worms that may or may not be present in the data packet. The rules that are set help the firewall in deciding whether the packet received is suitable for the host or not. The questions like “whether the packet is true or not?” and “whether the packet will cause harm or no harm at all?” are to be answered by the firewall and then as per the answers it decides the quality of the packet. This quality factor proves helpful in deciding the further fate of the data packet. If the firewall feels that the packet is suitable for the host then it is passed and forwarded to the respective OVSwitch by the SDN controller. On the other hand if it fails to pass, then it is straight away thrown back into the internet pool and left there to travel until its TTL comes to an end. There also may exist a possibility where in the packet keeps on re-arriving at the same firewall and getting rejected a multiple times by the same procedure.

3.5 THE ATTACK

This is the final testing step in the project.

Inhere on purpose a malicious attack will be performed on the firewall to check for its functionalities. The attack can be of any type and the firewall must be able to defend against it.

The attacks performed are:-

a)Syn Flooding:- In here a flow of continued SYN request is sent to the target so as to consume up the resources of the target.

b)Null attack: A null session is an anonymous connection to an inter-process communication network service on [Windows](#)-based computers.



4. CONCLUSION

Though Firewall has its own limitations, its a healthy and safe procedure for the betterment of networking in various countries round the globe. Not only our security is increased, but its various problems are solved due to it thus resulting in less preaching and disloyalty. It has a bright future and economic scope thus not only it ll generate employment but also it ll be a generator of money as well. Irrespective of the fact that with new technology new threats are released but dealing with that threats and breaking it down to the very core

thus ensuring integrity is the scope of this project.

5..References

1. Michelle Suh, Sae Hyong Park, Byungjoon Lee, SunheeYang Building Firewall over the Software-Defined-Network Controller SDN Research Section, ETRI (Elec-tronics and Telecommunications Research Institute),Korea
2. Jake Collings,Jun Liu An OpenFlow-based Prototype of SDN-Oriented Stateful Hardware Firewalls 2014, IEEE22nd International Conference on Network Protocols
3. Justin Gregory V. Pena and William Emmanuel Yu Development of a Distributed Firewall Using Soft-ware Defined Networking Technology Department of Information Systems and Computer Science Ateneo De Manila University Loyola Heights, Quezon City,Philippines
4. Thuy Vinh Tran,Heejune Ahn A Network Topology-aware Selectively Distributed Firewall Control in SDN Department of Electrical and Information Engineering Seoul National University of Science and Technology Seoul, Republic of Korea
5. <https://turbonomic.com/blog/ontechnology/sdn-software-defined-networking-primer-and-why-we-need-sdn/>
6. <https://www.pressreader.com/india/opensource-for-you/20160610/282157880544330>
7. <https://en.wikipedia.org/wiki/SDN>
8. <https://www.slideshare.net/lz1dsb/why-sdn>
9. <http://opensourceforu.com/2016/07/implementing-a-software-defined-network-sdn-based-firewall/>
10. <http://sdnhub.org/resources/useful-mininet-setups/>
11. <https://www.techopedia.com/definition/4038/packet-filtering>
12. <http://securityworld.worldiswelcome.com/packet-filtering-firewall-an-introduction>
13. <https://www.sdxcentral.com/sdn/definitions/sdn-controllers/opendaylight-controller/>
14. <http://www.necoma-project.eu/m/filerpublic/18/30/1830b40c-a2e2-4a0b-b83e-af8b2d846e61/imt-zhang-comnet2015.pdf>
15. <https://www.ietf.org/proceedings/91/slides/slides-91-i2nsf-0.pdf>

Plagiarism Report

Plagiarism Scan Report

Summary

Report Genrated Date	05 Apr, 2018
Plagiarism Status	100% Unique
Total Words	999
Total Characters	5909
Any Ignore Url Used	

Plagiarism Scan Report

Summary

Report Genrated Date	05 Apr, 2018
Plagiarism Status	100% Unique
Total Words	973
Total Characters	5714
Any Ignore Url Used	

Plagiarism Scan Report

Summary

Report Genrated Date	05 Apr, 2018
Plagiarism Status	100% Unique
Total Words	108
Total Characters	617
Any Ignore Url Used	