University of Bedfordshire

ASSIGNMENT TOP SHEET

Faculty of Creative Arts, Technologies & Science

Department of Computer Science & Technology

Student Ref. No **2016051**	Unit Code: **CIS017-3**
Unit Name: **Undergraduate Project**	Deadline for Submission(s) **Monday 17th May 2021**
Student's Surname Mohamed Haris	Student's Forename Mohamed Arshad
Unit Leader's Name: **Enjie Liu**	Supervisor: Ms. Dilushinie Fernando
Assignment Details: **Assessment 2: Final Report**	

Instructions to Student:

Please note: Work presented in an assessment must be the student's own. Plagiarism is where a student copies work from another source, published or unpublished (including the work of a fellow student) and fails to acknowledge the influence of another's work or to attribute quotes to the author. Plagiarism is an academic offence.

Work presented in an assessment must be your own. Plagiarism is where a student copies work from another source, published or unpublished (including the work of another student) and fails to acknowledge the influence of another's work or to attribute quotes to the author. Plagiarism is an academic offence and the penalty can be serious. The University's policies relating to Plagiarism can be found in the regulations at http://www.beds.ac.uk/aboutus/quality/regulations. To detect possible plagiarism we may submit your work to the national plagiarism detection facility. This searches the Internet and an extensive database of reference material including other students' work to identify. Once your work has been submitted to the detection service it will be stored electronically in a database and compared against work submitted from this and other universities. It will therefore be necessary to take electronic copies of your materials for transmission, storage and comparison purposes and for the operational back-up process. This material will be stored in this manner indefinitely.

I have read the above information and I confirm that this work is my own and that it may be processed and stored in the manner described.

Signature (Print Name): Arshad Haris Date: 17th May 2021

Extension deadline

CAAS agrees that the assignment may be submitted ____ days after the deadline and should be marked without penalty.

CAAS confirmation.....

Please leave sufficient time to meet this deadline and do not leave the handing-in of assignments to the last minute. You need to allow time for any system problems or other issues.

Mohamed Haris Mohamed Arshad

2016051

SDN BASED LOAD-BALANCING IN CLOUD COMPUTING WITH SECURITY ENHANCEMENT.

BSc (Hons) Computer Networking

Undergraduate Thesis Report

Department of Computer Science & Technology – University of Bedfordshire

Ms. Dilushinie Fernando

AY 2021

Abstract

In IT industry load-balancing involves a mixture of time management, commitment and mostly prioritization. Actually, the number of internet-connected devices like mobile, PC and electronic machines are rising day to day in industry, datacenters are expanding, cloud computing concept are growing simultaneously and computer network and security are becoming more complex. So, traditional network strategy is becoming increasingly challenging and inadequate. Therefore, it causes huge traffic in networking system. As a result, there is congestion, which decreases system efficiency, speed, performance and reliability. Also, happening packet loss in the system while transportation. As a consequence, a single server would be unable to accommodate this volume of traffic, necessitating the use of certain techniques to improve network efficiency in system. The use of a load balancer to spread network traffic among multiple servers could reduce the load on a single server. For this case, they improve network stability and performance. But programmability is missed by industry experts. Even there are traditional load balancers are in market but it cannot handle automatically, it was expensive, non-flexible and close vendor. Also, non-programmable because it has programmed but user can't editable or develop new one. So, industry introduced technology called Software Defined Network. It was programmable. Actually, SDN is a rapidly emerging technology that has piqued the interest of both industry and academia. SDN enables network engineers, cloud architect and system administrator to respond quickly to changing business demands by centralizing management of the entire system and network infrastructure. In comparison to a traditional network, SDN is able to make better use of resources and provides the ability to customize the network topology to meet our needs. In this project prime aim on SDN is to customize the process of load-balancing in cloud-based network with security manner. The control plane and the data plane are separated to achieve this concept. The aim of this research is using an SDN-controller Python Network Operating System (POX) to implement the least packet algorithm, which is used in traditional load balancer, to disperse load across servers. This paper discusses about benefits, limitation and existing similar system, implementation, test and evaluation.

Acknowledgement

At the beginning of the content, I would like to express my gratitude and thank to University of Bedfordshire (UK) and Sri Lanka Institute of Information Technology Academy (Pvt) Ltd to provide me this great opportunity to enhance and advance my career. Also, their resources gave me an extremely good learning experience with their dedicated and quality lecture panel. Next, I would like to honest thank and appreciate Dr. Gayana Fernando, the lecture in charge of the Undergraduate Project and Research methodologies and emerging technologies modules for leading me from the beginning of research up to the final submission of reports and assignments. Ms. Dilushinie Fernando, genuine and honest thank you for the tremendous assistance given and she has been the perfect supervisor who gave me the best and most accurate guidance. She assisted me during the development and documentation areas. Also, recommended to me with new ideas and techniques to complete and achieve this proposed project. Moreover, she also assisted me to successfully achieve my aim and objectives from the starting of the research up to the final submissions. In other word I can say she act like a mentor to my career. When I am felt struggle at some point, she helped me a lot for overcome from that struggle especially at the implementation process.

Very special thanks to Mr. Methaq Khamees (Iraq), For gave such a wonderful advice to implement the project and explanation on SDN topics and their future projects. Also, heartfelt thanks to Mr. Ajith Pasqual (Senior Lecturer - University of Colombo). For his best advice at the topic selection from the beginning of the project. An honest and genuine thanks to my family members and friend who supported and assistance me to successfully complete project on time. Finally, I would like to my thank my batch mates for their assists and support to complete this project.

Dedication

This Final thesis dedicated to my supervisor Ms. Dilushinie Fernando, the lecture in charge of our undergraduate project lesson Dr. Gayana Fernando, Networking students, my batch mates, family members, friends and all people who are interested in Software Defined Networking, Cloud computing, security and Load-balancing area.

Key words

Software Defined Network

Load Balance

Cloud Computing

Security

OpenFlow

POX

Mininet

List of Abbreviation

SDN	Software Defined Network
RPA	Robotic Process Automation
AR	Augmented Reality
VR	Virtual Reality
IT	Information Technology
DevOps	Development and Operation
QoS	Quality of Service
SDLC	Software Development Life Cycle
CLI	Command Line Interface
GUI	Graphical User Interface

Table of Contents

Abstract i
Acknowledgement ii
List Of Figures viii
List Of Tables ix
1 Introduction 1
1.1 Project Background 1
1.2 Project Aims and Objectives 3
1.3 Description of the artifact 3
1.4 Structure of Report 3
2 Literature Review and Market Research 6
2.1 Literature Review 6
2.1.1 Challenges in cloud research area 6
2.1.2 Challenges in load balance cloud computing 7
2.1.3 Why SDN involved in Information and communication industry 9
2.1.4 Security Impact in SDN areas 13
2.2 Research Gap/Conclusion 14
2.3 Market Research 15
3 Methodology and Design 17
3.1 Methodology 17
3.2 Planning 17
3.3 WBS 19
3.4 Gauntt Chart 20
3.5 Requirement Gathering and Analysis 20
3.5.1 Primary Data Gathering – Market research 20
3.5.2 Secondary Data Gathering – Literature review 20
3.6 Design 21
4 Implementation 23
5 Testing and Evaluation 38
5.1 Testing 38
5.2 Evaluation 65

6 Results and Discussion 67
6.1 Results 67
6.2 Discussion 69
7 Conclusion and Future work 71
7.1 Conclusion 71
7.2 Limitation 71
7.3 Benefit of the SDN based load-balancing system project 72
7.4 Future Work 73
8 References 74

List Of Figures

Figure 2-1: SDN Architecture 10
Figure 2-2: Flowchart for project system 13
Figure 2-3: SDN security threats and attack surfaces (Muhammet Fatih Akbas, Enis Karaarslan, Cengiz Güngör, 2016) 14
Figure 3-1: Prototype model (Wahab, 2009) 17
Figure 3-2: Network diagram for project 21
Figure 4-1: Oracle VM Virtual Manager for project 24
Figure 4-2: Mininet installation command and output. 25
Figure 4-3: System update 26
Figure 4-4: System upgrade 27
Figure 4-5: cleanup the pre-Mininet 28
Figure 4-6: List available version 29
Figure 4-7: testing the Mininet with topology creation 30
Figure 4-8: Remote XTerm testing the Mininet 30
Figure 4-9: Mininet's MiniEdit topology creates command and its output 31
Figure 4-10: MiniEdit interface for system topology creation. 32
Figure 4-11: MiniEdit topology for SDN based load-balancing system 32
Figure 4-12: Actual topology created via command 34
Figure 4-13: pick a server for new connection 35
Figure 4-14: List of algorithms 36
Figure 4-15: Wireshark capturing loopback 37

List Of Tables

Table 5-1: Test Case 01 39
Table 5-2: Test Case 02 40
Table 5-3: Test Case 03 41
Table 5-4: Test Case 04 42
Table 5-5: Test Case 05 43
Table 5-6: Test Case 06 44
Table 5-7: Test Case 07 48
Table 5-8: Test Case 08 51
Table 5-9: Test Case 09 54
Table 5-10: Test Case 10 57
Table 5-11: Test Case 11 61
Table 6-1: Load-balancing concept with servers, controller and users' interfaces..... 67
Table 6-2: multiple time requests from users and balancing technique..... 69
Table 6-3: least balance coding interface..... 69

1 Introduction

This chapter discusses the introduction to the thesis report which includes the project background, project aim, project objectives, description of artifact and also the structure of the thesis report.

1.1 Project Background

Cloud computing is the provision of computing resources or services like storage, servers, integration, application and more over the internet. Based on location cloud computing is categorized into public, private, hybrid and community cloud (Grance, October 7 .2009). As well as based on services it is categorized into IaaS (Infrastructure-as-a-service), SaaS (Software-as-a-service) and PaaS (Platform-as-a-service). It offers a versatile and quick solution way to keep data and files and restore them. Specifically, to make for the spreading number of large data sets and file available around the globe with users. Virtualization in technology act as the basis of the IaaS service distribution model to virtualize and efficiently provide cloud services in effective style. However, most of the latest existing research activities in the past years primarily concentrated on the productive use of the compute and storage services using the virtualization technology such as Xen (Xen, 2013), Kernel Virtual Machine (KVM) (Kernel-based Virtual Machine (KVM), 2013), VMWare (Vmware, 2013) and etc. Network automation, security, load balancing and virtualization of datacenter LAN (Local Area Network) and WAN (Wide Area Network) were not the primarily concentrated of almost all of the researcher and end users. Furthermore, cloud computing and virtualization technology forcing data center's operators and administrators can think beyond the establishment of their traditional network.

A progressive quantity of transmission data allows the growth of internet resources to more traffic between devices on the network, this leads to congestion and information packet loss (J. Saisagar, D. Kothari, R. Kothari, and V. Chakravarthy, 2017). There is no centralized control over traditional networks; with each component (routers, switches) has its own control in the network. Hence, each network component device forwards packet traffic by its configuration such as MAC (media access control) learning devices identification of forwarding data planes and tables (P. Goransson and C. Black and T. Culver, 2017). Although traditional network devices are decentralized, there is a great deal of administration work; devices need to be handled and reconfigured separately, causing confusion in network. With the creation of a good paradigm known as Software Defined

Networking (SDN), as in the traditional network, the control plane is moved from decentralized platforms to a centralized one. This enterprise concept optimizes data-designed facilities, this leads to a decrease in the cost of making switches and decreases the cost of generating switches. Required network management efforts by enabling automation management through programmability (F. Bannour, S. Souihi, and A. Mellouk, 2017).

There are three layers in SDN, which are infrastructure, data and control layer. The infrastructure layer includes network equipment (router, firewall, server and switches) supported by SDN protocol (OpenFlow, NetFlow, sFlow (Phaal, 2004)). Control layer representing the brain of the network that is able to determining the path of data packet and apply limitations on them and get innovative application layer created like load balancer and firewall etc. The application layer and control layer deployed by northbound API, which offers abstract view of the network infrastructure. As well as relation between control layer and data layer are known as the Southbound (API) Application Program Interface, which completed through a protocol such as OpenFlow. A load balancer is a mechanism that delivers network traffic between multiple systems servers to optimize the use of network resources, including reduced response time and enhanced bandwidth (F. Bannour, S. Souihi, and A. Mellouk, 2017). The traditional load balancer for the network is very costly and unique by vendor (U. Zakia, H. Yedder,, 2017). On the other hand, software load balancer is open for everyone and cheaper than hardware, but in addition to running, its performance based on the configured server specification compatibility with the operating system like updates and upgrades.

In this proposed project we consume N number of servers (for implementation 4 servers virtually create), OpenFlow switch, N number of users (5 virtual end devices) and controller (that one act as load balancer its work according to proposed algorithm). We create network infrastructure using these devices that show in fig 1. After implementation we testing, evaluate and compare with existing algorithms created by some researches.

1.2 Project Aims and Objectives

Aim

- To implement load balancing technique to SDN controller to get efficient and secure network infrastructure.

Objectives

- To identify and evaluate existing loadbalancing techniques implemented in SDN field network infrastructure.
- To gather information about SDN based loadbalancing to create new technique.
- To identify loadbalancing approach which suitable for SDN and design simulation system for new technique.
- To implement and analysis network get packet flow, response time and check how load is balancing with time and speed.
- After the implement compare with other loadbalancing approach and compare security performance with existing approaches and produce ideas to mitigate threat.

1.3 Description of the artifact

In IT industry, there are so many SDN based system and approaches available. Nowadays SDN uses and need in versatile area such as Radio network, 5G, AI, Mobile network, IOT, ML and Wireless LAN and WAN etc. Actually, researchers and IT professionals found there are major issues in SDN research area, which are scalability, load-balancing, security and Quality of service (QoS). In this SDN base load-balancing project researcher focus about load-balancing and security with the use o controller in cloud computing. In this complete project, researcher use least packet algorithm to avoid the above-mentioned issues. For project succeed, researcher used Mininet, Wireshark, MiniEdit, XTerm, POX and Oracle Virtual Box.

1.4 Structure of Report

This SDN based load-balancing in cloud computing with security enhancement thesis report divided into six key parts, each of which is organized to provide the reader and other researcher with a clear understanding of the report's contents and logical flow. Other than Title page, abstract, acknowledgement, dedication, table of contents, list of figures, list of tables and list of

abbreviation. Actually, six chapters will make up the report's outline. Each chapters discusses same ideology with different topics.

1. Introduction
2. Literature Review
3. Methodology and Design
4. Implementation
5. Testing and Evaluation
6. Results and Discussion
7. Conclusion and Future work

In the first chapter researcher discuss about project background, aim and objectives and description of artifact which means describe and discuss about project and why researcher select this topic. In second chapter researcher discuss about literature review which means explaining about related existing project according to their research papers, conferences proceedings, journal articles, books, websites and interviews from industrial experts. Around 70 resources were investigated to achieve the goal. Also, discussed existing technologies used in cloud computing and SDN in IT industry and background of SDN based load-balancing in cloud computing. Generally, focused on similar system developed by researcher. In chapter three researcher discussed about methodology and design. In this section researcher explain how project will be managed and designed according to a time period. The schedule will be generating using a Gantt chart and a Work Breakdown Structure (WBS). This segment will also discuss requirement gathering and specification them. Especially explain planning methodology used by researcher which is prototype. Also, primary and secondary data gathering and analysis. Furthermore, it includes tools and techniques used by researcher to achieve the focused project.

In implementation chapter researcher discuss about technologies like virtualization, Mininet, POX, Wireshark and remote controls. Even more, Linux commands, functions, error handling and troubleshooting discuss via aid of screenshots. Also, include development aspects, language and tools used to complete the load-balancing project. In addition, with the help of programming code, the implementation did take place will be demonstrated. In chapter five, testing and evaluation discuss by researcher. Actually, test cases and the evaluation results of the SDN based load-balancing concept-based system. In result and discussion chapter explain about importance of the

load-balancing system, by illustrating system's screenshots of the accurate interface. Also, discuss about system's weakness, reliability, integrity, struggle experienced by researcher and accuracy. In chapter seven determined the conclusion and future work of the system. The goals and targets that were met would be outlined in greater details here, along with the perception of the SDN based load-balancing project. This chapter also covered limitation and advantages of system and future work of the project expectation.

The references and appendices sections will also be covered in this thesis report. The references used to compile a thorough literature review for this load-balancing project have been mentioned in the reference chapter using Harvard referencing format. In addition, all supporting resources used in the system's implementations are contained in the appendices which are project poster, survey form and Gantt chart etc.

2 Literature Review and Market Research

2.1 Literature Review

Remember the days when to access stored files we had to bring disks, USBs and hard disk drives? Well, it changed times, and how! The way companies organize, scale and process large-scale applications and extract value from data has been revolutionized by cloud computing. The International Data Corporation (IDC) estimates that global expenditure on public cloud services and technology will hit $160 billion in 2018, up from 23.2% in 2017. It also estimates that spending will hit $277 billion in 2021 at a 21.9% percentage compound annual growth rate (CAGR) (trianz, 2020). Actually, cloud computing technology is gaining so much attention and demand. It is very rapidly being adopted in many organizations. A real and perceived lack of protection is one of the main obstacles to the cloud. There are several topics in cloud computing research that can be taken further to get a beneficial production. The research areas are Edge computing, Green cloud computing, cloud cryptography, cloud analytics, cloud scalability, service model, Big data, cloud security, Load balancing, cloud deployment model, mobile cloud computing and cloud computing platforms. In this proposed research we choose load balancing and security. As well as provide instruction to how we implement SDN on it.

2.1.1 Challenges in cloud research area

As the popularity of cloud has grown, it has the major influence on the cloud data centers due to millions of requests which arrive at a time this strengthens the need for the addition of emerging networking resources, services and tools, if it not available at requested demand time. When the load is not precisely balanced, this leads to the incorporation of new services from network, which is mismanagement and wastage. Therefore, there is also a need to balance the load on the links for contact, which means communication links. Furthermore, the management of such large data sets calls for several approaches to simply and streamline practices and should provide users with acceptable levels of efficiency therefore certain part of the cloud need to be investigated. Improving the use of data and uploading for the consumers result. So, load balancing and task scheduling are major issues in cloud computing industry. Task scheduling is the mechanism by which incoming requests (tasks) are organized in a certain way so that the resources available are properly used (Gawali, M.B., Shinde, 2018). In Dhumal's research they said, since cloud computing is the technology that provides internet-based services and resources, which users must

submit their online requests (Dhumal, 2020). Load balancing algorithms have been widely studied in different ways. Nonetheless, with cloud environments, certain environments there are additional issues and they must be tackled and gave solution to the problems.

Algorithms for load balancing are categorized as static and dynamic algorithms. Static algorithms are suitable mainly for highly stable environments and can capable of creating great outcomes in these settings. Nonetheless, they are usually not versatile and not capable of matching the dynamic changes to the parameters with the execution time. It is more versatile and provide an understanding multiple kinds of parameters in the system all subsequent to at run time (Rimal, B. Prasad, E. Choi, I. Lumb, 2009). These algorithms help to get better result but some gives lack of ineffectiveness.

In this paper we investigate existing load balancing algorithm which suit to cloud environment. Before the review we need to investigate main issue and challenges in cloud and how that algorithm impact to the area and performance. So first we discuss challenges in cloud load balance approach.

2.1.2 Challenges in load balance cloud computing

A significant number of tools consist of cloud computing. Organizing these resources and allocating requests from users to relevant resources needs legitimate scheduling and relevant formats in research area (Calheiros, Rodrigo N, Rajiv Ranjan, Anton B and C AF De Rose, 2011). Therefore, before actually jumping into the load balancing approaches should take into account complication and issue that might occur because of the nature of processes. Several problems can be experienced when designing the optimum load balance technology techniques. The goal of this study is to address the challenges to be faced when proposing the technique of appropriate load balancing. Few research approaches are developed for the intranet only where nodes can be strongly marked and it is possible to prevent communication delays (Randles, M., D. Lamb, A. Telab-Bendiab, 2010). The major obstacles are in the load balancing method design, which handles the load across the network infrastructure. In this case to reduce the response time, increase start making, respond to fault tolerance and reach great levels of individual requirements availability, some techniques should operate correctly with the diffusion of cloud servers. Generally, cloud platform contains a multiple range of virtual machines. Every VM may be capable of same dimension, which they are computing power, bandwidth, storage and memory. These

homogeneous virtual machines are called setup. Depending on the dimension this parameter different from cloud platform layout, this contributes to heterogeneous virtual machines. The response time of each respond case generated by users would rely on the capacity of virtual machine, which request is assigned. Since a large number of cloud users and vendors, which user can perform task in cloud. As well as there will be vast number of VM. Some task needs to maintain the restriction of context while maintaining the limit of context in the implementing cloud. Almost all load balancing strategies should be kept in mind prior to assigning response and request to the virtual machine. It plays major impact in that area in the overall performance and management.

The biggest challenge is that it is very difficult to develop an optimal load balancing technique. To meet all user requirements in order to enhance each metric in load balancing. It is only a slight exchange between various factors to satisfy the same factors specification of cloud platform users (S. Abed, D.S. Shubair, 2018). Many users would be interested with much less reaction time over managing the priority of execution of request. However, some users may demand strict order of execution as per the priority of response over between them. Some users are concern about enhance the make-up system and high level of availability. Therefore, certain exchange should be made in order to produce the more acceptable and available results. To enhance the load balancing technique between available factors and improve its best, also improve the effective load balancing technique to response period and manage the utilization of services. In the industry there are different kinds of complex load balancing algorithm available based on time with the output and the number of assignments they need to do while execution of services. These algorithms contribute to additional time for processing at more complicated operation. In addition, enforcing such load balancing techniques we need more control, monitoring service, overhead coordination and delay actions which help to bottleneck that reject overall system performance. Techniques for load balancing should be constructed with worst case scenarios considered. If a node fails, for example how can a load occur? if the task longer than average time to complete. How can a load balancing technique move such a job to another node, if necessary, for execution? So, these challenges made huge impact on Information technology and communication industry.

There are static and dynamic load balancing algorithms. Static algorithm delegates the tasks to the nodes that depends only on the node's ability to process new nodes solicitations. The approach is based exclusively on previous understanding of the resources and capabilities of the nodes. These

will encompass nodes communication performance. They usually don't recognize dynamic changes of these characteristic at run time. Also, unable to adjust or adapt during runtime to load changes. Dynamic load balancing algorithms provide control of the various features of the nodes' capability and network bandwidth in the network. It relies on mixture of awareness based on the knowledge obtained abouts the nodes in runtime resource collected and selected nodes for process mission. It is assigned the task and reassigned dynamically based on calculated and gathered information. So, it us harder implement become at progress of task we can't constantly monitoring the network. However, they are more precise and may result in more effective balancing of loads. In this proposed project also consider the static and dynamic load balancing.

2.1.3 Why SDN involved in Information and communication industry

Generally, Software Defined Networking (SDN) offer centralized and opened method for network management by separating both control plane and data plane. Actually, controller act as software, which responsible for control plane. So, if there is too much load, it means traffics on a particular controller, its suboptimal will occur issue or congestion in network infrastructure. Due to this problem, the network will offer facilities in an improper way. In this case OpenFlow offer a route to communicating switches with multiple controllers so that it is possible to balance the load of the controllers. So, vendor-based manufacture devices are not providing issues because its programmable and flexible.

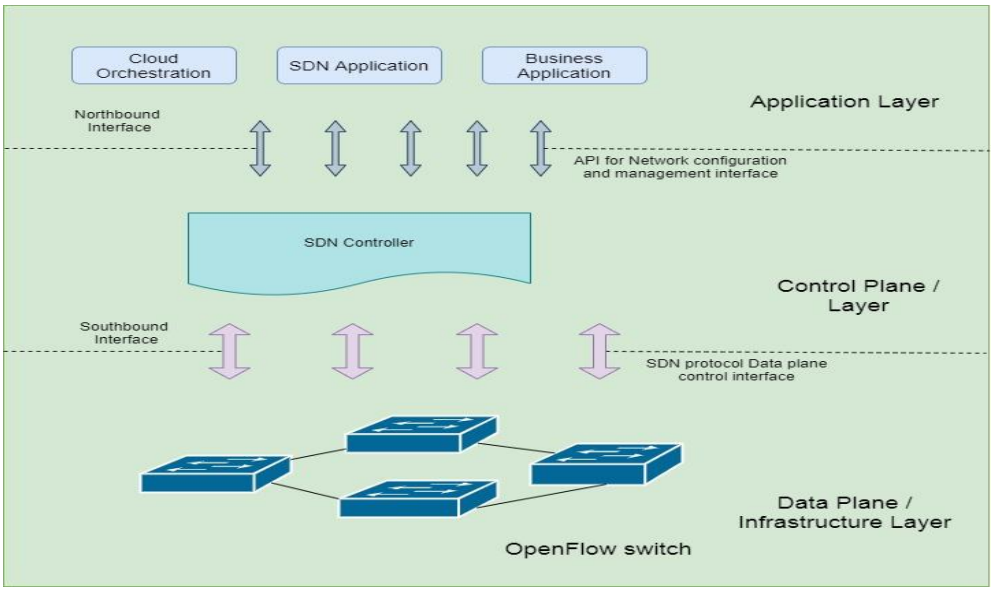


Figure 2-1: SDN Architecture

Open Networking Foundation (ONF) Specified Software Defined Networking (SDN) has recently years been announced and emerging methodology that opens the route for virtualization the network infrastructure and services in a demand way of information and communication technology (Open Networking Foundation (ONF), 2013). This gives an abstract idea to the implementations entire system, which lives in upper layer. Traditionally the network equipment such as switches, servers and routers have control, data and management plane although the principle of control and data plane is independently decoupled. Logic for the control plane is implemented as a part of software that exists in the network devices they are situated in the server, which is in data plane. The decoupling of control and logic of the data plane has changed the network services and resources into automation, programmable, data analytical, security monitoring and network monitoring. As well as it is highly scalable, available, versatile and flexible network infrastructure based on business needs and user's requirements at workplace. In addition, mainly SDN substitutes for the features and functionality of networking devices as just forwarding appliance (OpenFlow Consortium, 2013). Some question come over user's mind, which is to forwarding traffic or packet in control plane how and where it functions, so we can come with simple answer, controller. Basically, control plane has logic. It is implemented it in software program called controller. OpenFlow is the protocol for communicate between the network devices and controller (N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J., 2008). Most popular software defined networking controllers are Floodlight

(Floodlight, 2013), Beacon (Beacon, 2013), NOX (NOX, 2013), POX (POX Manual Current documentation, 2019) and OpenDayLight (Floodlight, 2013), which are market trending controllers some are working on windows or Linux, but some are working on both like OpenDayLight. In this proposed project we use POX because it is python based and all open-source operating system distros like Ubuntu and kali Linux are support to that controller.

McKeown et al. suggested the first and foremost SDN standard known as OpenFlow (N. McKeown et al., 2008) in 2008 at Stanford University. The OpenFlow protocol plays a critical role in the SDN architecture; it enables network creativity and has one or more flow tables used in switches and routers for redistributing data forwarding (ONF, n.d.). Flow tables have flow entries in each OpenFlow switch. In flow entries each entry determines the manipulation and forwarding of incoming packets to the target location in network infrastructure (N. Joshi and D. Gupta,, 2019). Based on algorithm implemented and usage in distributing network traffic, load balancing in SDN based networks can be classified. The shortest algorithm is a random load balancing technique (W.Prakash, 2019), which randomly forward traffic between servers without taking any quality of services (QoS) factors into account. There are several drawbacks to this method, a one server can be overloaded. The next technique is to use a Round-Robin algorithm to disperse the traffic between nodes equally (S. Kaur, K. Kumar, J. Singh, and N. Ghumman, 2015). Due to its simplicity this technique is one of the most common algorithms used in load balancing (H. Uppal and D. Brandon, 2010). M. Koerner and O. Kao proceeding in IEEE conference, this algorithm does not take into account any QoS factors on servers that may result in weak connections and low bandwidth links to forward traffic (M. Koerner and O. Kao, 2012). A further common load balancing method is weighted Round-Robin, which allows some servers to get a greater share of total traffic in this algorithm (S. Vyakaranal and J. Naragund,, 2019). Heterogeneous servers, various connection quality or security constraints have been used successfully by this algorithm (G. Tiwari, V. Chakaravarthy, and A. Rai, 2019), but J. Singh proposed, this algorithm need to set the weight manually while the network infrastructure will alter mostly during the runtime in the realistic situation in network (Singh, 2016). Least connection load balancing technique guides the flow to the server with a minimal number of available transactions (Y. Shengsheng, Y. Lihui, L.Song, and Z. Jingli, 2003). M. Elgili suggest, this algorithm transmits load efficiently and evenly between servers and when the number of clients and traffic staircase demands reliable servers (M. Elgili, 2017). The least bandwidth algorithm dynamically distributes network traffic where traffic

forwarded to the server with the minimum utilization network traffic (L. Padilha and D. Batista, 2019).In load balancing, it collects bandwidth information about server and make judgments based on data obtained, so it is far more precise but difficult to enforce. The IP hash (Internet Protocol Hash Load Balancing) is another solution (P. Suwandika, M. Nugroho, M. Abdurahman, 2018). It is testing the incoming packet and compares the IP in the control log, otherwise it will be forwarded to a new server if it is checked and updates the flow table. Also adopts the traffic of IP and may not take into account network traffic server load, so it can lead to inefficient and unequal access.

In addition, the SDN-aided web load balancing mechanism based on server analytics. This algorithm selects the appropriate dedicated server on the traffic of switch port. Mainly it is counted by number of bytes provided and the server's response time. Although the whole research (SD-WLD), increase performance in lower response time, but it cannot be used in service port-based transfer traffic as it does not manage the packet header (K. Soleimanzadeh, M. Ahmadi, M. Nassiri, 2019). (Montazerolghaem, 2019) A. Montazerolghaem recommend Session Initiation Protocol (SIP) load balancing focused on SDN is another algorithm. This technique employs SDN to deliver SIP networks with a modern architecture because that is easy to customize and modify. SIP networks capable of request making operators and servers processing those requests. Based on minimum number of requests, the algorithm delivers traffic between servers as a server load. But it does not manage the size of requested data into account, so it can generate an unequal load in network. Actually, SIP used heavily in Voice Over Internet Protocol (VOIP).

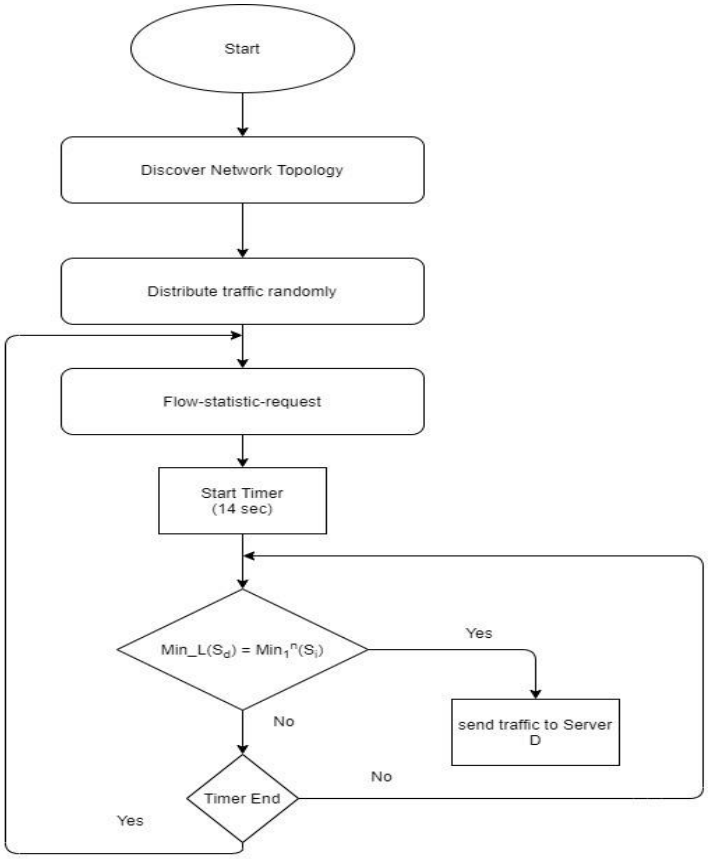


Figure 2-2: Flowchart for project system

In previous methods had lot of disadvantages like packet loss, congestion and some load unbalanced issue. So proposed project work manages the constraints by parsing the packet header and calculating the load on each server based on the destination IP. In heterogeneous and homogeneous servers, the SDN based least packet algorithm approach is effectively used because it defines the existing load on the system. This approach offers a research basis focused on some application like firewall and Denial of services. These techniques interpreted the packet header which helps the network engineer and administrators to establish policy-based information and access collected from packet header.

2.1.4 Security Impact in SDN areas

Software Defined Networking (SDN) has been developed process for creating and manages administering networks, but also has security risk it means vulnerability provided by network. It raised by unauthorized hackers or authorized people, which people target for money or damage company reputation to personal or public reason. SDN offer a range of features that make it easy

to prevention of attacks like DoS, Man-in-the-middle. Also prevent some other attack with basic work. So, in this proposed project we investigate about security vulnerability and how we can mitigate it with secure manner. Also compared existing security researches for clarification. Some algorithms can easily beat by hackers. That was a challenge to make the system efficiently.

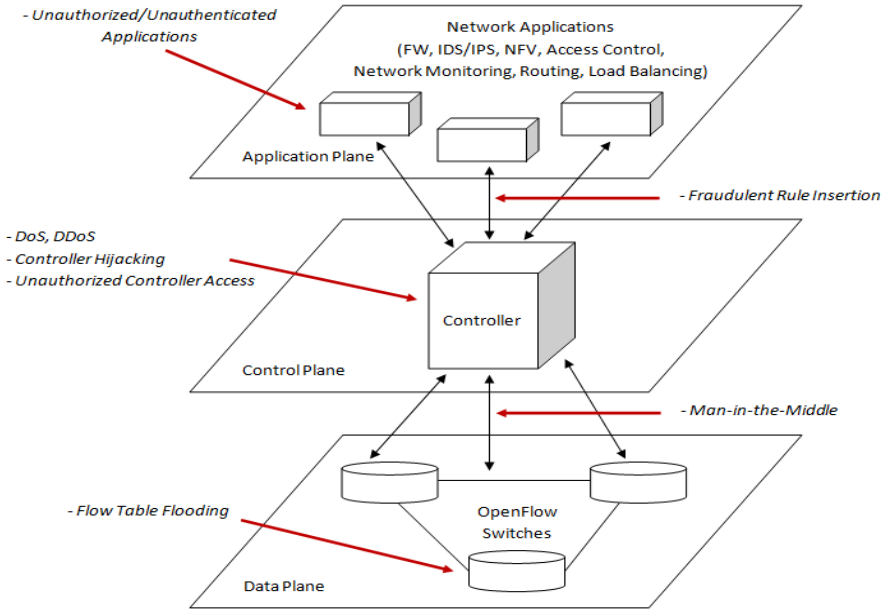


Figure 2-3: SDN security threats and attack surfaces (Muhammet Fatih Akbas, Enis Karaarslan, Cengiz Güngör, 2016)

2.2 Research Gap/Conclusion

Today it is impossible to efficiently do a project or provide a concept-based system to information technology industry without having conducted market research. Market research is a continuous process for collecting or gathering, investigating and interpreting information about a specific project conducted by companies or individual. Before do a project implementation we must investigate the topic environment, which means how trend or famous it is in industry. Also, analysis about how companies adopt the concept for their day-to-day process. Market research making analysis all information and assets, which is about including in particular area, customer, companies and competitors. In order to achieve successful operation for output and target the audience. In this proposed project we analysis key topics, which are related to the project. Topics are SDN, OpenFlow, cloud computing, network security, load balancing and controllers. We conduct questioner Google form about SDN based load balancing approach how impact in

information technology industry and how people knowledge, understand and familiarity with automation side.

2.3 Market Research

Today it is impossible to efficiently do a project or provide a concept-based system to information technology industry without having conducted market research. Market research is a continuous process for collecting or gathering, investigating and interpreting information about a specific project conducted by companies or individual. Before do a project implementation we must investigate the topic environment, which means how trend or famous it is in industry. Also, analysis about how companies adopt the concept for their day-to-day process. Actually, market research making analysis all information and assets, which is about including in particular area, customer, companies and competitors. In order to achieve successful operation for output and target the audience. In this proposed project we analysis key topics, which are related to the project. Topics are SDN, OpenFlow, cloud computing, network security, load balancing and controllers. We conduct questioner Google form about SDN based load balancing approach how impact in information technology industry and how people knowledge, understand and familiarity with automation side.

According to survey majority of student know and familiar with SDN, load balance and cloud computing. Also, Network engineers, Software engineers and Data Scientist familiar and give their feedback and experience with SDN. University of Moratuwa senior lecturer Ajith Pasqual shared his experience via social media. His opinion is in SriLanka, SDN based projects are very rare and it is high cost for Sri Lanka's economy but in Europe and Emirates they use much more in their work process. Especially in Emirates their major business is power plant. So, if there are autonomous it increases their product easily. Also, he mentioned mobile networking, hybrid cloud, 5G, AI and ML are automating via SDN and SD-WAN.

In 2021 the global SDN market value was expected to reach 13.8 billion U.S dollar in size. In 2013, it was 0.41 billion U.S dollar (Statista, 2017). So, 2013 to 2021 SDN market increase rapidly, which means lot of corporate companies, private and government sectors use SDN for their work process because it is autonomous. In market there are lots of controllers to control network. POX,

Beacon, Cherry, OPENDAYLIGHT and floodlight are example for controllers. POX is an open source OpenFlow/SDN controller based on python. For quicker development and prototyping of new network applications, POX is used in IT industry. With the Mininet virtual machine, the POX controller comes pre-installed. So, in the proposed project we use POX for control. We can transform dumb OpenFlow devices in to hub, switch, load balancer, firewall devices using the POX controller. The POX controller makes it possible to run OpenFlow/SDN experiments quickly. According to actual or experimental topologies, POX can be passed on various parameters, allowing you to run experiments on real hardware, testbeds or Mininet simulator (Sukhveer Kaur, Japinder Singh, Navtej Singh Ghumman, 2014). In market NOX controller also use because it acts as a forum for network control and offering a high-level programmatic interface for the management and creation of applications for network control. One of the largest open source SDN controller is OpenDaylight, which is helping to lead this transition. According to our survey 60% people familiar to work with OpenDaylight. Because it is Linux based. It is flexible and open framework for networks of any size scale to be personalized and automated.

3 Methodology and Design

3.1 Methodology

In this project we used prototype model for go forward the project successfully. Actually, prototype model is a software development phase, which is designed, tested and reworked until an appropriate acceptable prototype is accomplished. It is an iterative process of testing and error that takes place between creator and user. As well as in this project we divided by version. In first version design the network according to gathered requirement and implement it via software. We monitor outcome and investigate its flows. Also identify errors and modifications. After we move to second version. In that case same as first version but with the modification identify at that stage. If the error occurs in second stage, we move to version three. So, this process iteratively forwarded until good solution occur in the network.

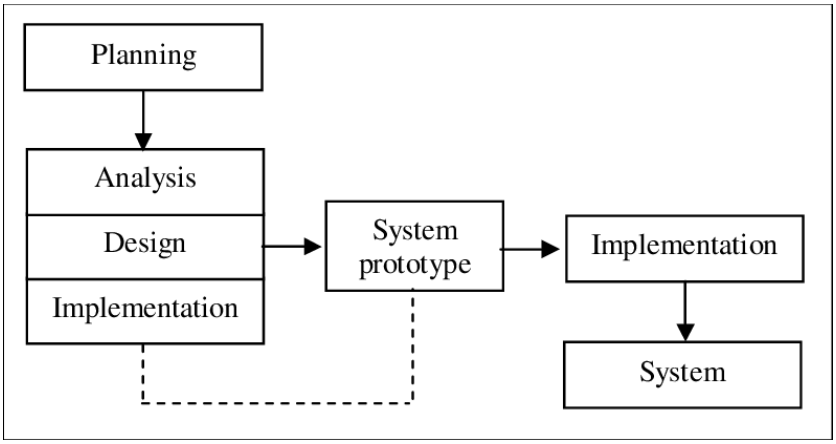


Figure 3-1: Prototype model (Wahab, 2009)

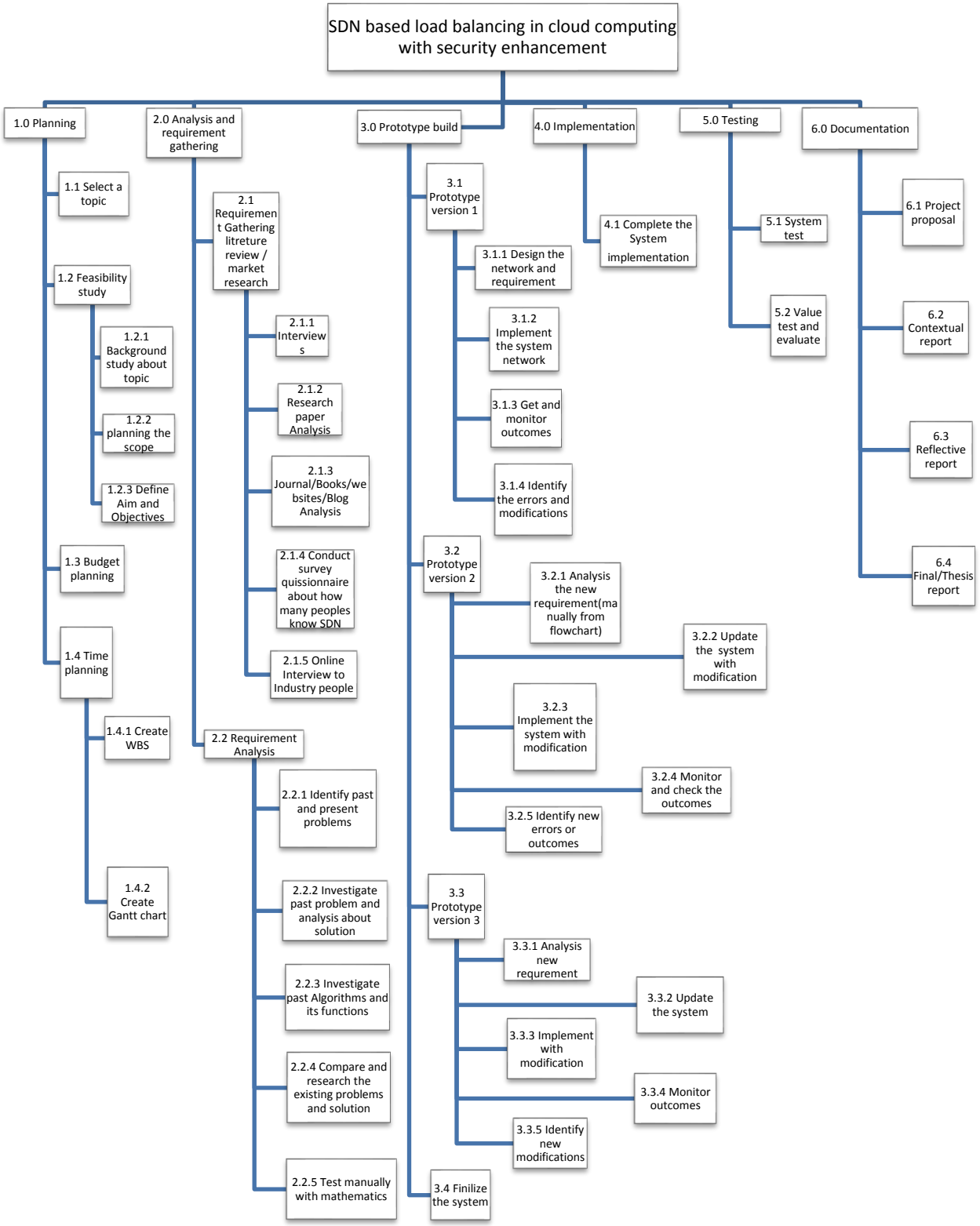
3.2 Planning

In industry, government or private businesses or small businesses necessitate planning, to increase their sale, achieve target and earn high profit, success and reputation. In simple term we may assume, planning bridges the distances between where we are now and where we want to reach the target. Also keep in mind plans are always generates fixed time allocation as business cannot go on planning continuously. It fulfills project aim and objectives with the time limit. Usually, planning is important when beginning a project. As well as it is one of the primary components in a system design and implementation project. Especially Work Breakdown Structure (WBS) and

Gantt chart played significant role in project. Above mentioned chapter of the project plan, Gantt chart is quite detailed with time and periods. The project is divided from beginning to end into all facets of the project. Such as separation will make it easier to finish the job within the duration.

Before research about the project, we studied about current trends in information technology industry. Along with that task we select a topic which is our proposed Software Defined Networking area. After we planned to do feasibility study about topic. In the feasibility study we research about books, journal articles, websites and existing reports. As well as technology topic. After according to triple constraint, we plan and divide the scope. Define aim and provide objectives about proposed. Overall, all plans have a milestone to achieve with required time and beneficial cost.

3.3 WBS


SDN based load balancing in cloud computing with security enhancement
1.0 Planning
2.0 Analysis and requirement gathering
3.0 Prototype build
4.0 Implementation
5.0 Testing
6.0 Documentation
1.1 Select a topic
1.2 Feasibility study
1.2.1 Background study about topic
1.2.2 planning the scope
1.2.3 Define Aim and Objectives
1.3 Budget planning
1.4 Time planning
1.4.1 Create WBS
1.4.2 Create Gantt chart
2.1 Requiremen t Gathering litreture review / market research
2.1.1 Interview s
2.1.2 Research paper Analysis
2.1.3 Journal/Books/we bsites/Blog Analysis
2.1.4 Conduct survey quissionnaire about how many peoples know SDN
2.1.5 Online Interview to Industry people
2.2 Requirement Analysis
2.2.1 Identify past and present problems
2.2.2 Investigate past problem and analysis about solution
2.2.3 Investigate past Algorithms and its functions
2.2.4 Compare and research the existing problems and solution
2.2.5 Test manually with mathematics
3.1 Prototype version 1
3.1.1 Design the network and requirement
3.1.2 Implement the system network
3.1.3 Get and monitor outcomes
3.1.4 Identify the errors and modifications
3.2 Prototype version 2
3.2.1 Analysis the new requirement(ma nually from flowchart)
3.2.2 Update the system with modification
3.2.3 Implement the system with modification
3.2.4 Monitor and check the outcomes
3.2.5 Identify new errors or outcomes
3.3 Prototype version 3
3.3.1 Analysis new requrement
3.3.2 Update the system
3.3.3 Implement with modification
3.3.4 Monitor outcomes
3.3.5 Identify new modifications
3.4 Finilize the system
4.1 Complete the System implementation
5.1 System test
5.2 Value test and evaluate
6.1 Project proposal
6.2 Contextual report
6.3 Reflective report
6.4 Final/Thesis report

3.4 Gauntt Chart

Gantt charts are using to dividing the activities according to the time frame. In the left of the plan is consisting of the actions identified. As well as the top of the chart contains proper time allocation. Each of the action is represented by a bar line as to the timeline. To control the time and work activities, the SDN based load balancing system was created Gantt chart. The Gantt Chart that was attached in Appendix A for future purpose.

3.5 Requirement Gathering and Analysis

Requirement Gathering plays a second role in the Software Project. Requirement gathering are gathered as primary data and secondary data.

3.5.1 Primary Data Gathering – Market research

Primary data gathered using market research. After choose a topic we briefly analyzed on internet and some research papers. After we planned to do a questioner via Google form and share with university students, Network, Software Engineers and Data Analyst. In the form we collected information about how people familiar with SDN, OpenFlow, Cloud computing, load balance and network security. Also, we get respondent's email and LinkedIn, because we can individually connect and get more information about our key topics in proposed project. Therefore, I contact Mr. Ajith Pasqual, who is founder and CEO of Paraqum Technologies (Private) LTD and Senior lecturer at University of Moratuwa SriLanka. He shared his experience in SDN and its based projects his own projects and his student's projects. After thorough information we moved to find from existing reports and journals etc.

3.5.2 Secondary Data Gathering – Literature review

After find primary data we investigate our project topic using the resources such as research papers, gather journal articles, conference proceedings, books, websites and existing system provide by technical people. This research is done to find out what kind of SDN load balancing approaches are used in network or IT industry. Investigate about existing algorithm approach for controllers like Round-Robin, weighted Round-Robin and least connection etc. Also gather and analyze about methodology they are used. What kind of software use to implement the system's testing and evolution. Also, investigate how they planned the system and artifact so on. Also collect

information about what kind of programming language used for achieve the target and develop the system network.

3.6 Design

The design process aims to create comprehensive requirements that focus the physical solution to the IT needs of the customer. Actually, researchers need a good knowledge of the overall research process to design and undertake a research study about their project. For this project generally need a design like topology of network infrastructure. It should perform good idea and blueprint for whole project. It gains very useful for start to end implementation process.

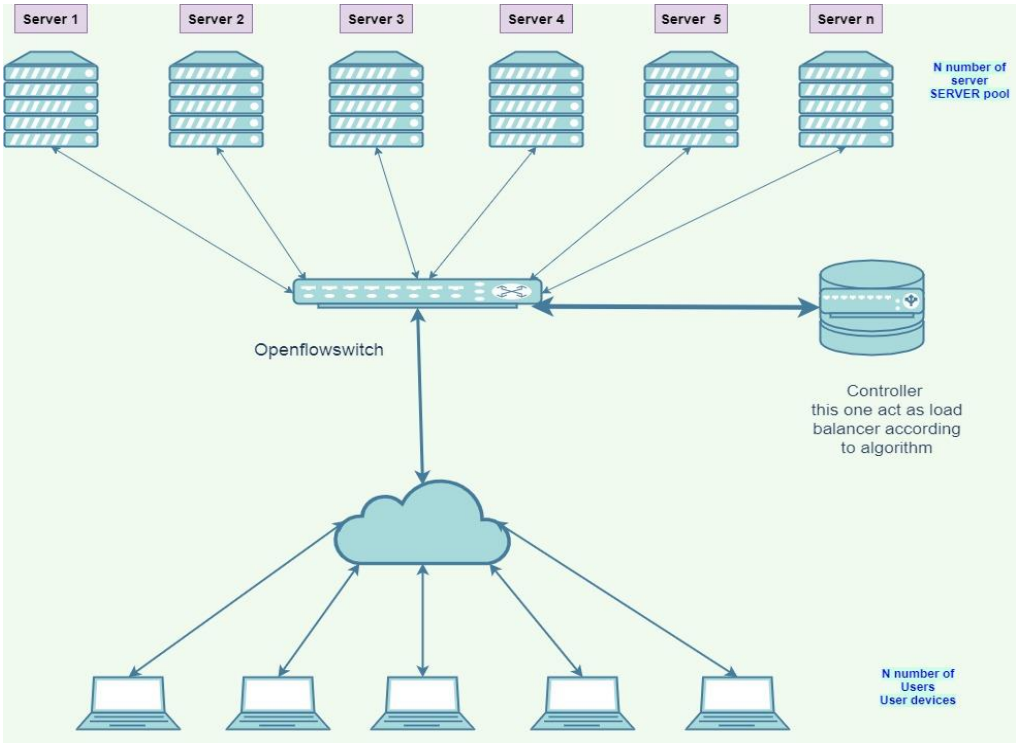


Figure 3-2: Network diagram for project

After the plan, requirement gathered and analysis we need to design the project. Above picture shown about project design, which means it physical components, packet flow and functions. In order to this project design, we have server pool (N number of servers), Users (N number of users), OpenFlow switch and controller. Server pool is help for load the balance storage according to algorithm, Users indicates that, who want to access and use the system, OpenFlow switch is data

switch activated by OpenFlow that communicates and respond to an external controller over the OpenFlow channel. According to one or more flow tables and group table, it handles packet lookup and forwarding. The OpenFlow switch communicates with the controller through the OpenFlow switch protocol and controller control the switch in order to user activities. This project is cloud based project, so all user access from the cloud. Generally, controller is the highlight part in this project design, because load balance can happen or guide from that device according to least packet algorithm.

4 Implementation

In this chapter discusses about implementation process of SDN based load-balancing in cloud computing with security enhancement research project. It implements on virtual environment and used some remote work to successfully complete. Also, discuss about Linux commands and their usage for the project.

For implementation process researcher used virtualizing software called Virtual Box for the environment. First researcher downloads newest Oracle Virtual Box software. It is a cross-platform virtualization application. Whether they are running on Mac, Windows, Solaris and Linux operating systems. Virtual Box create guest operating system called virtual machine. It not damaging changes or conflict to the host operating system. So, researcher install Oracle Virtual Box newest version on Windows operating system. For this project researcher identified three ways to start the project and some ways are not fitted to the host for work process. Download and install virtual box in to windows OS (researcher use Windows 10 Pro).

Option 1 - After, install Mininet VM and import into the Virtual Box. It is speed up Mininet installation. That VM works on Mac, Windows and Linux through not only Virtual Box but also VMware, QEMU and KVM. After download the Mininet VM researcher did the few steps to customize the setup. It is an OVF (Open Virtualization Format) virtual machine image which can be imported by most virtual machine interface. Normally we can import a .ovf file by double clicking it. Sometimes we have experienced problems importing the .ovf file. As a solution we simply install a new VM of the right form and use the .vmdk file as the new VM's virtual hard disk. After select "settings," and then add an alternate host only network adapter for logging into the VM image and start the virtual machine. That Mininet VM contains Wireshark, Mininet and Nmap. It was installed default by providers. Hardware specification not support in some application. So, researcher changes to another approach.

Option 2 – After, install Ubuntu Server and configure basic settings. Server not contains Wireshark and Mininet like Mininet VM. So, researcher installs application one by one after server machine basic configuration. Ubuntu Server was CLI mode. So, researcher installs Tkinter for GUI mode. Researcher not interested for this option because, error happened in some times which means for the machine's hardware specification little bit poor.

Option 3 – After the virtual box installation researcher insert the Ubuntu desktop iso image in to virtual box for get virtual machine. Like ubuntu server, ubuntu desktop also not have pre-installed Wireshark and Mininet. But researcher satisfy with the OS because, its user-friendly interface. So, all implementation done with option 3's idea. Let's see the implementation process one by one.

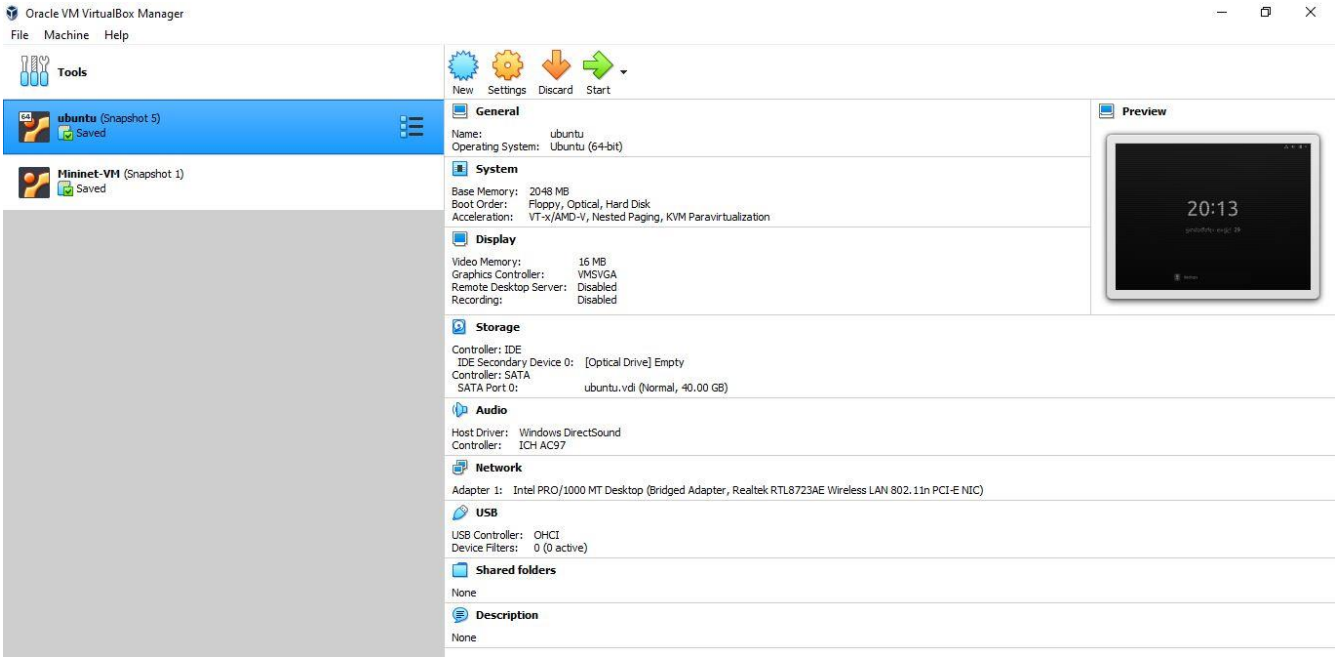


Figure 4-1: Oracle VM Virtual Manager for project

First researcher determines the issue in the installation. Likewise, GUI issues, non-supported command issues and hardware specification issues. So, researcher used option 3 for implementation process. Let's see the commands used for the Mininet installation. After Ubuntu desktop installation researcher opens the terminal. First type

```
$ sudo apt-get install mininet
```

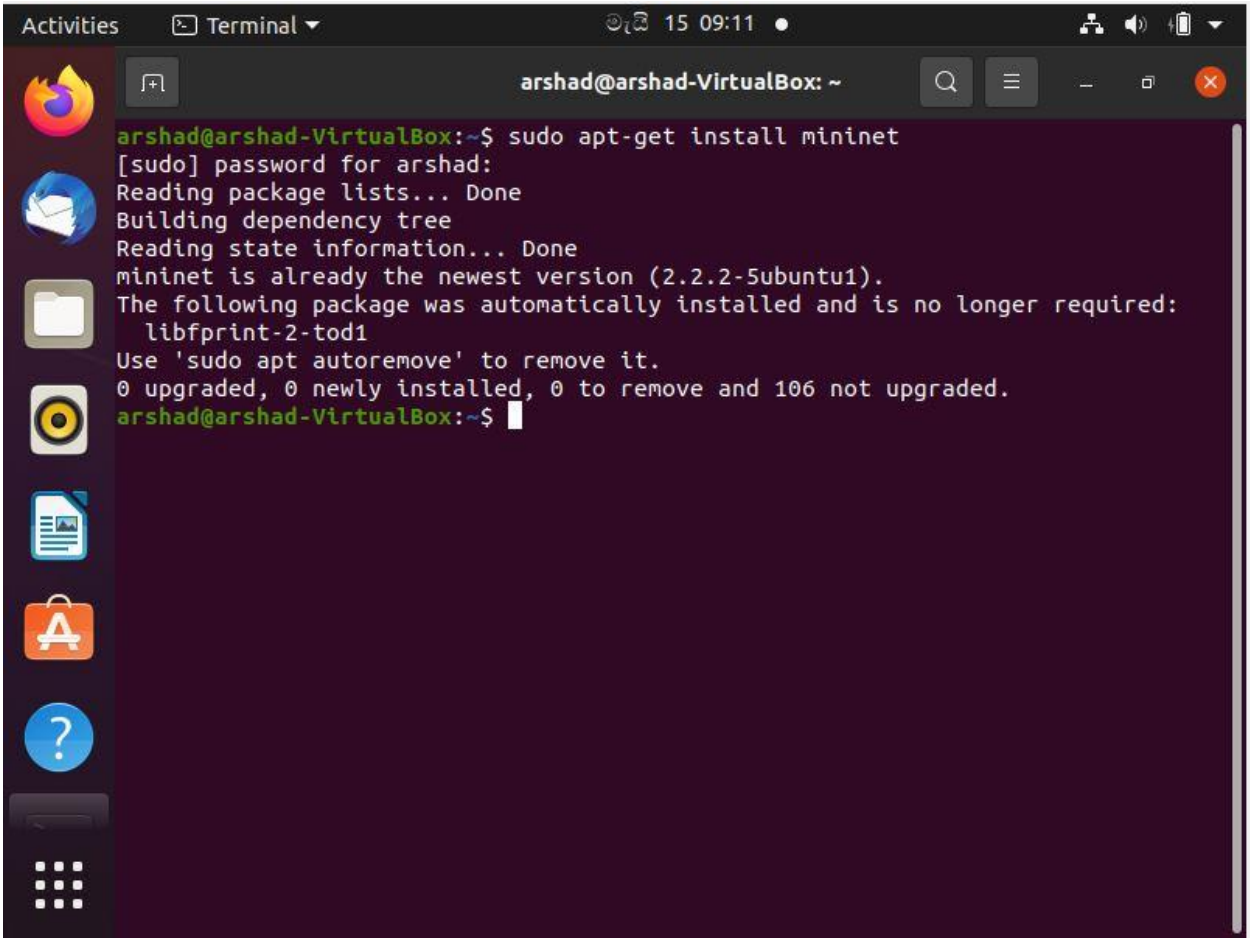


Figure 4-2: Mininet installation command and output.

After researcher did download and install system updates. At this stage researcher experienced storage issue because at default Ubuntu installation it taken 20GB storage for hard disk. When we update the system, it gains more storage space to the system. So, researcher deletes the existing Ubuntu desktop virtual machine and reinstalls it along with 40GB storage. After re install Mininet and update according to below command

```
$ sudo apt-get update
```

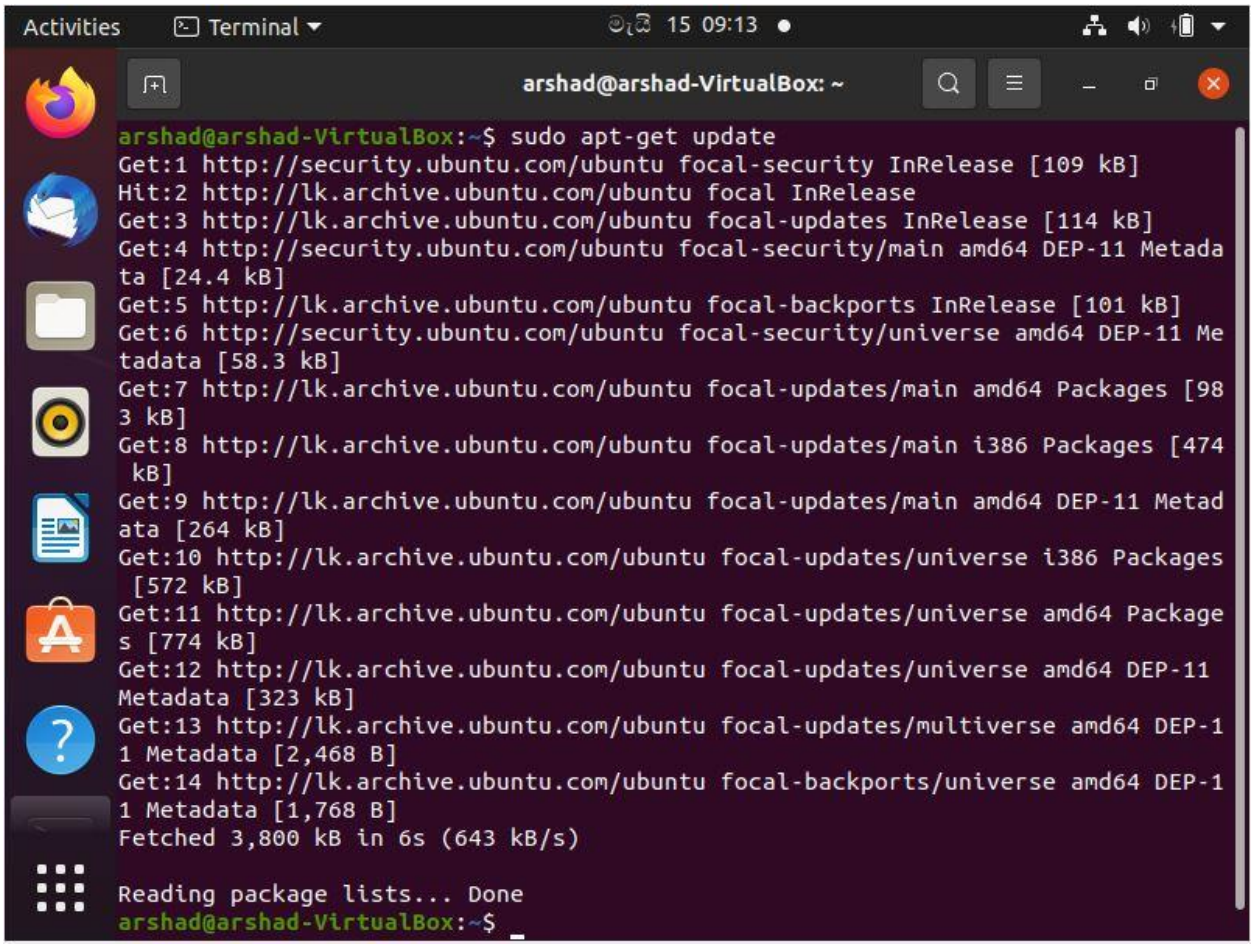


Figure 4-3: System update

This command doesn't necessarily install new versions of software. Rather than, it updates the package lists for upgrades for packages that need upgrading existing one and for new packages that have just been added to the repositories. Downloads and updates package list from repositories to provide information about the most recent releases of packages and their dependencies.

```
$ sudo apt-get upgrade
```

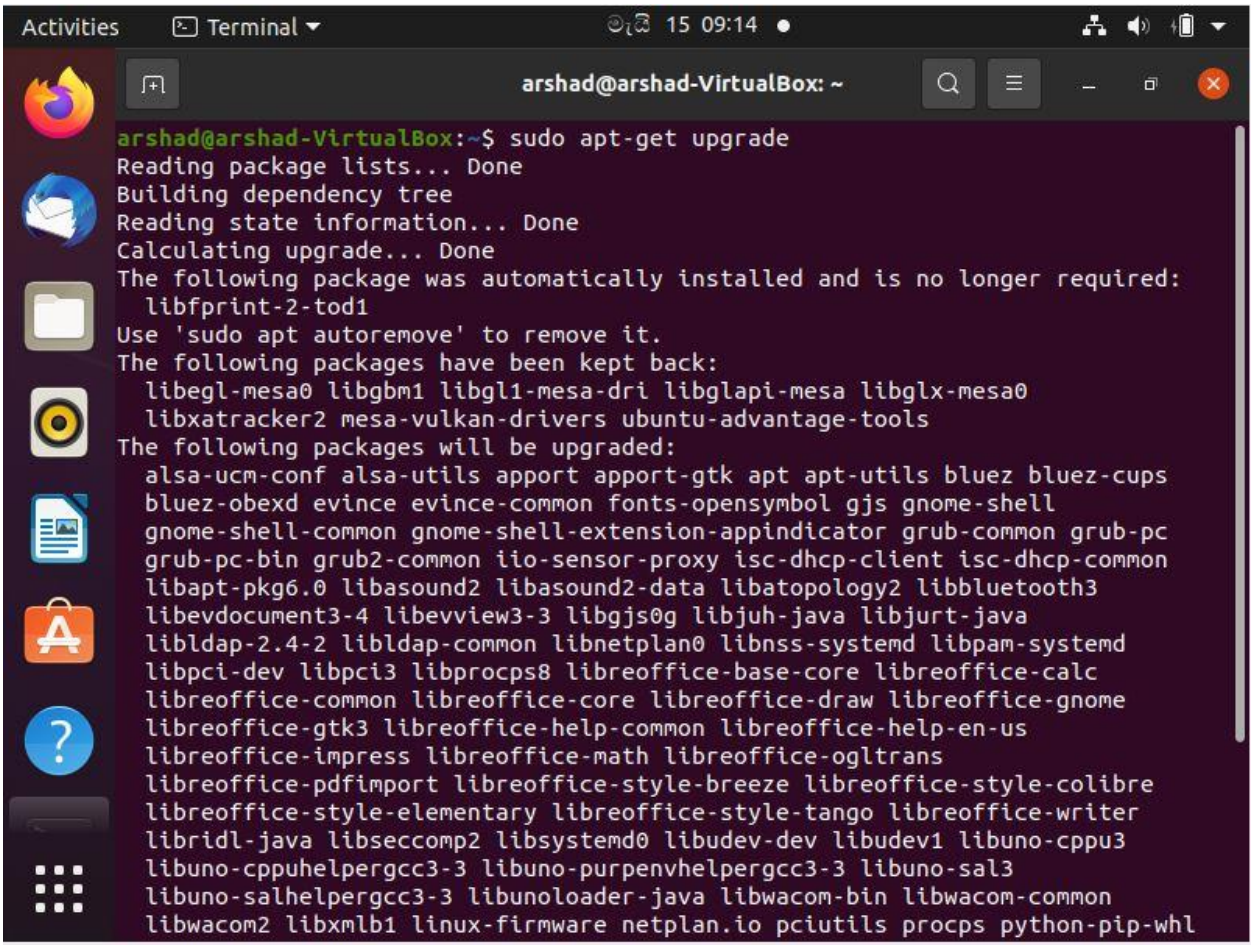


Figure 4-4: System upgrade

This command help to fetch new version copies of packages already installed on the system if APT is aware of them by way of apt-get update.

```
$ sudo apt-get dist-upgrade
```

It is performed same task as apt-get update, but it will also sensibly manage the dependencies. So, it can delete or install redundant packages or add new one from the internet.

```
$ sudo mn -c
```

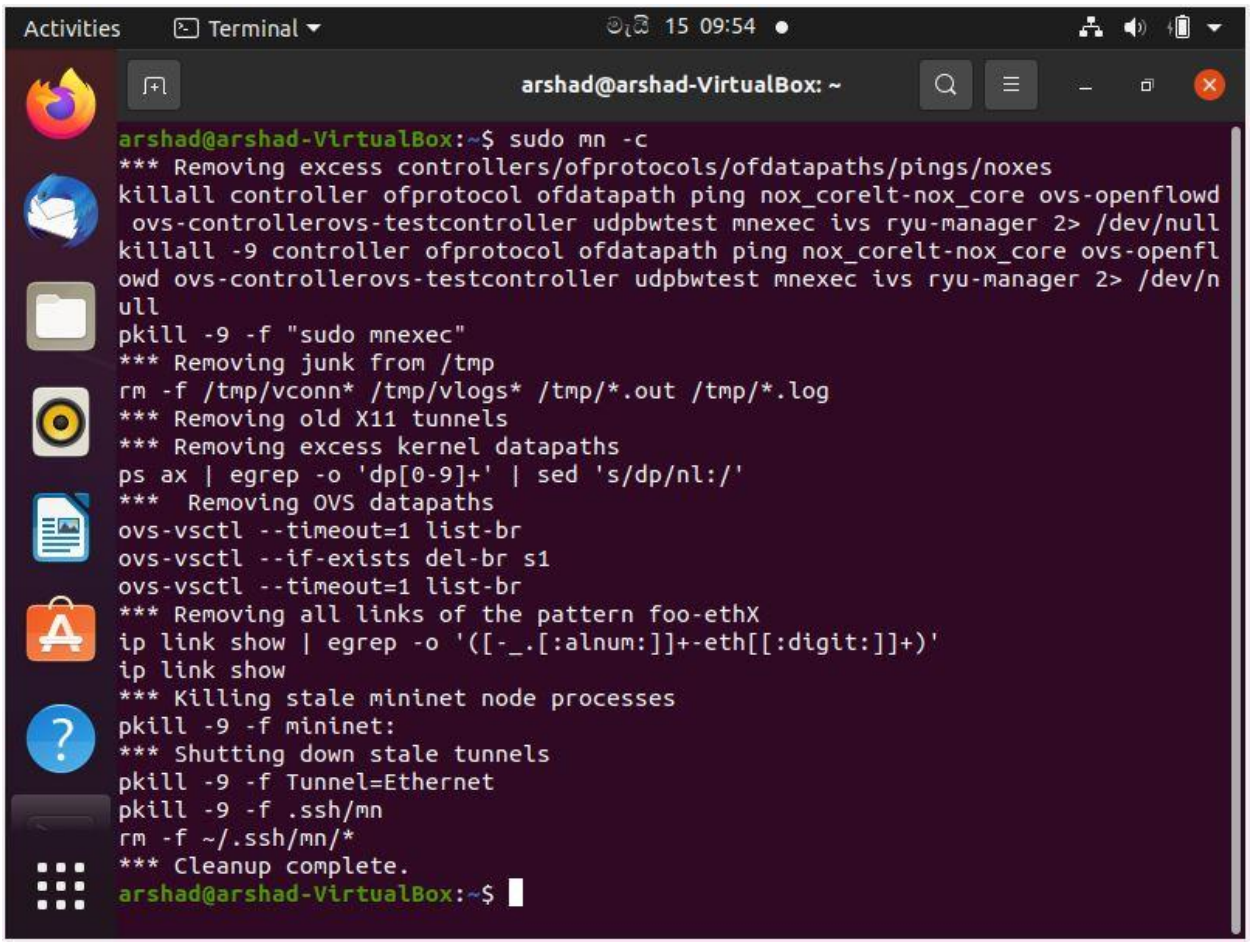


Figure 4-5: cleanup the pre-Mininet

```
$ sudo apt-get install git
```

Git is compatible with the most popular operating systems such as Mac OS, Windows and Linux. In reality, git is pre-installed on the majority of Mac and Linux machines. If we have already installed git just opened terminal and type git version it prompt detailed information about git or it give alert you that git is an unknown command. Git is open source and free distributed version control framework that can accommodate anything from simple to largest projects with ease, efficient, speed and effective. Also, it is simple to understand and use with a light weight and lightning-fast efficiency. Git was created with the aim of versioning the linux operating system but it's only natural that it's simple to set up to run on linux. Also, we should use the package management tool which came with linux release distribution to install git. Git packages provide access using apt. It's a smart thing to make sure you have the most recent update installed which means running the latest version. To do just that, open a command prompt shell and execute the

command to ensure is up-to-date. To install git finally type above mentioned command. Also, it prompt output completely. For verification we can use git version command. In this project researcher use git to download the Mininet 2.2.0 source code.

After researcher clone git source code for Mininet installation.

```
$ git clone git://github.com/mininet/mininet

$ cd mininet

$ git tag # list available versions
```

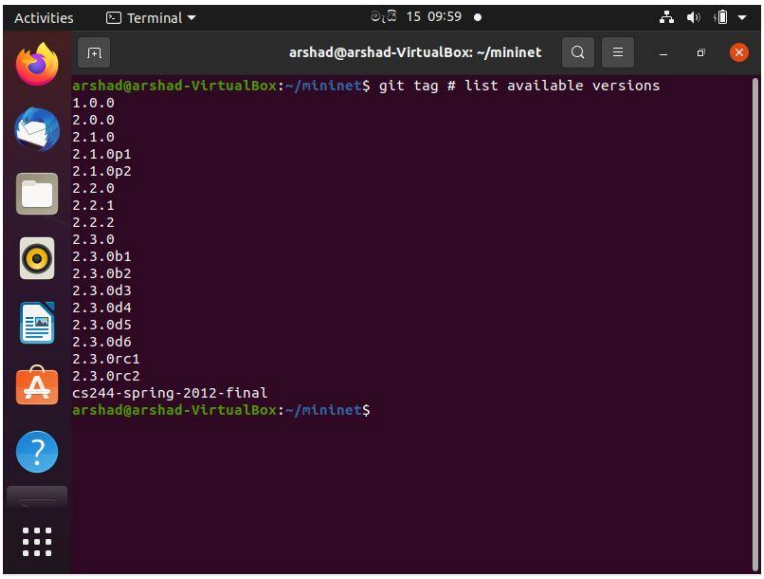


Figure 4-6: List available version

```
$ git checkout -b Lastversion

$ cd..

$ ~/mininet/util/install.sh -a
```

For test purposes researcher run the following command.

```
$ sudo mn
```

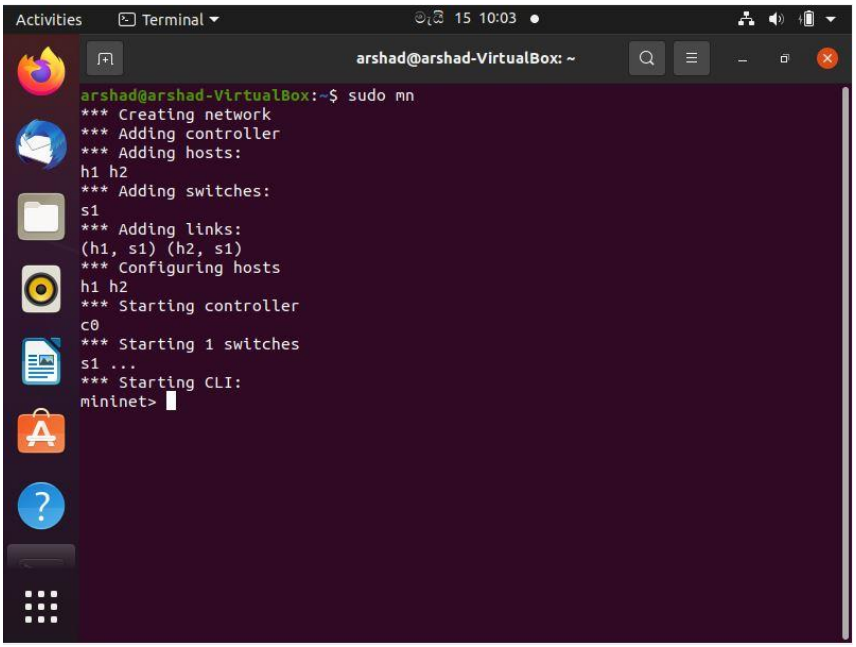


Figure 4-7: testing the Mininet with topology creation

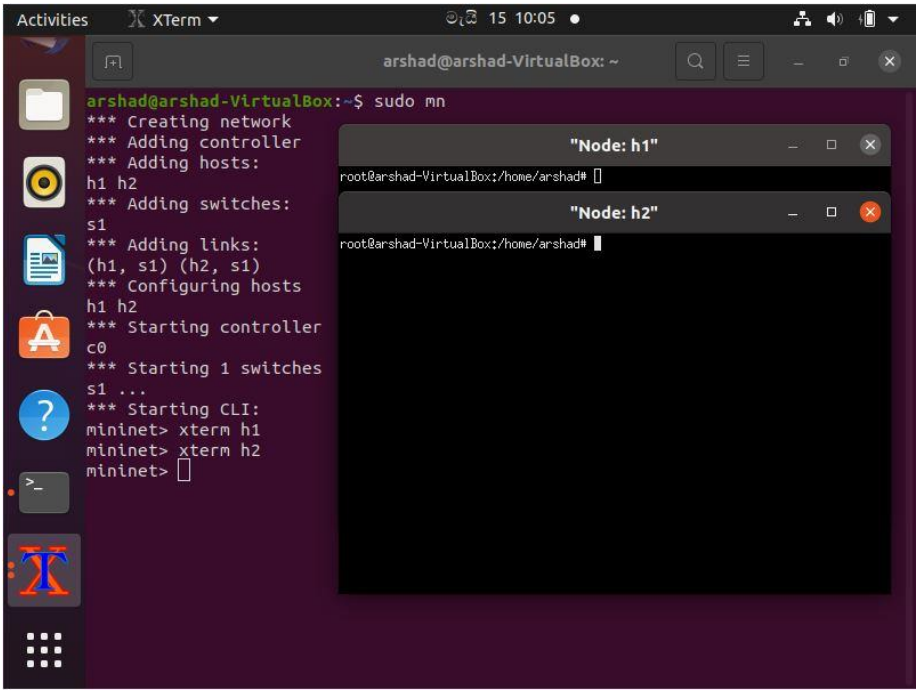


Figure 4-8: Remote XTerm testing the Mininet

After Mininet installation researcher moved to topology creation for project. Actually, researcher create topology using two methods. General method is command method which is not create using devices and cables. Let's see the example.

Other method was creating using MiniEdit. Actually, Mininet network simulator contains MiniEdit. It is simple GUI editor for Mininet technology. Also, it was experimental and concept-based tool conceived to demonstrate how Mininet can indeed be expanded. Mininet's examples folder contains the MiniEdit script. Execute the following command to start MiniEdit. Generally, MiniEdit requires root privilege to execute. So, researcher used to start sudo command for execution.

```
$ sudo ~/mininet/examples/miniedit.py
```

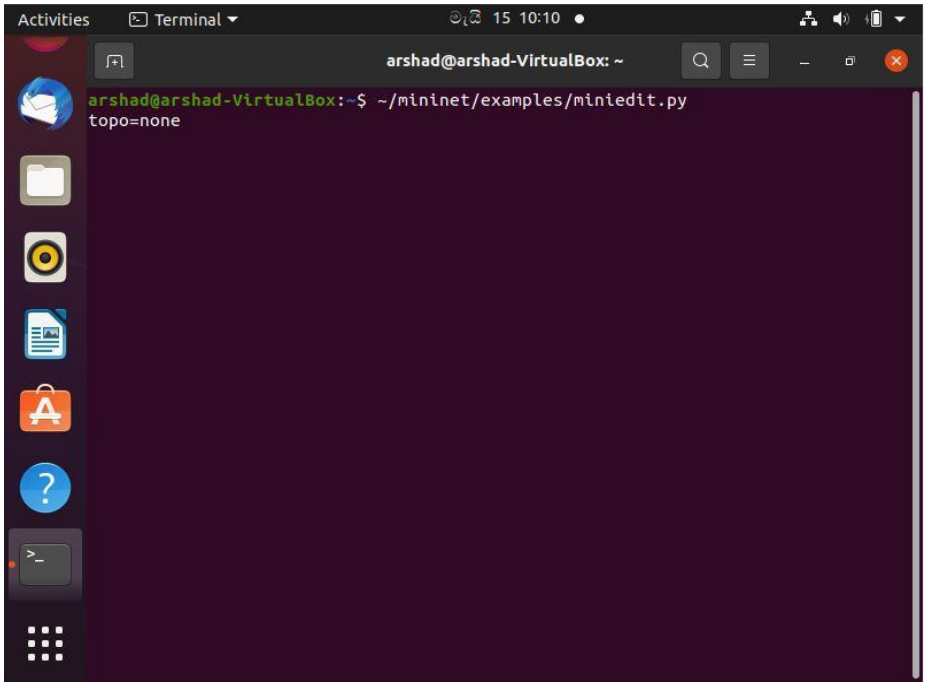


Figure 4-9: Mininet's MiniEdit topology creates command and its output

Actually, MiniEdit has simple user interface. It looks like number of tool icons and a menu bar. Tools are select tool, Host tool, switch tool, Router, controller, net link cable and OpenFlow switch tool. Also, include simulation run state and stop state.

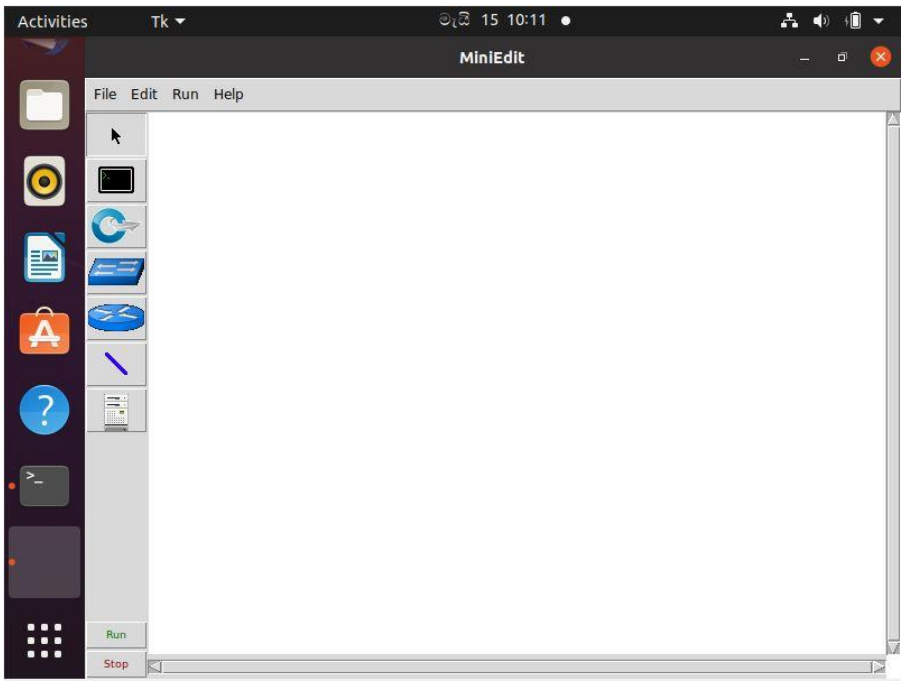


Figure 4-10: MiniEdit interface for system topology creation.

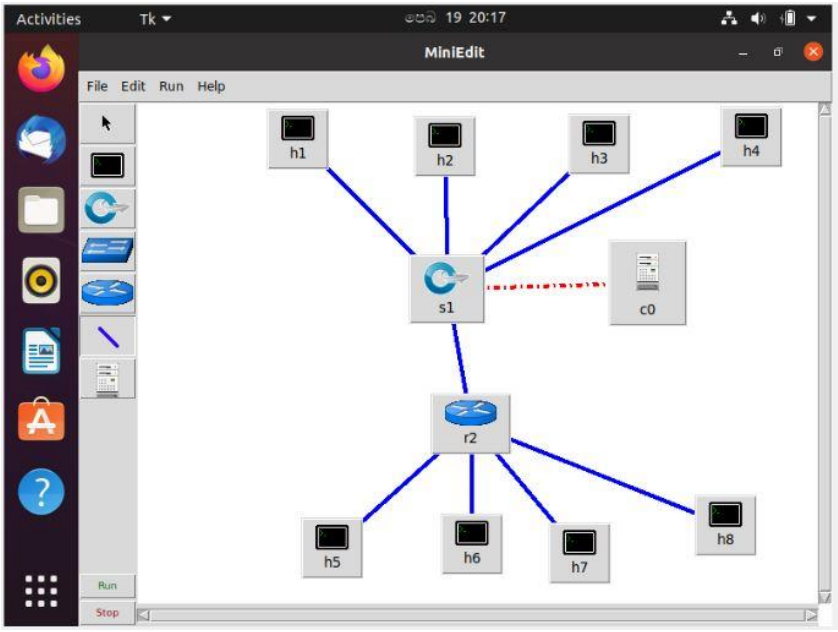


Figure 4-11: MiniEdit topology for SDN based load-balancing system

After the Mininet's MiniEdit topology creation researcher move to the POX implementation progress. In this case researcher demonstrate basic software -defined networking concepts using POX SDN controller, Mininet network simulator and POX component. Actually, SDN based load-balancing project's main portion is controller. Because controller was main concept which means

load-balancing did by controller. For controller researcher used POX. It provides platform for interacting with SDN switches using the OpenFlow or OVSDB protocol. Using the python programming language, developer can build an SDN controller with POX. It is widely used learning and analysis platform for Software Defined Networks and network application or implementation programming. Using the stock components that come with POX, it can be used immediately right away as a standard basic SDN controller. By creating or designing more components, developer can build a more dynamic and complex SDN controllers. Developer can also create network apps that's use POX's API. POX components are python programs that can be from the command line when POX is initiated. In SDN, these modules enforce network features and functionality. Also, POX comes with plenty of pre-installed stock components. The ~/pox/pox directory contains the code for each component. The goal of all SDN controllers and POX are let gave permission to user to code their own applications which use the controller abstraction layer or intermediary between network infrastructure machinery and application of network. Above mentioned option 1 contains POX. It pre-installed on the Mininet VM image. But researcher manually downloads and installed POX according to below steps.

First researcher identifies, working with POX as a git repository is the easiest and best method to go. Furthermore, researcher find another way which is tarball or zipball. But researcher recommended git repository because its popularity and version control efficiency. POX support Windows, Mac OS and Linux. It can be used with the CPython which is standard python interpreter in IT industry. Also, it is support to PyPy. Because, it was faster than CPython. Even more, it has easy portability. Actually, researcher experience POX requirement issue while installation which was Python 3 not supported for the project. So, researcher get help from mentors and post an issue on social media and share with groups for get a solution. After get solution which is remove Python3 and run-on Python 2.6. After its work properly.

```
$ git clone http://github.com/noxrepo/pox
```

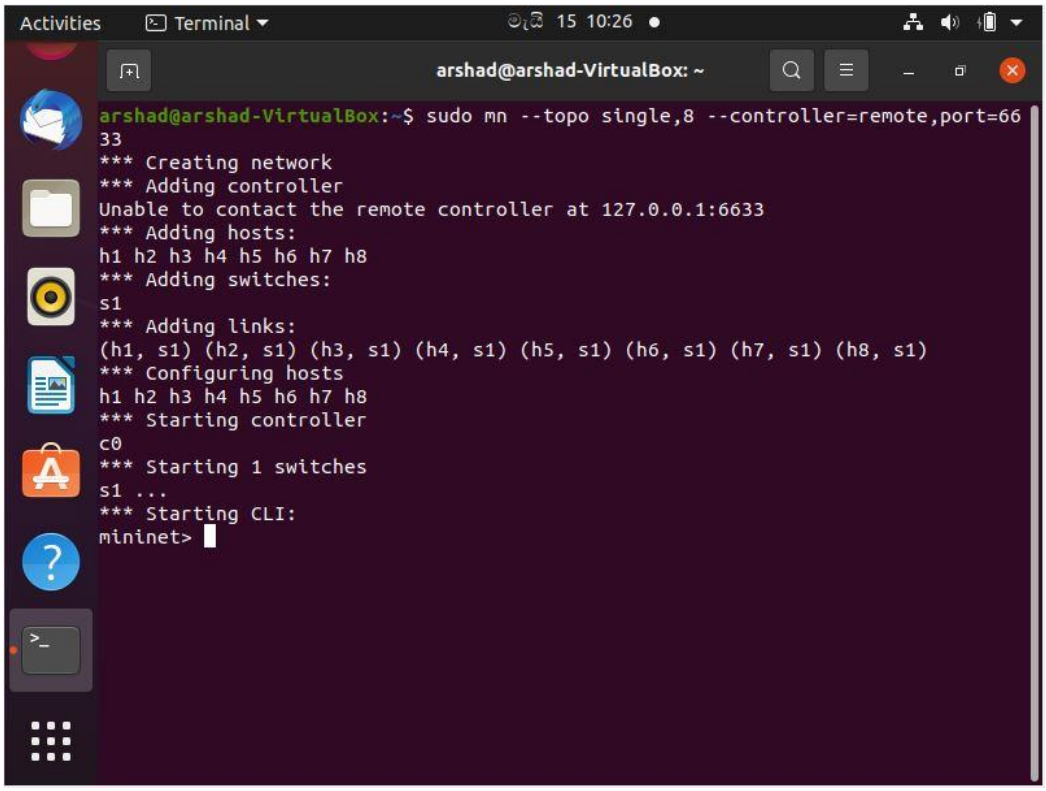


Figure 4-12: Actual topology created via command

After researcher use XTerm command to remotely access the servers, controller and users. In test cases all the access and load-balancing concept thoroughly explain by researcher with the help of system's screenshots.

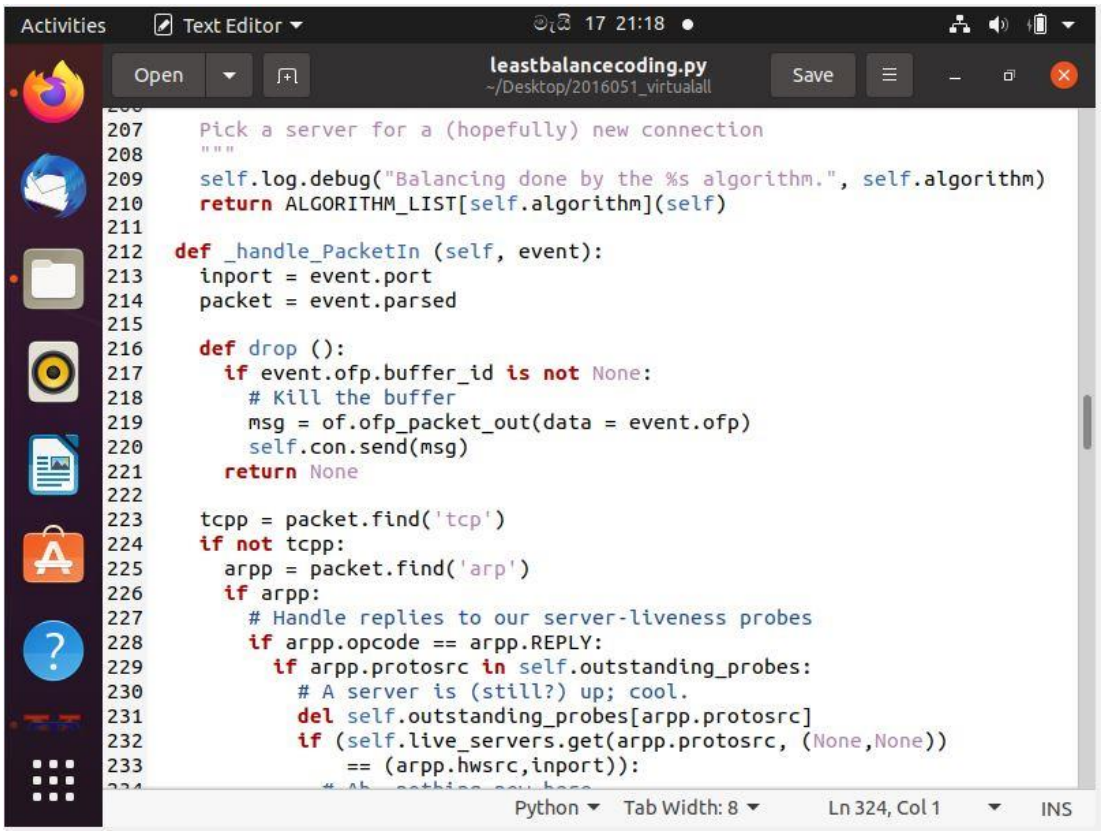


Figure 4-13: pick a server for new connection

In this coding segment, researcher showed the systems' process, which means how system balances the load according to the algorithm. First server is selected the load or number of packets came to that server. Actually, minimum load of selected server equal to the available server, in order to forward the packet to particular server. The OpenFlow connection begins with a handshake between the controller and the switches, during which the controller discovers the switches' presence. The controller uses the link layer discovery protocol (LLDP) packet to figure out how the switches are linked in topology. The controller sends an OPFT-FLOW-MOD message to the switches, instructing them to resend any LLDP received to the controller. The controller would then send a PACKET-OUT message to the switches with LLDP as the payload. PACKET-OUT is resent to all ports except the incoming one by each turn. When a switch receives an LLDP packet, it sends an OPFT-PACKET-IN message to the controller. This mechanism is used to discover topology. Since there is no history of the load on servers at startup and they all have the same load, the POX controller can initiate traffic in a random fashion after discovering the topology like hosts, switches and links. After that, POX sends a "Flow-statics-request" to the

switch every 14 seconds to collect load statistics on each port. Flow-statics-received is how the OpenFlow switch responds to the controller. The flow-statics-received command has been changed to obtain the total number of loads processed by each server. The load balancer then chooses the server with the fewest packets, and the controller prepares the necessary flow rules for the server specified in the switch to send the incoming traffic using the OPFT-FLOW-MOD message.

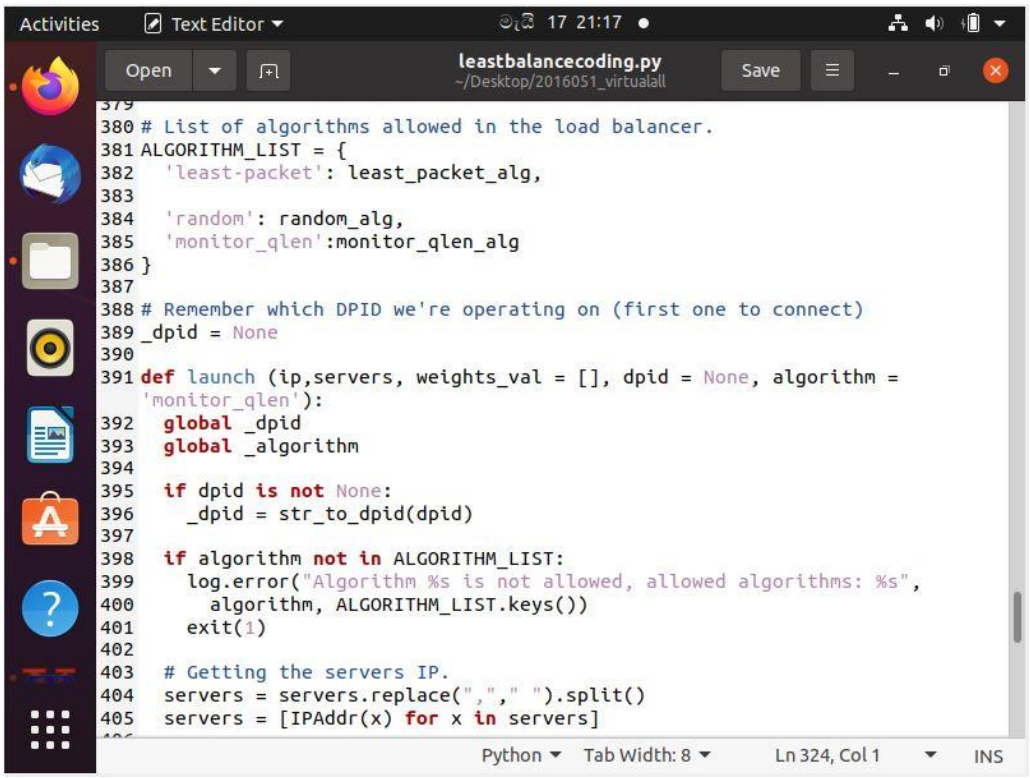


Figure 4-14: List of algorithms

After balance the load according to researcher's concept, after then moved to the analyzing the network flow. Which means researcher main aims and one objective was security enhancement. For this case researcher used Wireshark. Most probably Wireshark is a network packet analyzing tool which appears to be trying to capture network packets and display that packet data in interface. Also, it is open-source network analyzer tool. It was available on UNIX, windows and Mac OS. Usually, it captures live packet data from a network interface, display packets with very detailed protocol information, save packet data captured, filter packets on many criteria, search for packet on many criteria, export some or all packets in a number of capture file format and import packets

from text files including dumps o packet data. Actually, Wireshark interface looks like divided in to three panes which are packet list, packet details and packet bytes. In the first pane, a list of packets in the current capture file is shown. The packet number, time caught, packet source and destination, packet protocol and some general information contained I the packets are shown as columns in a row. The second pane shows details about a single packet in a hierarchical format. Also, to see all of information gathered regarding a single packet, click the "collapsed and expanded" button. The third pane shows a packet in its raw, unprocessed form and contains encoded packet data.

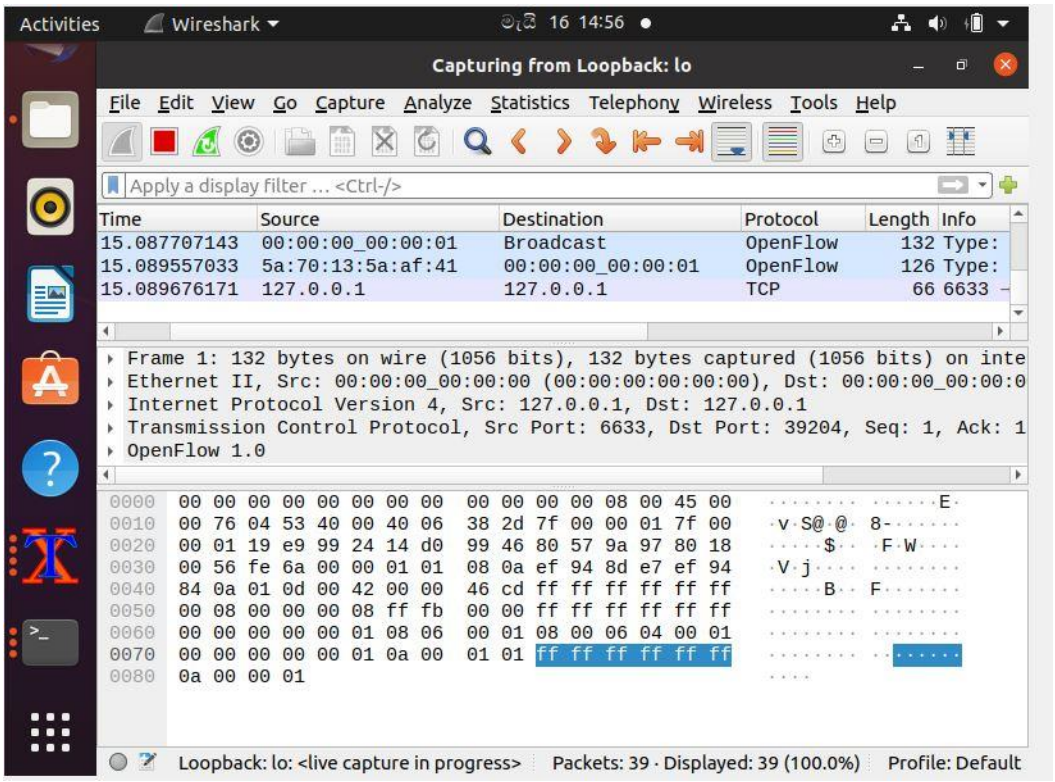


Figure 4-15: Wireshark capturing loopback

5 Testing and Evaluation

5.1 Testing

System testing is a way of determining whether the final system product meets the intended specifications and ensuring that it is free of defects or free from errors. In comparison to real specifications, system testing's aim is to find bugs, holes or gaps and incomplete requirements. So, testing and Evaluation phase is final step of SDLC. In this project researcher used prototype methodology to succeed the final product. To determine that, prototype has a testing panel in its every user evaluation and refining prototype period. Many current computer systems are so complicated and operate in such an interdependent technology environment that thorough inspection is impossible. Furthermore, any of the test cases have been done with a certain pass rate, they stop testing. Also, when the research budget is out, they stop testing. In information technology world there are so many tools for testing purpose. Even more there are two type of testing which are manual and automated. In this research, manual test done by researcher. Test cases are executed by manually by researcher to identify bugs and error in system. But type of testing tool is test data preparation, static analysis, test execution, configuration management, performance analysis, test management and coverage management. Actually, before the final product which means complete researcher's concept was reached the system, environment and components, number of tests to be carried out to achieve goal. For the network operation such as Mininet and pox controller operation taken lots of test to get goal. Every piece of Linux command tested by researcher because every command important to the project's aim and objectives. Also, get expected outcomes are achieved through the development phase such as least packet load-balancing algorithm concept. More over integration testing are taken place because remote controllers activated to the network operation.

Table 5-1: Test Case 01

TEST CASE ID	TC01
Test description	Topology creation and controller equal to remote. XTerm use or access remote. Port number is 6633. Researcher use only one controller
Test steps	1) Type the command 2) Click enter button
Test data	Devices, that wants to develop system with remote access.
Expected result	Switch, hosts, controller and servers are created as researcher expect from the command.
Actual result	As expected by researcher
Status	Pass

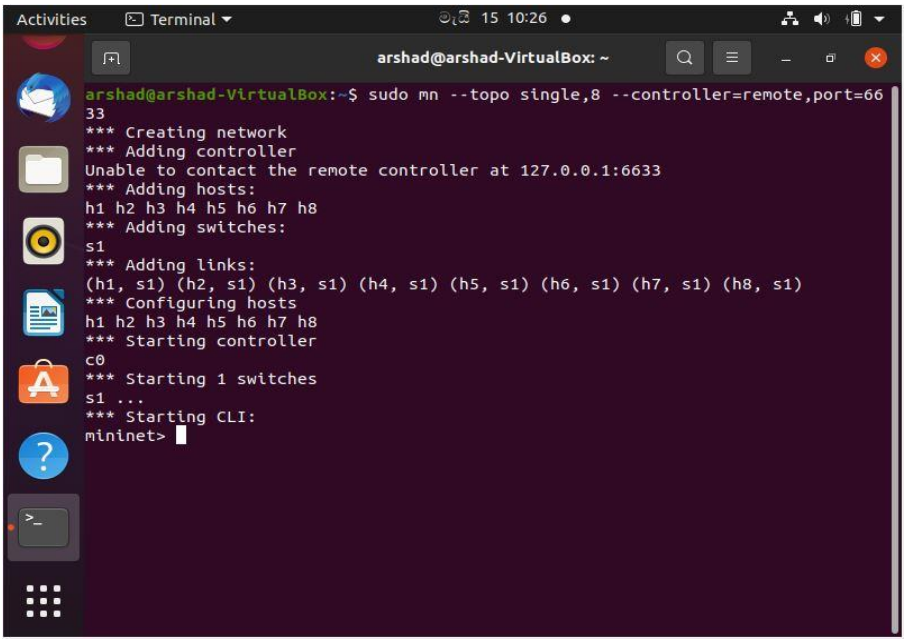


Table 5-2: Test Case 02

TEST CASE ID	TC02
Test description	Server creation with according to design topology. Node h1 and the Node h2 are simple HTTP servers with port 80.
Test steps	1) Type the command in starting CLI Mininet> position 2) Click enter button
Test data	Command = Mininet> xterm h1 h2
Expected result	Node h1 and Node h2 command line interfaces are prompting with black terminal
Actual result	As expected by researcher
Status	Pass

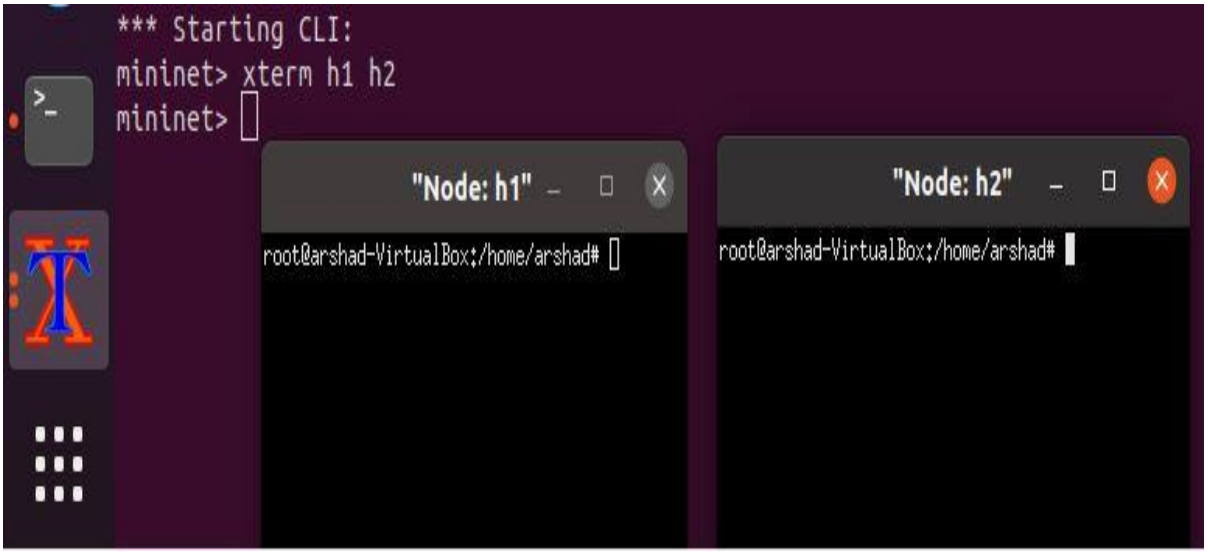


Table 5-3: Test Case 03

TEST CASE ID	TC03
Test description	Server creation with according to design topology. Node h1 and the Node h2 are simple HTTP servers with port 80.
Test steps	1) Type the command 2) Click enter button
Test data	Command = $python –m SimpleHTTPServer 80
Expected result	Serving HTTP on 0.0.0.0 port 80..

	Was ready to serving and waiting for load.
Actual result	As expected by researcher
Status	Pass
	

Table 5-4: Test Case 04

TEST CASE ID	TC04
Test description	Pox controller implementing with IP assign servers Node h1 and Node h2. Also, IP assign for controller with port number 6633.
Test steps	1) Type the command 2) Click enter button
Test data	Command = $ pox/pox.py log.level –DEBUG misc_ip_loadbalancer --ip=10.0.1.1 –servers=10.0.0.1,10.0.0.2
Expected result	Servers Node h1 and Node h2 up. Also, ready to balance the load. Controller worked as a load balancer. So, controller IP is 10.0.1.1.
Actual result	As expected by researcher
Status	Pass

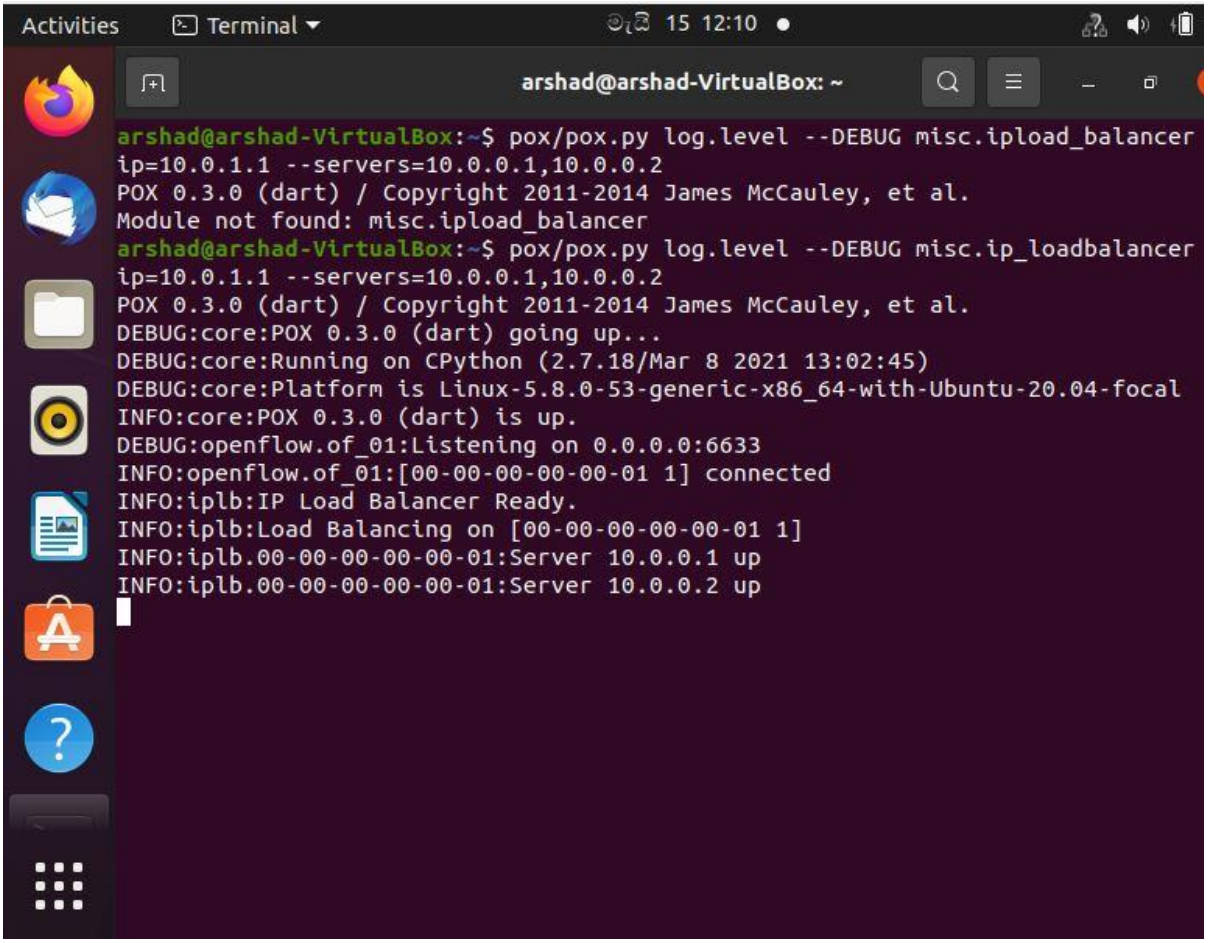


Table 5-5: Test Case 05

TEST CASE ID	TC05
Test description	User creation with according to design topology. Node h3, h4, h5, h6, h7 and the Node h8 are users.
Test steps	1) Type the command in starting CLI Mininet> position 2) Click enter button

Test data	Command = Mininet> xterm h1 h2 h3 h4 h5 h6 h7 h8
Expected result	Node h3, h4, h5, h6, h7 and Node h8 command line interfaces are prompting with black terminal
Actual result	As expected by researcher
Status	Pass

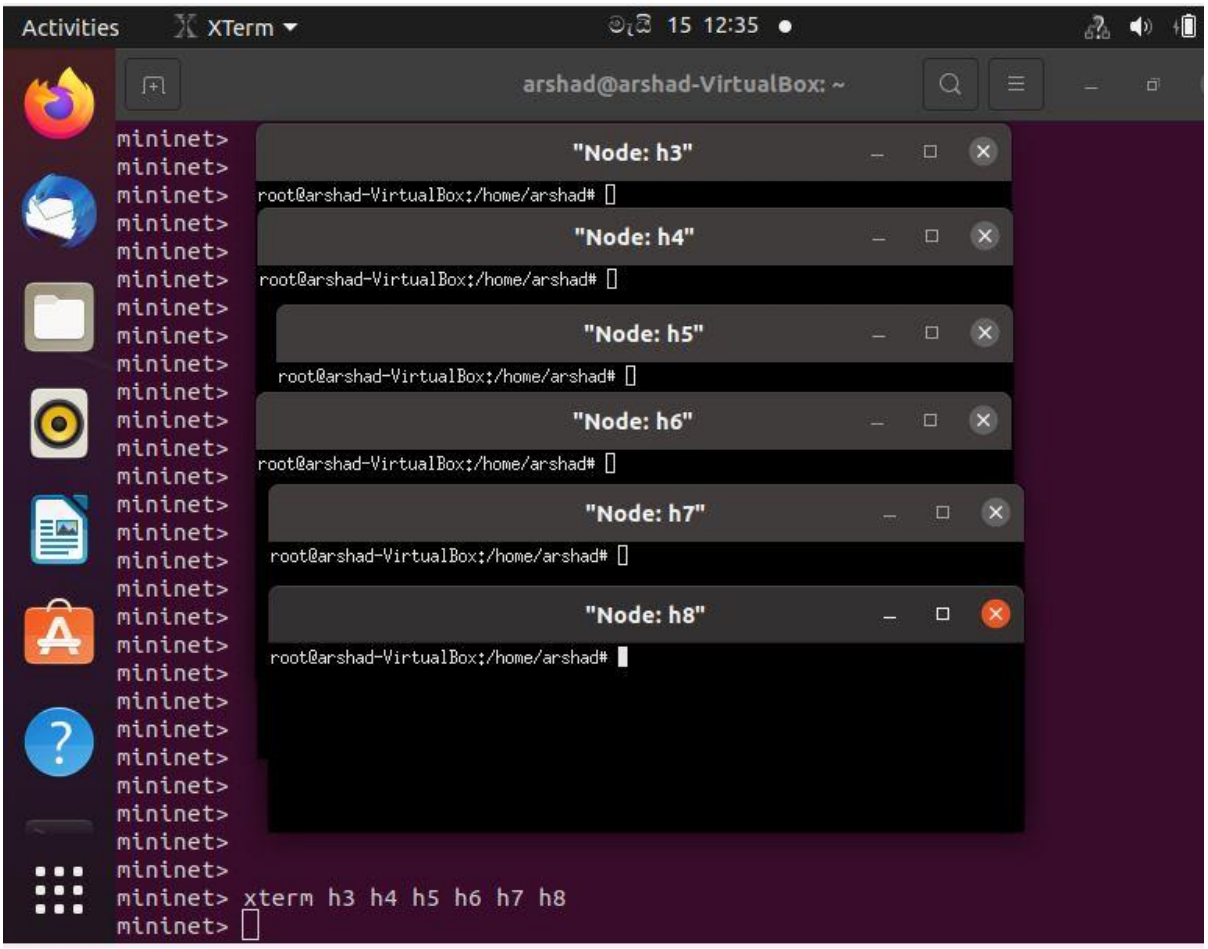
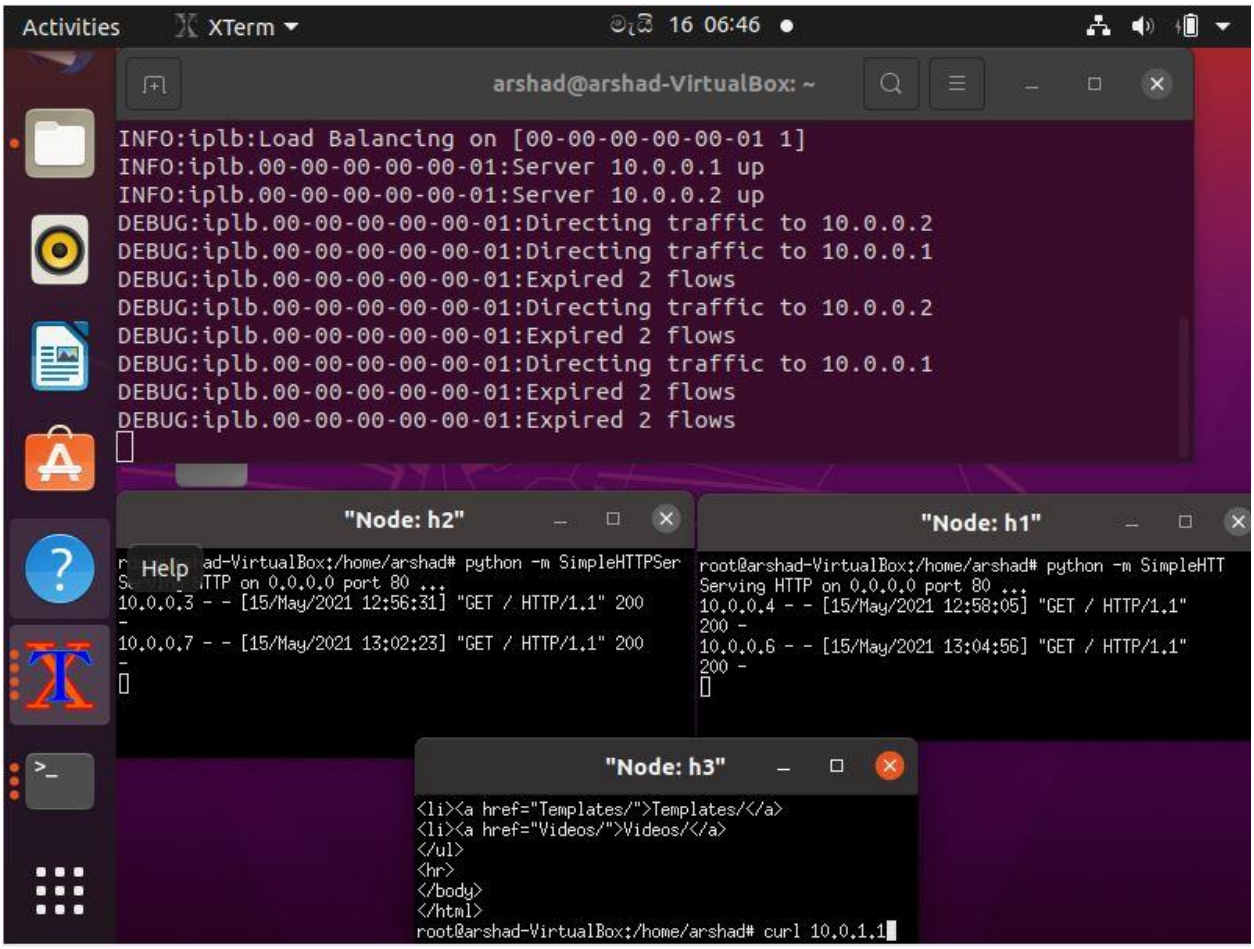


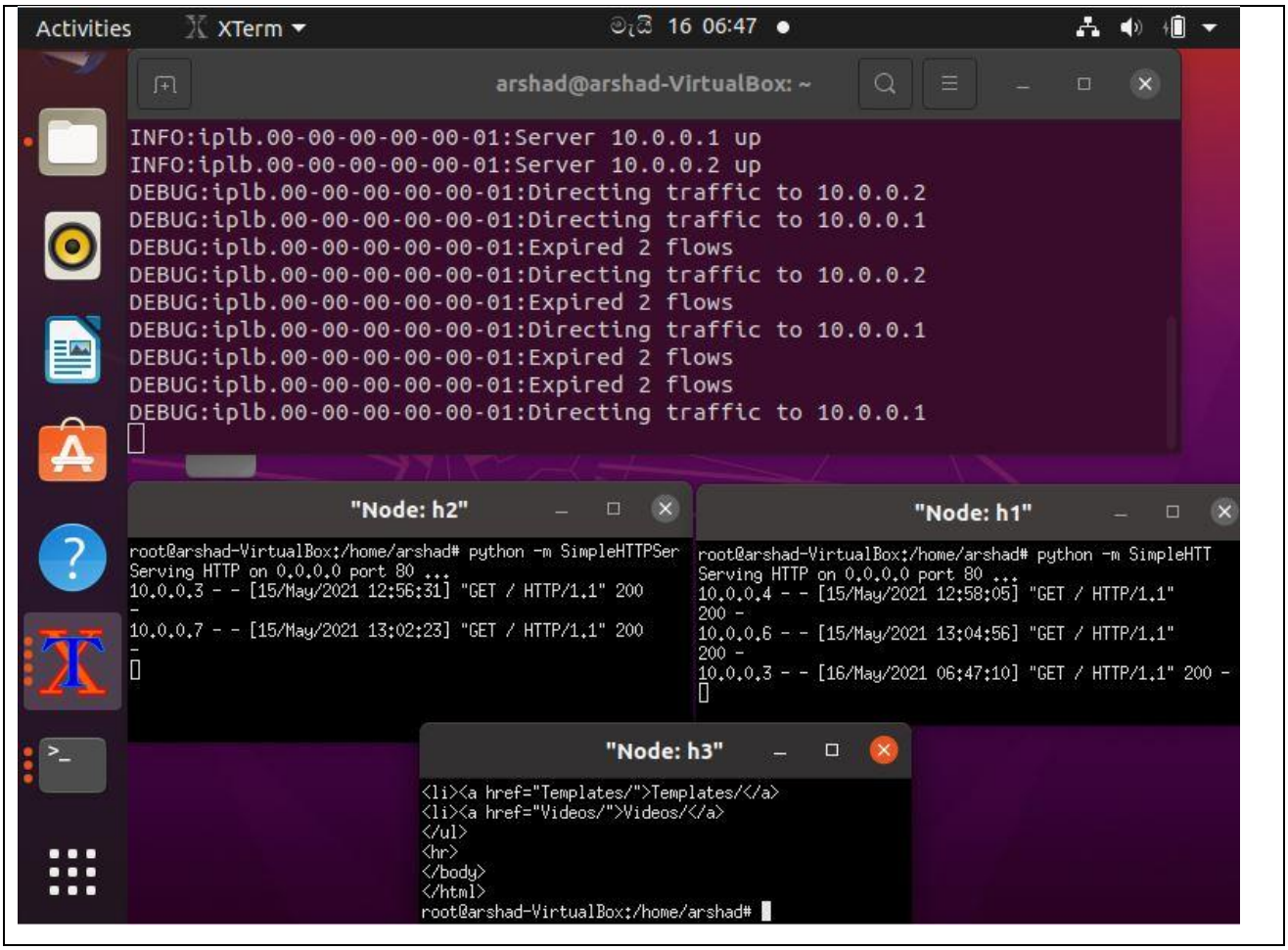
Table 5-6: Test Case 06

TEST CASE ID	TC06

Test description	When curl 10.0.1.1 command executed in Node h3, at terminal suddenly we can see the directing traffic. Also, in the Node h1 or Node h2. Actually, for the testing purpose we execute curl command multiple times and see the result (third image showed the load balancing which one host request for the network). Expected result is concept of the project.
Test steps	1) Type the command in remote Node h3 terminal 2) Click enter button
Test data	Command = curl 10.0.1.1
Expected result	Balance the load in Node h1 and Node h2 with the help of controller. When we gave multiple request in Node h3, it react in both severs.
Actual result	As expected by researcher
Status	Pass


Activities
XTerm
16 06:46
arshad@arshad-VirtualBox: ~
INFO:iplb:Load Balancing on [00-00-00-00-00-01 1]
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.1 up
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.2 up
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Expired 2 flows
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Expired 2 flows
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Expired 2 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 2 flows
"Node: h2"
Help
10.0.0.3 - - [15/May/2021 12:56:31] "GET / HTTP/1.1" 200 -
10.0.0.7 - - [15/May/2021 13:02:23] "GET / HTTP/1.1" 200 -
"Node: h1"
root@arshad-VirtualBox:/home/arshad# python -m SimpleHTT
Serving HTTP on 0.0.0.0 port 80 ...
10.0.0.4 - - [15/May/2021 12:58:05] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [15/May/2021 13:04:56] "GET / HTTP/1.1" 200 -
"Node: h3"
Templates/
Videos/

<hr>
</body>
</html>
root@arshad-VirtualBox:/home/arshad# curl 10.0.1.1


Activities
XTerm
16 06:47
arshad@arshad-VirtualBox: ~
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.1 up
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.2 up
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Expired 2 flows
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Expired 2 flows
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Expired 2 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 2 flows
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
"Node: h2"
root@arshad-VirtualBox:/home/arshad# python -m SimpleHTTPSer
Serving HTTP on 0.0.0.0 port 80 ...
10.0.0.3 - - [15/May/2021 12:56:31] "GET / HTTP/1.1" 200
-
10.0.0.7 - - [15/May/2021 13:02:23] "GET / HTTP/1.1" 200
-
"Node: h1"
root@arshad-VirtualBox:/home/arshad# python -m SimpleHTT
Serving HTTP on 0.0.0.0 port 80 ...
10.0.0.4 - - [15/May/2021 12:58:05] "GET / HTTP/1.1"
200 -
10.0.0.6 - - [15/May/2021 13:04:56] "GET / HTTP/1.1"
200 -
10.0.0.3 - - [16/May/2021 06:47:10] "GET / HTTP/1.1" 200 -
"Node: h3"
Templates/
Videos/

<hr>
</body>
</html>
root@arshad-VirtualBox:/home/arshad#

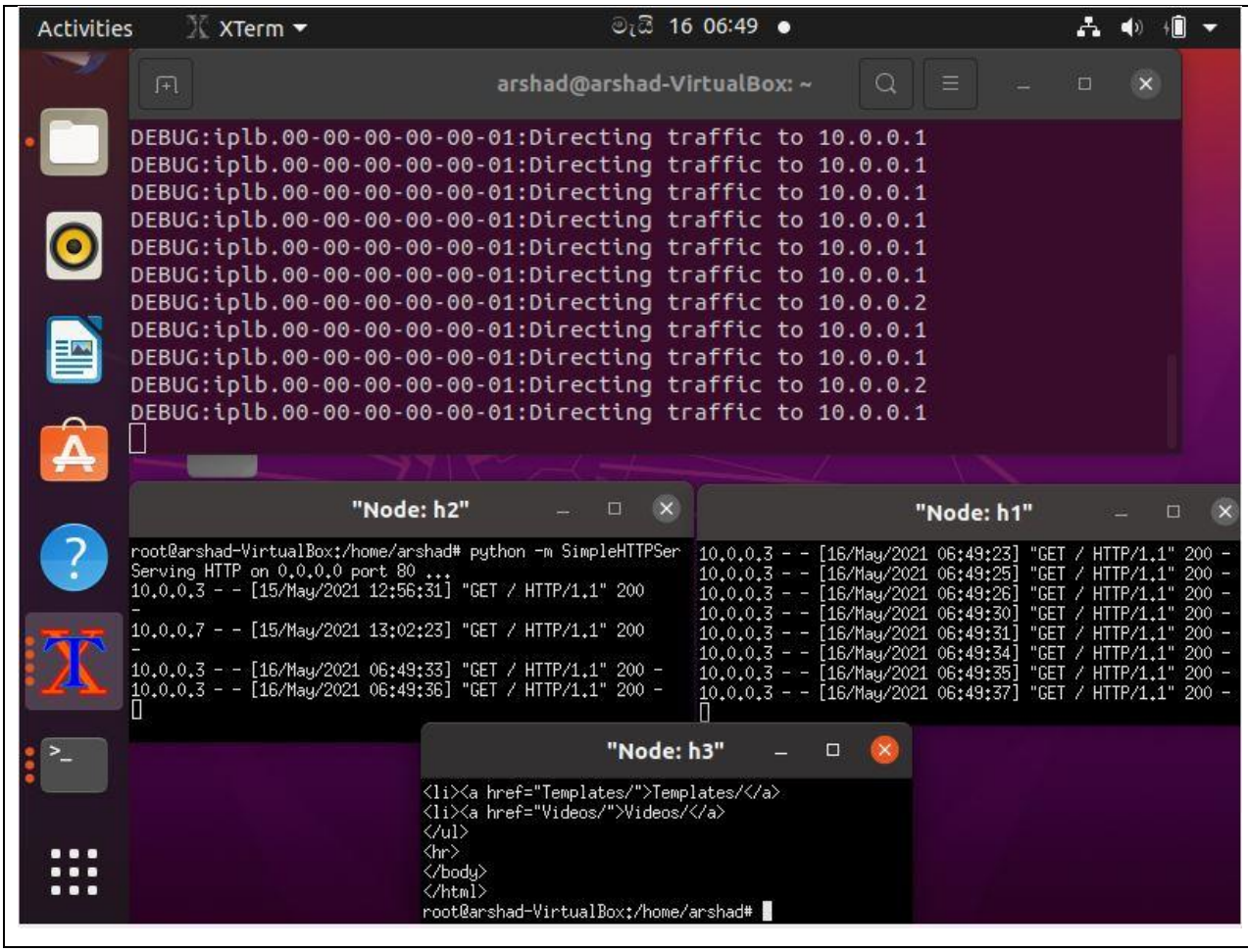
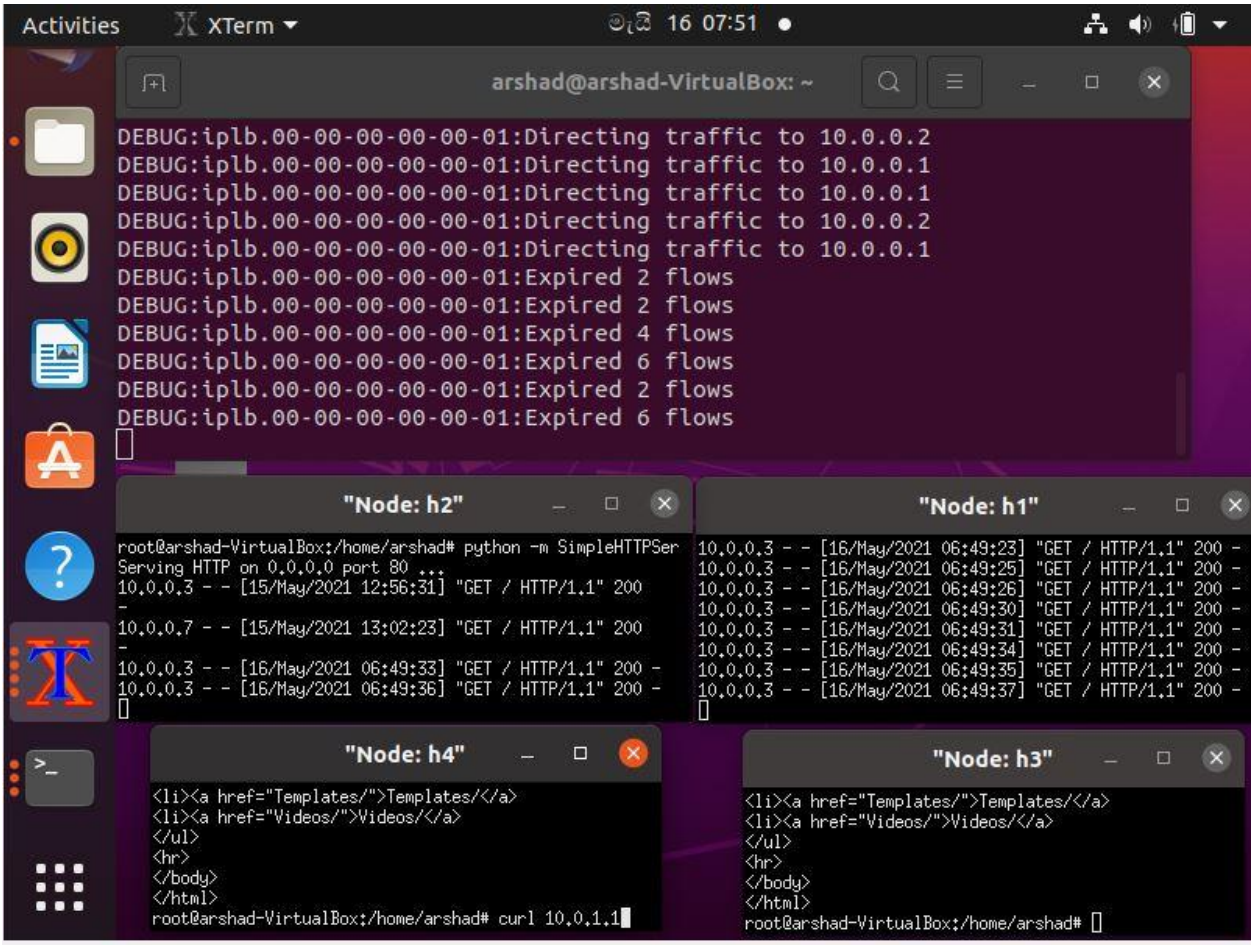
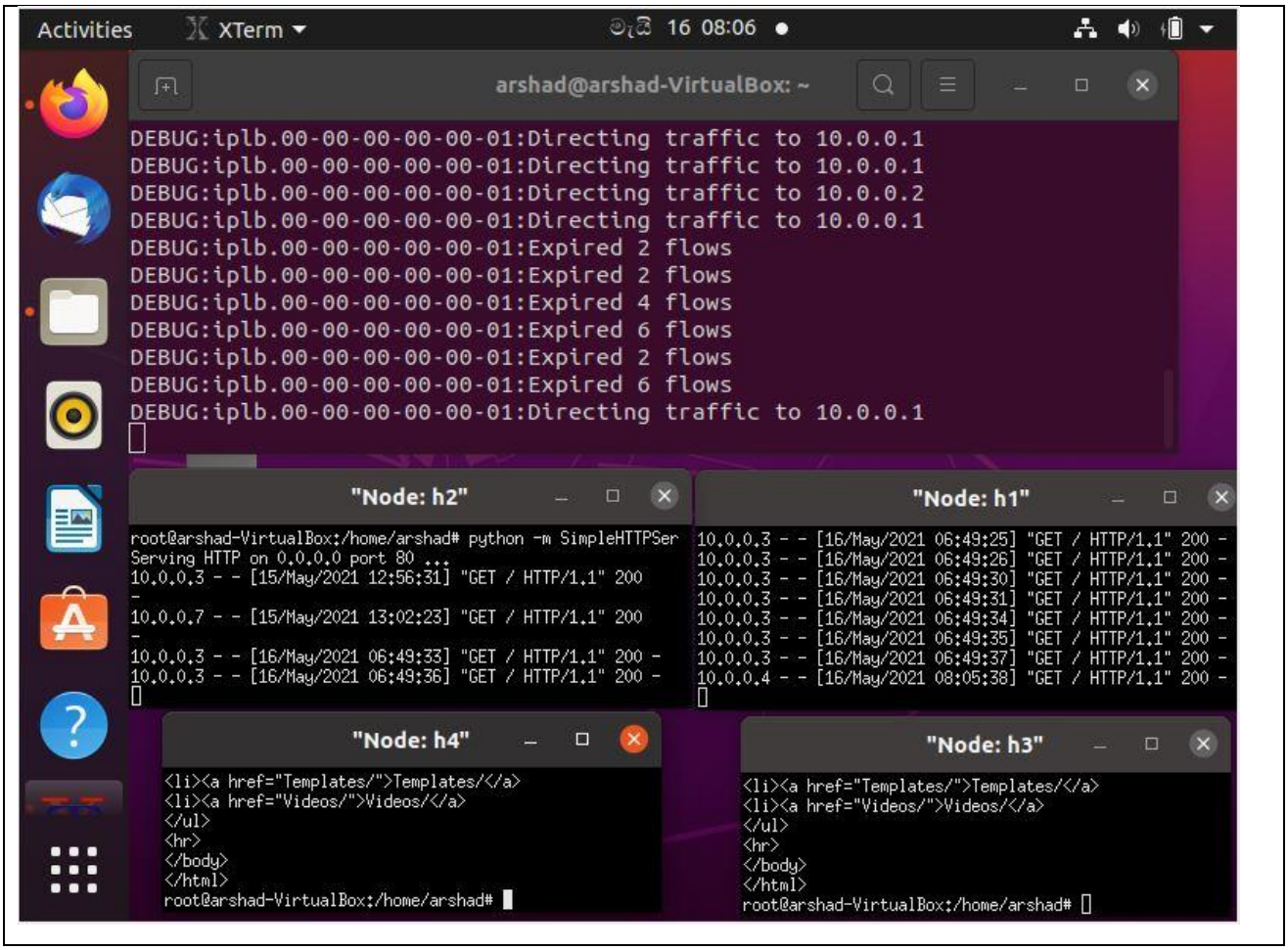


Table 5-7: Test Case 07

TEST CASE ID	TC07
Test description	When curl 10.0.1.1 command executed in Node h4, at terminal suddenly we can see the directing traffic. Also, in the Node h1 or Node h2. Actually, for the testing purpose we execute curl command multiple times and see the result (third image showed the load balancing which one host request for the network). Expected result is concept of the project.

Test steps	1) Type the command in remote Node h4 terminal 2) Click enter button
Test data	Command = curl 10.0.1.1
Expected result	Balance the load in Node h1 and Node h2 with the help of controller. When we gave multiple requests in Node h4, it react in both severs.
Actual result	As expected by researcher
Status	Pass




Activities
XTerm
16 08:06
arshad@arshad-VirtualBox: ~
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Expired 2 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 2 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 4 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 6 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 2 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 6 flows
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
"Node: h2"
root@arshad-VirtualBox:/home/arshad# python -m SimpleHTTPSer
Serving HTTP on 0.0.0.0 port 80 ...
10.0.0.3 - - [15/May/2021 12:56:31] "GET / HTTP/1.1" 200
-
10.0.0.7 - - [15/May/2021 13:02:23] "GET / HTTP/1.1" 200
-
10.0.0.3 - - [16/May/2021 06:49:33] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:36] "GET / HTTP/1.1" 200 -
"Node: h1"
10.0.0.3 - - [16/May/2021 06:49:25] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:26] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:30] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:31] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:34] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:35] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:37] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:05:38] "GET / HTTP/1.1" 200 -
"Node: h4"
Templates/
Videos/

<hr>
</body>
</html>
root@arshad-VirtualBox:/home/arshad#
"Node: h3"
Templates/
Videos/

<hr>
</body>
</html>
root@arshad-VirtualBox:/home/arshad#

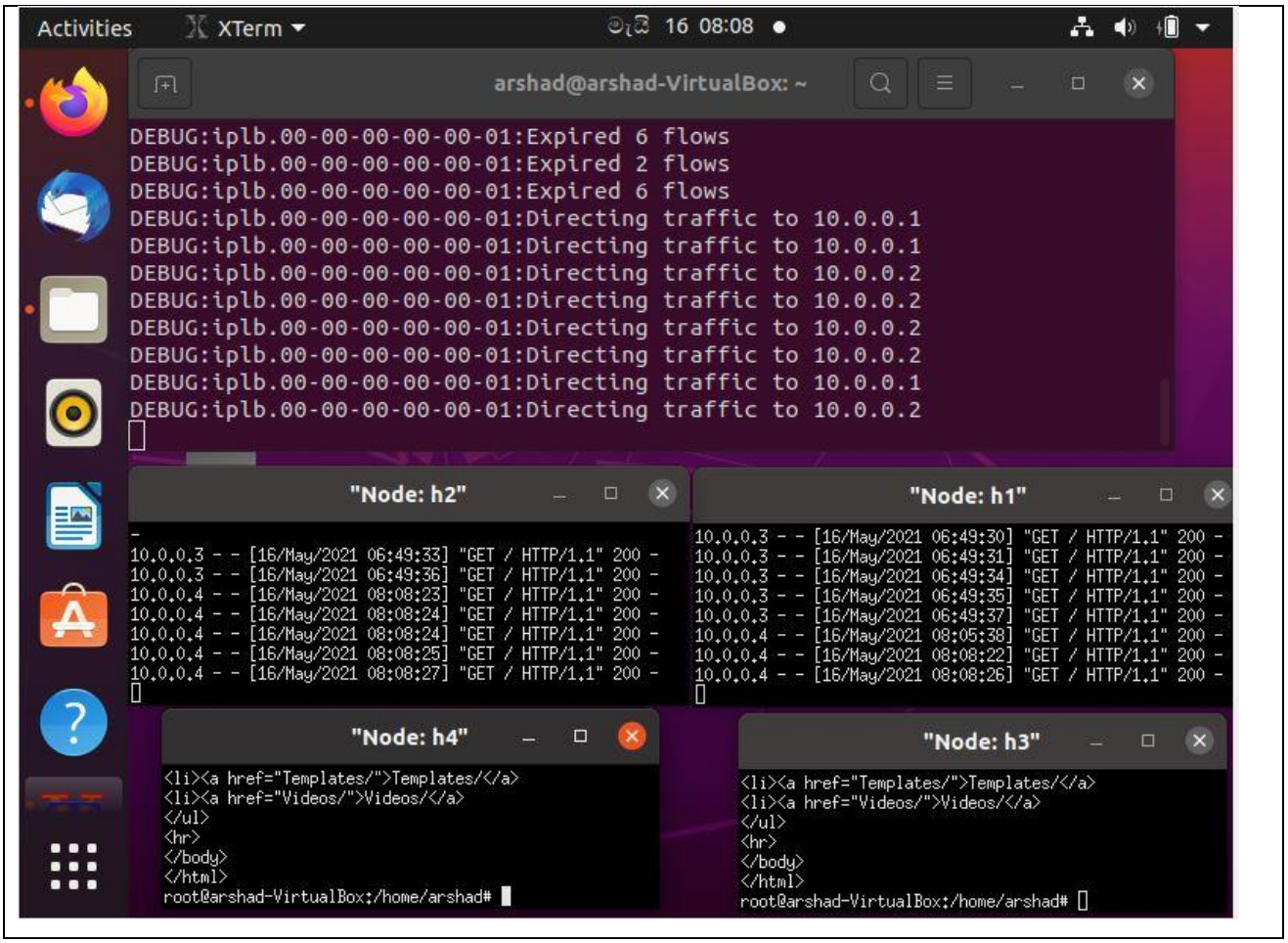
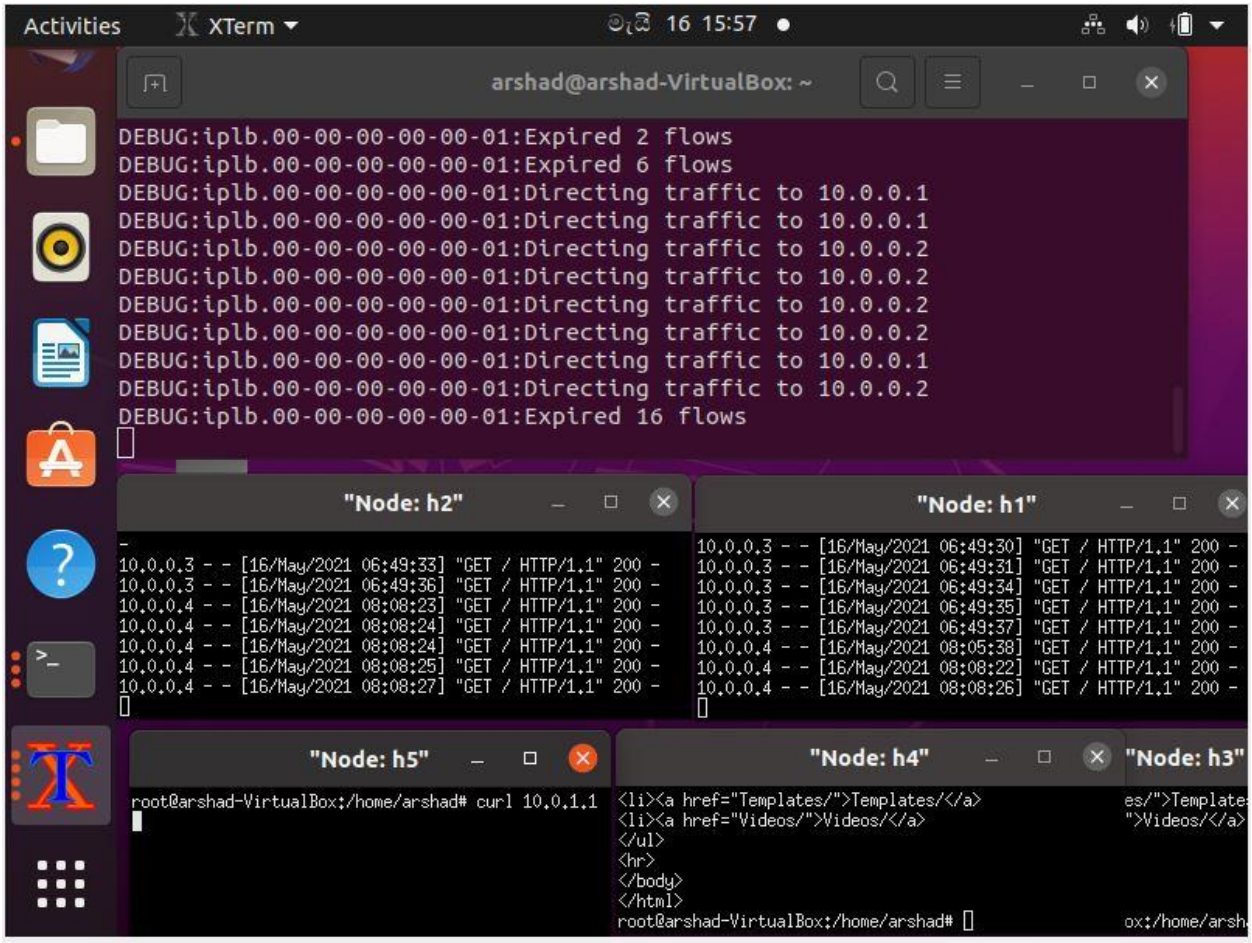
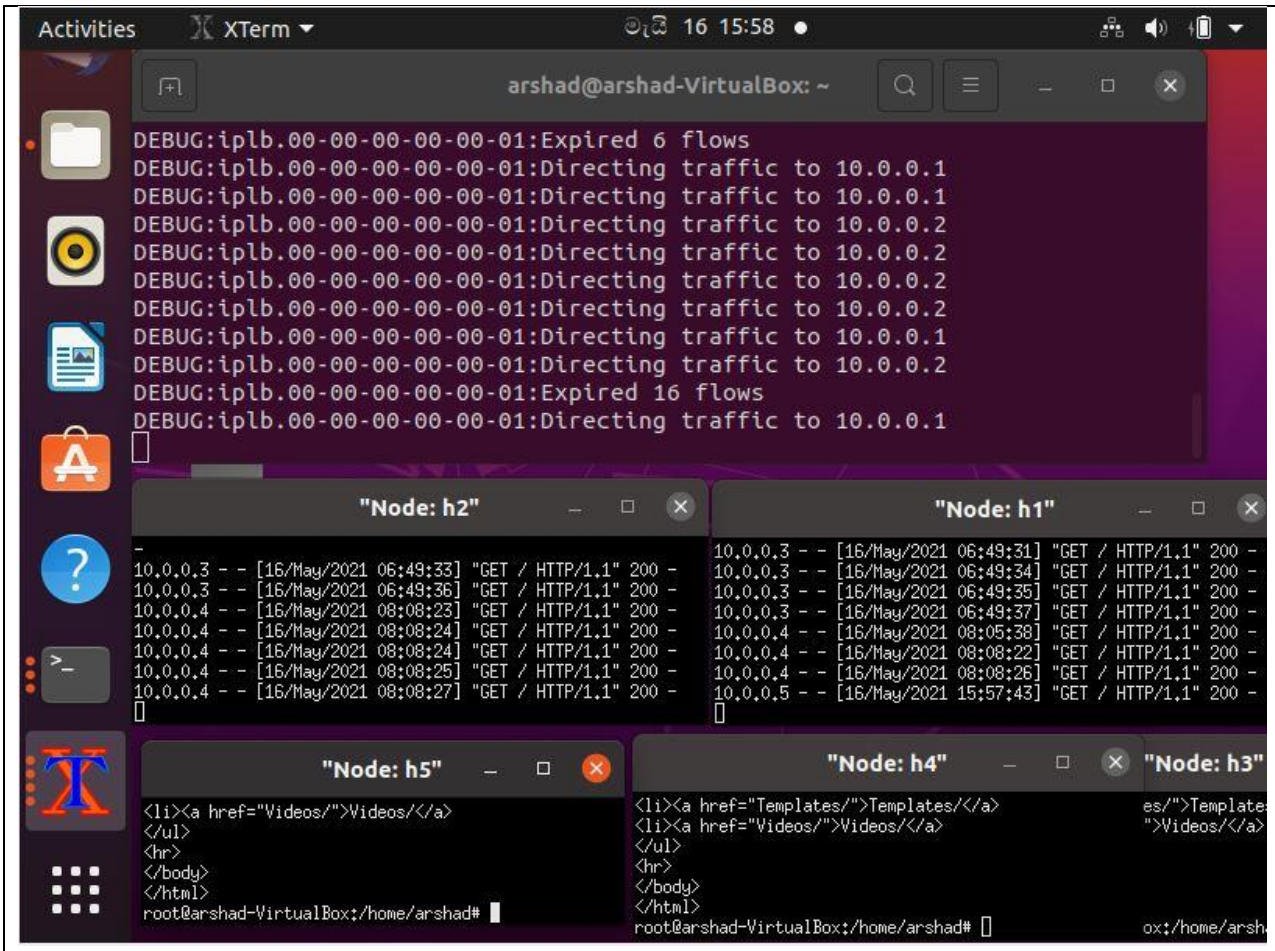


Table 5-8: Test Case 08

TEST CASE ID	TC08
Test description	When curl 10.0.1.1 command executed in Node h5, at terminal suddenly we can see the directing traffic. Also, in the Node h1 or Node h2. Actually, for the testing purpose we execute curl command multiple times and see the result (third image showed the load balancing which one host request for the network). Expected result is concept of the project.

Test steps	1) Type the command in remote Node h5 terminal 2) Click enter button
Test data	Command = curl 10.0.1.1
Expected result	Balance the load in Node h1 and Node h2 with the help of controller. When we gave multiple request in Node h5, it react in both severs.
Actual result	As expected by researcher
Status	Pass




Activities
XTerm
16 15:58
arshad@arshad-VirtualBox: ~
DEBUG:iplb.00-00-00-00-00-01:Expired 6 flows
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Expired 16 flows
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
"Node: h2"
10.0.0.3 - - [16/May/2021 06:49:33] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:36] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:23] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:24] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:24] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:25] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:27] "GET / HTTP/1.1" 200 -
"Node: h1"
10.0.0.3 - - [16/May/2021 06:49:31] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:34] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:35] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:37] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:05:38] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:22] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:26] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:57:43] "GET / HTTP/1.1" 200 -
"Node: h5"
Videos/

<hr>
</body>
</html>
root@arshad-VirtualBox:/home/arshad#
"Node: h4"
Templates/
Videos/

<hr>
</body>
</html>
root@arshad-VirtualBox:/home/arshad#
"Node: h3"

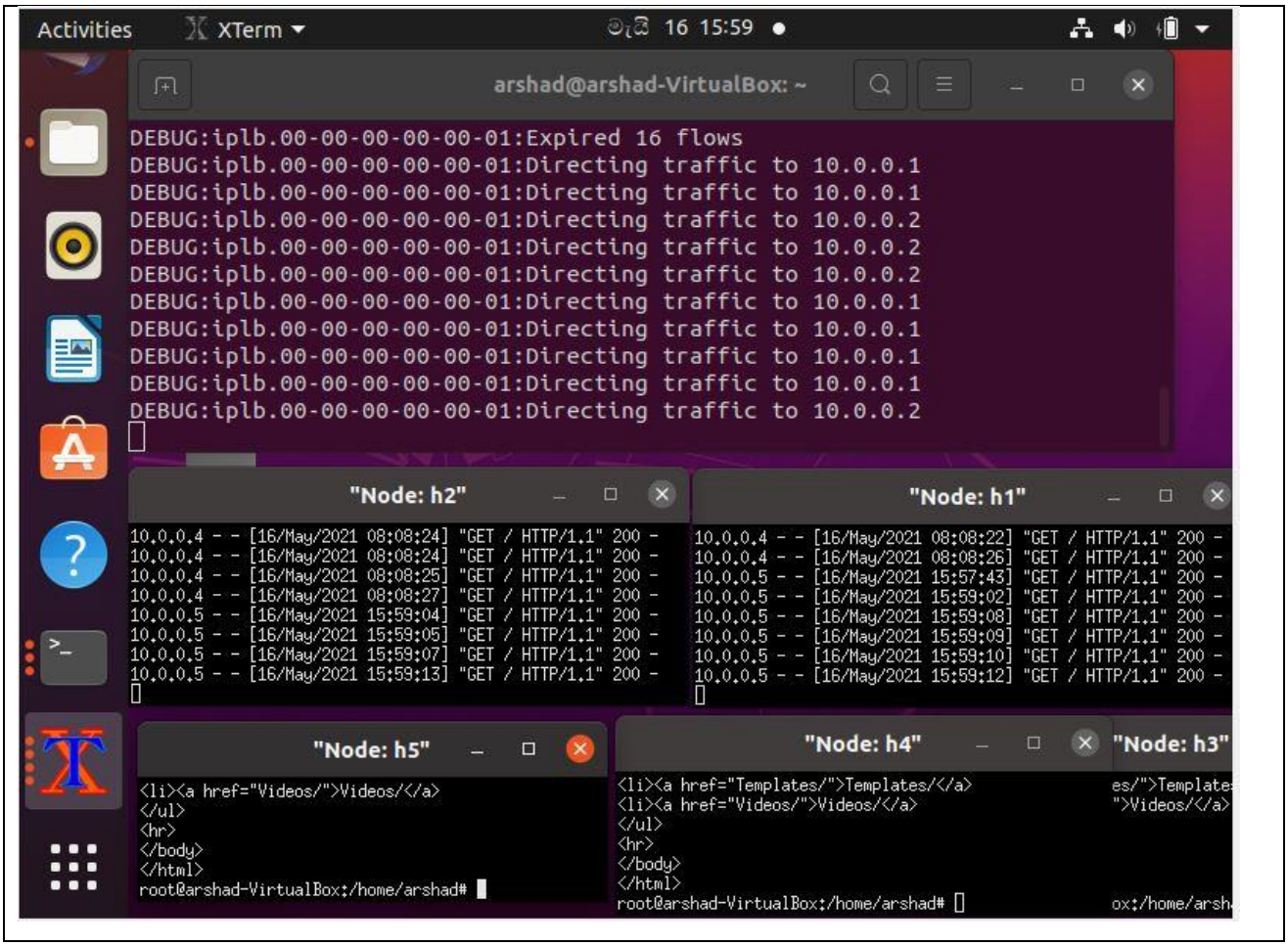
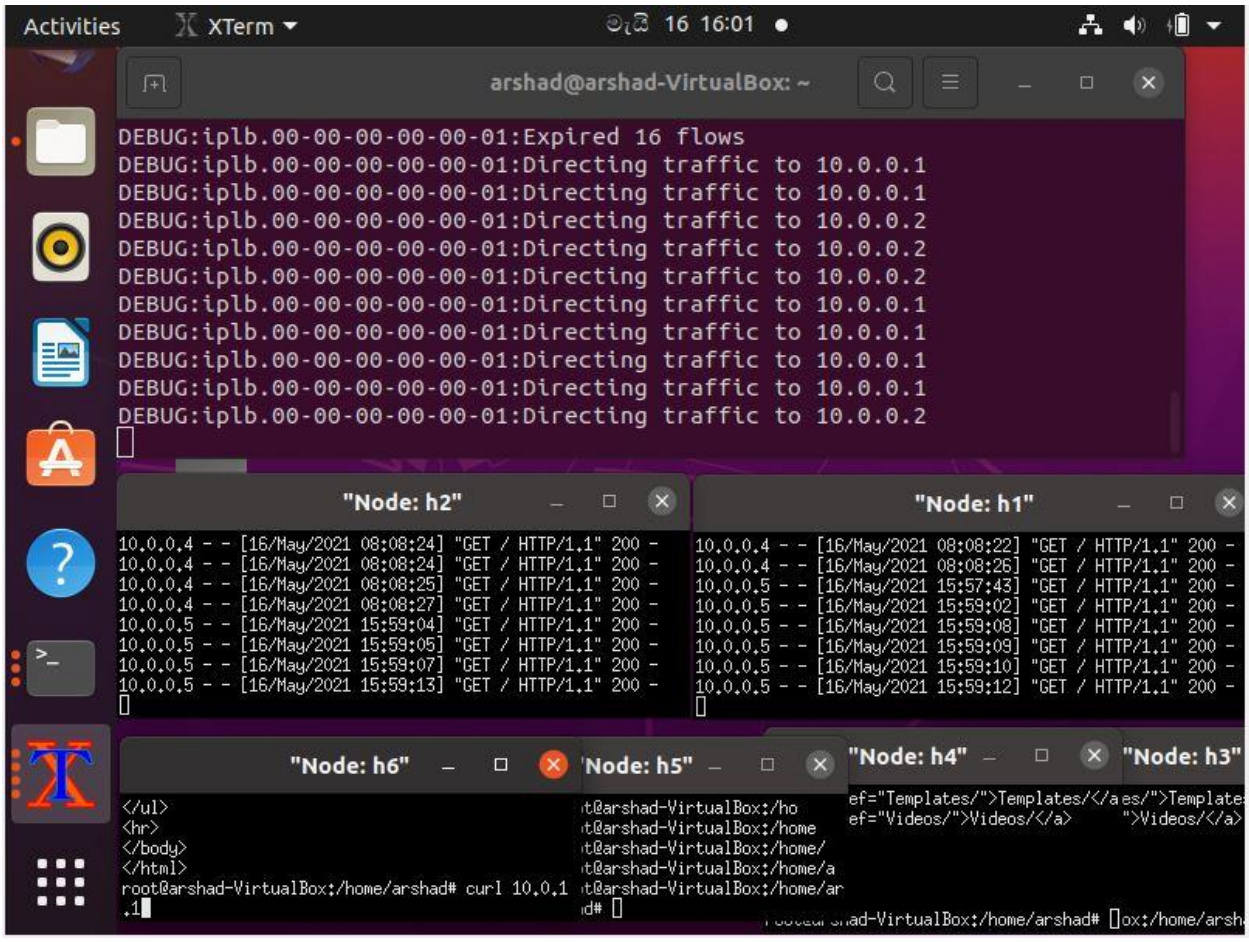
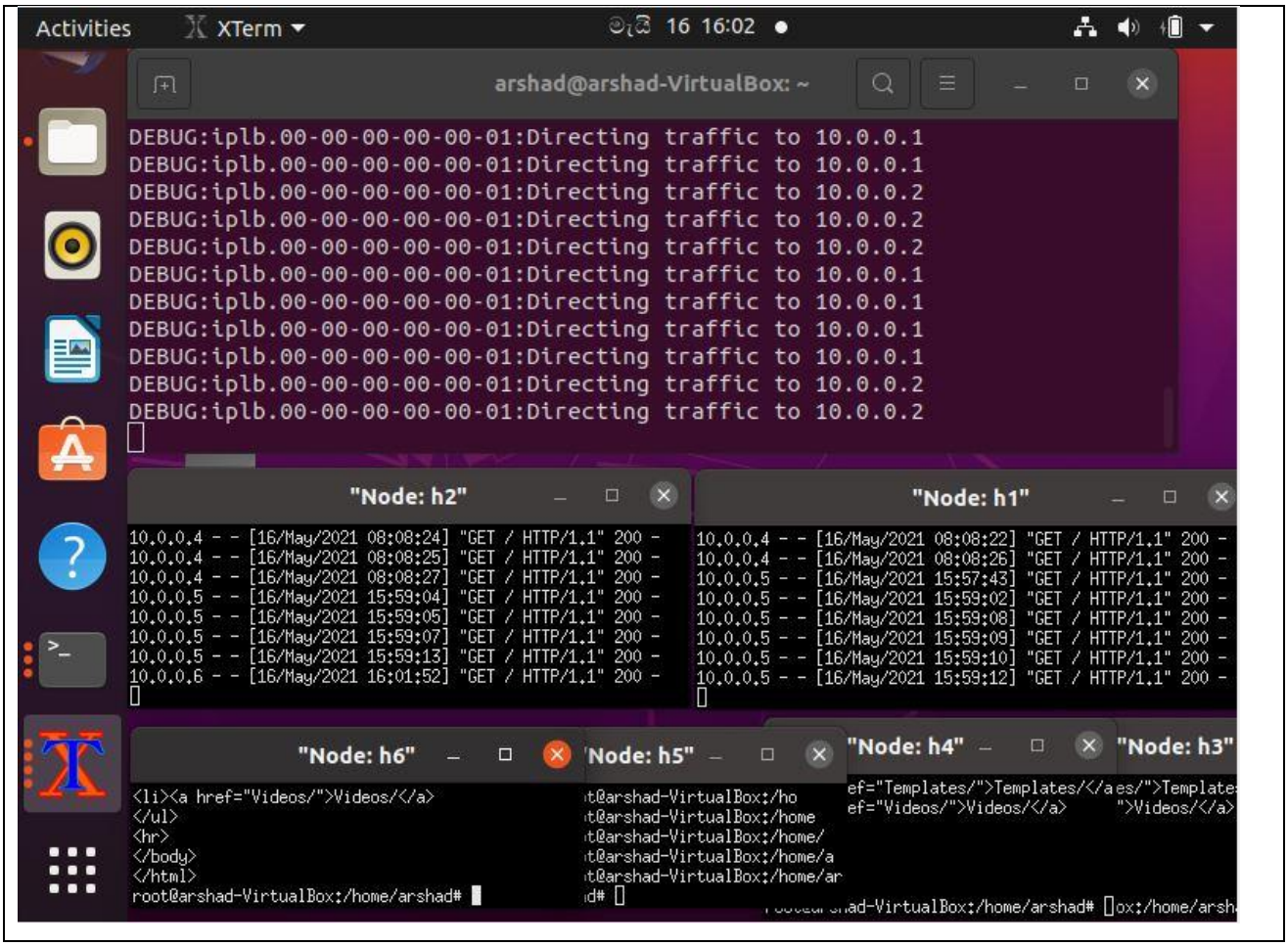


Table 5-9: Test Case 09

TEST CASE ID	TC09
Test description	When curl 10.0.1.1 command executed in Node h6, at terminal suddenly we can see the directing traffic. Also, in the Node h1 or Node h2. Actually, for the testing purpose we execute curl command multiple times and see the result (third image showed the load balancing which one host request for the network). Expected result is concept of the project.

Test steps	1) Type the command in remote Node h6 terminal 2) Click enter button
Test data	Command = curl 10.0.1.1
Expected result	Balance the load in Node h1 and Node h2 with the help of controller. When we gave multiple request in Node h6, it react in both severs.
Actual result	As expected by researcher
Status	Pass




Activities
XTerm
16 16:02
arshad@arshad-VirtualBox: ~
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
"Node: h2"
10.0.0.4 - - [16/May/2021 08:08:24] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:25] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:27] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:04] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:05] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:07] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:13] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:01:52] "GET / HTTP/1.1" 200 -
"Node: h1"
10.0.0.4 - - [16/May/2021 08:08:22] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:26] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:57:43] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:02] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:08] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:09] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:10] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:12] "GET / HTTP/1.1" 200 -
"Node: h6"
Videos/

<hr>
</body>
</html>
root@arshad-VirtualBox:/home/arshad#
"Node: h5"
"Node: h4"
"Node: h3"

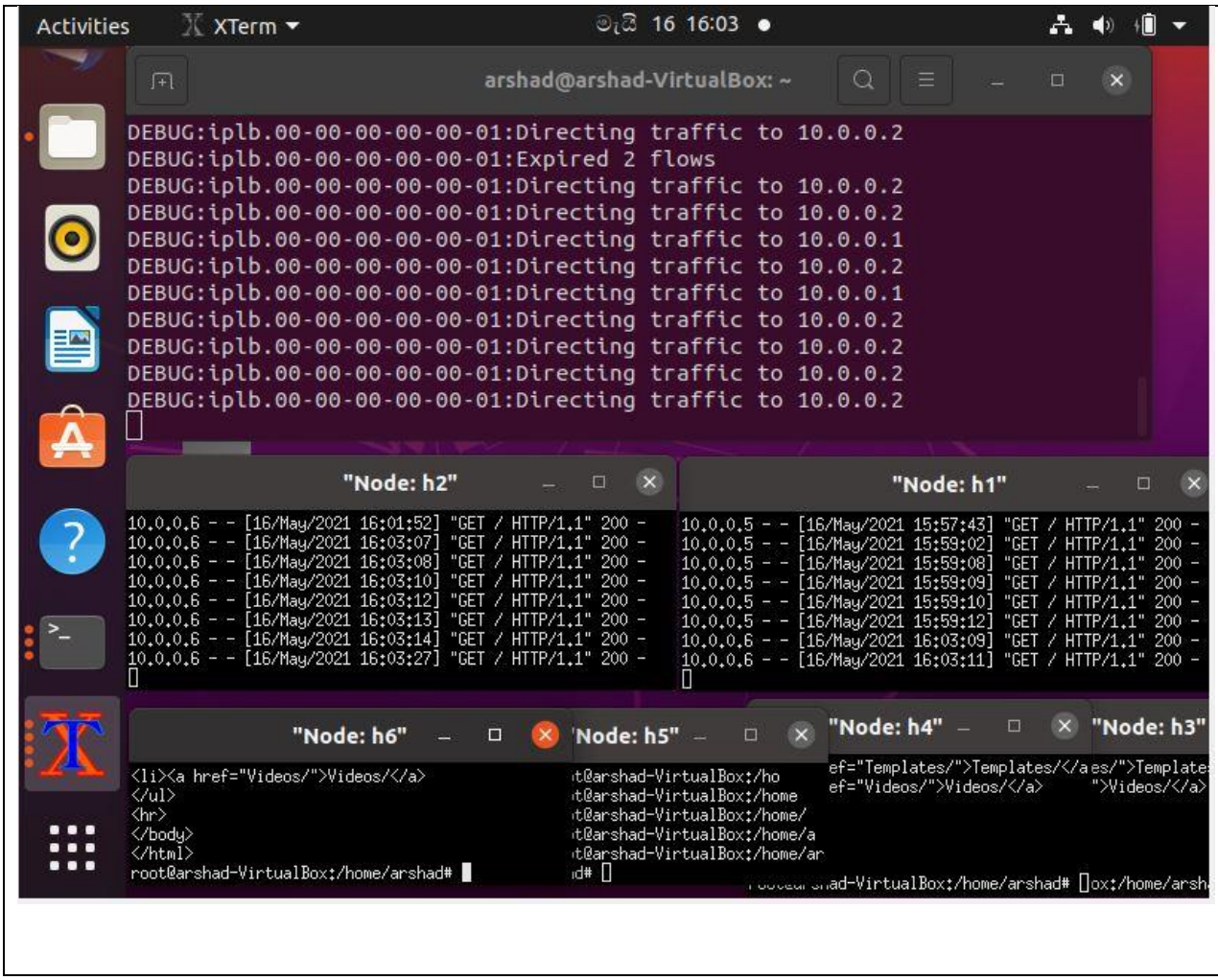
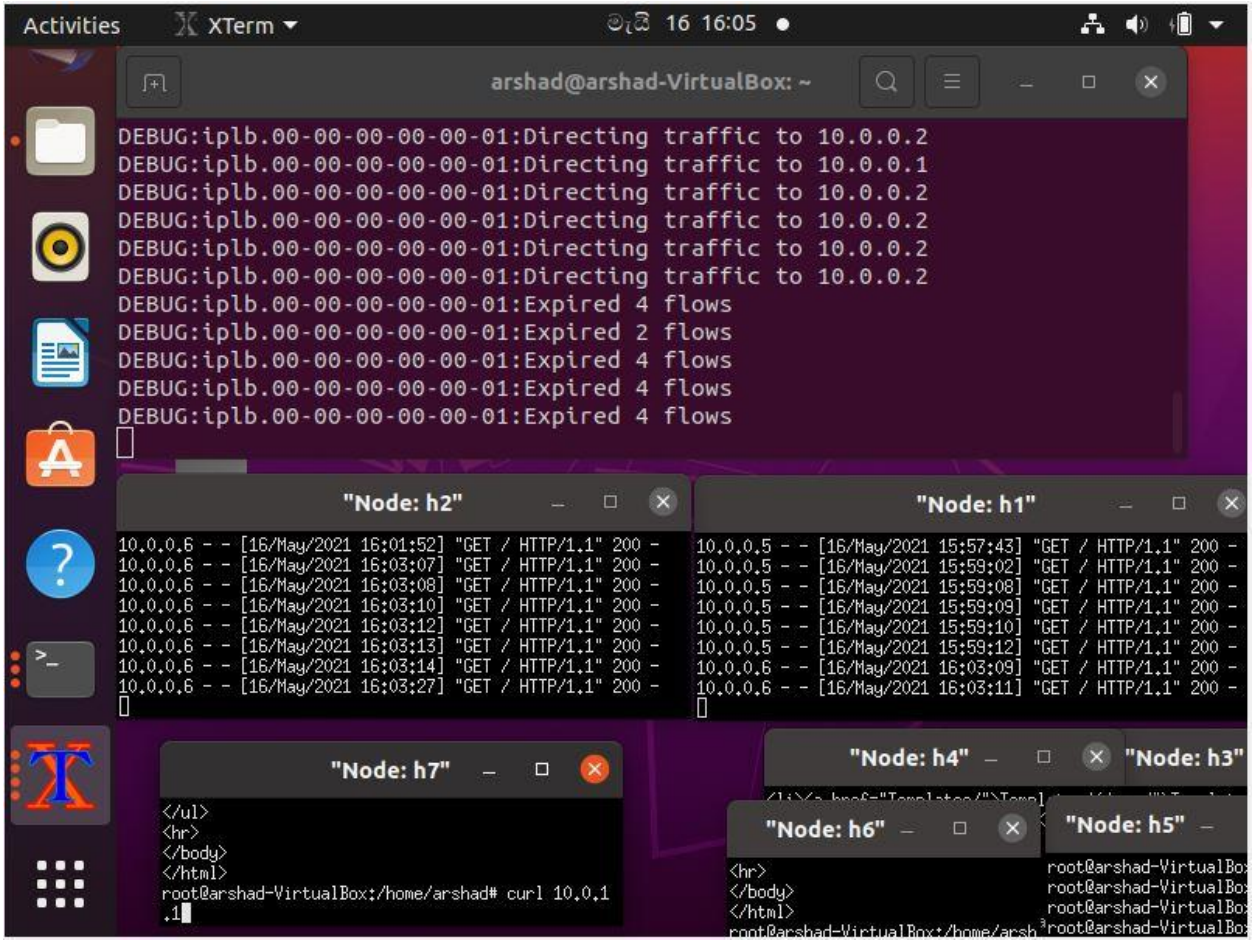


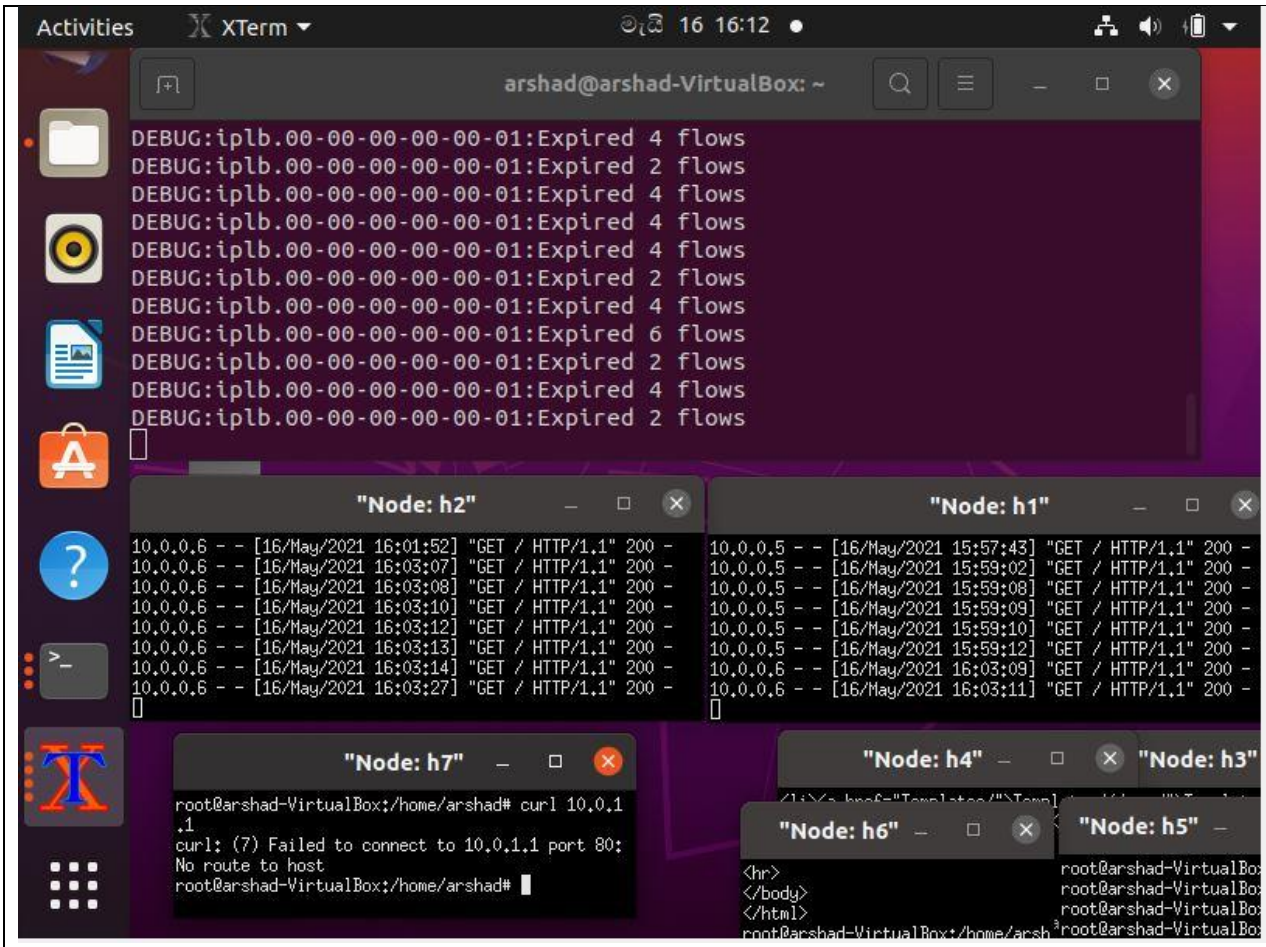
Table 5-10: Test Case 10

TEST CASE ID	TC10
Test description	When curl 10.0.1.1 command executed in Node h7, at terminal suddenly we cannot see the directing traffic. Also, in the Node h1 or Node h2. Actually, for the testing purpose we execute curl command multiple times and see the result (third image showed the load balancing which one host request for the network). Expected result is concept of the project.

Test steps	1) Type the command in remote Node h7 terminal 2) Click enter button
Test data	Command = curl 10.0.1.1
Expected result	Balance the load in Node h1 and Node h2 with the help of controller. When we gave multiple requests in Node h7, it react in both severs.
Actual result	Expected result was not get by researcher
Status	Fail, because at Node h7 user misunderstand and two times click the error key. So, network congestion happened. Third picture show miss behave command.


Activities
XTerm
16 16:05
arshad@arshad-VirtualBox: ~
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Expired 4 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 2 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 4 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 4 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 4 flows
"Node: h2"
10.0.0.6 - - [16/May/2021 16:01:52] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:07] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:08] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:10] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:12] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:13] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:14] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:27] "GET / HTTP/1.1" 200 -
"Node: h1"
10.0.0.5 - - [16/May/2021 15:57:43] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:02] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:08] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:09] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:10] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:12] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:09] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:11] "GET / HTTP/1.1" 200 -
"Node: h7"

<hr>
</body>
</html>
root@arshad-VirtualBox:/home/arshad# curl 10.0.1
.1
"Node: h4"
"Node: h3"
"Node: h6"
<hr>
</body>
</html>
"Node: h5"
root@arshad-VirtualBox:/home/arshad
root@arshad-VirtualBox:/home/arshad
root@arshad-VirtualBox:/home/arshad


Activities
XTerm
16 16:12
arshad@arshad-VirtualBox: ~
DEBUG:iplb.00-00-00-00-00-01:Expired 4 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 2 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 4 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 4 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 4 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 2 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 4 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 6 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 2 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 4 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 2 flows
"Node: h2"
10.0.0.6 - - [16/May/2021 16:01:52] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:07] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:08] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:10] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:12] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:13] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:14] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:27] "GET / HTTP/1.1" 200 -
"Node: h1"
10.0.0.5 - - [16/May/2021 15:57:43] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:02] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:08] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:09] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:10] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:59:12] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:09] "GET / HTTP/1.1" 200 -
10.0.0.6 - - [16/May/2021 16:03:11] "GET / HTTP/1.1" 200 -
"Node: h7"
root@arshad-VirtualBox:/home/arshad# curl 10.0.1
.1
curl: (7) Failed to connect to 10.0.1.1 port 80:
No route to host
root@arshad-VirtualBox:/home/arshad#
"Node: h4"
"Node: h3"
"Node: h6"
"Node: h5"
</hr>
</body>
</html>

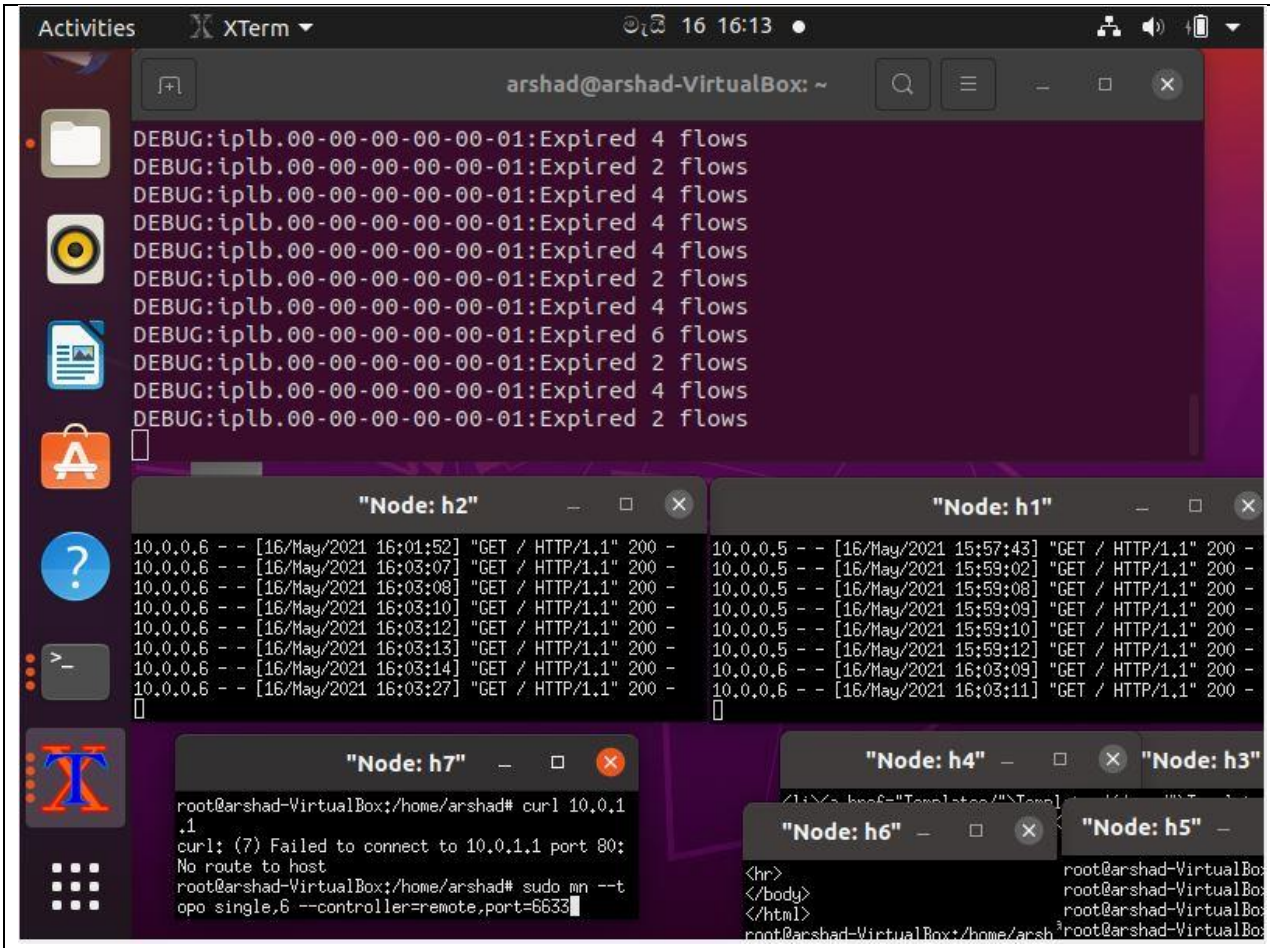
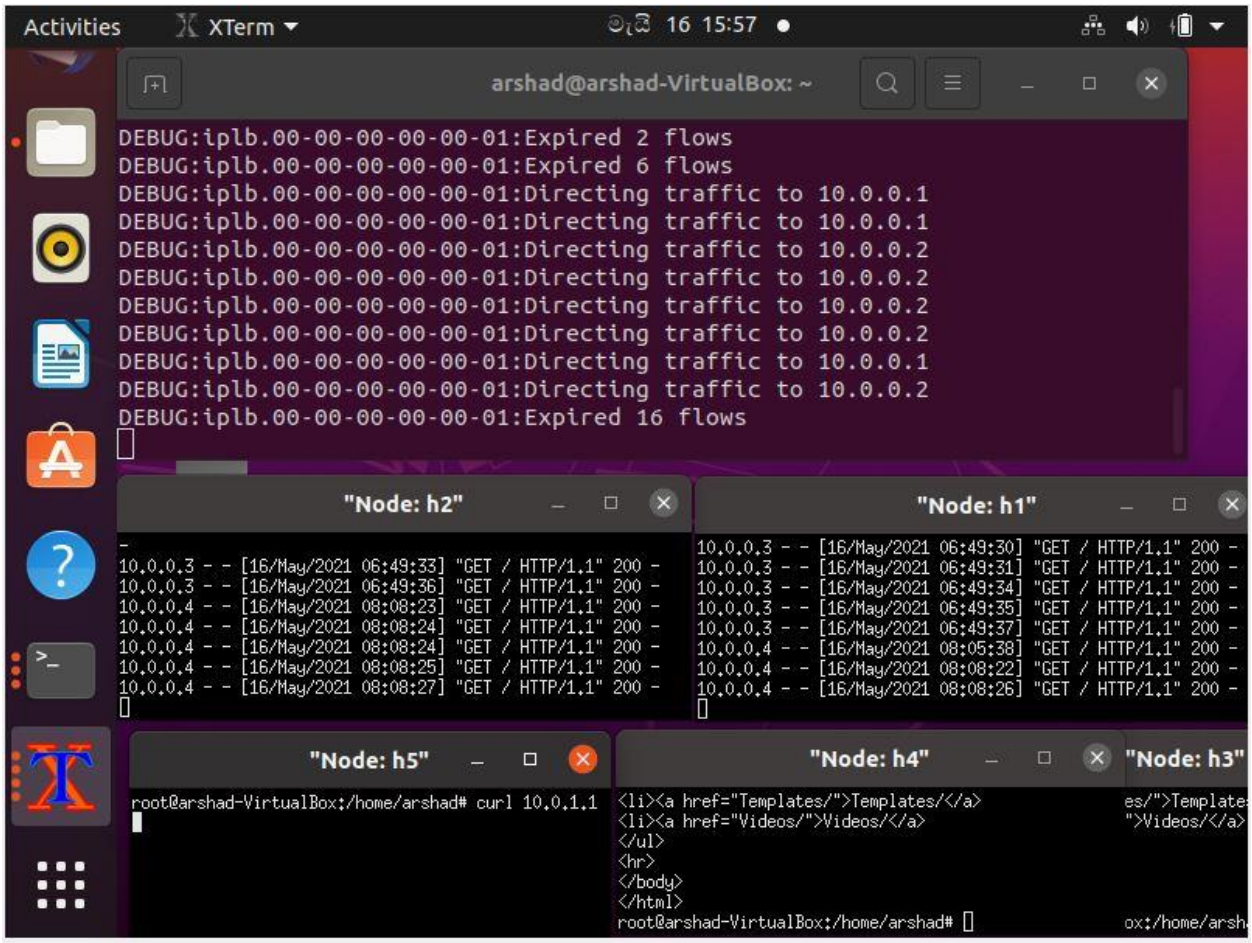


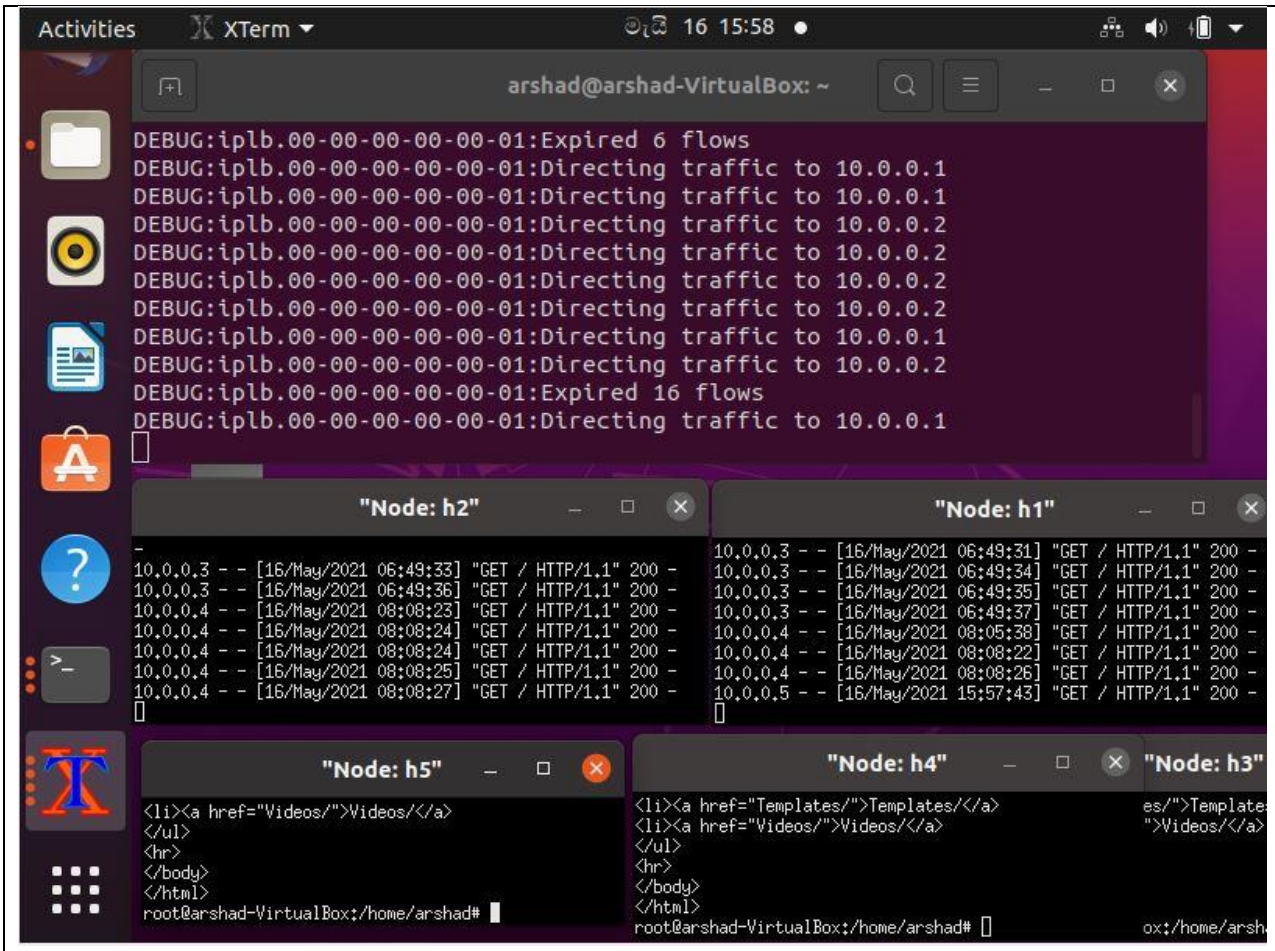
Table 5-11: Test Case 11

TEST CASE ID	TC11
Test description	When curl 10.0.1.1 command executed in Node h8, at terminal suddenly we can see the directing traffic. Also, in the Node h1 or Node h2. Actually, for the testing purpose we execute curl command multiple times and see the result (third image showed the load balancing which one host request for the network). Expected result is concept of the project.

Test steps	1) Type the command in remote Node h8 terminal 2) Click enter button
Test data	Command = curl 10.0.1.1
Expected result	Balance the load in Node h1 and Node h2 with the help of controller. When we gave multiple request in Node h8, it react in both severs.
Actual result	Not expected result appear
Status	Fail, because at Node h8 user misunderstand and two times click the error key. So, network congestion happened. Third picture show miss behave command.

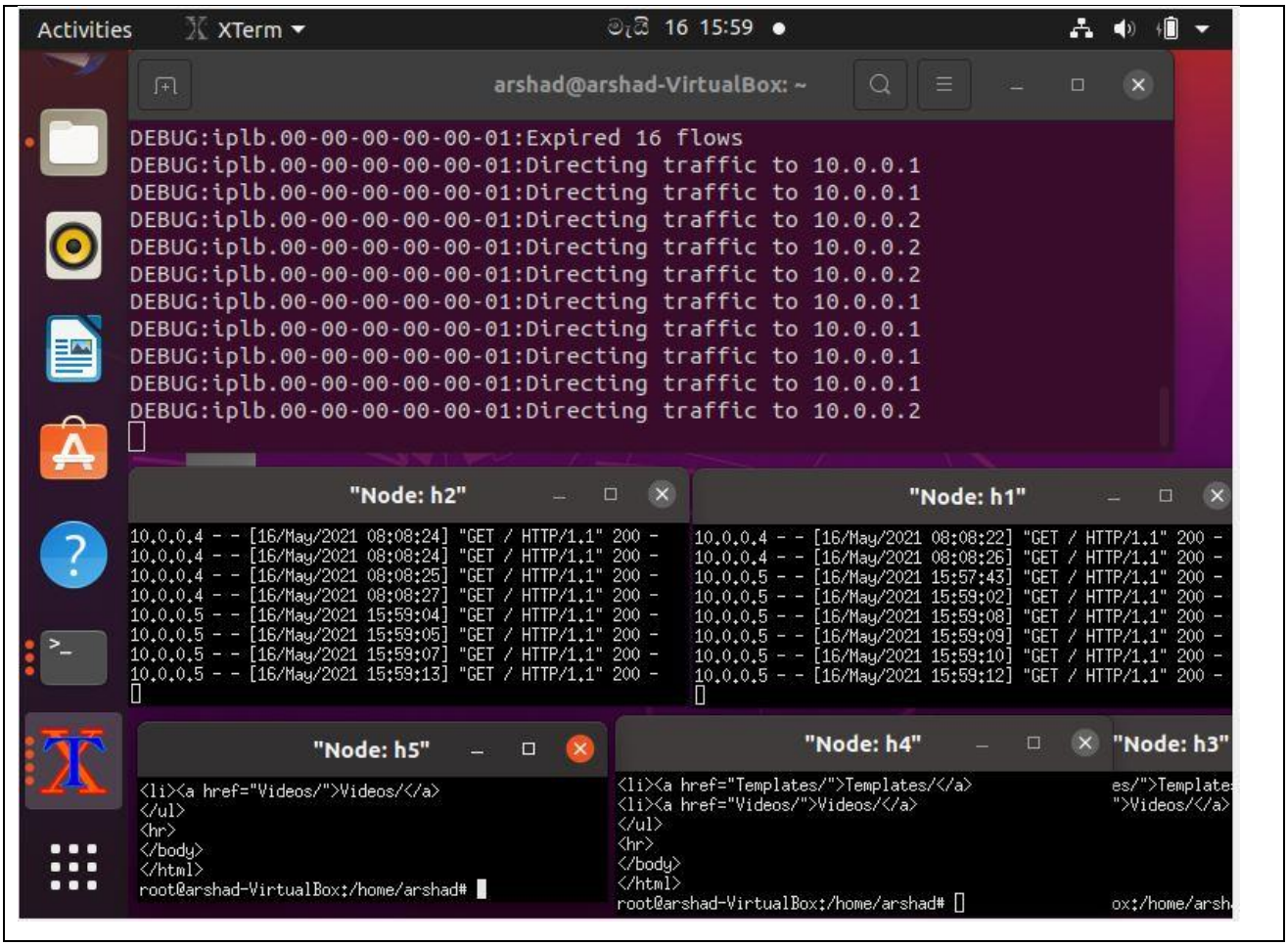

Activities
XTerm
16 15:57
arshad@arshad-VirtualBox: ~
DEBUG:iplb.00-00-00-00-00-01:Expired 2 flows
DEBUG:iplb.00-00-00-00-00-01:Expired 6 flows
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Expired 16 flows
"Node: h2"
-
10.0.0.3 - - [16/May/2021 06:49:33] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:36] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:23] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:24] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:24] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:25] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:27] "GET / HTTP/1.1" 200 -
"Node: h1"
10.0.0.3 - - [16/May/2021 06:49:30] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:31] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:34] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:35] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:37] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:05:38] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:22] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:26] "GET / HTTP/1.1" 200 -
"Node: h5"
root@arshad-VirtualBox:/home/arshad# curl 10.0.1.1
"Node: h4"
Templates/
Videos/

<hr>
</body>
</html>
root@arshad-VirtualBox:/home/arshad#
"Node: h3"
es/">Template
">Videos/
ox:/home/arsh


Activities
XTerm
16 15:58
arshad@arshad-VirtualBox: ~
DEBUG:iplb.00-00-00-00-00-01:Expired 6 flows
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Expired 16 flows
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
"Node: h2"
-
10.0.0.3 - - [16/May/2021 06:49:33] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:36] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:23] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:24] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:24] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:25] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:27] "GET / HTTP/1.1" 200 -
"Node: h1"
10.0.0.3 - - [16/May/2021 06:49:31] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:34] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:35] "GET / HTTP/1.1" 200 -
10.0.0.3 - - [16/May/2021 06:49:37] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:05:38] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:22] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [16/May/2021 08:08:26] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [16/May/2021 15:57:43] "GET / HTTP/1.1" 200 -
"Node: h5"
Videos/

<hr>
</body>
</html>
root@arshad-VirtualBox:/home/arshad#
"Node: h4"
Templates/
Videos/

<hr>
</body>
</html>
root@arshad-VirtualBox:/home/arshad#
"Node: h3"



5.2 Evaluation

Actually, the researcher was gathering requirements and analyzed from articles, websites journal and reports etc. Actually, that time researcher noticed that IT industry was expecting this kind of concept, which can balance dynamic wised, increase system performance and decrease packet loss. For this researcher conducted questioner survey goggle form for statistics. Appendices B Google form, victim for the statistic. Generally, in industry numbers of algorithms were introduce to balance the load in the network but least packet load-balancing algorithm little bit different from it because, its efficiency, security, scalability and effectiveness were high. Therefore, after completing the SDN based load-balancing project, the system was given to set o users like IT professionals, researchers, lecturers and computer science students. For the covid-19 pandemic situation researcher can't go for the company location. Also, IT experts they not interested to give their valuable time to spent in the concept. So, researcher plans to do online demonstration and get

feedback from them. But researcher actually evaluate via social media at the start of the project. According to the user's feedback, more users agree and satisfy with concept but they not agreed with security side because, nowadays ransom wares and malwares are effectively attack on corporate companies, home PCs and shops. Also, in the system researcher highly focus on physical security rather than logical security. Furthermore, users said supportive feedback for the uniqueness of the system. Below table illustrated summary of the SDN based load-balancing in cloud computing with security enhancement concept-based system.

In the form researcher, gave option level questions and text box to give idea to the increase value to the system. So, researcher plan more testing and feedback session after the covid-19 pandemic issue.

6 Results and Discussion

This section will provide a review of the SDN based load-balancing project. In previous segment researcher explain about testing and evaluation. So, in this section researcher need to prove the testing result like an argument. First in result segment, researcher will explain description about main findings about project, whereas the discussion segment, explain the results for readers and future researchers. Also, it's given significance of the findings. Furthermore, researcher explains about technical issues faced at start to finish the project. Moreover, troubleshooting the issues also discuss in this section.

6.1 Results

This segment discovers the results of the SDN based load-balancing project. Actually, researcher's machine, hardware specifications were very poor for the implementation. However, researcher defeated the struggle. As an IT professional we should take the responsibility when trouble happened. Because, every minute important to the company and the business strategy. This project is also, same scenario concept but logically different.

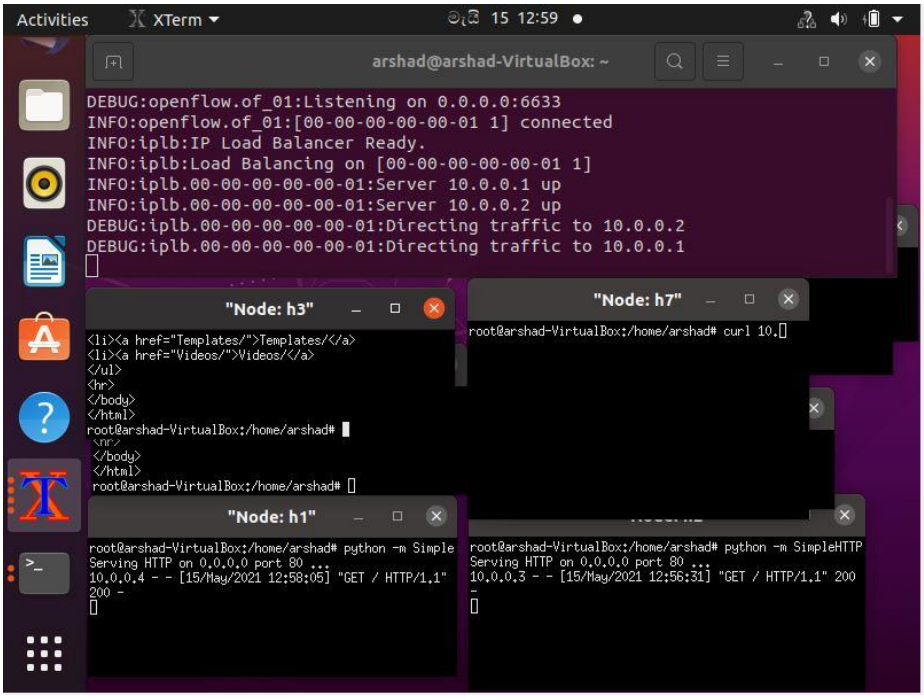


Table 6-1: Load-balancing concept with servers, controller and users' interfaces

For the SDN based load-balancing project researcher need proper internet connection to balance the load. Actually, first researcher determines planning and designing of the project. Moreover, researcher installs Mininet and creates topology using that. Researcher found two ways to create topology. One was using MiniEdit, other one using command. After assign IP to servers and hosts which used in network topology in the project. As a result, Node h1 and h2 are servers. Also, Node h3, h4, h5, h6, h7, h8 are host users in the project. For the connection, researcher used remote connection, which accommodated by XTerm. For the controlling scenario POX is recommend by researcher because it is popular than other controllers. First, researcher installs pox using GitHub and git command and clone in the system. As a result, POX was ready for the work with one controller, two servers and six hosts with assigned IP address and controller's port number is 6633. To gain a good result, researcher used simple HTTP server for fulfill the cloud computing key word. User one by one executed curl command and sees the balancing cases. After, users execute multiply and see the servers and controller acceptance. As a result, according to least packet algorithm successfully balance the load with secure manner, which means according to topology at Node h3, h4, h5, h6 are request HTTP through curl 10.0.1.1. It is controller's IP. So, figure shows the successful result. It shows smoothness of the controller. Also, illustrate how controller was balance the load to server 1 (Node h1) and server 2 (Node h2).

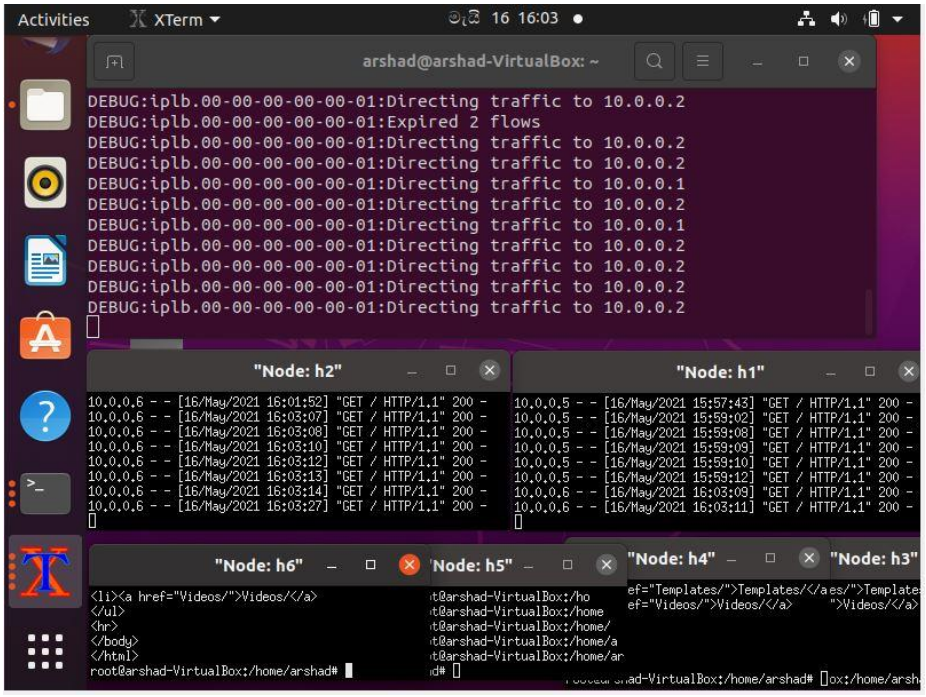


Table 6-2: multiple time requests from users and balancing technique

Researcher used Python language to provide programmed coding according to algorithm. In other word researcher can say python script. For the least packet algorithm researcher used python because, it was trending in automation and IT industry. Python, easy for coding and its can be understandable and readable from every user and future researches. Leastbalancecoding.py is researcher's coding script for the project. It is a concept of the project for load-balancing.

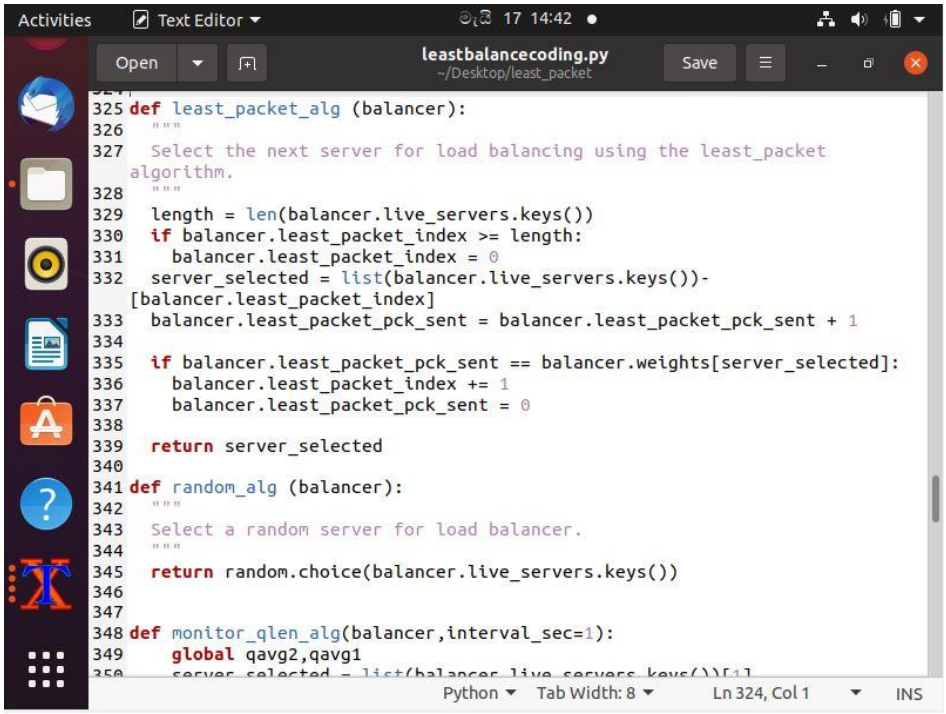


Table 6-3: least balance coding interface

6.2 Discussion

In this segment researcher explain about SDN based load-balancing system efficiency, security and effectiveness. Also, explain about technical issue faced while implementation and testing. First researcher focused on efficiency. Basically, as an IT professional, researcher should practice first to implement the system. So, first researcher create topology basic wised and test it on remotely. After its success, researcher made exact system and test it. At that time researcher examine the issues very well and implementation step was little bit easy to him. Each proposed functions and features tested twelve times to get output and analyzed the different between tests. As a result, ten times output was same as expected. So, researcher determines system was success. Therefore, most of users rated it 75%. Actually, researcher did the system with two algorithms because, to check

the speed and throughput. One is least packet, it is proposed and completed project's algorithm. Other one is Round-Robin. In this two algorithms researcher noticed the best efficiency on least packet because its speed in balancing technique. So, users recommend the system with high reliability.

In this system, researcher experienced lot of technical issues while system implementation. First issue happened in while configure Mininet VM. It is not supported to researcher's hardware specification. For the solution, researcher alternatively develop system using Ubuntu Desktop and install Mininet and other technologies separately. In other issue was remote connection which was curl command not supported. After install curl with help of command in Linux Community Hub's foot path. Linux commands are case sensitive. So, while implementation silly mistakes happen.

7 Conclusion and Future work

7.1 Conclusion

After seven months of long process, researcher completed the SDN based load-balancing system successfully with lots of hard work and dedication. According to prototype methodology, researcher finished the project before deadline. For that, researcher use WBS and Gantt chart. Actually, SDN is an emerging technology and now days IT experts focused on it. Also, researchers provide solutions approaches to security, load-balance and cloud computing technologies. The main problem was single server cannot handle the load. So, user needs multiple servers. But it's expensive and internet users are increasing day to days. Generally, traditional load-balancers are non-programmable and close vendor. That case, SDN arrived and takes responsibility to increase network performance.

In this project research aim was to implement a load-balancing technique to SDN controller to get efficient and secure network infrastructure. Technique was least packet algorithm, which is unique to load-balancing technology. For the system, internet connection was compulsory. Actually, researcher identified and evaluate existing load balancing techniques according the objective of implemented SDN network infrastructure. For improve the key concept to the system researcher spent around four months for gather information about SDN based load-balancing to create new technologies. Researcher identify the load-balancing approach which suitable for SDN and design simulation system for new algorithm concept using Mininet and MiniEdit. To implement and analyse network get packet flow and response time using Wireshark and check how the load is balancing with time and speed using POX controller. After the implement compares with other load balancing approach like Round-Robin and compare security performance with existing approaches and produce ideas to mitigate the threat. So, researcher successfully completed the system within the time period. According to user feedback system were very efficient and effective.

7.2 Limitation

A system approach can often misdiagnose the problem, or miss-prescribe the solution due to its limitation. So, SDN based load-balancing had some limitation. There are few areas to consider for include value to the system. This SDN based system is concept-based system, which was load-

balancing in cloud computing, so user need to have proper internet connection. Researcher include "proper internet connection", which means for the load-balance server used HTTP protocol, without internet connection proper load-balancing possibility really low because its speed, throughput and time. So, expected concept does not work under the internet connection problem.

In other case researcher's main aim and objectives was security enhancement, which means logically or physically intruders can't access the network without administrator's permission. So, concept was two major areas, which are cloud computing and security. Also, SDN based so user concentrate programmed coding and it's working flow. Actually, concept-based project limitation responsibility was handling by administrators of companies and workspaces.

7.3 Benefit of the SDN based load-balancing system project

In this project user can get multiple benefit. According to the algorithms and the keyword readers can gained the benefit by eye catching. Actually, benefit means profit or an advantage gained from something like a place, business, project, charity and gift. But in this case benefit gained from SDN based load-balancing project. So, user used Software Defined Networking, it's a technology which was hardware independent, provide high availability, scalability, agility, versatile network performance and the safest security. Actually, in this project SDN used to load-balancing, because traditional load-balancers close vendor, hardware dependent and non-programmable. If user's workspace was multilevel, that was expensive. But, traditional load-balancer has fixed algorithms, which cannot edit or changed by administrator or users. For this reason, SDN come to stage. So, this project contains benefit of the SDN technology. Researcher's main concept was load-balancing. So, in this project researcher used lest packet load-balance algorithm, it was increase network performance. Because, it's dynamic concept and speed and throughput were high. Server selected based number of packets forwarded to particular server according to the algorithm. After, it selects minimum load server by dynamic wised. So, no packet loss or congestion happened in the system. Security enhancement is one of the features. So, logical and physical wised better security shield are in the project. Researcher only used port 6633 for controller. So, researcher activates port security and includes access control for it.

7.4 Future Work

In this generation lot of technologies were introduce and working successfully with them. ML, AI, automation, Industry 4.0, IOT, RPA, Data Science, AR, VR, Block chain and Edge computing are emerging and trending technologies which are gave easiness for human's day to day life. As an IT professional researcher need to know about SDN. Also, load-balancing in systems for speed work process at workspace. Actually, SDN is emerging idea in the world; it gave network infrastructure flexible, agile, programmable and manageable with the help of dividing the control plane and data plane. Nowadays, network automation impacts the IT industry, which means DevOps concept base strategy of business activity. Generally, companies want to increase their product with the help of digital world. Also, they want speed and the efficient in the service process. Even, banks and government offices also, reduce the time and speed up the services. In this case IT people wants to balance the load for efficient work process. Actually, when the load is high on the network, congestion and packet loss happened simultaneously in networking system. So, SDN based load-balancing concept help in this situation. This project concept provides the basis for future researchers and studies. To load balance cases, analyze about traffic packet header and forwarded to define services and investigate about its behavior like packet flow and security. Another approach is configuring some automation application technology manage with ansible, chef or puppet and deploy docker containers to project and orchestrate with Kubernetes services. For the future research implement those ideas and investigate the performance and its management skills. Additionally, add for hosting process at Azure, Amazon Web Service and Google cloud platform. Another future approach is implementing multi-controller instead of a single controller to prevent simple point failure in the system network. Researcher will assume he can gain speed, good throughput and services. Another future research approach is developed algorithm input to investigate about IP filtering in controller for avoid DoS and DDoS attacks in network. Another approach is preventing congestion and increase Quality of Service (QoS), which means in network load is greater than network capacity, congestion will occur as well as it impacts to network quality, so delay happen in network and day to day work collapse. For this case researcher recommended to increase capacity and efficiency via algorithms.

8 References

(2013). Retrieved from Floodlight: http://www.projectfloodlight.org/floodlight/.

(2013). Retrieved from Beacon: htt[2s:llo[2enflow.stanford.edu/dis[2lax/Beacon/Home.

(2013). Retrieved from NOX: : htt[2:llnoxre[2o.org/wp/.

(2019). Retrieved from POX Manual Current documentation: https://noxrepo.github.io/pox-doc/html/

Calheiros, Rodrigo N, Rajiv Ranjan, Anton B and C AF De Rose. (2011). CloudSim:a toolkit for modeling and simulation of cloud computing. In *Practice and experience 41.1* (pp. 23-50).

Dhumal, S. (2020). *Load Balancing in Cloud Computing.* scholarworks.

F. Bannour, S. Souihi, and A. Mellouk. (2017). *Distributed SDN ControlSurvey, Taxonomy and Challenges," IEEE Communication. Survey. Tutorials, vol.20, issue 1.*

G. Tiwari, V. Chakaravarthy, and A. Rai. (2019). Dynamic load balancing in software defined networking. *Int. J. Eng. Adv. Technol., 8*(5), pp. 2706–2712.

Gawali, M.B., Shinde. (2018). *Task scheduling and resource allocation in cloud computing using a heuristic approach.* Retrieved from https://journalofcloudcomputing.springeropen.com/articles/10.1186/s13677-018-0105-8

Grance, P. M. (October 7 .2009). *"NIST definition of cloud computing".* NIST.

H. Uppal and D. Brandon. (2010). *OpenFlow Based Load Balancing.* Retrieved 2020, from https://courses.cs.washington.edu/courses/cse561/10sp/project_files/cse561_openflow_project_report.pdf

J. Saisagar, D. Kothari, R. Kothari, and V. Chakravarthy. (2017). SDN enabled packet-based load-balancing (PLB) technique in data center networks. In D. K. J. Saisagar, *ARPN J. Eng. Appl. Sci.* (pp. vol. 12, no. 16, pp. 4762–4768).

K. Soleimanzadeh, M. Ahmadi, M. Nassiri. (2019). SD-WLB: An SDN-aided mechanism for web load balancing based on server statistics. *ETRI Journal, 41*(2), pp. 179-206.

Kernel-based Virtual Machine (KVM). (2013). Retrieved from http://www.linux-kvm.org/.

L. Padilha and D. Batista. (2019). Effectiveness of Implementing Load Balancing via SDN. Brazil: In Proceedings of the 37th Brazilian Symposium on Computer Networks and Distributed Systems.

M. Elgili. (2017). Load Balancing Algorithms Round-Robin (RR), Least-Connection and Least Loaded Efficiency. *GESJ: Computer Science and Telecommunications, 1*(51), pp25-29.

M. Koerner and O. Kao. (2012). Multiple service load-balancing with OpenFlow. Belgrade, Serbia: in 2012 IEEE 13th International Conference on High Performance Switching and Routing.

Montazerolghaem, A. (2019). *SIP Server Load balancing Based on SDN.* Retrieved 2020, from https://arxiv.org/ftp/arxiv/papers/1908/1908.04047.pdf

N. Joshi and D. Gupta,. (2019). A Comparative Study on Load Balancing Algorithms in Software Defined Networking,". *International Conference on Ubiquitous Communications and Network Computing UBICNET.* India.

N. McKeown et al. (2008). OpenFlow: Enabling innovation in campus networks,. Comput. Commun Rev., vol. 38, no. 2,.

N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. (2008). OpenFlow: enabling innovation in campus networks. In *SIGCOMM Comput. Commun. Rev., vol. 38, no. 2* (pp. pp. 69-74).

ONF. (n.d.). Retrieved 2019, from OpenFlow Switch Specification 1.4.0: https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-spec-v1.4.0.pdf

Open Networking Foundation (ONF). (2013). *Software-Defined networking: the new norm for networks.* Retrieved from https:llwww.opennetworking.org/images/stories/downloadslopenflow/wpsdn-newnorm.pdf.

OpenFlow Consortium. (2013). Retrieved from http://openflowswitch.org

P. Goransson and C. Black and T. Culver. (2017). Software Defined Networks a comprehensive approach, ch 4. In Boston: Morgan Kaufmann, *How SDN Works* (pp. pp. 61–79).

P. Suwandika, M. Nugroho, M. Abdurahman. (2018). Increasing SDN Network performance using load balancing schema on webserver. Bandung, Indonesia: in IEEE 2018 6th International Conference on Information and Communication Technology (ICoICT).

Phaal, P. (2004). Retrieved 2019, from http://sflow.org/sflow version 5.txt.

Randles, M., D. Lamb, A. Telab-Bendiab. (2010). A Comparative Study in Distributed Load Balancing Algorithms for Cloud Computing. Perth, Australia: IEEE 24th International Conference on Advanced Information Networking and Applications Workshops (WAINA).

Rimal, B. Prasad, E. Choi, I. Lumb. (2009). A taxonomy and survey of cloud computing systems. *5th International Joint conference on INC, IMS and IDC.* IEEE.

S. Abed, D.S. Shubair. (2018). Enhancement of task scheduling technique of big data cloud computing. In *Advances in Big Data Comput. and Data Commun.Syst* (pp. 1-6).

S. Kaur, K. Kumar, J. Singh, and N. Ghumman. (2015). Round-robin based load balancing in Software Defined Networking. New Delhi, India: 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom).

S. Vyakaranal and J. Naragund,. (2019). *Weighted Round-Robin Load Balancing Algorithm for Software-Defined Network.* Retrieved 2020, from https://doi.org/10.1007/978-981-13-5802-9_35

Singh, J. (2016). Weighted Round-Robin Load Balancing Using Software Defined Networking. *in International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE), 6*(6), pp. 621–625.

Statista. (2017, July). Retrieved 2021, from Software-defined networking (SDN) market size worldwide from 2013 to 2021: https://www.statista.com/statistics/468636/global-sdn-market-size/

Sukhveer Kaur, Japinder Singh, Navtej Singh Ghumman. (2014). Network Programmability Using POX Controller. *International Conference on Communiction,Computing & System.* Ferozepur, Punjab, India.

trianz. (2020). Retrieved from trianz cloud evalution: https://www.trianz.com/insights/revolution-that-is-cloud-computing#1

U. Zakia, H. Yedder,. (2017). Dynamic Load Balancing in SDN-Based Data Center Networks. *18th IEEE annual information technology, electronics and mobile communication conference (IEMCON).* Canada.

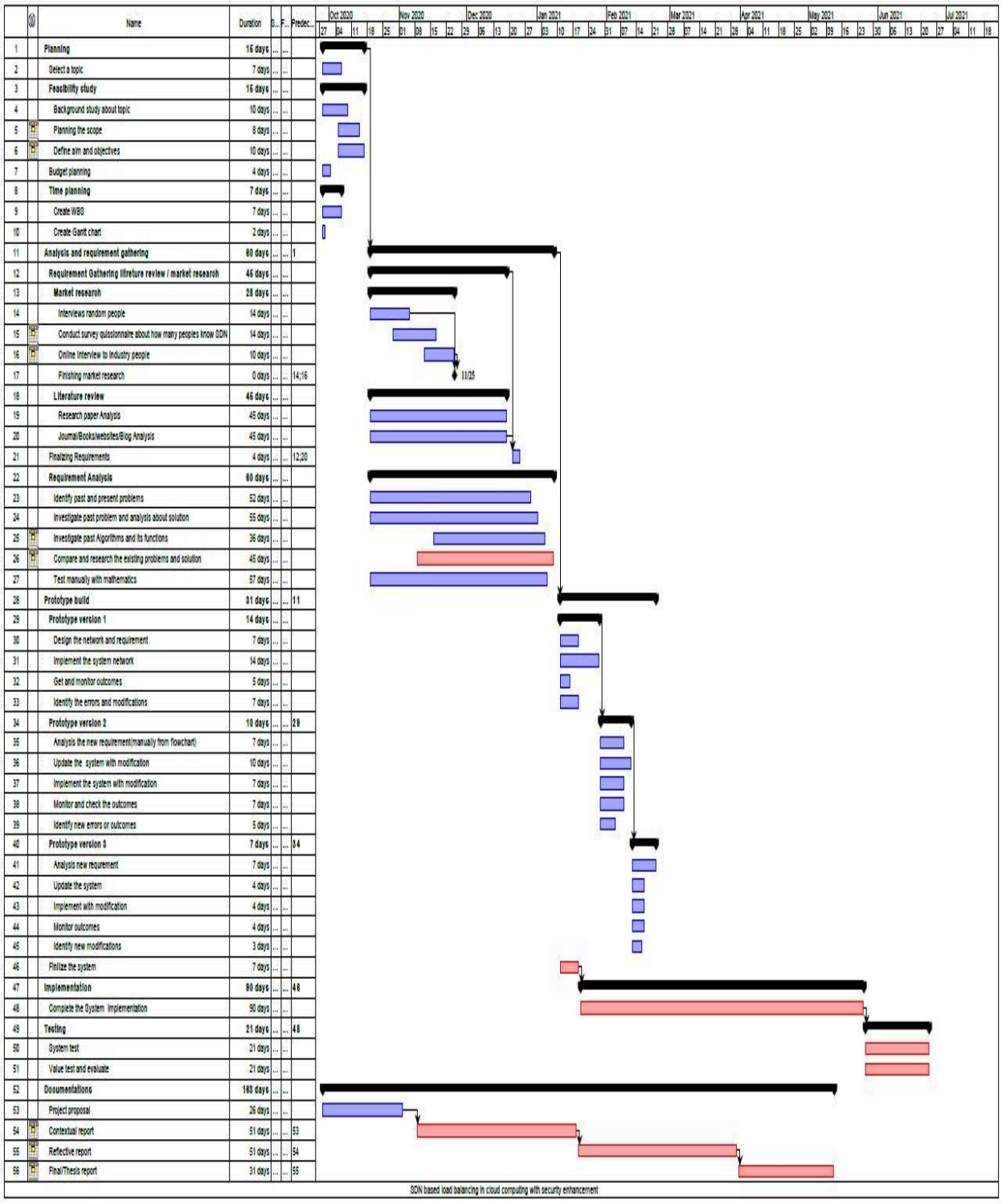
Vmware. (2013). Retrieved from http://www.vmware.coml.

W.Prakash. (2019). DServ - LB: Dynamic server load balancing algorithm. *International Journal of Communication Systems - Wiley Online Library, 32*(1,2019).

Xen. (2013). Retrieved from : http://www.xen.org/.

Y. Shengsheng, Y. Lihui, L.Song, and Z. Jingli. (2003). A variable weighted least-connection algorithm for multimedia transmission. *Journal of Shanghai University, 7*, pp 256-260.

Appandices - A



	Name	Duration	Predec...
1	Planning	16 days	
2	Select a topic	7 days	
3	Feasibility study	16 days	
4	Background study about topic	10 days	
5	Planning the scope	8 days	
6	Define aim and objectives	10 days	
7	Budget planning	4 days	
8	Time planning	7 days	
9	Create WBS	7 days	
10	Create Gantt chart	2 days	
11	Analysis and requirement gathering	60 days	1
12	Requirement Gathering litreture review / market researoh	46 days	
13	Market researoh	28 days	
14	Interviews random people	14 days	
15	Conduct survey quisionnaire about how many peoples know SDN	14 days	
16	Online interview to industry people	10 days	
17	Finishing market research	0 days	14;16
18	Literature review	46 days	
19	Research paper Analysis	45 days	
20	Journal/Books/websites/Blog Analysis	45 days	
21	Finalizing Requirements	4 days	12;20
22	Requirement Analysis	60 days	
23	Identify past and present problems	53 days	
24	Investigate past problem and analysis about solution	55 days	
25	Investigate past Algorithms and its functions	36 days	
26	Compare and research the existing problems and solution	45 days	
27	Test manually with mathematics	57 days	
28	Prototype build	91 days	11
29	Prototype version 1	14 days	
30	Design the network and requirement	7 days	
31	Implement the system network	14 days	
32	Get and monitor outcomes	5 days	
33	Identify the errors and modifications	7 days	
34	Prototype version 2	10 days	29
35	Analysis the new requirement(manually from flowchart)	7 days	
36	Update the system with modification	10 days	
37	Implement the system with modification	7 days	
38	Monitor and check the outcomes	7 days	
39	Identify new errors or outcomes	5 days	
40	Prototype version 3	7 days	34
41	Analysis new requirement	7 days	
42	Update the system	4 days	
43	Implement with modification	4 days	
44	Monitor outcomes	4 days	
45	Identify new modifications	3 days	
46	Finlize the system	7 days	
47	Implementation	90 days	48
48	Complete the System Implementation	90 days	
49	Testing	21 days	48
50	System test	21 days	
51	Value test and evaluate	21 days	
52	Documentations	160 days	
53	Project proposal	26 days	
54	Contextual report	51 days	53
55	Reflective report	51 days	54
56	Final/Thesis report	31 days	55

Appandices B – Google Form

Research about Software Defined Networking based load balancing approach impact in the cloud computing and ICT industry.

This form is conducted by BSc (Hons) Computer network final year student for an undergraduate project. project is SDN based load-balancing in cloud computing with security enhancements. This survey all about market research.

* Required

1. Name *

2. Gender *

Mark only one oval.

- ◯ Female
- ◯ Male
- ◯ Prefer not to say
- ◯ Other: _____

3. Email *

4. Company / University / Institute / School name

5. Which explains your current role? *

Mark only one oval.

- ◯ student
- ◯ System Admin / Engineer
- ◯ Software Developer / Engineer
- ◯ Network Admin / Engineer
- ◯ Data Scientist / Analyst
- ◯ Other: _____

6. How long have you been employed in your organization?

Mark only one oval.

- ◯ Less than 1 year
- ◯ 1 to 5 years
- ◯ 6 to 10 years
- ◯ 11 to 20 years
- ◯ More than 20 years

7. Did you familiar with cloud computing? *

Mark only one oval.

- ◯ Yes
- ◯ No

14. What controllers are industries mostly used in SDN based projects?

Check all that apply.

- [ ] NOX/POX
- [ ] Beacon
- [ ] cherry
- [ ] OPENDAYLIGHT
- [ ] Project Floodlight
- [ ] Other: _____

15. OpenFlow is one of the most well-known and widely used protocols for communication between the Data Plane and Control Plane in an SDN is OpenFlow. It is a standardized communication interface between the control and forwarding levels and allows direct access to the forwarding functions of switches or routers. Access can be both direct and virtualized.

Mark only one oval.

- ◯ Yes
- ◯ No
- ◯ Don't know
- ◯ Other: _____

16. Do I need to use OpenFlow in software-defined networks

Mark only one oval.

- ◯ Yes
- ◯ No

1/12/2021 Research about Software Defined Networking based load balancing approach impact in the cloud computing and ICT industry.

17. computer security is only helping to prevent viruses and malware.

Mark only one oval.

- ◯ Strongly Disagree
- ◯ Disagree
- ◯ Neutral
- ◯ Agree
- ◯ Strongly Agree

18. what security attacks you most experienced in your day to day life?

Check all that apply.

- [ ] DoS
- [ ] phishing attack (whale, spear)
- [ ] malware
- [ ] DDoS
- [ ] MITM
- [ ] Other: _____

19. LinkedIn

_____

https://docs.google.com/forms/d/1BmNmkG7XyJBX4N4MbLr0fUv7h-MR3XMpUx0he28/edit

Appendices C - POSTER

SDN Based Load-Balancing in Cloud Computing with Security Enhancement

Author: Arshad Haris (2016051) | Supervisor: Ms. Dilushinie Fernando
BSc(Hons) Computer Networking

University of Bedfordshire

Introduction

In industry technologies and internet usage rapidly increased. As a result, there is congestion, which arise packet loss and decrease system efficiency and performance. So single server cannot handle the huge traffic. So, industry need load-balancing approach in cloud computing with secure manner. Moreover, there are so many approaches provide by researchers to balance the load such as Roud-Robin, weighted Roud-Robin and least bandwidth algorithms. In this project introduce least packet load-balancing idea to the IT industry.

Technologies Used

- Oracle Virtual Box
- Wireshark
- XTerm
- Ubunthu OS
- POX Controller

Research Problem

The traditional network's load-balancers are expensive, close vendor and non-programmable. Also hardware depended.So, if we want multiple servers, it's so expensive and some times its not support to the network. Security must in this generation.

Research Aim

To implement a load-balancing technique to SDN controller to get efficient and secure network infrastructure.

Solution

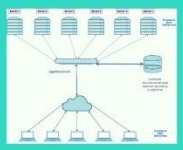
SDN based loadbalancing is solution given by researcher. The use of a load-balancer to distribute network traffic among multiple serverscould decrease the load on a single server give availability and scalability and efficiency in network. So, project concept that utilized load-balancer is programmable.

Results

Research Objectives

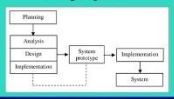
- To identify and evaluate existing load balancing techniques implemented in SDN field network infrastructure.
- To gather information about SDN based load-balancing to create new technologies.
- To identify the load-balancing approach which suitable for SDN and design simulation system for new techniques.
- To implement and analyse network get packet flow, response time, and check how the load is balancing with time and speed.
- After the implement compares with other load balancing approach and compare security performance with existing approaches and produce.
  ideas to mitigate the threat.

Topology



Methodology

Project used prototyping methodology because it is built, tested and reworked as necessary until an acceptable outcome achieved from which complete system. the prototype is iterative and takes trial-and-error process between user and company. So missing functionality and errors one detected easily and prototype can be reused in future for projects.



Features

* Analysis of the outcome results from network traffic and comparison with small, medium, and large loads.
* Monitor and Analysis security check the network.
* If threat will detect give physical or logical solution.

Benefits

- Increase network and system performance.
- High availability, integrity, agility and scalability.
- Programmable. So, hardware independent.
- Good security concept like port security
- No congestion or packet losses happen. Also, it was dynamic.

References

t Dhumal, S., 2020. Load Balancing in Cloud Computing, scholarworks: s.n.
F. Bannour, S. Souihi, and A. Mellouk, 2017. Distributed SDN ControlSurvey, Taxonomy and Challenges," IEEE Communication. Survey. Tutorials, vol.20, issue 1, s.l.: s.n.
G. Tiwari, V. Chakaravarthy, and A. Rai, 2019. Dynamic load balancing in software defined networking. Int. J. Eng. Adv. Technol., 8(5), p. pp. 2706–2712.