Efficient Controller Positioning in SDN

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Technology
in
Computer Science and Application

Submitted By
Pratibha Kanwar
Reg. No: 601634012

Under the supervision of:
Dr. Rajesh Kumar
Professor

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY
PATIALA – 147004

June 2018

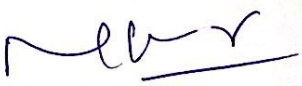
Certificate

I hereby certify that the work which is being presented in the thesis entitled, *"**Efficient Controller Positioning in SDN**"*, in partial fulfillment of the requirements for the award of degree of Master of Technology in *Computer Science and Application* submitted in Computer Science and Engineering Department of Thapar Institute of Engineering and Technology, Patiala, is an authentic record of my own work carried out under the supervision of *Dr. Rajesh Kumar* and refers other researchers' work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for the award of any other degree of this or any other University.

Pratibha Kanwar

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.



Dr. Rajesh Kumar
Professor
(Computer Science and Engineering Department)

Acknowledgement

First and foremost, I would like to express my sincere gratitude to my supervisor and my mentor **Dr. *Rajesh Kumar*** who has supported and guided me throughout my thesis work. He has been providing me with his invaluable advice and encouragement from the very beginning of my M.Tech. Program. Without his unfailing support and belief in me, this thesis would not have been possible. I appreciate Computer Science and Engineering Department of Thapar Institute of Engineering and Technology for providing the necessary research facilities. I am also thankful to Dr. Maninder Singh, Head CSED and also Dr. Sanmeet Bhatia, PG Coordinator and all the respected faculty members of the department for their teaching and guidance. I would also want to extend my obligation towards Nava Nalanda Central Library for providing access to the prominent research journals.

I would like to express my deepest gratitude to my parents and friends for their whole hearted cooperation. They have always offered me guidance and support when I needed the most.

Most of all, I am grateful to Mr.Vanshaj Rana for his understanding, patience, encouragement and for pushing me farther than I thought I could ever go. He kept me going, and this research would not have been possible without him.

Abstract

There has been a major advancement in the technology as it is changing at a faster rate every day, developing things better and increasing the efficiency by generating new generation technology. Software Defined Networks (SDN) have developed a lot since it came into the existence and is still evolving, giving new and better application prospects. Controller Placement Problem is a very important prospect of SDN networking and this research aims at contributing something efficient and better than the already existing work. Controller Placement is one of the major concerns of SDN networking, designing and placement are essential as to get the best fit solutions for the efficient functioning of a network. Traffic Engineering (TE) is an important network application in Software Defined Networks (SDN) which contributes by providing the measurements and the management of network traffic, designing better routing mechanisms for analyzing and predicting the network flow. This research focuses over the controller placement in SDN for efficient networking throughputs and to perform Traffic Engineering (TE) for analyzing and predicting the flow in the network to sort the traffic management and load balancing in SDN along with some of the important factors which can be improvised and used for an overall smart performance of the network.

Table of Content

List of Figures

List of Tables

List of Abbreviations

- FIB – Forwarding Information Base
- SDN – Software Defined Network
- RIB – Routing Information Base
- GUI – Graphical User Interface
- DALB – Dynamic and Adaptive Algorithm for Load Balancing
- RCD – Reliable Controller Deployment Strategy
- WAN – Wide Area Networks
- API – Application Program Interface

CHAPTER 1

Introduction

Software Defined Network (SDN) is a paradigm that brings change by breaking vertical integration and separating the network's control logic from the underlying routers and switches promoting logical centralization of network control and the ability to program the network. SDN moves the architecture from traditionally built distributed model to a more centralized approach. SDN allows the decoupling of control plane and the data plane. There can be a number of advantages such as a centralized controller which is appropriate for managing the complete network layout by a manually scripted control application and switches that requires the forwarding of packets based over the flow table entries. SDN has two main elements i.e. decoupling of data (infrastructure plane) and control plane and the programmability (ability to improve network flexibility, dynamically providing traffic flows and providing application level quality of service).

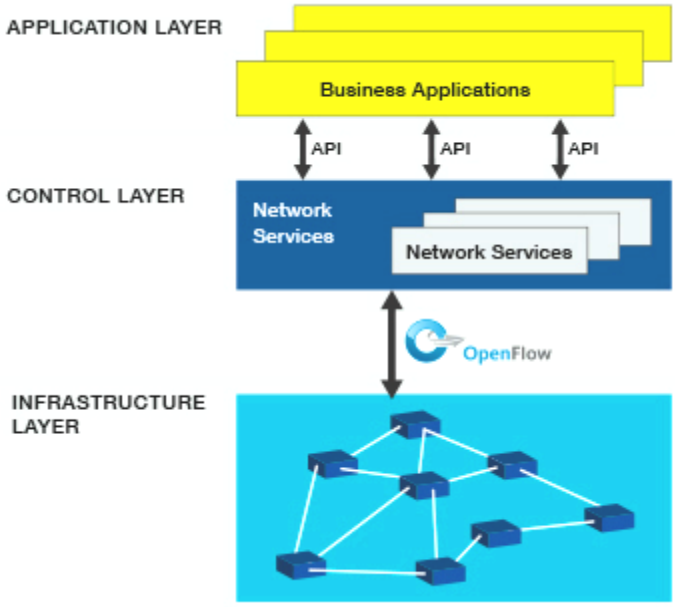


Figure 1.1 Different Layers in SDN [15]

Figure 1.1 depicts the three main layers of the SDN architecture, containing the application layer which consists of the business applications, the control layer containing the network services and the infrastructure layer. SDN offers a simplified means to dynamically control multiple simple switches via a single controller

program. Software Defined Network technology is a term composed of several series of network technologies which aims at building a network structure more flexible and effectively functional. It is an advance strategy of networking that helps to eliminate the complex and static behavior of the distributed network layout by using standards-guided abstraction between the data forwarding plane and the control plane, containing both physical and virtual devices. SDN technology enables improvisation in the network layout and automation along with the reduction in the expenses of the network operations being performed. If we use an already market existing standardized infrastructure plane abstraction protocol i.e. OpenFlow, users can freely access to any of the infrastructure plane devices, since almost all the available network hardware are addressable by the most common abstraction protocol. In SDN architecture OpenFlow stands as the first standard communicating factor present between the control and forwarding layer. OpenFlow allows to directly access and also the modification of the forward plane and the devices in the network i.e. switches and routers. Working in an OpenFlow based environment is defined as any device which tends to have any sort of communication with a Controller must be supporting the OpenFlow protocol. By using this interface, Controller gains the access for making the suitable changes to the entries in the flow-table allowing the network administrator to let the partitioning of the traffic and control flows for the appropriate performance.

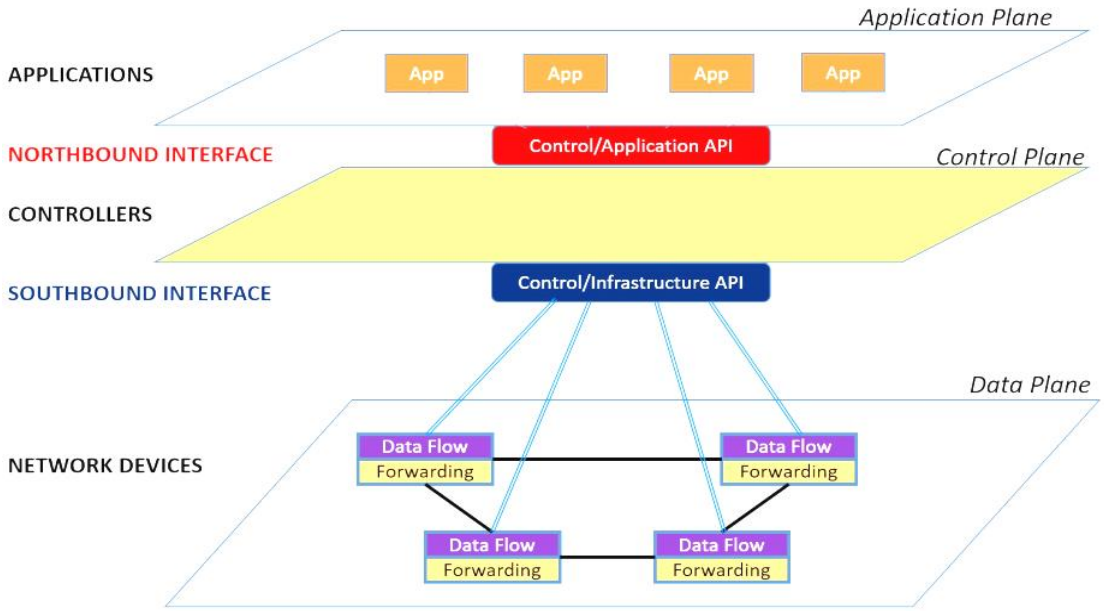


Figure 1.2 Layers Infrastructure and Network Devices in SDN [15]

Figure 1.2 depicts the infrastructure of the layers in the network along with the network devices contained in the data plane all linked to the control plane via the infrastructure Application Program Interface (API) and the application plane via the application API. The design of a SDN controller is a very important concern and must be given proper attention while building. Several controllers have been developed and their performances have been evaluated. This evaluation has filtered some of the best and average working controllers i.e. the best performing and the best fitted controllers with several criteria followed for the evaluation process. SDN controllers act as an application in the networking field and helps managing the traffic to enable efficient and automated networking. SDN controllers are all protocol based, such as the OpenFlow protocol that provides the access to the servers for indicating the switches about where to deliver the packets. There are several controllers into existence and we will discuss some of them on the basis of their performances, protocol support, load balancing, network monitoring, traffic engineering, GUI and platform support and a few more features. Controller's performance in a network is based upon its various attributes which defines a controller's overall efficiency. There is a huge importance for a controller within an SDN structure along with the diversity of structure and the implementation within the research field, it required to benchmark all the given choices against the entirely different performance indicators. The controllers within a network aim to contribute for the success or the failure of SDN.

There are multiple controllers available in the market. These controllers can be compared for their efficient functioning and further can be used within a network for the suitable purpose in the field of networking. Controller building aimed at enhancing the SDN and to increase the management skills within a network. Most of these controllers built are OpenFlow based and are helpful in having a better control over the network with various other advantages, providing the best fit alternatives to the network performance related issues or failures.

Table 1.1 contains a list of ten controllers, already existing in the market and gives a comparison amongst the various features of each controller based upon their performance, network monitoring, load balancing, GUI and platform used etc. This comparative study reflects some of the most and the least efficient controllers out of the all ten listed below. All these controllers aim at contributing for the betterment as well as more effective networking in SDN.

TABLE 1.1: List of SDN Controllers using OpenFlow Protocol

CONTROLLERS	PURPOSE	OPENFLOW SUPPORT	LOAD BALANCING	NETWORK MONITORING	TRAFFIC ENGINEERING	GUI	PLATFORM SUPPORT	DITRIBUTED/ CENTRALISED
Open Daylight	Aimed at enhancing software-defined networking (SDN) by offering a community-led and industry-supported framework for the Open Daylight Controller, which has been renamed the Open Daylight Platform.	support for the Open Flow protocol	Yes (Supports completely)	Yes (Supports completely)	Yes (Supports completely)	Web based	Linux, MAC OS & Windows	Distributed
Ryu	Designed to increase the agility of the network by making it easy to manage and adapt how traffic is handled.	support for the Open Flow protocol	No support	Yes (Supports completely)	Partial support	Python based	Most supported on Linux	Centralized
Floodlight	Works with the Open Flow protocol to orchestrate traffic flows in a (SDN) environment. Enables businesses to better adapt to their changing needs and have a better control over their networks.	support for the Open Flow protocol	No support	Yes (Supports completely)	Partial support	Web/Java based	Linux, MAC OS & Windows	Centralized
NOX	Serves as a network control platform that provides a high level programmatic interface for management and the development of network control applications.	support for the Open Flow protocol	No support	Partial support	Partial support	Python +QT4	Most supported on Linux	Centralized
Trema	Provides everything needed to create Open Flow controllers in Ruby. It provides a high-level Open Flow library and also a network emulator that can create Open Flow-based networks for testing on your PC.	support for the Open Flow protocol	No support	Partial support	Partial support	C, Ruby based	Linux, MAC OS & Windows	Centralized
Iris	Open flow-based SDN controller designed to solve scalability and availability issues of SDN.	support for the Open Flow protocol	Yes (Supports completely)	Yes (Supports completely)	Yes (Supports completely)	Web based	Linux, MAC OS & Windows	Centralized
POX	Allow users to write their own applications that uses controller as an intermediary or abstraction layer between network applications and the network equipment.	support for the Open Flow protocol	Yes (support completely)	Partial support	Partial support	Python +QT4	Most supported on Linux	Centralized
ONOS	It can scale out to accommodate a physically distributed system of devices. For overall system visibility & management, ONOS provides a relatively straightforward GUI.	support for the Open Flow protocol	No support	Yes (Supports completely)	Partial support	Web based	Linux, MAC OS & Windows	Distributed
Mul	Designed for performance, reliability and availability which is the need of the hour for deployment of SDN in mission-critical networks. It is also highly flexible, modular and easy to learn.	support for the Open Flow protocol	--	Yes (Supports completely)	Yes (Supports completely)	Web based	Most supported on Linux	Centralized
RUNOS	It is fully user space controller with high functionality, easy to develop your apps, relatively high performance comparing with existing controllers.	support for the Open Flow protocol	Yes (Supports completely)	Yes (Supports completely)	--	Web based	Most supported on Linux	Distributed

1.1 The Traditional Approach

In the traditional approach there is a coupling of control plane and data plane, a traditional router or a switch contains a high velocity data plane where the flow packets are directed and the control plane contains the functions for managing routing, forwarding, service quality, access control etc.

This requires the configurations to be done manually. There is also a single path from the source of communication flow to its destination which affects the performance as well.

The major factor that led to the slow network integration has always been integrated closed proprietary industry. This tired structure of specialized applications over specialized control program over the specialized hardware leads to a very little configuration and customization options. It is quite difficult and almost impossible to develop and test new protocols and prototypes due to this rigid environment.

The need for flexible and directly programmable agile networks which emerged as a challenge after the introduction of cloud computing, distributed database and other resource intensive applications demands an abstraction for the TCP/IP in order to make the network manage centrally and to appreciate the global application development of the network research groups.

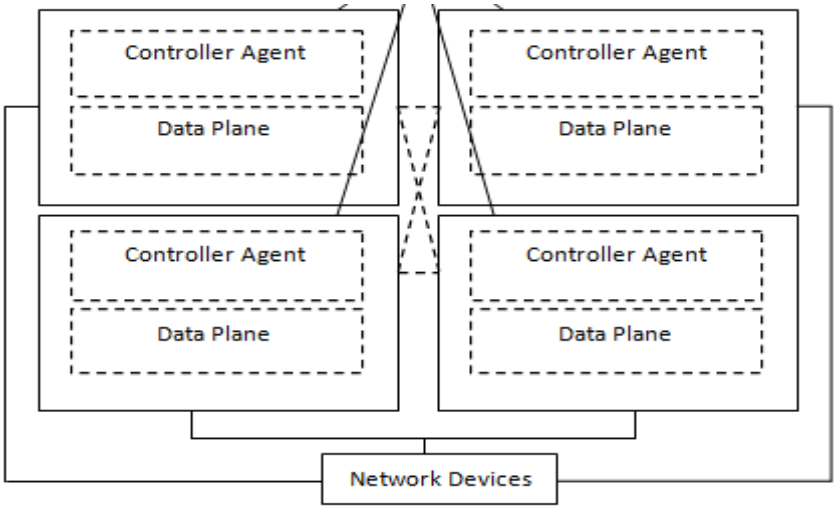


Figure 1.3 Traditional Networks

Figure 1.3 depicts the internal structure for the network devices in the traditional networks containing the controller agent and the data plane all combined together within the devices.

1.2 The Modern Approach

The need for easily accessible and efficiently programmable networks led to the partitioning idea of both the data and control plane and also the abstraction of TCP/IP mainly targeted for the centralized control and for the global application building process. In the modern approach by decoupling both the planes is a major feature of SDN and so by partitioning them it resulted in a better structured software environment for simplifying data plane and developing network wide abstractions. This split architecture is called as software defined networks. The process is similar to server virtualization i.e. the process of building multiple virtual machines and decoupling them from the physical servers or the network virtualization, creating virtual networks those are decoupled by physical networks.

This approach benefits by increasing the configuration accuracy and consistency and enables an administrator to manage the network as if it were a single device.

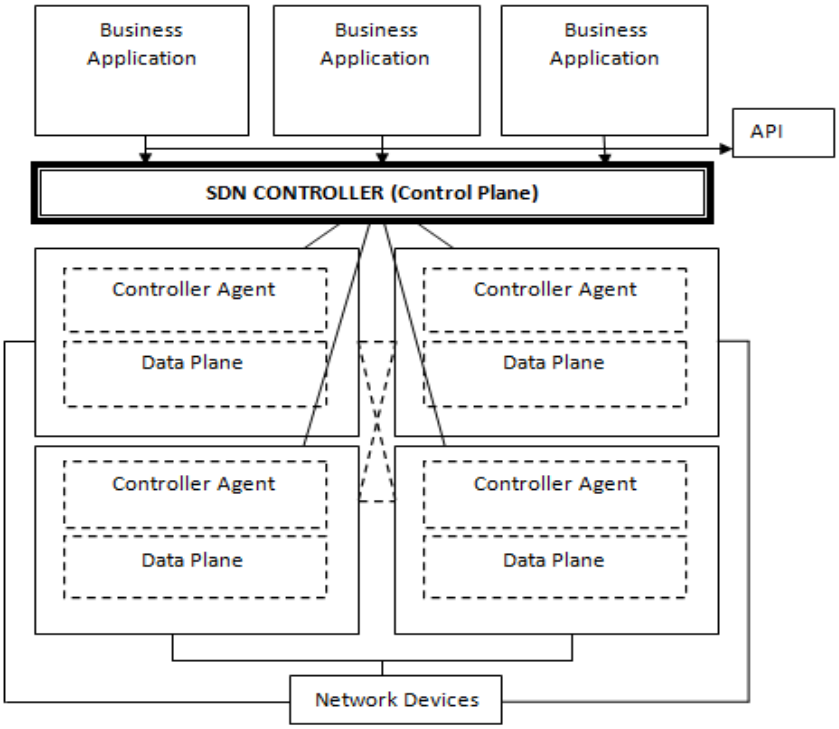


Figure 1.4 SDN Modern Architecture

Figure 1.4 depicts the modern architecture in the software defined networks, containing the controller agent and the data plane combined together in the network devices. All these network devices are further connected to the control plane and then to the business application through the application program interface.

SDN has two main elements:

- Separation of data and control plane
- Programmability (ability to improve network flexibility, dynamically providing traffic flows and providing application level quality of service).

1.2.1 The Control Plane

It contributes to the creation of a dataset which helps to generate the flow entries for the forwarding plane used by the data plane for pushing the datagram in between the incoming and the outgoing ports of a device. The routing functions or the topology tables are the ones used for the storing purpose of the network information. The forwarding table entries are updates and maintained by the Forwarding Information Base (FIB).

1.2.2 Data Plane

It mainly manages the incoming flow by using some of the operations applying multiple authorization and authentication tests. The FIB generated by the control plane helps to process the datagram which is verified and is further processed by the data plane. When there exist a problem regarding the FIB entries, it is pushed onto the control plane for processing it further with the help of Routing Information Base (RIB). These RIB tables are at a number of location software, CPU, GUI etc.

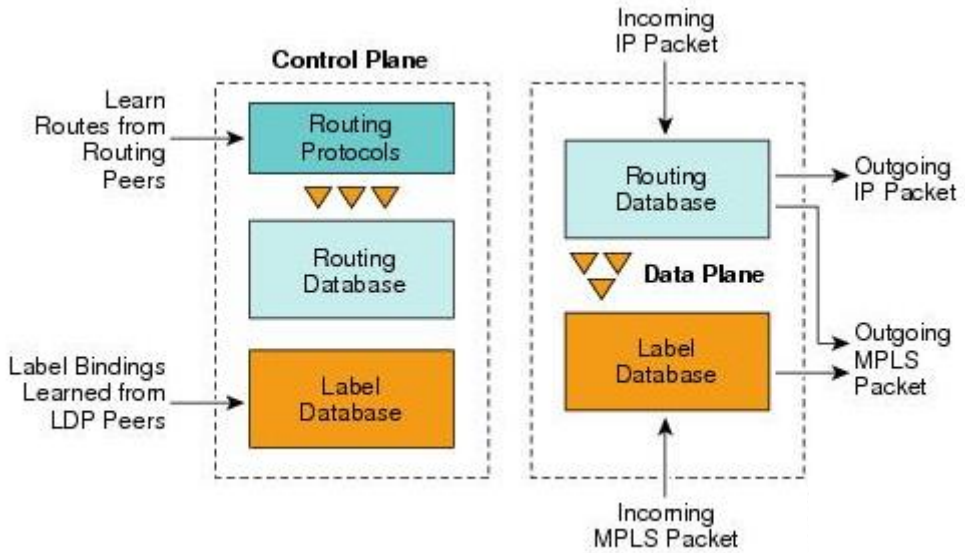


Figure 1.5 Control and Data Plane [15]

Figure 1.5 depicts the control plane and the data plane containing the routing database along with the routing protocols and the label database.

1.3 SDN Architecture

SDN is a structure built in a way that it does not have static capabilities and is quite flexible and cost-efficient. It tends to be an ideal network layout for higher bandwidth and spontaneous nature of applications.

1.3.1 Different Layers in the SDN Architecture

- **Infrastructure layer**

  This layer contains all the virtual network devices as well as the physical devices i.e. the switches and routers. The network devices use the OpenFlow to implement the traffic forwarding rules.

- **Control layer**

  The control plane provides a centralized global view to the entire network. The communication within the network is sorted by the usage of OpenFlow protocol.

- **Application layer**

  It basically helps in the interaction of the control layer by the use of network services and the tools. It provides an open interface to have a communication with other layers in the network layout.

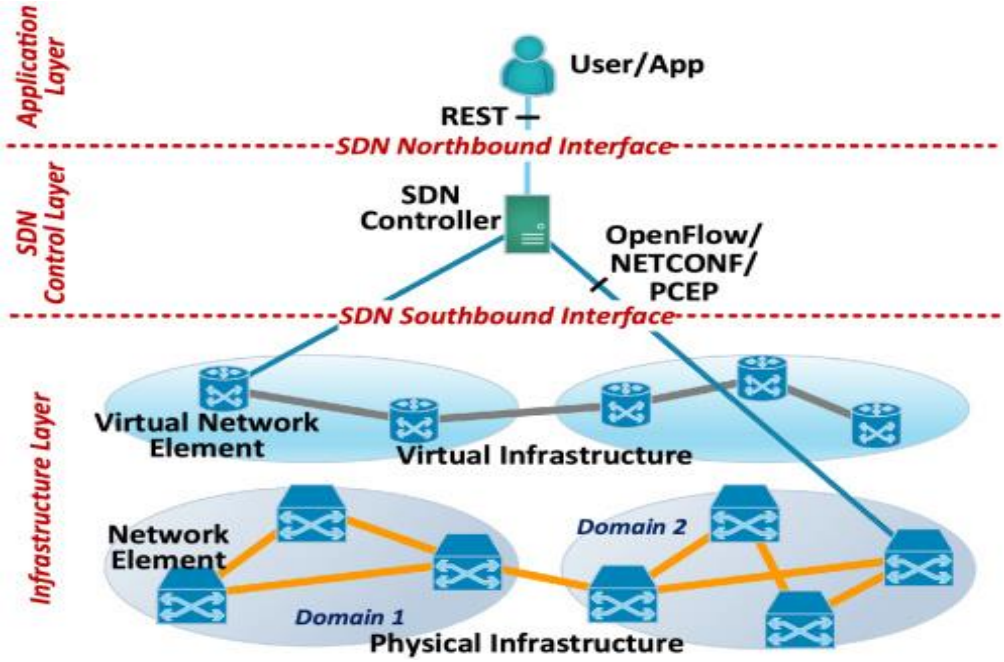


Figure 1.6 SDN Architecture [15]

Figure 1.6 depicts the detailed layer structure along with all the devices and the elements contained in each layer and their connections to the each layer respectively.

1.3.2 Characteristics of an SDN Layout

- It is directly Programmable and is quite efficient.
- It is very agile in its behavior in the network.
- Centralized in nature and is easy to be managed.
- It is convenient and programmatically configuration.
- It is open standard based and vendor neutral.

1.4 Motivation

There exists many research works that have already discussed about the controller placement issues. Most of the existing solutions to the problem are derived out by applying some already existing algorithms which helps in the improvements of the previous outcomes ever derived. The purpose and motivation behind this research work is not to pick any of the algorithm for the improvisation but to generate something from a basic structure designed functioning properly to check the behavior of the network layouts re-created by repositioning the physical nodes, by blocking and unblocking the links in between and obtain some optimal solutions to the problems on the basis of some strongly build architecture with least structural flaws.

CHAPTER 2

Literature Review

Wang *et al.* [3] "**The Controller Placement Problem in Software Defined Networks: A Survey**", the study reveals the targeted areas for discussion. The latency rate for the switches, the development cost, energy consumption and the reliability of the network are all the major factors affecting the controller placement. This paper contains different categories of challenges compiling all the major factors along with the improvised solutions. Despite the effort there are many factors which still are affecting the controller positioning problem and are required to be dealt with for a better and reliable solution. The authors have proposed some solutions for the betterment of the network with the help of some algorithms and improving the functioning of the controllers in a network to get reduced latency rates and a better cost efficient network.

Heller *et al.* [4] "**The Controller Placement Problem**", the concept explains that it moves the control logic off packet processing devices onto the external controllers. The study contains the discussion over the issues regarding the scalability and the performance of a network. These factors were introduced due to the decoupled control plane and a comparison has been made of these factors with the traditional system.

The main targeted areas are the topology structure and the number of controllers required in a network along with their accurate position in the structure for a better performance. The solution is provided by examining propagation latency and the topology structure.

The proposed solutions covered in this research work include the reformed network topology along with the reaction bounds (upper/lower bounds for traffic etc) and the metric choices for a given network.

All possible solutions by the authors are already explained for the problem of topology and controllers requirements for a better performance. There can be the development of some more efficient algorithms or techniques for generating some more efficient outcomes.

Zhou *et al.* [5] "**The Load Balancing Strategy for SDN Controller based on Distribution Decision**", the concept explains that the load balancing can be dealt by deploying the super controllers and these super controllers are mainly the reason for

balancing the load of all the other controllers in the network. There is a controller node which is of centralized decision in nature that collects all the information regarding the load of rest of the controllers and then it further evaluates if the load balancing is required or it need not to be launched. It is good way to monitor the load but the performance might get affected and results into a less reliable network. The performance of the node is limited by memory, CPU power and bandwidth. In this paper, load balancing strategy or the algorithm used is the dynamic and adaptive algorithm for load balancing (DALB) for SDN controllers which is used for sorting the load issues.

One may find another algorithm for load balancing but surely need to review the already existing algorithms for a better understanding and to carry the process any further.

Hu *et al.* [6] "**Reliability-aware Controller Placement for SDN**", the decoupling of the control and the forwarding plane has been performed in this research carried out by the authors which introduces the reliability design issues for SDN control network. As the separation takes place between the control plane and the forwarding plane, it may lead to the packet loss and reduced performance level.

The main area of target is how to create a SDN to maximize the reliability of the control network. In this paper, placement algorithms are also developed. Also there is a study about the impact of the control network by using real topologies. By studying and understanding the proposed aspects, the placement algorithms and the impact of control network we might improve the reliability of SDN control network. The simulated annealing algorithm has been provided as the best optimal solution.

Finally it has been concluded that there is a need of structuring the controller's physical position in SDN to maximize reliability of control network. Placing too many or even a few controllers also reduces the reliability and hence there can be made some improvisation in the structuring process so as to generate some alternative best fit solutions as well.

Mckeown *et al.* [7] "**OpenFlow: Enabling Innovation in Campus Networks**", OpenFlow protocol is the one which is based upon an Ethernet switch, with an interface which is standardized in nature and helps in adding and removing the flow entries from an internal flow-table. To allow the researchers to evaluate the ideas within the settings of the real-world traffic, OpenFlow might serve as a useful component. The study reveals that the OpenFlow provides us with an open protocol to

program the flow entries in the flow tables of different routers and switches. Controllers are used to control the addition and removal of these flow entries. OpenFlow is not really common but if the vendors open up to this technology then it will help by allowing switches, each with their own interfaces and scripting language to easily manage all the functionalities using a single protocol. OpenFlow Switch is one of the main components, consisting of one or more than one flow tables and a group table, which are all responsible for the packet lookups and forwarding process along with an OpenFlow channel to an external controller.

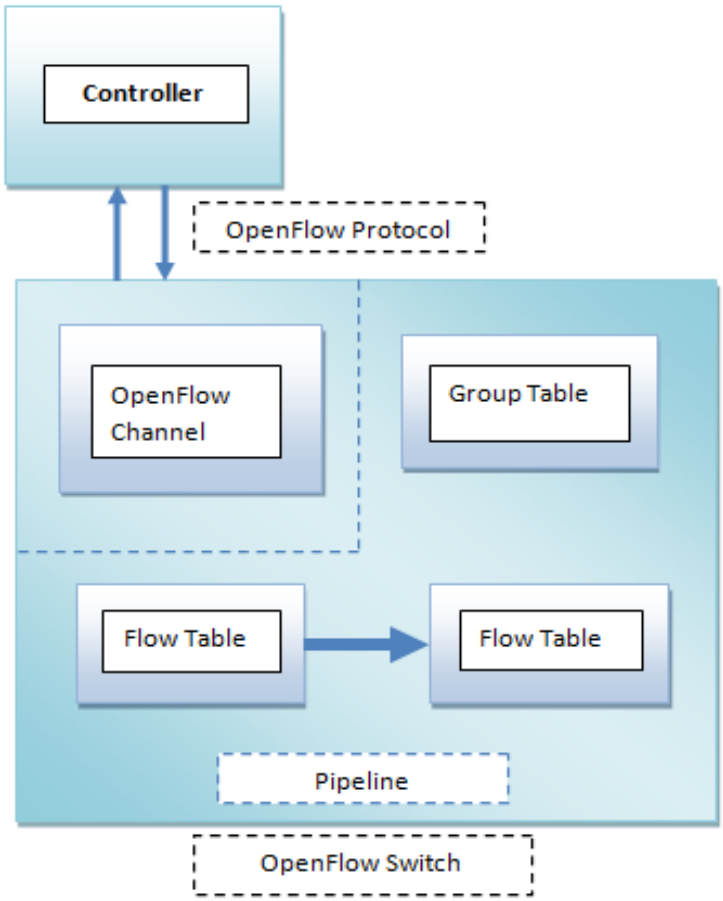


Figure 1.7 OpenFlow Switch

Figure 1.7 depicts the OpenFlow switch which contains the multiple flow tables and the group table, connected to the controller via the OpenFlow protocol.

The purpose of the switch is to communicate with the controller and the controller further manages the switch via the OpenFlow protocol. The controller can add, update and delete flow entries from the flow tables. Each flow table contained in the switch is comprised of a set of flow entries and each entry contains a number of matching packets and the instructions to apply to those matching packets in the flow entries.

Albert *et al.* [11] "A **Comparative Evaluation of the Performance of Popular SDN Controllers**", a comparative study has been made based upon several open source controllers on the basis of their performance factors. The major areas focused in this research work are related to the latency issues and the throughput using an OpenFlow tool Cbench. The study reveals the overview regarding the feature-based qualities of several controllers for determining the most efficient controller in addition with the improvements made for the latency issues in the network. The study mainly proposes that one must completely understand the functioning of the controller before deploying it into the architecture.

Li-Chun *et al.* [12] "**SDN-enabled Traffic-aware Load Balancing for M2M networks**", the main area of concern is the cluttered traffic issues in the machine -to - machine networks and hence it proposes a solution for this traffic management problem by implementing some of the load balancing techniques to relieve the heavy flow throughout the network. It helps to understand the load balancing scenario for machine-to-machine networks which is quite important for satisfying the Quality of service needs by flow rerouting and flow identification. The flow tables are used for transferring the traffic to the appropriate server and the tables are updated when the delay in the performance increases more than the quality of service threshold. This research is purely based upon how the traffic can be efficiently managed, identified and rerouted.

Jianhui *et al.* [13] "A **Reliable Controller Deployment Strategy Based on Network Condition Evaluation in SDN**", the problem statement explains the issues regarding the reliability of controller placement in the network. It studies the scheme for how a controller can be positioned in the network for an efficient performance and throughput. It evaluated the network conditions for generating the deployment scenario in the network for its efficient functioning. The evaluation parameters discussed are the node reliability and the path quality. The main purpose is to have high scalability and low packet loss ratio. This has been dealt by applying the RCD strategy for obtaining the optimal solutions.

CHAPTER 3

Problem Statement

Literature review states that there has been a lot of research work over the controller positioning problem, which led to the consideration of the various factors responsible for the optimal functioning of a network. The literature mostly contains the strategies and the algorithms which already exist. The comparative study amongst different strategies shows the improved results and some better functioning network structures. There has been a discussion over the controller positioning in a network and the factors that are getting affected due to the positioning, but the solutions provided are not really derived from the structural behavior of the network. All the work existing shows some great improvements in the SDN but most of them are based upon the existing techniques or the strategies. These strategies are used as the deriving factor for the scope of improvements. The sole purpose of this research is to contribute for the betterment of the functioning of a SDN by keeping a proposed structure which will be completely derived from a basic structure formed, just by studying and understanding the network behavior built differently for performing the same functions in a network.

3.1 The Controller Placement Problem

To find an optimal solution for the controller placement problem is quite a time consuming process. To generate the real time results for this problem is very challenging. As the result a network can be divided into many sub-networks for a better quality of service consideration. These several sub-networks are needed to be dealt accordingly in terms of their placement, flow management, cost efficiency and hence combining all the factors and many more to get optimized solution to all the problems combined in a network.

There are various categories in this problem which have been targeted earlier and do have a scope for improvisation. Some of these are listed below:

- Minimize the development cost of the network structure.
- Generating a fault tolerant network.
- Reducing the network latency rate.
- Load balancing in a network.
- Time efficiency of a network.

The controller placement problem focuses over the position of the controllers in a network i.e. the placement of the controllers in such a way that the network covers all the targeted areas and provides the optimal solution for improvising these targeted areas providing a more efficient and reliable network.

3.2 The Development Cost of the Network

There exist two approaches to deal with the development cost related issues, statically and dynamically.

3.2.1 Static Approach

It calculates the optimal number of the controllers, the location and also the type of the controller. It keeps the check for the interconnection amongst the controllers and the switches to reduce the cost effectiveness. Controllers are quite expensive and if more of these are placed in a network, it will be cost inefficient and will result into a heavy as well as a not so cost effective layout. We try to resolve this problem by using the minimum number of controllers with a better management and in a more effective manner generating optimal results.

3.2.2 Dynamic Approach

It is less complex than the static approach. It runs in real time and also periodically calculates the overall load of the network and compares it with the controller capacity. In this approach controllers can be dynamically added as well as deleted. The overall load in a network can be balanced by reducing the load over any controller or a switch by redirecting the flow and by creating an effective network layout by choosing optimal solutions for the creation process targeting the major effective areas of a network that requires improvisation.

3.3 Fault Tolerant Network

There are two existing solutions to make the existing network more reliable. Following are the two methods:

3.3.1 Create Robust Trees

The controllers are placed at the root of these trees. Robust trees are able to minimize the loss of management due to node or link failures. This tree contains the balanced number of nodes and so the load is usually balanced in this scenario.

3.3.2 Controllers Positioning

The physical positioning of the controllers affects the reliability factor. If the controllers are not positioned accurately and are at uneven distance, might increase the round trip time for the packets pushed over the network. Controllers at improper distance with more round trip time will more frequently result in the loss of packets, reducing the reliability of the network. To generate more reliable network we need to establish a network with the best fit solutions for the controller positioning problems.

3.3.3 Fault Tolerant Network

Controllers placed in a robust tree structure will result in a better and a more efficiently managed network layout. This scenario will help to maintain the fault tolerance of the layout, evenly distributing the load resulting in a negligible drop rate and making the flow to change its path if required without any interruption or failure of the network.

3.4 Network Latency

Network latency mainly consists of the two main fragments namely, packet transmission latency and the controller processing latency.

3.4.1 Packet Transmission Latency

This is the ratio of the packet size and the transmission rate of the link per node. Latency is nothing but the time taken by the packet to get transferred from one designated point to some other defined point (source point to the destination point). An efficiently working network has the minimum packet transmission latency i.e. more time efficient network.

3.4.2 Controller Placement Latency

This mainly focuses over the controllers bearing load and tracks down the load so as to overcome the controller's latency problems. The main aim is to reduce the latency between controllers and switches in a WAN. The reduced packet transmission latency contributes to this scenario and helps in the reduction of the latency in the controllers and the switches. For further decrease and management of the controller latency issues it can be managed by locating or relocating the controller appropriately in a network.

3.5 Load Balancing

The literature review states that the load balancing is one of the major concerns in a network and this can be dealt by deploying the super controllers in SDN, which are responsible for balancing the load of all the controllers. There is a centralized decision controller node that collects the load information of other controllers and then it further evaluates if the load balancing is required or it need not to be launched. It is good way to monitor the load. Load balancing can also be dealt in various other ways, by creating or by recreating a layout that acts more efficiently than any other existing layout in practice till date. The recreation process may require some of the major evaluations within a network layout already existing on the basis of time taken by the flow to transmit within a layout or the location of the switches along with the locations of the controllers in a layout. The restructuring process will also consider the behavior of the initially designed layouts. These layouts will be showing variations according to the positioning of the links, the switches and the controllers in the network, which might help us to determine the optimized solution to our positioning problem and for all the factors related to it in a network. Load balancing is done on each node in a network to make it work more efficiently i.e. balancing in terms of flow management on every switch and the controllers in a layout.

3.6 Time Efficiency of the Network Layout

In a network time efficiency can be measured in terms of packet transmission from the source to the destination in some time (in seconds). There should not be any drop rate in an effectively functioning network providing a fast transmission on every node present in the layout. In a network, fast packet transmission rate will contribute towards a time efficient network.

CHAPTER 4

Implementation

I aimed at developing a network layout that provides the optimal solution to the various targeted factors namely: latency, cost effectiveness, time efficiency, load balancing and fault tolerance in a network. The purpose of creating this layout is to find an optimized solution to the various factors providing a better reliable and overall efficient network.

4.1 Platform used for Research Purpose

The layout is created in the MiniNet, an emulator which contains the collection of the switches, hosts, controllers and links. MiniNet runs over standard software of Linux. It provides simple and easy environment of python API for experimenting and creating networks. MiniNet Editor is fast, runs real programs and allows the creation of a customized topology and this is where I created a layout for my research purpose. Below is the interface of the MiniNet Editor:

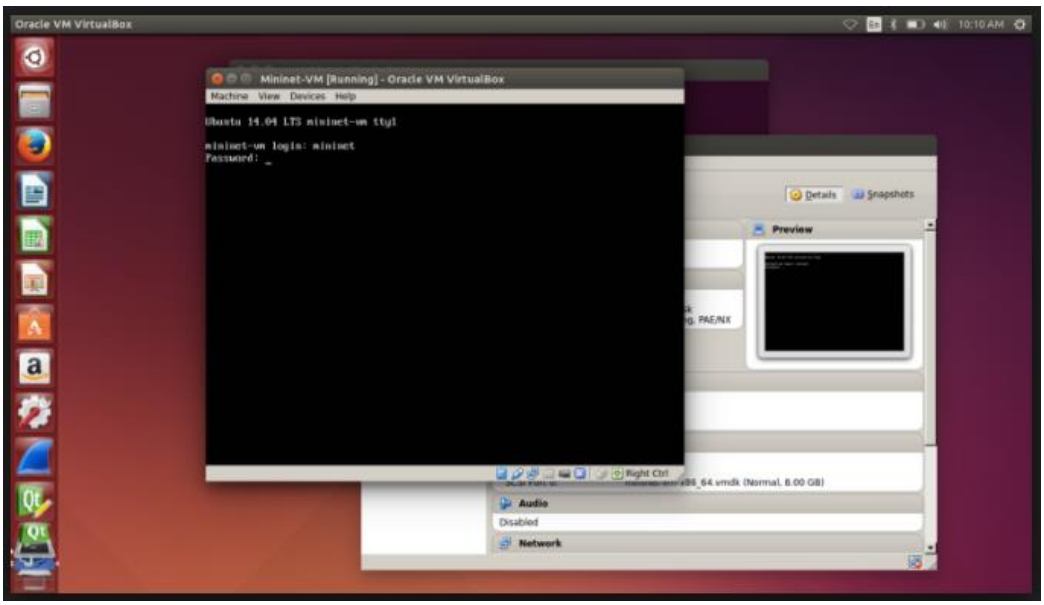


Figure 4.1 MiniNet Interface

Figure 4.1 depicts the MiniNet Interface, the platform over which the network structure is built and is programmed to provide some specific set of services within a network.

MiniNet Editor lets us create topologies using the python API. POX controller has been used for the research purpose and creating a layout using a few of the POX controllers, some switches, links and hosts all linked together to form a network layout. POX has been used along with the python programming language to manage a SDN controller and researching software defined networks to obtain the optimized solutions for my targeted research factors. POX Controller is a python based open source SDN Controller. POX was launched in the year 2014, allows users to write their own applications that uses controller as an intermediary or abstraction layer between network applications and the network equipment. POX supports for the Open Flow protocol and partially supports the features of traffic engineering and the network monitoring. POX functions over a centralized system and uses LINUX as its platform. The most important requirement for this system working properly is that the POX Controller is able to communicate with the MiniNet Editor.

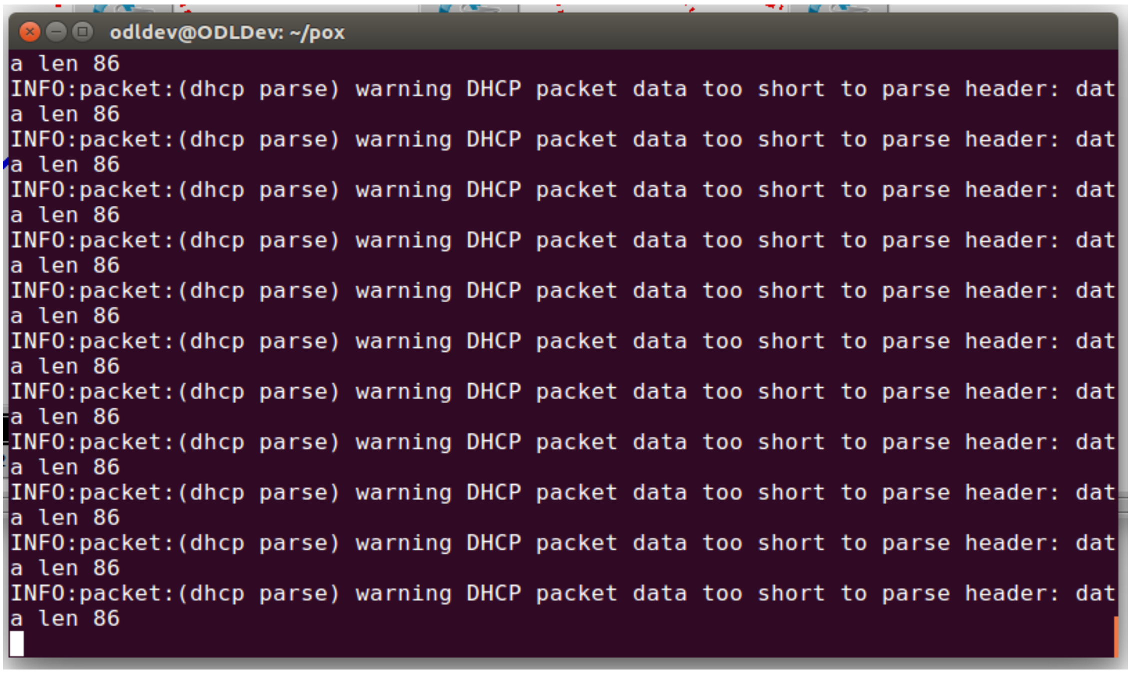


Figure 4.2 POX Running In MiniNet

Figure 4.2 depicts the POX controller running over the MiniNet, establishing the connections for the functioning of all the physical entities present in the network.

POX Controller consists of three fragments namely, the listener, control logic and the messenger.

- **Listener:** It contains the event that the user wants the controller to listen so that the further process could be worked upon.
- **Control Logic:** The logic part helps to distinguish between flows and can help the user by providing an appropriate action for a specific flow.
- **Messenger:** Allows the user to send the message to any switch for adding any new additional rule in the OpenFlow table.

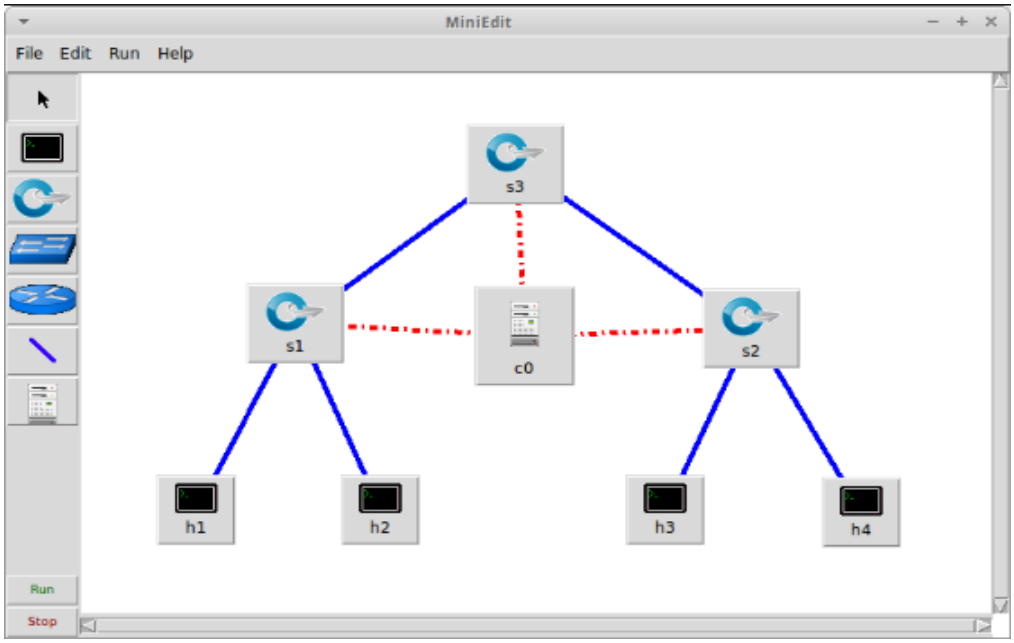


Figure 4.3 POX MiniNet Controller

Figure 4.3 depicts the sample network layout created with a POX controller c0, used for developing the connections within the structure.

POX controller is one of the efficient controllers which enables rapid development, quick prototyping and is growing at a faster rate. The proposed network layout uses the efficient POX controller because of its effective functioning and support for the network monitoring, load balancing and also the traffic engineering.

POX controllers were initially designed to allow the users to write their own applications that uses the controllers as an intermediary between the network equipment and the network applications.

4.2 Reduction in the Development Cost

The overall development cost of the whole network layout could be reduced by following either a static or a dynamic approach or both the approaches combined together for a better solution to our problem.

First, the cost of the network has been reduced by using the static approach, by eliminating the controllers from the network which were not necessarily required. The overall performance of the network was not affected by the elimination of a few controllers as the traffic passing through these eliminated controllers could be easily rerouted and balanced with no drop rates. Controllers are quite expensive in the market, more controllers you install in the system more it will cost along with the increase in the overall cost of the network which is not an economical option for the development process. By following the static approach there are reduced number of controllers which reduces the overall cost. There is also a need of increasing the number of links in the structure for the better and more accurate functioning of the network. The traffic passing through the eliminated controllers is now balanced by redirecting it through the links on to the other controllers with lesser load. The load balancing technique is being followed for balancing the flow so that the reduction of some physical entities would not result into a loss for the network.

Development Cost is also affected by following the dynamic approach in the network. Due to the increased number of links in the network it allows the flow to redirect itself onto the controllers with lesser load automatically providing the fault tolerance. The load remains balanced in the network by comparing the network load with the load distributed over each controller and the switches in the network. It also manages to eliminate the unnecessary controllers resulting into a better and a cost efficient network. This creates an effective network layout by choosing optimal solutions for the creation process targeting the major effective areas of a network that requires improvisation.

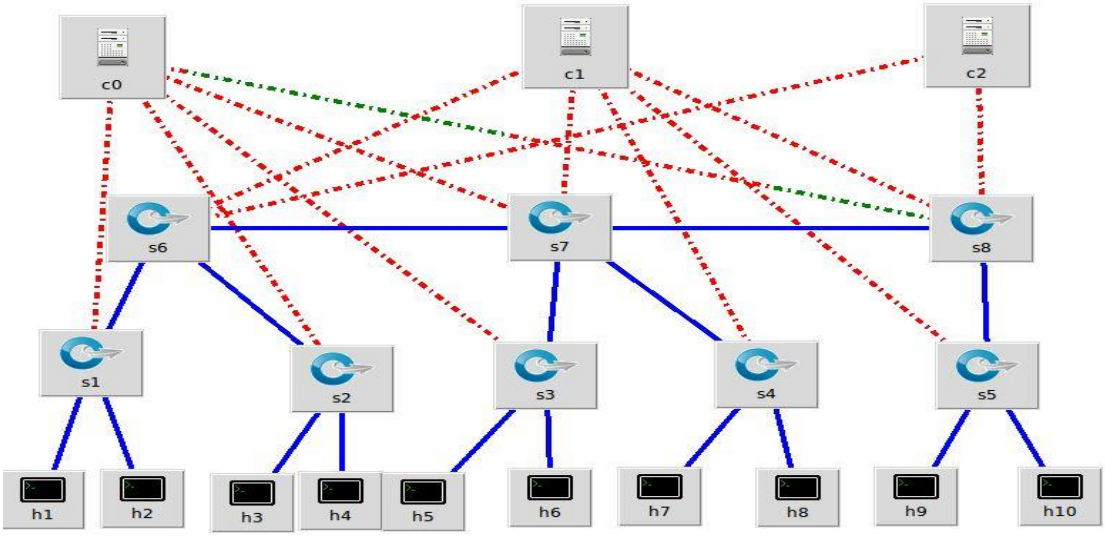


Figure 4.4 Proposed Layout with POX Controllers

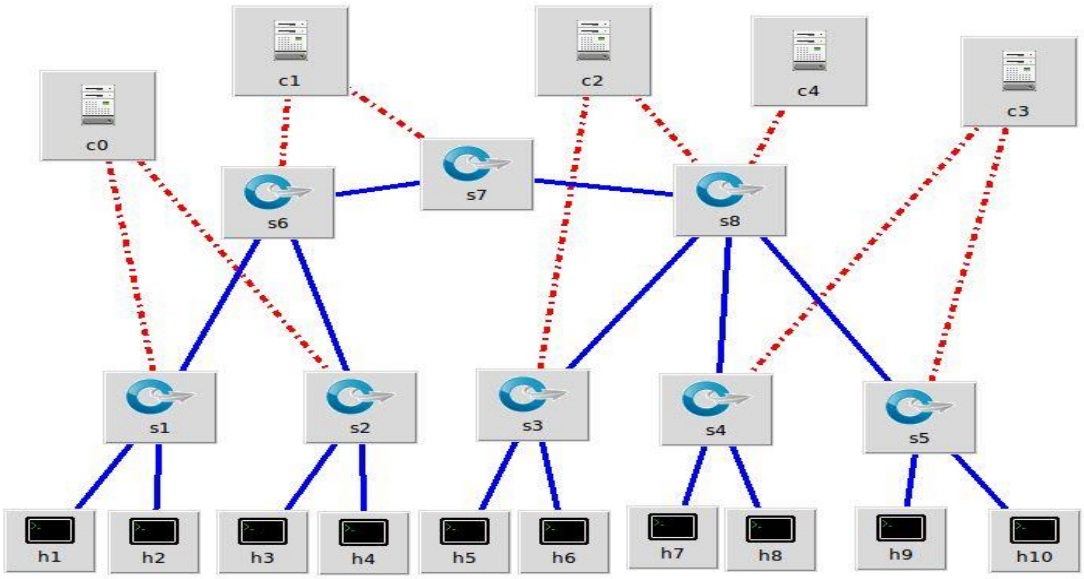


Figure 4.5 Initial Layout with POX Controllers

Figure 4.4 and Figure 4.5 depicts the proposed network layout with some better and improved results for the various factors affecting the software defined networks, containing ten hosts and eight switches all connected via a few number of increased links for the load balancing purpose and making the layout more cost efficient and the initially designed layout with more number of controllers deployed, all linked to the switches and the hosts via the links resulting into an inefficient network.

4.3 Establishing a Fault Tolerant Network

Fault tolerant network is the one which enables itself to work efficiently and appropriately even if there is any occurrence of an event resulting into the failure of any of the physical node in the network. The initially designed network was not fault tolerant as in case of occurrence of any failure the whole network was getting affected and the flow was not able to get redirected neither was able to manage the proper functioning of the network. Although there were no drop rates and outcomes were successfully getting generated but the network load was not getting properly distributed over the controllers and the switches. The overall designed layout was resulting into the inappropriate balancing scenario and into half of the topology failure. In order to make the network fault tolerant there was a need of generating a robust tree structure which could provide us with some outcomes in a better and a more efficiently managed network layout. Controllers positioned accurately and at an even distance might decrease the round trip time for the packets pushed over the network. Controllers at improper distance with more round trip time will more frequently result in higher drop rates for the packets transmitted. This reduces the reliability of the network. By distributing the load evenly in the network with the help of some restructuring strategy, it results in a negligible drop rate and this scenario also led to the flow to change its path if required with the help of links provided in case of any node failure without any interruption or failure of the network.

This topology is created in the form of robust trees due to their property of being able to minimize the loss of management due to node or link failures by having a balanced scenario of having equal distribution of nodes and being fault tolerant.

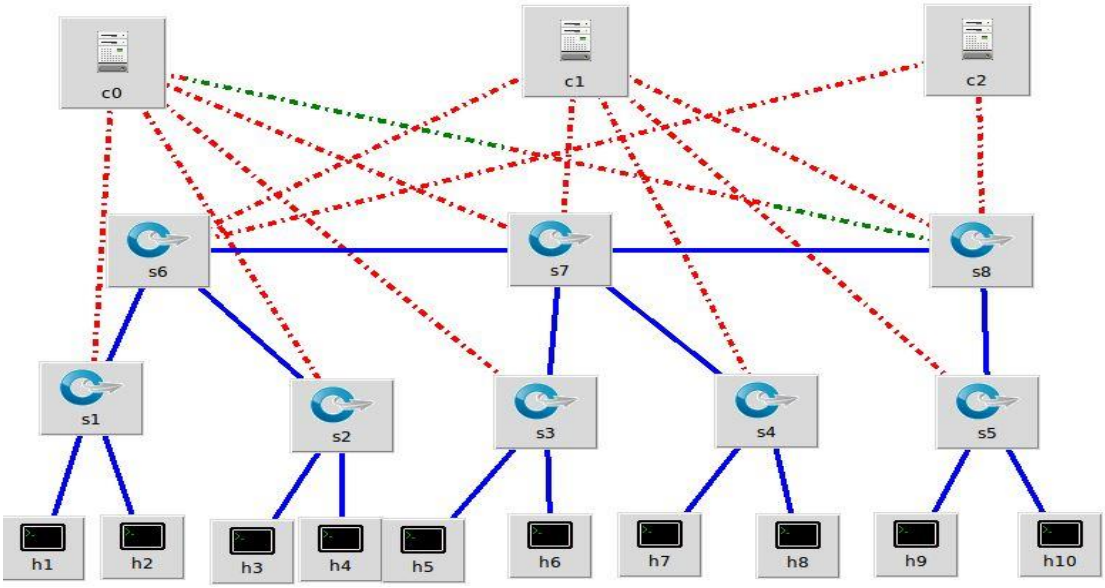


Figure 4.6 Fault Tolerant Network

Figure 4.6 depicts the fault tolerant network which contains the three POX controllers namely C0, C1 and C2. All the three controllers are connected to the eight different switches which are further connected to ten hosts. The network is fault tolerant as all the switches are linked to more than one controller so that in case of any failure occurring at any controller the flow gets redirected through another controller. Thus in case of a node failure there won't be a network failure as all the traffic gets rerouted and the network is working as efficient as it was working before the fault even occurred.

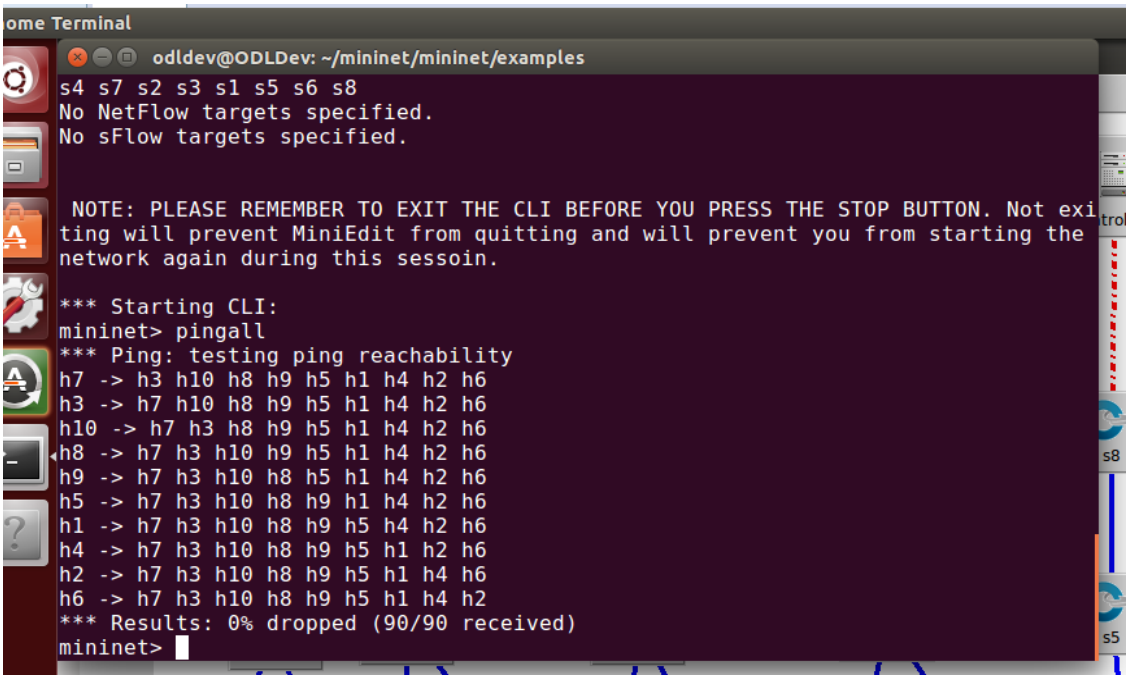


Figure 4.7 100% Success Rate for the Flow Transmission

Figure 4.7 depicts the hundred percent success rates for the flow transmission within a network by passing the pingall command in the terminal**.** The result above shows a zero percent drop rate and a (90/90 packets received) for the flow passing in the network layout.

4.4 Reduction in the Network Latency

This network has dealt with two major portions of the network latency namely, the packet transmission latency and the controller placement latency.

4.4.1 Packet Transmission Latency

Improvising the network resulted in the reduction of packet latency. Packet latency is calculated as the ratio of the packet size and the transmission rate of the link per node. After the reconstruction of the network the time taken by the packet to get transferred from a source point to the destination got reduced to a large extent in comparison to the initially designed network. The reduction in the latency occurred due to the efficiency of the new recreated layout as there are increased number of links for load sharing at faster rate and better performance of the network. There is a proper flow transmission with no drop rates and the evenly balanced nodes. Thus the transmission time reduced and the latency reduced respectively. Below the graph shows the reduction in the packet transmission latency for a particular node (switch1) and there are the latency graphs for every node in this network layout.

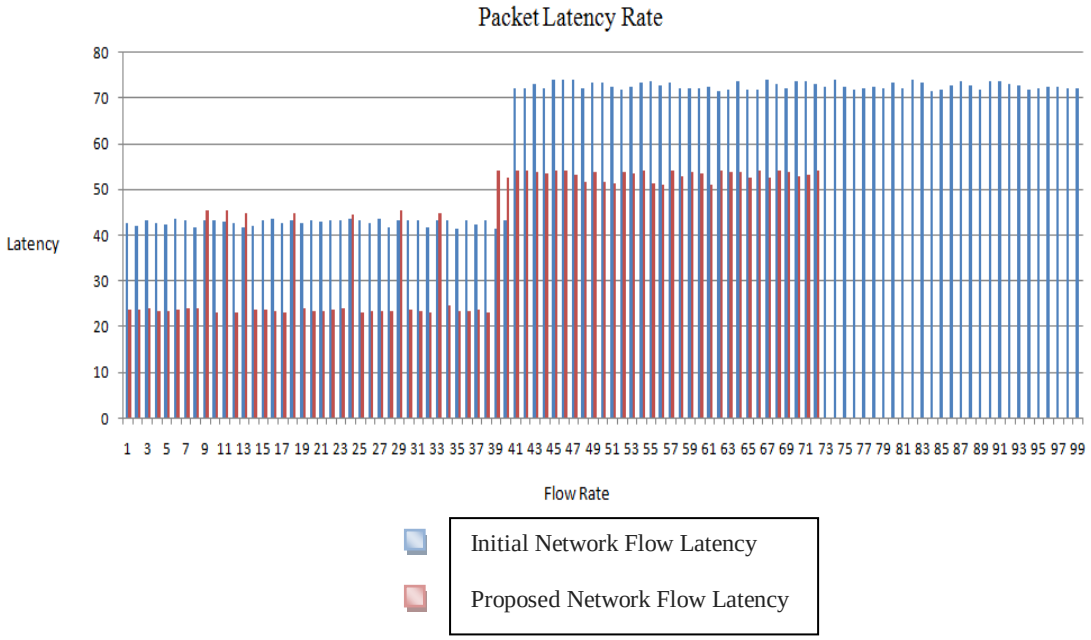


Figure 4.8 Packet Latency for Switch 1

Figure 4.8 depicts the comparison for the packet transmission latency rate of both the initially designed and the proposed network layout, for switch 1.

4.4.2 Controllers Placement Latency

The layout of network plays an essential role in the reduction of the load over the controllers. Controllers are the main nodes of the layout designed as a network. These nodes requires a proper controller placement solution to get an efficient network functioning scenario which could provide with the better transmission delay solution and helps to decrease the latency issues in the network. Thus, I have repositioned the layout by targeting the time as the main factor for defining a better evolved network and hence generated the outcomes which are statistically giving less transmission delay for the proposed network as compared to the transmission delay of the initial network layout.

Repositioning of the nodes is based upon all the five main factors of this research on controller placement problem. All of these factors target the time as their main prediction factor for the recreation of the proposed layout, which improves the overall efficiency of the network.

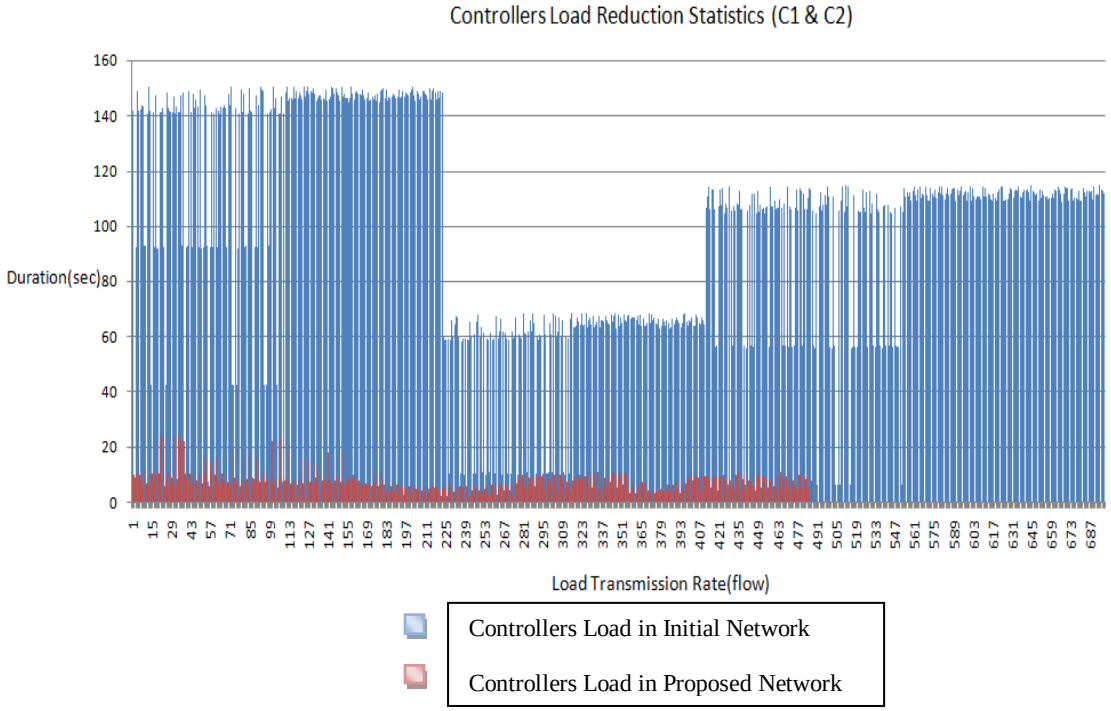


Figure 4.9 Controller Load Balancing and Reduced Transmission Rate

Figure 4.9 depicts the load reduction over the controllers in the proposed network in comparison to the initially designed network layout. The outcomes for the proposed network layout are quite efficient and show lesser time variations for the transmission of the packets.

4.5 Performing Load Balancing over the Network

The controller placement problem deals with the load balancing factor as an important issue and a major concern. Load balancing is one of the major requirements in this network. Traffic monitoring and the traffic management contribute for the flow to be balanced. This key factor in the network layout had to undergo through the following methodology:

- **Topology Creation:** Initially a layout of a network was created to work upon and to further carry out the research for the required outcomes.
- **Topology Implementation using POX**: POX Controllers are installed and run to get the flow monitored through the links per node in the network.
- **Generating Dataset**: Once the network finishes tracing the flow passing through the network layout, the data is being recorded into the excel sheets for each node present in the layout.
- **Target Factor for Prediction Value**: The dataset generated had various fields and required the data filtration by targeting in particular some of the major areas of the dataset. I chose time as the prediction value for the mining process to get the filtered data from raw dataset previously generated.
- **Mining**: The dataset undergoes the process of data mining to generate the filtered dataset as per the main prediction factor to get the outcomes which will be used further for restructuring the network layout. The main targeted field for the filtration process is the duration for the transmission in the network. To generate optimal solution to the issues in the controller placement problem data mining helps to fetch the data which can be evaluated further for proceeding to find the optimized results.
- **Load Check**: The filtered dataset record gets checked for each node to determine the load per node after the data filtration process.
- **Fault Tolerant**: Load balancing is a necessary factor to be dealt with but before we further precede with that the network needs to be fault tolerant so

that the data gets the targeted factors optimized. Thus it could be managed to restructure the network layout effectively in terms of cost and fault tolerance.

- **Proposed Layout**: Evaluating all the main factors for the research and by targeting the prediction factor the final layout is re-created till the optimized results are generated.

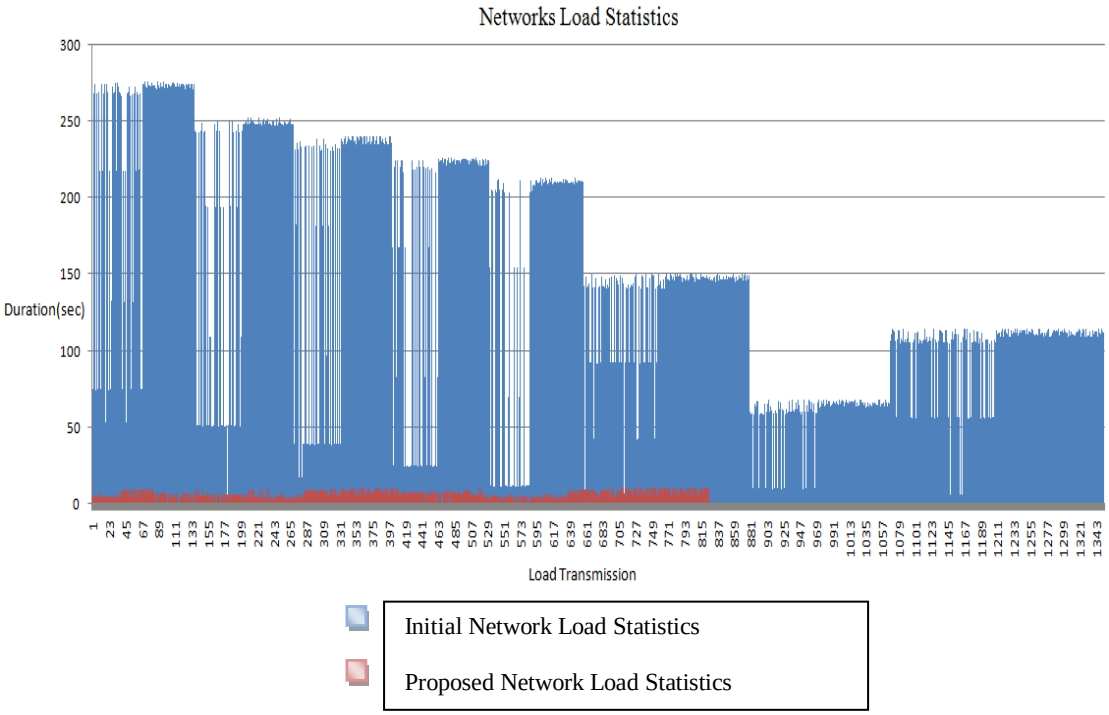


Figure 4.10 Initial and Final Network Load Transmission

Figure 4.10 depicts the comparative study for the initially designed and the proposed layout. It states that the controller placement problem has finally obtained the optimal solution to all its research factors, mainly in terms of the transmission time and the load distribution resulting into an efficient functioning network with the better and optimized results.

4.6 Improved Time Efficiency of the Network Layout

The time efficiency of a network is measured in terms of the rate of transmission of the packets flowing in the network. The proposed network layout provides us with some better and improved results in terms of transmission rate, latency, cost effectiveness and load balancing. The restructured layout is much more efficient and reliable than the initially designed network layout; it is fault tolerant and is even capable of having a better flow management. The efficiency of the proposed network is also measured in terms of the load distribution throughout the network without any occurrence of a fault or the loss of packets flowing in the network making it more reliable and more efficient.

CHAPTER 5

Results

The evaluation of the research problem statement has been done upon the basis of the all the key factors affecting the controller positioning problem in the network layout. Outcomes and the statistics have been generated by implementing the network topology created and then recreated for a better performance. The purpose of creating and generating these outcomes is to establish a network which is capable of serving most efficiently and appropriately.

5.1Initial and Proposed Network Layout

The initial network layout consists of five POX controllers along with eight switches and ten hosts. All the nodes in the network are connected to each other through the links in between the nodes. This is the base topology used for deriving the proposed topology for generating some optimal results and making the network work efficiently.

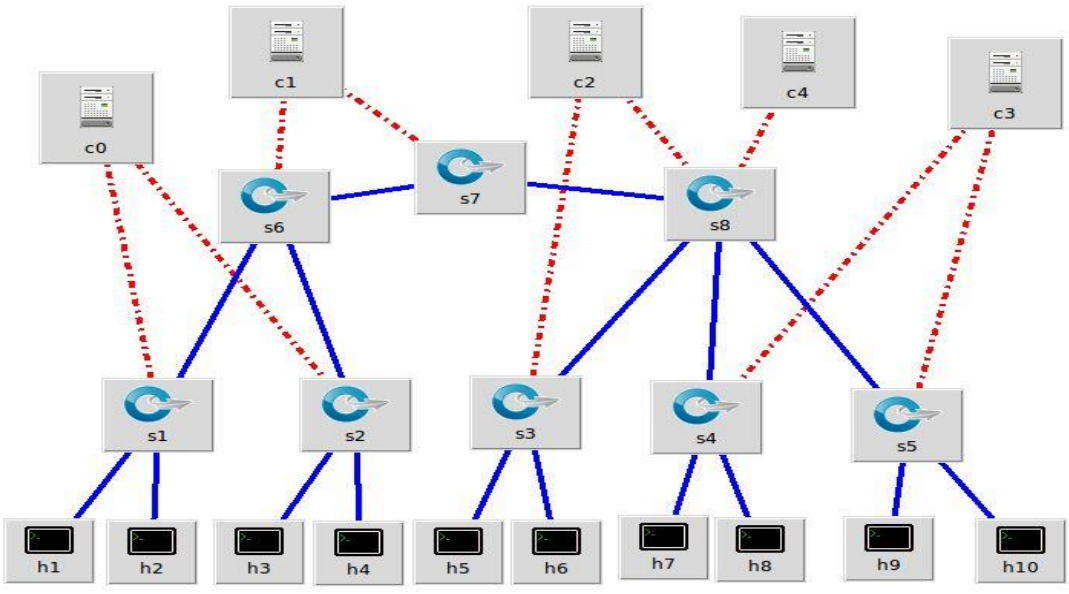


Figure 5.1 Initially Designed Network Layout

Figure 5.1 depicts the initially designed network layout consisting of five POX controllers along with eight switches and ten hosts. There are lesser numbers of links

which results in the lesser fault tolerance in the network which reduces the performance of the network.

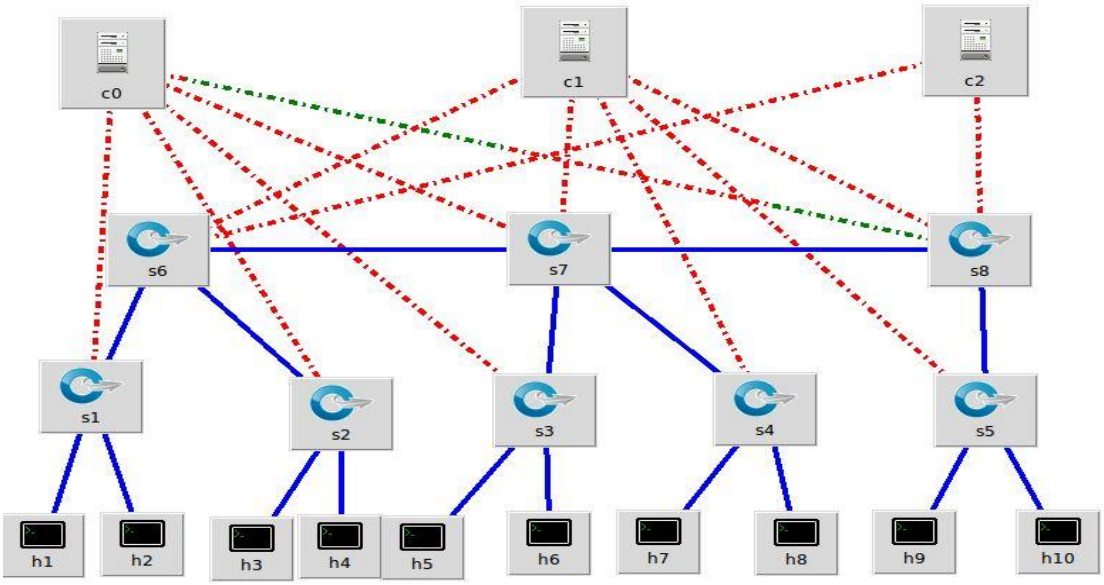


Figure 5.2 Proposed Network Layout

Figure 5.2 depicts the proposed network which contains the three POX controllers namely C0, C1 and C2. All the three controllers are connected to the eight different switches which are further connected to ten hosts. The network is fault tolerant as all the switches are linked to more than one controller so that in case of any failure occurring at any controller the flow gets redirected through another controller. Thus in case of a node failure there won't be a network failure as all the traffic gets rerouted and the network is working as efficient as it was working before the fault even occurred. The proposed network layout is much efficient in comparison to the initially designed network layout.

5.2 Load Transmission for Initial and the Proposed Network

The overall network traffic passing through both the initially built network and the proposed network are being compared in terms of the transmission rate of the flow. The network flow has been monitored so that the flow management scenario can be applied to the proposed network layout and the optimized results are produced in terms of flow getting transmitted at a faster rate and is balanced with lesser time variations.

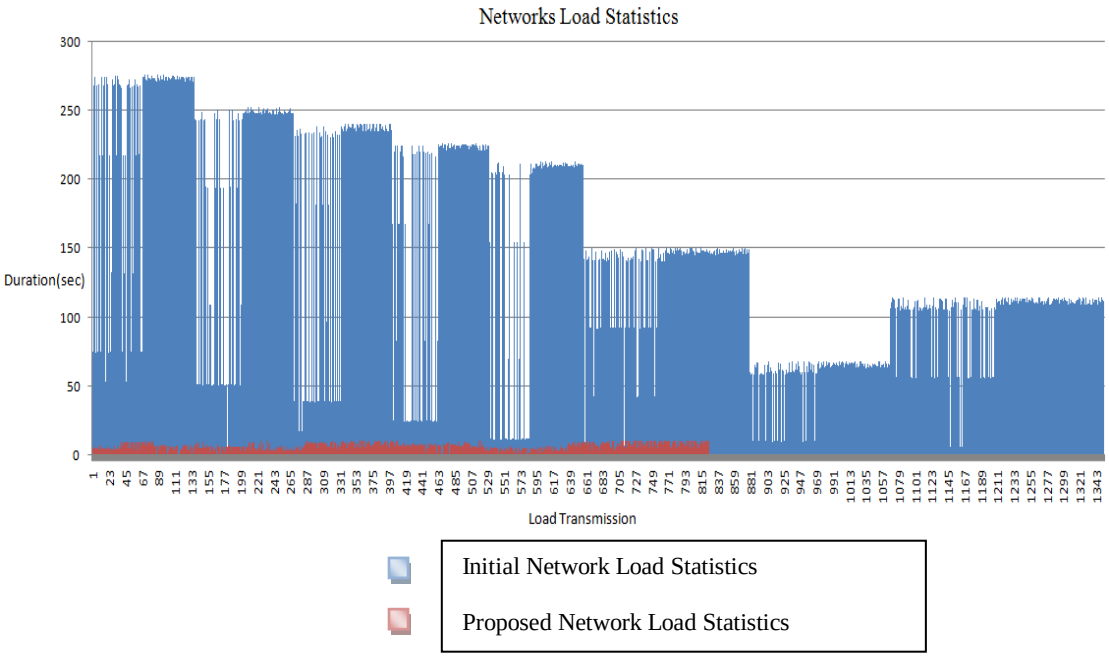


Figure 5.3 Network Load Transmissions

Figure 5.3 depicts that the controller placement problem has finally obtained the optimal solution to all its research factors, mainly in terms of the transmission time and the load distribution resulting into an efficient functioning network with the better and optimized results.

5.3 Controllers Load Reduction

The graph depicts the load passing through the controllers in the initial network and the proposed network. After reforming the network I obtained the optimized results for the restructured network layout. The two main controllers getting overloaded i.e. C1 and C2 were eliminated and the load was redirected to the other controllers. Load balancing occurred with the help of multiple links used for the load distribution process, resulting into a balanced network layout.

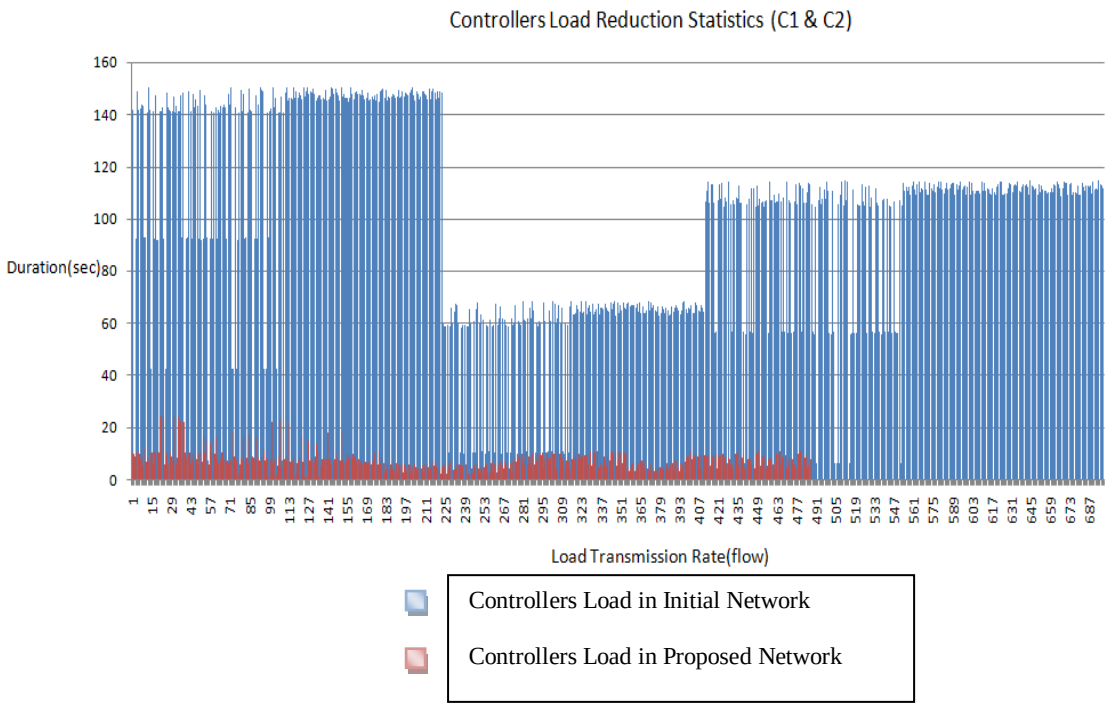


Figure 5.4 Controllers Load Distribution

Figure 5.4 depicts the load reduction over the proposed network in comparison to the initially designed network layout. The outcomes for the proposed network layout are quite efficient and show lesser time variations for the transmission of the packets.

5.4 The Packet Transmission over the Switches in the Proposed Network

This depicts the packet transmission rate for every node present in the network on the basis of duration (in seconds) and the amount of flow passing through the nodes. The graphs generated show the transmission rate per node over the x-axis for all the switches and the time taken for the end to end transmission over the y-axis i.e. the time taken by the traffic to pass through each of these switches.

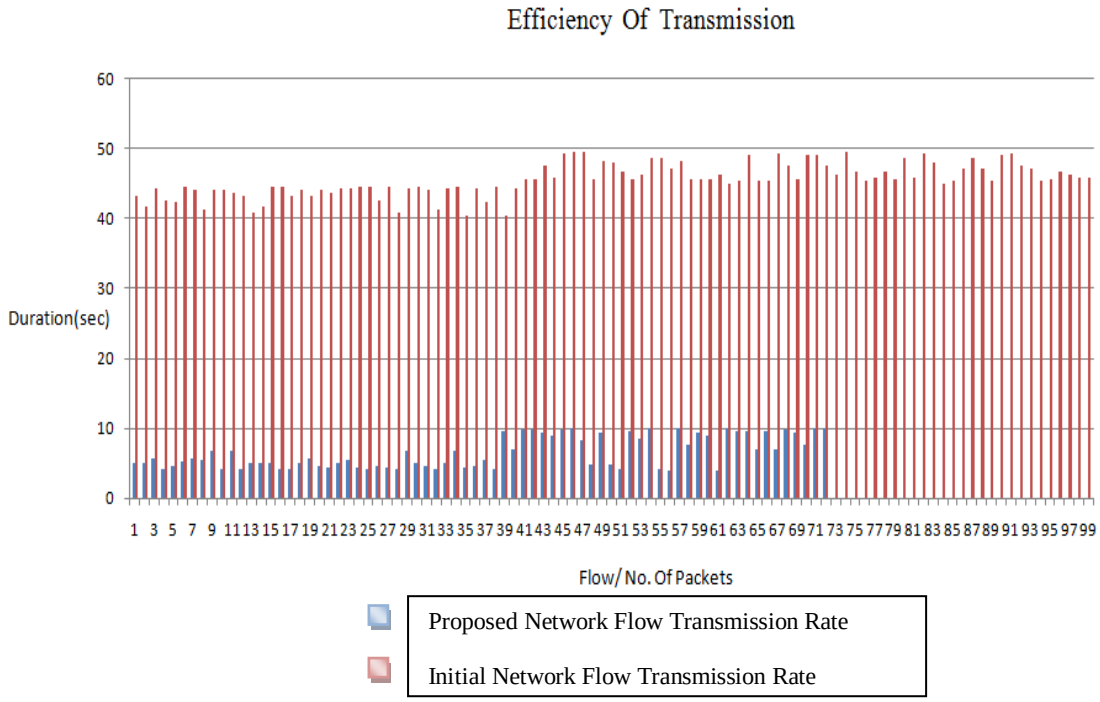


Figure 5.5 Packet Transmission Rate of Switch 1

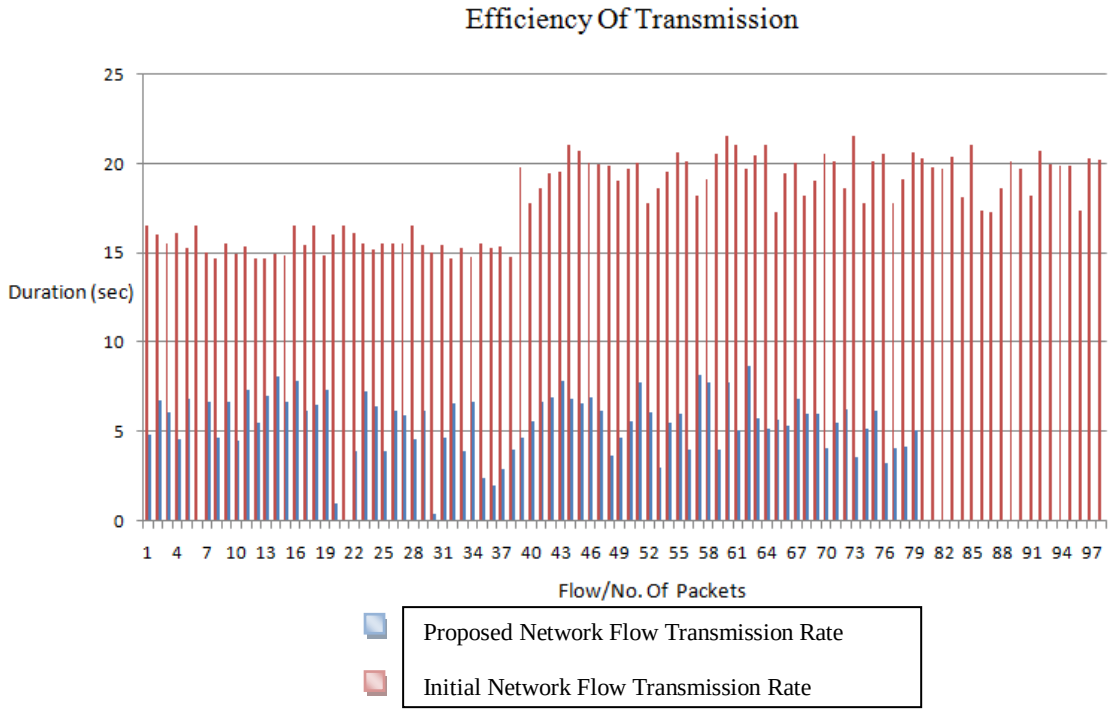


Figure 5.6 Packet Transmission Rate of Switch 2

Figure 5.5 and Figure 5.6 depicts the comparison for the network flow transmission rate of both the initially designed and the proposed network layout, for switch 1 and switch 2.

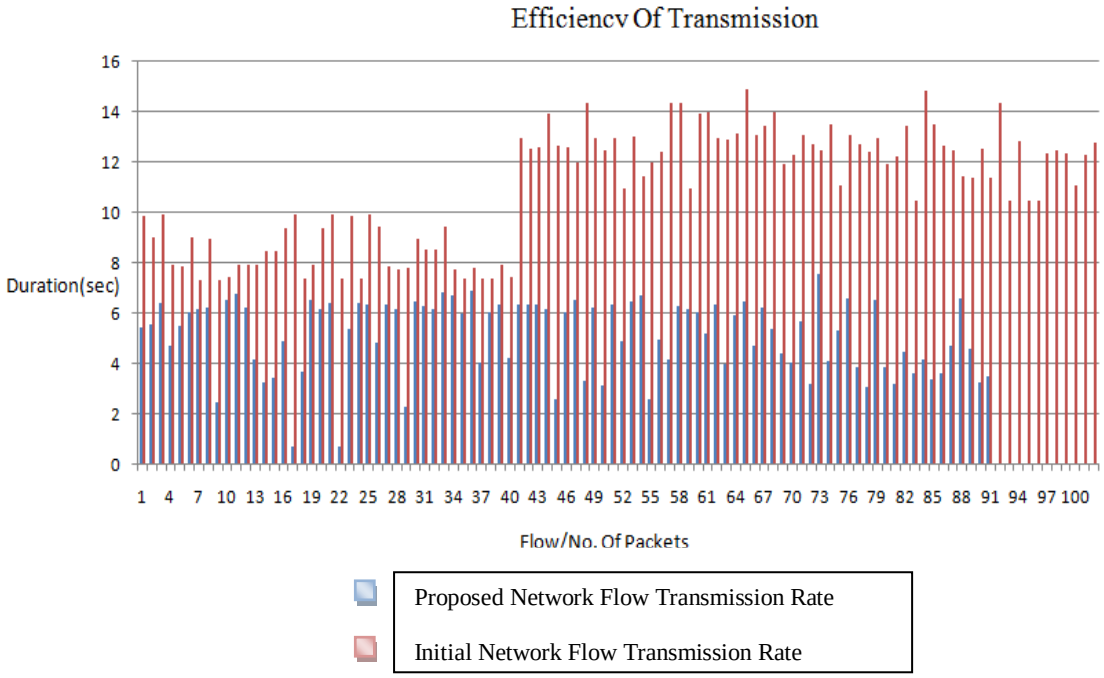


Figure 5.7 Packet Transmission Rate of Switch 3

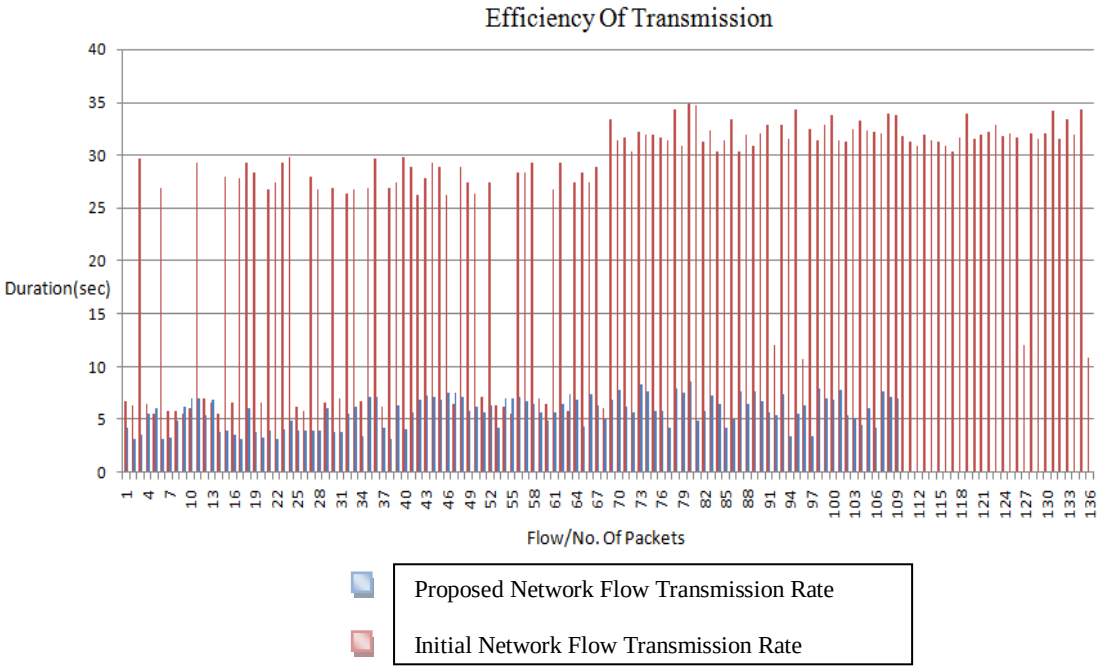


Figure 5.8 Packet Transmission Rate of Switch 4

Figure 5.7 and Figure 5.8 depicts the comparison for the network flow transmission rate of both the initially designed and the proposed network layout, for switch 3 and switch 4.

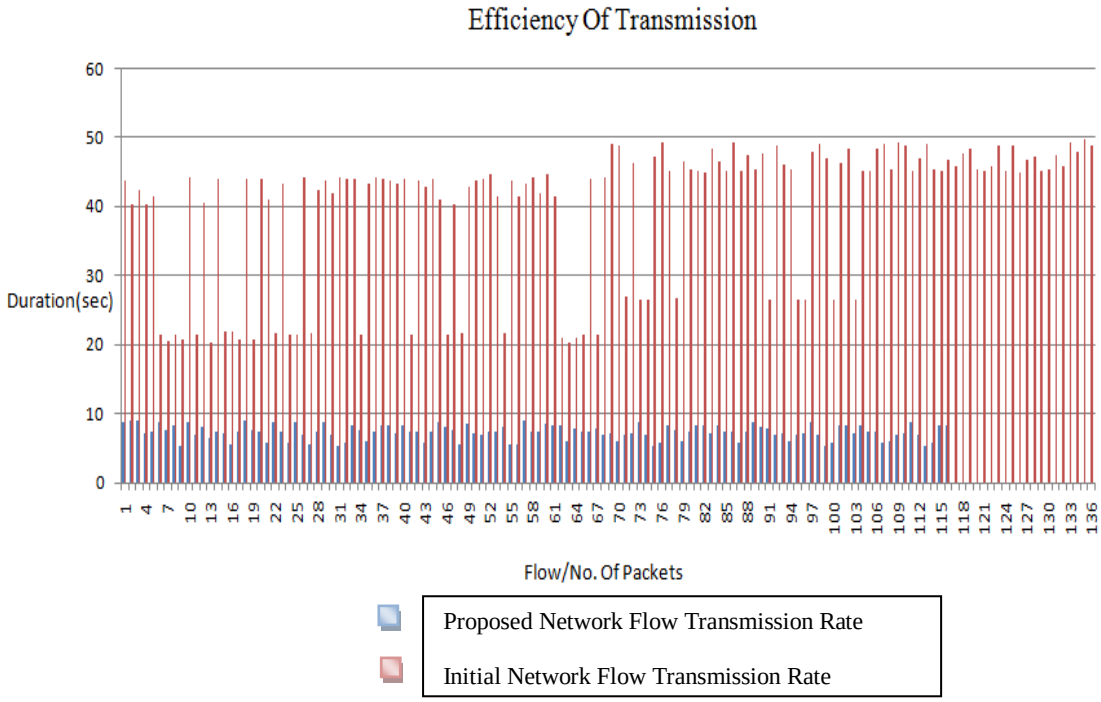


Figure 5.9 Packet Transmission Rate of Switch 5

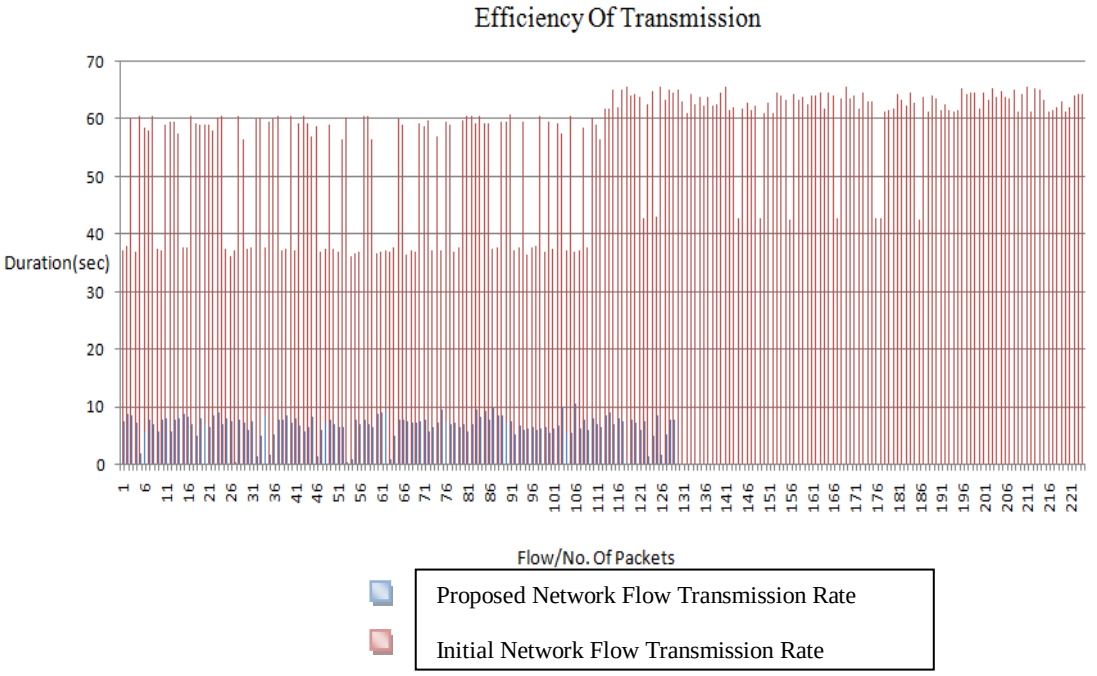


Figure 5.10 Packet Transmission Rate of Switch 6

Figure 5.9 and Figure 5.10 depicts the comparison for the network flow transmission rate of both the initially designed and the proposed network layout, for switch 5 and switch 6.

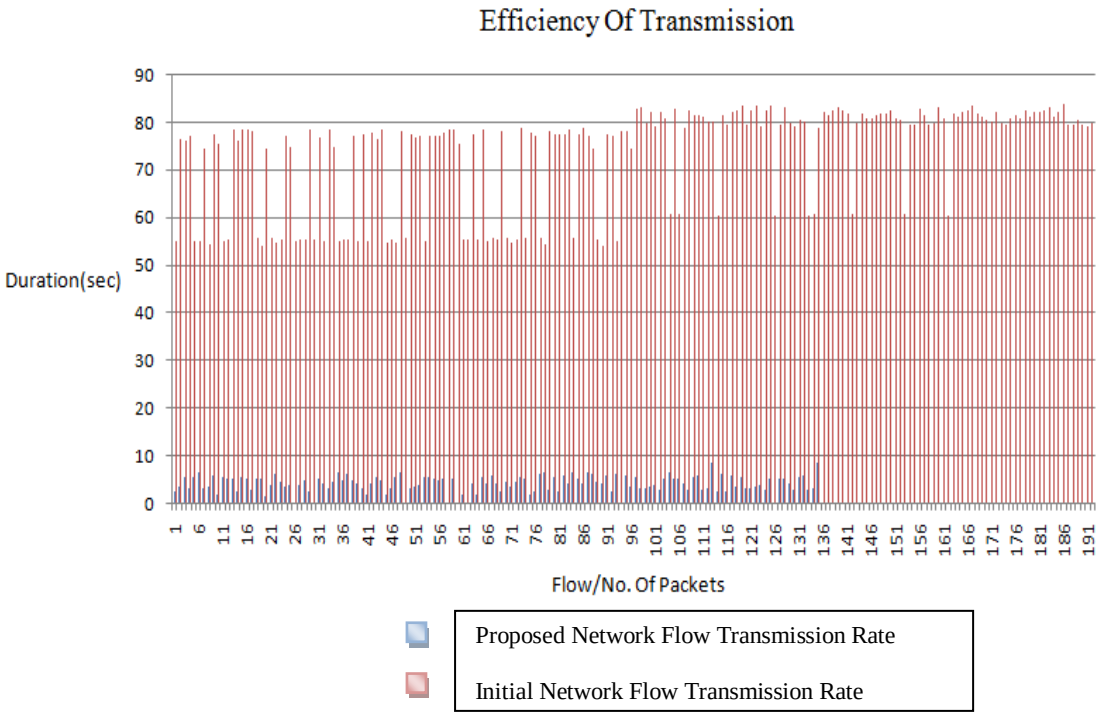


Figure 5.11 Packet Transmission Rate of Switch 7

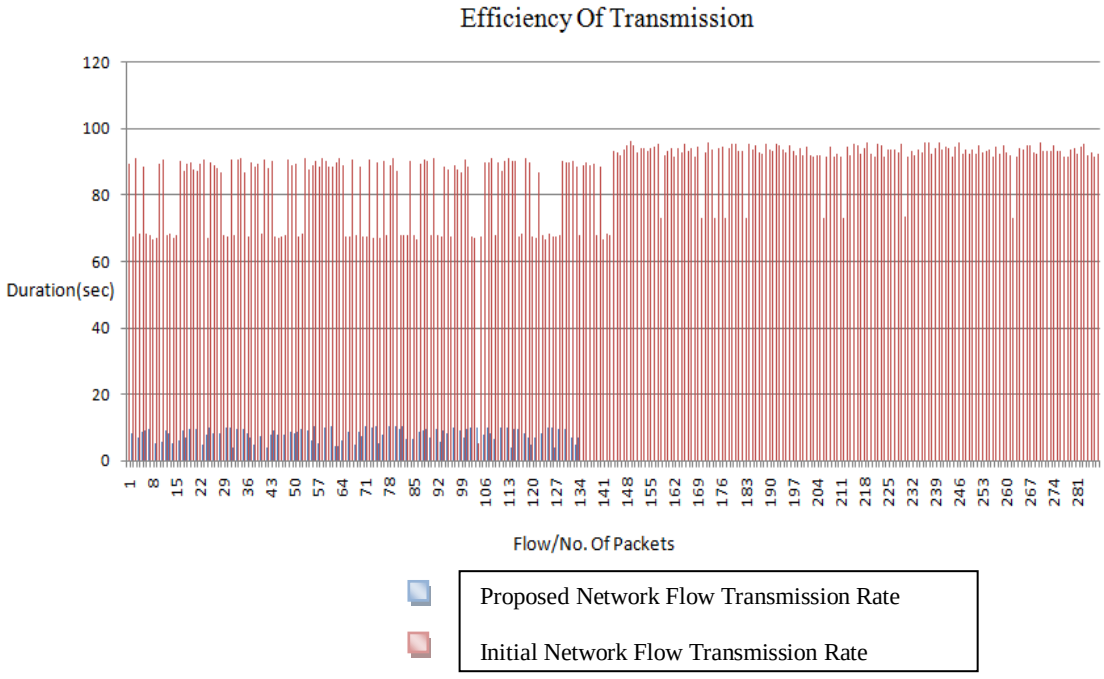


Figure 5.12 Packet Transmission Rate of Switch 8

Figure 5.11 and Figure 5.12 depicts the comparison for the network flow transmission rate of both the initially designed and the proposed network layout, for switch 7 and switch 8.

5.5 Packet Transmission Latency for the Switches in the Network

It is the ratio of the packet size to the transmission rate. It depicts the outcomes for all the eight switches in the network layout and provides the end to end transmission latency rate for the initial network and the proposed network. The x-axis shows the flow rate for both the networks and the y-axis shows the latency.

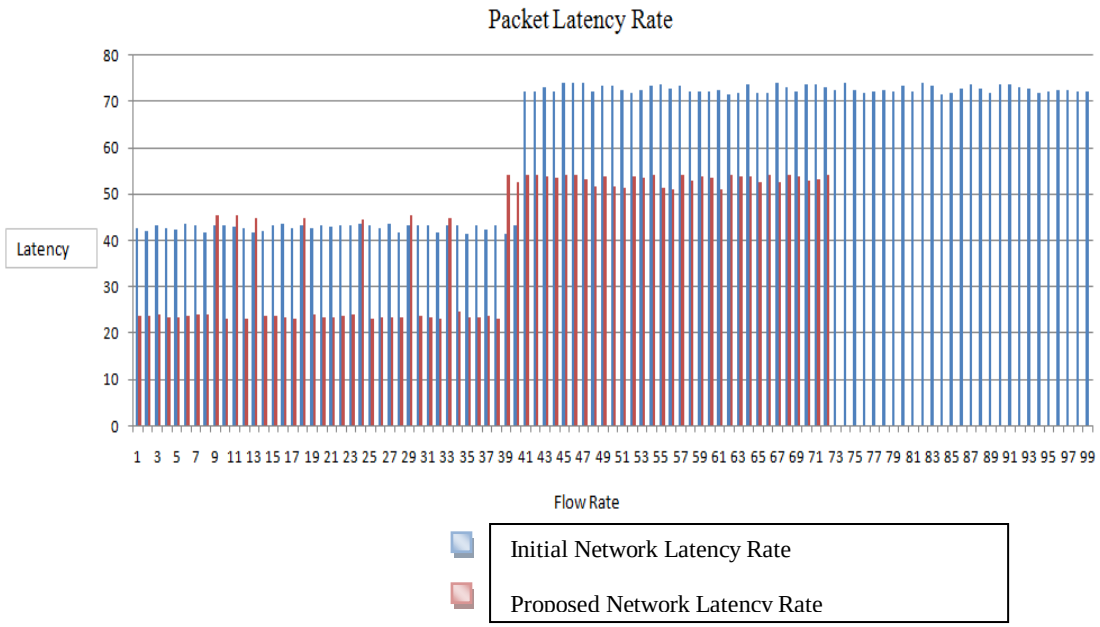


Figure 5.13 Packet Latency of Switch 1

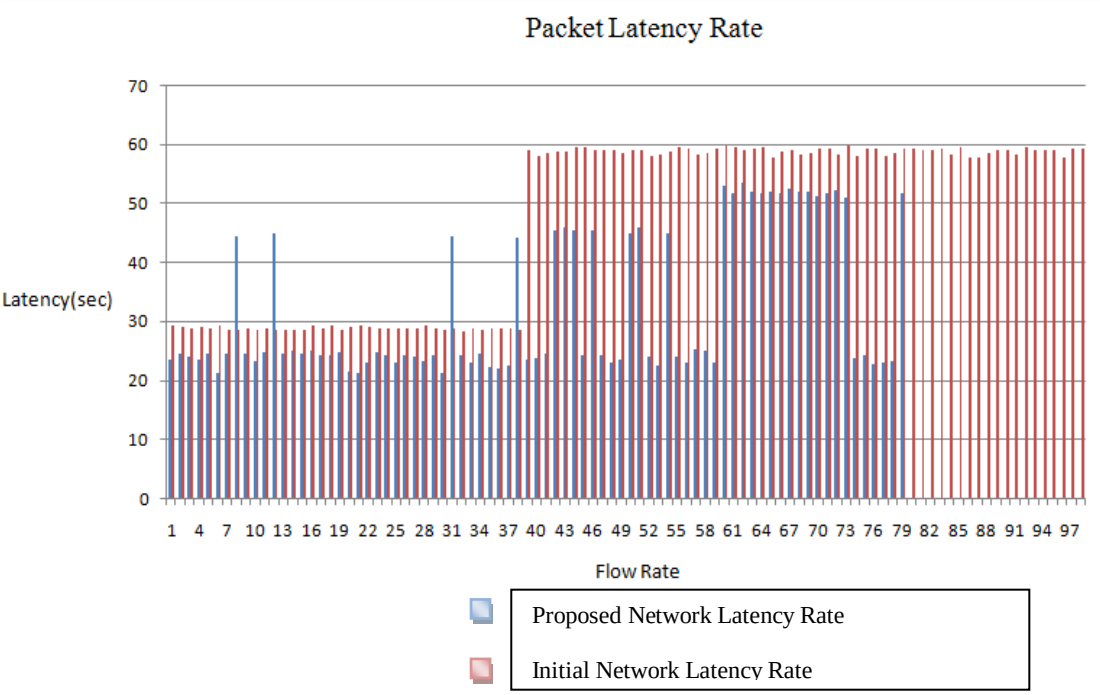


Figure 5.14 Packet Latency of Switch 2

Figure 5.13 and Figure 5.14 depicts the comparison for the packet transmission latency rate of both the initially designed and the proposed network layout, for switch 1 and switch 2.

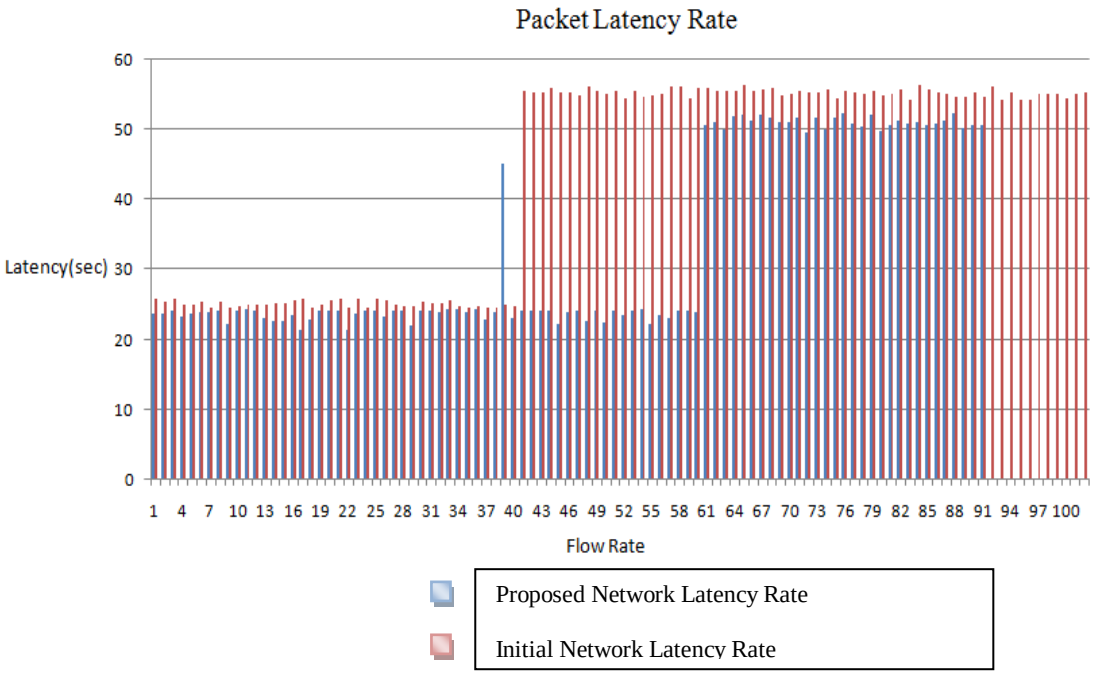


Figure 5.15 Packet Latency of Switch 3

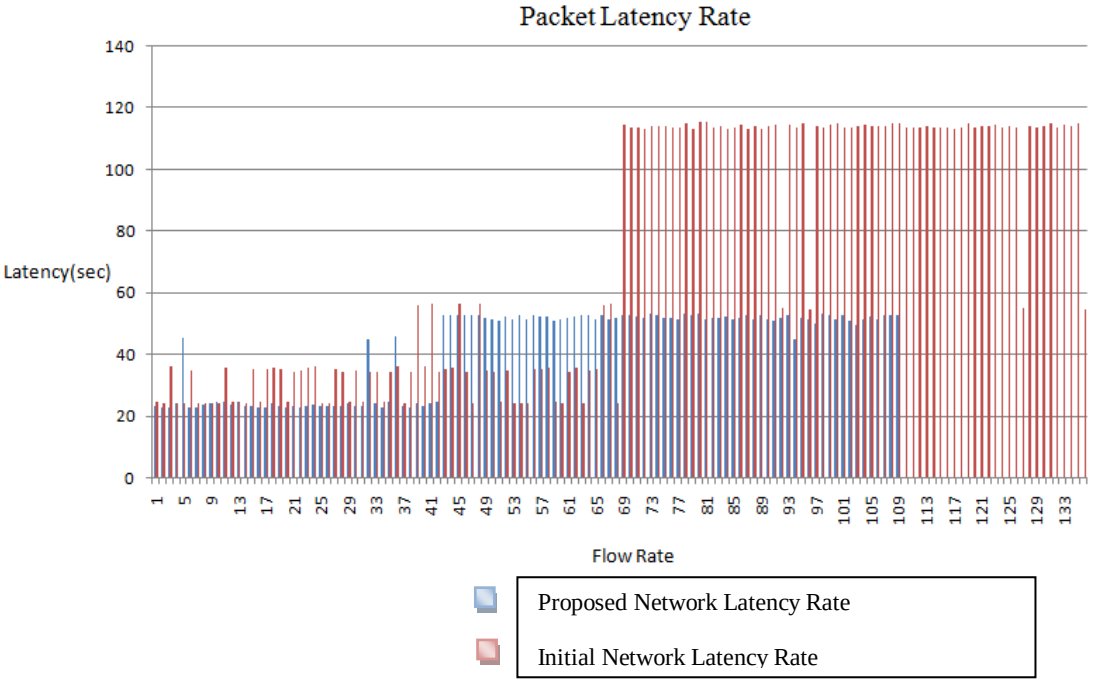


Figure 5.16 Packet Latency of Switch 4

Figure 5.15 and Figure 5.16 depicts the comparison for the packet transmission latency rate of both the initially designed and the proposed network layout, for switch 3 and switch 4.

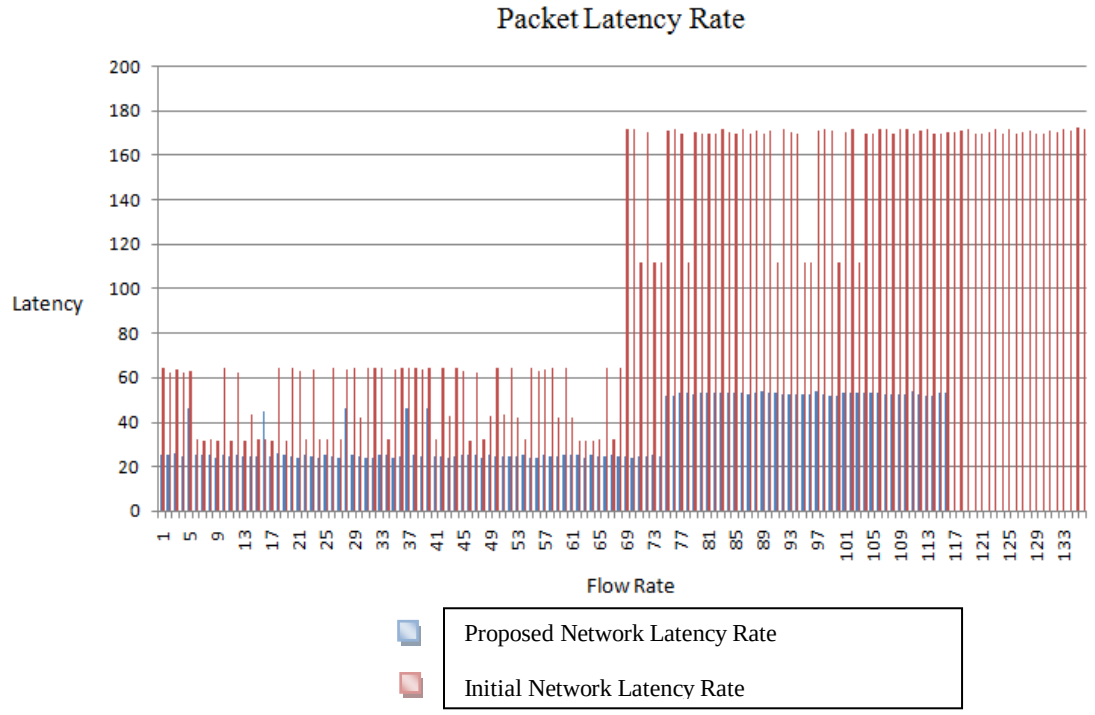


Figure 5.17 Packet Latency of Switch 5

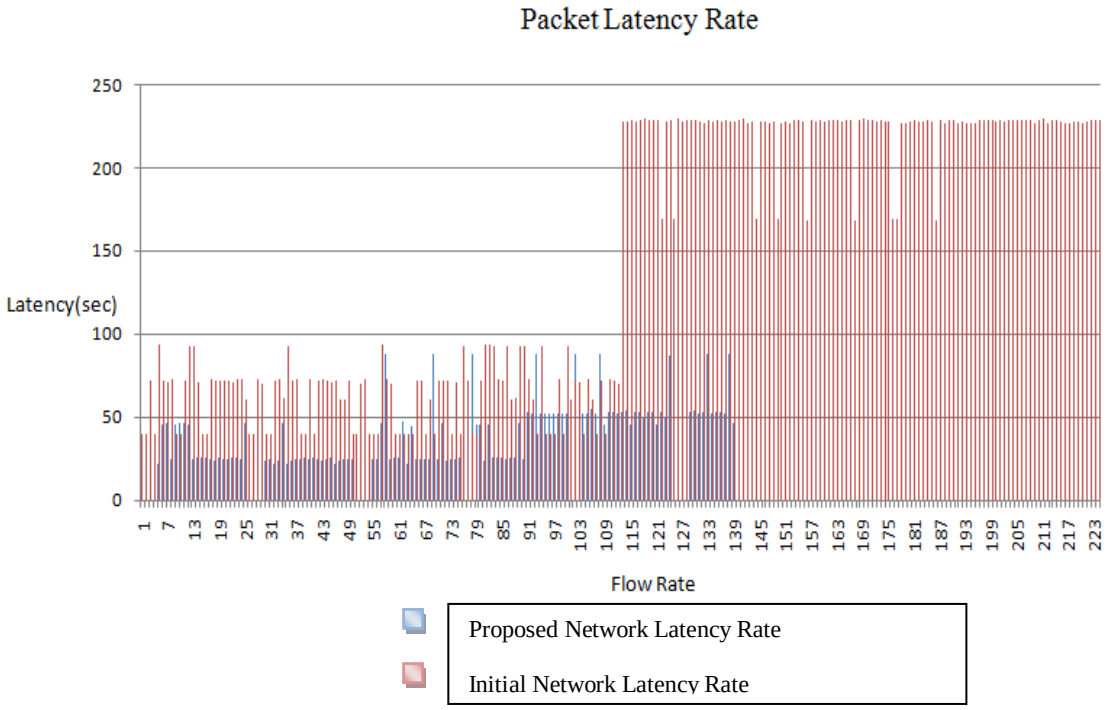


Figure 5.18 Packet Latency of Switch 6

Figure 5.17 and Figure 5.18 depicts the comparison for the packet transmission latency rate of both the initially designed and the proposed network layout, for switch 5 and switch 6.

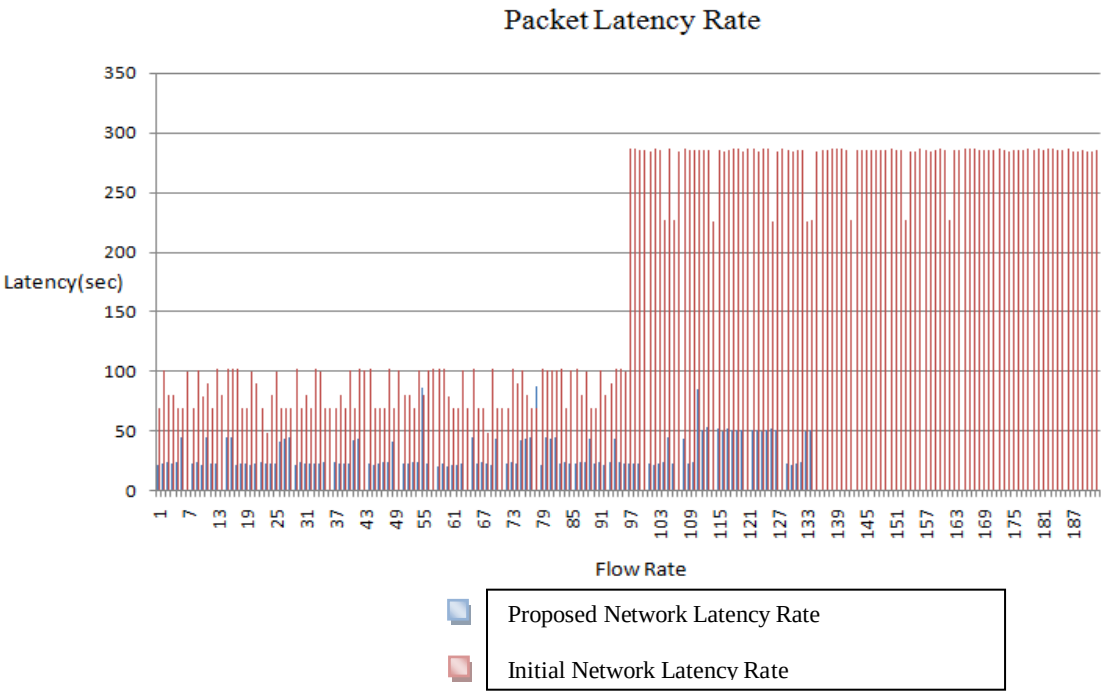


Figure 5.19 Packet Latency of Switch 7

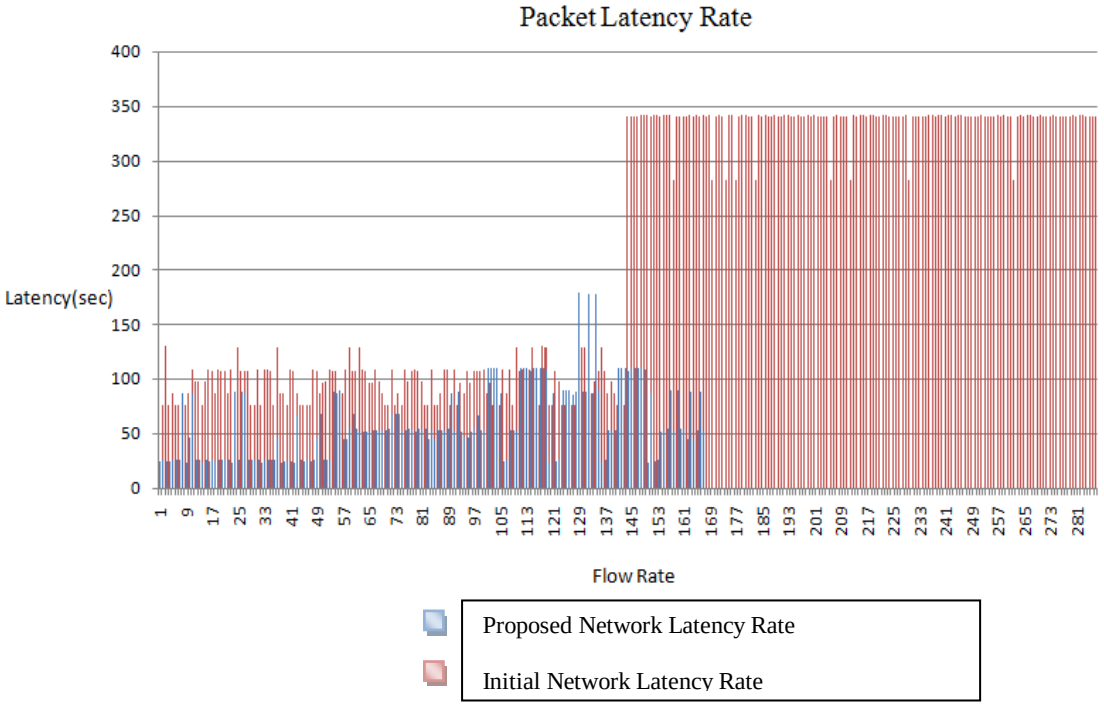


Figure 5.20 Packet Latency of Switch 8

Figure 5.19 and Figure 5.20 depicts the comparison for the packet transmission latency rate of both the initially designed and the proposed network layout, for switch 7 and switch 8.

CHAPTER 6

Conclusion and Future Scope

6.1 Conclusion

Controller positioning has always been a major concern in the networking. Multiple controllers must be restructured and repositioned to best fit the network need which is the future trend for SDN to achieve the quality of service on large scale SDN networks. The equilibrium between network performance and cost reduction should be fully evaluated. There is a need for an efficient algorithm need to be designed for quality of service. The factors affecting the Controller Placement in SDN must be reconsidered for their performance in the network for improvisation or regenerating some of the existing solutions and to provide with some alternatives to deal with the controllers failures and achieve a better network topology. There are multiple factors associated with the controller placement which require improvisation to deal with the upcoming revolution in SDN. Time efficiency and proper load distribution over a fault tolerant network layout was the targeted area which is successfully dealt with but there is always a scope of evolving something new with the technology developing better each day.

6.2 Future Scope

Controller Placement Problem in SDN networks has a wide area for research associated with multiple factors that can be targeted for further evolution of more effective network layouts. These improvised and upgraded network layouts can be very useful for building better wide area networks for faster transmission capabilities and also are cheaper in terms of development cost. In future there is a scope for the creation of a network layout which might deal with more factors affecting the functioning of these networks in SDN providing multiple services and being effective for the wide area usage.

Video Link

Video link on YouTube for the presentation over the Controller Placement Problem in SDN.

https://www.youtube.com/watch?v=oKLXS6

Appendix

Pratibha Kanwar and Dr. Rajesh Kumar "*Controller Placement Problem in SDN*," published in IJSRCSAMS journal, Vol.7, no.4, pp. 1-11, 2018.

References

[1] Nadeau, T. and Gray, K. " *SDN: software defined networks*." Beijing: O'Reilly, pp. 18-28, 2013

[2] O.N.F., "Software-defined networking: The new norm for networks," *ONF White Pap.*, vol. 2, pp. 2–6, 2012.

[3] G. Wang, Y. Zhao, J. Huang and W. Wang, "The Controller Placement Problem in Software Defined Networking: A Survey," in *IEEE Network*, vol. 31, no. 5, pp. 21-27, 2017.

[4] B. Heller, R. Sherwood, and N. Mckeown, "The controller placement problem," *in first workshop on Hot topics in software defined networks - HotSDN* , vol.19, p. 7, 2012.

[5] Y. Zhou, M. Zhu, L. Xiao, L. Ruan, W. Duan, D. Li, R. Liu, and M. Zhu, "A Load Balancing Strategy of SDN Controller Based on Distributed Decision," *in IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications*, no. 978, pp. 851–856, 2014.

[6] Y. Hu, W. Wendong, X. Gong, X. Que and C. Shiduan, "Reliability-aware controller placement for Software-Defined Networks," *in IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*, Ghent, pp. 672-675, *2013.*

[7] N. McKeown *et al.*, "OpenFlow: Enabling Innovation in Campus Networks," in *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, p. 69, 2008.

[8] S. Azodolmolky, "*Software defined networking with OpenFlow: get hands-on with the platforms and development tools used to build OpenFlow network applications"*. Birmingham, UK: Packt Pub., 2013.

[9] B. Heller, "OpenFlow Switch Specification 1.0.0," *Current*, vol. 0, pp. 1–42, 2009.

[10] B. P. R. Killi and S. V. Rao, "Capacitated Next Controller Placement in Software Defined Networks," in *IEEE Transactions on Network and Service Management*, vol. 14, no. 3, pp. 514-527, Sept. 2017.

[11] L. Mamushiane, A. Lysko and S. Dlamini, "A comparative evaluation of the performance of popular SDN controllers," *in Wireless Days Conference(WD)*, Dubai, pp. 54-59, 2018. 2018

[12] Y. J. Chen, L. C. Wang, M. C. Chen, P. M. Huang and P. J. Chung, "SDN-Enabled Traffic-Aware Load Balancing for M2M Networks," in *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1797-1806, June 2018.

[13] T. Hu, J. Zhang, L. Cao and J. Gao, "A reliable controller deployment strategy based on network condition evaluation in SDN," *2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, Beijing, pp. 367-370, 2017.

[14] S. Rout, S. S. Patra, B. Sahoo and A. K. Jena, "Load balancing in SDN using effective traffic engineering method," *International Conference on Signal Processing and Communication (ICSPC)*, Coimbatore, pp. 452-456, 2017.

[15] Images Reference Link
https://www.google.co.in/search?q=sdn+architecture&prmd=inv&source=lnms&tbm=isch&sa=X&ved=0ahUKEwifjfnbtPrbAhUWTI8KHfukAAsQ_AUIESgB&biw=414&bih=628

Thesis file

ORIGINALITY REPORT

7%	2%	5%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Guodong Wang, Yanxiao Zhao, Jun Huang, Wei Wang. "The Controller Placement Problem in Software Defined Networking: A Survey", IEEE Network, 2017 Publication	2%
2	Zhou, Yuanhao, Mingfa Zhu, Limin Xiao, Li Ruan, Wenbo Duan, Deguo Li, Rui Liu, and Mingming Zhu. "A Load Balancing Strategy of SDN Controller Based on Distributed Decision", 2014 IEEE 13th International Conference on Trust Security and Privacy in Computing and Communications, 2014. Publication	1%
3	www.cnblogs.com Internet Source	1%
4	www.hit.bme.hu Internet Source	1%
5	Morreale, . "OpenFlow", Software Defined Networking, 2014. Publication	1%