Improving Data Center Network Performance Using SDN

Gezahegn, Mamo

BAHIR DAR UNIVERSITY

INSTITUTE OF TECHNOLOGY

SCHOOL OF RESEARCH AND POSTGRADUATE STUDIES

FACULTY OF ELECTRICAL AND COMPUTER ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

Improving Data Center Network Performance Using SDN

By

Gezahegn Mamo

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Masters of Science in Computer Engineering

Advisor

Dr. Abebe Tesfahun (Ph.D)

July 26, 2022

Bahir Dar, Ethiopia

BAHIR DAR UNIVERSITY

INSTITUTE OF TECHNOLOGY

SCHOOL OF RESEARCH AND POSTGRADUATE STUDIES

FACULTY OF ELECTRICAL AND COMPUTER ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

Approval of Thesis for Defense

I hereby certify I have supervised, read, and evaluated this thesis titled "Improving Data Center Network Performance Using SDN" prepared by Gezahegn Mamo Negewo under my guidance. I recommend the thesis to be submitted for oral defense.

Dr. Abebe Tesfahun (Ph.D)	[illegible]	27-07, 2022
Name of Advisor	Signature	Date

BAHIR DAR UNIVERSITY

INSTITUTE OF TECHNOLOGY

SCHOOL OF RESEARCH AND POSTGRADUATE STUDIES

FACULTY OF ELECTRICAL AND COMPUTER ENGINEERING

Approval of Thesis for Defense Result

I hereby certify the changes required by the examiners have been carried out and incorporated in the final thesis.

Name of Student: Gezahegn Mamo Negewo Signature: [illegible] Date: 20-08-2022

As member of the Board of Examiners, we examined this thesis entitled "Improving Data Center Network Performance Using SDN" by Gezahegn Mamo Negwo. We hereby certify that the thesis is accepted as fulfilling the thesis requirements for the award of degree of Master of Science in "Computer Engineering".

Board of Examiners

Name of Advisor	Signature	Date
Dr.Abebe Tesfahun (Ph.D)	[illegible]	20-08-2022
Name of External Examiner	Signature	Date
Dr.Addisalem Genta (Ph.D)	[illegible]	20-08-2022
Name of Internal Examiner	Signature	Date
Dr.Sosina Mengistu (Ph.D)	[illegible]	20-08-2022
Name of Chairperson	Signature	Date
Mr.Molla Atanaw (MSc)	[illegible]	23/02/2015EC
Name of Chair Holder	Signature	Date
Mr.Molla Atanaw (MSc)	[illegible]	23/02/2015E
Name of Faculty Dean	Signature	Date
Mr.Tewodros Gera (MSc)	[illegible]	

[illegible] (PhD)

Tewodros Gera Workineh
[illegible] Computer
[illegible] Faculty Dean

Bahir Dar Institute of Technology [illegible] Engineering

II

DECLARATION

This is to certify that the thesis entitled “Improving Data Center Network Performance Using SDN”, submitted in partial fulfillment of the requirements for the degree for Master of Science in Computer Engineering under the Faculty of Electrical and Computer Engineering, Bahir Dar Institute of Technology, is a record of my original work carried out by me and has never submitted to this or any other institution to get any other degree certificates. The assistance and help I received during the course of the investigation have been duly acknowledged.

Gezahegn Mamo Negewo	[illegible]	July 26,2022
Name of the Candidate	Signature	Date

ACKNOWLEDGEMENTS

First of all, great thanks to Almighty God for his mercy in all my dreams and the strengthen me to accomplish this thesis. Secondly, I would like to thank and express the deepest appreciation and gratitude to my honourable advisor, Dr.Abebe Tesfahun (Ph.D), for his willingness, valuable guidance, and encouragement toward successful completion of this study. Last but not least, great thanks to my coordinator, Molla Atanaw (MSc) for consistent follow-up, passionate support and effort made indeed.

Finally, I would like to express my sincere gratitude to my mother Ilfe Biyo and my wife Nedhi Kinfe, who help me with the success of my life.

ABSTRACT

In this digital age, newly emerged network services and applications have made a huge demand for network performance. A large amount of traffic being generated from this very fast-growing rate of network traffic has made traditional network technologies reach their limit of capabilities. This huge traffic often causes frequent link congestions, and imbalanced traffic loads and network performance degradation issues on traditional Data Center Networks (DCN) services.

To mitigate these increasing demands and challenges, there have been numerous Information Technology (IT) temporary solutions developments in the past decade which do not fix the necessity of modern network problems. So that creative and effective methods of handling multiple applications and services with flexible traffic management and control mechanisms are still needed. In this study, Software Defined Networking (SDN) is used to improve the performance of network services, applications and systems as a whole. The traffic needs to be analysed by identifying network patter.

From the test results, we can conclude that using the proposed new architecture, Software Defined Networking (SDN) on existing Internet subscriptions, it is possible to achieve an average response time for Mail service 111.3ms, Web service is 99.7ms, for social media applications 110.5ms and for VoIP applications 113.62ms. The cumulative response time for SDN-based DCN for Internet services and applications is 108.78ms, which has 32.22ms relative improvement and with better throughput for all network traffic being transmitted in typical DCN.

Key words: **DCN, QoS, SDN**

TABLE OF CONTENT

Contents **Page**

ACKNOWLEDGEMENTS .... **IV**

ABSTRACT .... **V**

LIST OF FIGURES .... **VIII**

LIST OF TABLES .... **IX**

LIST OF ACRONYMS .... **X**

CHAPTER ONE: INTRODUCTION **1**

1.1 Background 1

1.2 Statement of the Problem 3

1.3 Research Questions 4

1.4 Objective 5

1.5 Scope and Limitation of the Study 6

1.6 Significance of the Study 7

1.7 Methodology of the Study 8

1.8 Organization of the Research 9

CHAPTER TWO: REVIEW OF LITERATURE **10**

2.1 Overview of Data Center 10

2.2 Overview of Data Center Network 11

2.3 Overview DCN Architecture 12

2.4 Overview DCN Performance Limiting Factors 12

2.5 Overview DCN Performance Improving Mechanisms 17

2.6 Overview of Modern DCN: SDN 21

2.7 Related Work 28

2.8 Research Gap 28

CHAPTER THREE: RESEARCH DESIGN AND METHODOLOGY 30

3.1 Case Description 30

3.2 Methodology Flowchart 31

3.3 Offline Data Collection from Existing Network 32

3.4 Data Center Network Environment Setup and Tools Used 33

3.5 Proposed DCN Design and Implementation Scenarios 39

CHAPTER FOUR: RESULTS AND DISCUSSIONS 42

4.1 Network Traffic Analysis 42

4.2 Existing Network Traffic Performance Analysis and Interpretation 42

4.3 Throughput Comparison Graph in Mininet 45

4.4 Delay Comparison of Both Legacy and SDN Based DCN 47

4.5 Summary of Finding 49

CHAPTER FIVE: CONCLUSION AND RECOMMENDATION 50

5.1 CONCLUSION 50

5.2 RECOMMENDATION 51

REFERENCES 52

APPENDIXES 58

I. Traditional DCN Architecture Topology Python Script 58

II. SDN Architecture Based DCN Topology Python Script 61

LIST OF FIGURES

Figure 2. 1: SDN Architecture [6] 22
Figure 2.2: Components of OpenFlow Logical Switch [47] 24
Figure 3. 1: Existing NMI DCN High Level Design 30
Figure 3. 2: Methodology flowchart for Improving DCN Performance Using SDN 31
Figure 3. 3: Simulation Environment Setup 33
Figure 3. 4: MiniEdit GUI 34
Figure 3. 5: ODL GUI Page 37
Figure 3. 6: ODL Main Working Environment 38
Figure 3. 7: Proposed Legacy DCN Design in Miniedit 39
Figure 3. 8: Proposed SDN Based DCN HLD 40
Figure 4. 1: Delay in (RTT)-July 1 to July 31 2021 43
Figure 4. 2: Packet Loss-July 1 to July 31 2021 44
Figure 4. 3: Throughput from July 1 to July 31 2021 44
Figure 4. 5:SDN Based and Traditional DCN Throughput 46
Figure 4. 6: Delay Comparison for both Traditional and SDN Based DCN 47

LIST OF TABLES

Table 1: Error Classification in Traditional DCN (Xia et al., 2017) 13

Table 2.1- OpenFlow Controllers Comparisons(Huang & Dong, 2020).. 23

LIST OF ACRONYMS

DCN	Data Center Network
ICT	Information and Communication Technology
IETF	Internet Engineering Task Force
ISP	Internet Service Provider
IT	Information Technology
LAN	Local Area Network
NMI	National Meteorological Institute
NOS	Network Operating Systems
NPM	Network Performance Monitoring
ODL	Opendaylight
OLA	Operation Level Agreement
OSGI	Open Services Gateway Initiative
QoS	Quality of Service
RTT	Round Trip Time
SDN	Software Defined Networking
SLA	Service Level Agreement
TCP	Transmission Control Protocol
ToR	Top of Rack
UDP	User Datagram Protocol
UI	User Interface
VM	Virtual Machine
WAN	Wide Area Network

CHAPTER ONE: INTRODUCTION

1.1 Background

In the last few years, the world is being live in dynamic change due to the technological advances. These technologies introduce new concepts, challenges and ways of solving problems in a manner that have been not seen before for the last two decades and ago. Especially in IT, the introduction of Internet and networking technologies such as of Cloud Computing, Artificial Intelligence (AI), Big Data, Internet of Things have provided the growth of network traffic coming into the data center from various devices(Mujib & Sari, 2020)(D. A. Popescu & Moore, 2021)(Yao et al., 2014)(Xue et al., 2015)(Wang et al., 2014)(Shirmarz & Ghaffari, 2020)(Balodis et al., 2013)(Govindarajan et al., 2014).

Basically, organizations can have or rent IT infrastructure to run their business-related applications and services across several networking devices in Data Center Network (DCN). These networking devices execute many protocols to deal with many network services and business applications. To deliver these services, network engineers and administrators should closely monitor and adjust network traffics in their data center. However, in traditional DCN, the policies are manually configured because of hardware dependent and the absence of a common standard to interoperate between vendor's devices. This makes network management and performance improvements a challenging issue to the network operators and researchers. Nowadays, the number of clients is alarmingly growing due to the rapid growth in the number of applications and services. These make the traditional DCN not ideal and insufficient to easily avail network services and applications(Xia et al., 2017)(Bitar et al., 2013).

Because of the rapid growth in the number of applications and services, network congestion occurs when traffic generated by network users exceeds the available bandwidth. In such circumstances, not all the packets sent by the sources can be immediately relayed on the route towards their destination, causing packet loss in transmission, poor QoS and overall

DCN network performance degradation. The performance of Internet service is closely linked to TCP protocol where it is associated with its congestion control mechanism using network parameters such as throughput, loss, delay and jitter implemented in QoS(Ignaciuk & Bartoszewicz, 2013) (Thombre, 2018).

To address this growing demand and challenge, numerous temporary solutions, such as the development of new protocols, network segmentation, and topology changes, have been implemented. However, these solutions do not address the necessity of modern networks. But effective methods of handling multiple critical applications and services with flexible traffic management and control mechanisms are still needed to give special treatment for network traffic flows during their transmissions. A good understanding of the DCN would enable us to design a more efficient, cost-effective, reliable and sustainable DCN. Hence, data center owners should adapt new concepts to stay up to date, be competitive and make sure they are not stalled in a deprecated technology no longer in use. It is also important to improve network traffic flows in their DCN in automated fashion to meet the rapid service delivery, scalability, and customized service level agreements (SLA) or Operational Level Agreement (OLA)(Wang et al., 2014) (Akter et al., 2020)(Nguyen et al., 2019)(Sugeng et al., 2015).

In this study, the most recent technique, SDN are applied to improve DCN of NMI. The proposed ML models are being used to classify Internet traffic and categorized into a number of traffic classes in certain DCN based on various parameters. This enables to learn and determine network traffic patterns, manage network traffic and meet organizational network services utilization SLA parameters while browsing, mailing, social interaction and browsing is a challenge.

Additionally, Software Defined Networking (SDN) platform, on the other hand, enables effective and efficient network resource utilization and centralized network management, which adds intelligence to control through consistently and holistically managing networks with the utmost flexibility and speed to improve network performance and provide services such as traffic engineering, QoS optimization, and congestion management.(Han et al., 2014)(Chang & Wang, 2015)(Hafeez et al., 2017)(Kathiravelu, 2016)

1.2 Statement of the Problem

The National Meteorology Institute (NMI) is one of the federal government offices of Ethiopia which anticipated to centrally disseminating services such as daily weather forecasts and regional meteorological information dissemination from a total of 919 active meteorological stations. As per interviews conducted with its ICT experts, the institute is utilizing its IT infrastructure effectively and aimed to increase its network services performance, which are being seen as very low comparing to the service needs. An observation is also made and confirmed that the data center has legacy network infrastructure with huge amount of traffic coming in to data center. Having the experience of network performance degradation has a major impact on critical organizations. It causes undesirable condition such as traffic propagation delay, response speed throughout that diminish QoS, causes customer dissatisfaction and service interruption.

1.3 Research Questions

The aim of this study is to identify and analyze collected samples of existing network services, as well as to propose, and recommend technologies that add some intelligence to create reliable DCN by answering the following main questions.

- Which network services are consuming network bandwidth?
- How to analyze existing network services and application performance improvement and have reliable DCN?
- How to mitigate DCN performance bottlenecks through software-based networking approach?
- Which platform is hotly being used to mitigate DCN performance bottlenecks through easier provisioning and monitoring

1.4 Objective

1.4.1 General Objective

The general objective of this study is to improve DCN performance using SDN.

1.4.2 Specific Objectives

The specific objectives of the study are:

- To analyze existing network applications and services' traffic performance levels
- To identify network services categories that consume network bandwidth,
- To propose new DCN topology using SDN
- To compare real-time network services traffic legacy and proposed modern network service performance

1.5 Scope and Limitation of the Study

1.5.1 Scope

Traditional networking is not inherently flawed, but, nowadays, there are some business challenges that require different solutions, services and applications to meet customer and business needs. To achieve this requirement, many organizations invest their capital to increase the performance of the network in the data center. However, in this study, the researcher has only focused on the improvement of enterprise DCN services and applications performance through traffic prediction and propose new platform with topology in the case of NMI to be implemented on simulated environment.

1.5.2 Limitation

In this study;

- Only one-month network traffic was collected for a dataset in this study.
- A limited number of network metrics are selected for the reference
- Only traffic is exchanged between local networks and external networks.
- The proposed platform and topologies are implemented in
  the simulation environment.

1.6 Significance of the Study

Nowadays, the way we communicate for social and business issues are almost dependent on Information and communication technologies (ICT) that assist to live digital living for users to browse, mail, chat and calls. Accordingly, network needs have grown dramatically due to the advance of IT technologies. These can produce network congestion in DCN node because of carrying more data that its capacity. This introduces QoS degradation in connectivity and response speed throughout a given network and reduces the performance of network applications and services. This can affect traditional it is difficult to provide QoS for each application which is difficult to the analysis and diagnosis to maintain consistent SLA or OLA.

Thus; this research work contributed to organization not to commit to specific manufacturers that may block the development their DCN in the future. Two new topologies have proposed using new paradigm, SDN architecture which becomes the most alarming topic with innovative trends and providing various application(Valencic & Mateljan, 2019). The proposed designs will be demonstrated and evaluated using network emulation tools to define a logical network infrastructure.

Finally, both theoretical and practical contributions towards improving the performance of network traffic flow have provided. The theoretical contributions of this study are the researchers use as reference to study on the related topics using defined metrics as well as the proposed framework for further studies. The practical guide will have used for understanding how to plan, analyze, evaluate, design, implement, operate and improve the performance of network service. Further studies have also suggested enhancing the performance of network services.

1.7 Methodology of the Study

In order to achieve the objectives of this research, the researcher has employed research methodologies listed below:

- Review of Related Literature: To understand the subject matter, we have conducted a literature review to assess the major issues and concepts in the field done so far in the area of DCN service performance improvement on relevant literature (books, journals, articles, conference papers, research reports and web documents) pertaining to the research under consideration.
- Selection of Metrics of Network Services Performance: This provides information about the network utilization, the type of network services using the network, application flows that exist between the network nodes, the top applications services talkers on the network.
- Data Collection and Analysis: The Solarwinds tool is used as data collection method from existing system. Analysis has performed to understand the existing network services performances.
- Propose new architecture and design for DCN: This study is aimed to propose and adopt the currently leading DCN architecture, SDN to enhance network services performance.
- Design and simulation of the proposed network architecture has designed after the knowledge is developed and clearly understood the solution to the identified problem.
- The simulation environment has done on the laptop which the Mininet, mini-edit, Opendaylight (ODL) controller, are installed on oracle virtual box as hypervisor to run Ubuntu 20.04 Linux operating system as test bed

1.8 Organization of the Research

This research is organized in five chapters; the first chapter is concerned with introducing background, statement of the problem, basic research questions, objectives, significance of the study, scope and limitation of the study. The second chapter is a review of literature; it has the conceptualization and framework of network service performance and related works. Chapter three covered on the methodology of the study. The fourth chapter includes results and discussion or data analysis. The last chapter five includes conclusion and recommendation that will be used for future work.

CHAPTER TWO: REVIEW OF LITERATURE

The researcher has reviewed different related literature on previously conducted research dissertations, books, journal articles, conference proceeding, and other reliable source of knowledge from reviewed to have detailed understanding on the present research to understand the problem and emphasize the research gap in relation to the study. This gives a review of earlier contributions on improving network performance. This chapter has also included an overview existing network. Some papers are reviewed on traditional networking limitations and proposed the migration of a platform to maintain dynamism in the network adaptation to the changes in traffic SDN. This enables to achieve the objective this study to have DCN with enhanced capabilities on services in rapid response to changing business needs or user demands.

2.1 Overview of Data Center

A data center is defined as a facility that centralizes and houses different hardware, software and virtualized systems for IT operations. Those systems such as storage systems for backup, computing components run the applications, server farms, network infrastructure which interconnects between data center components using fiber or copper links to the outside world through routers, switches, and various information security elements, such as firewalls. These devices are contained in racks distributing power systems in room having cooling system, physical security system, safety solution and operational staff who monitor operations and maintain IT operations (Wang et al., 2014).

The data center can contain physical and virtualized IT infrastructures and to run different services. It can be private, public, or virtually private. Where private data center is dedicated to one enterprise as it is typically constructed, owned and operated by a single organization for the sole use of supporting their own internal purposes. It can be located either on-premises or off-premises at a site chosen for connectivity, power, and security purposes. Public data center that can be rented out to individual companies to host their applications or use and pay as they use and the provider manages it where virtually private partially act as both private and public data centers. As it often house an organization's business-critical data, services applications, and communication, the security, availability and the network services performances information are among any top priorities and it’s essential to improve DCN services performance (Wang et al., 2014)(Bitar et al., 2013)(Li et al., 2014).

2.2 Overview of Data Center Network

A DCN consists of physical of networking resources used for switching, routing, load balancing, virtual networks and protocols. Those component work together to achieve a common objective and serve these applications and connecting all users and systems on a local area network (LAN) to applications in the data center and Internet as well as facilitating access to network from remote side. It is the backbone infrastructure that used to house existing and emerging applications and services such as web search, file sharing, email, real-time applications with continuously increasing computing requirements(Balodis et al., 2013)(Nguyen et al., 2019).

Since these growths in data has immensely impacted organizations it is very important to accommodate the needs to meet Service Level Agreement (SLA) and Operation Level Agreement (OLA) requirements and keep promises by identifying network users’ demands and behaviors on the network in DCN infrastructure through managing network traffic flows and uses the DCN infrastructure in efficient and effective manner by prioritizing needs and realizing a QoS is very critical(Alrokayan et al., 2015).

2.3 Overview DCN Architecture

Data center networks can be implemented as switch-only architectures in which packet forwarding using switches and server- only architectures which uses servers for packet forwarding to run applications(Xia et al., 2017), and to forward packets between servers. Network engineers and administrators choose and evaluate the designs in cost effective wise. Since these architectures impact the performance of applications and services, switch-only fat tree data center architecture is used in this study.

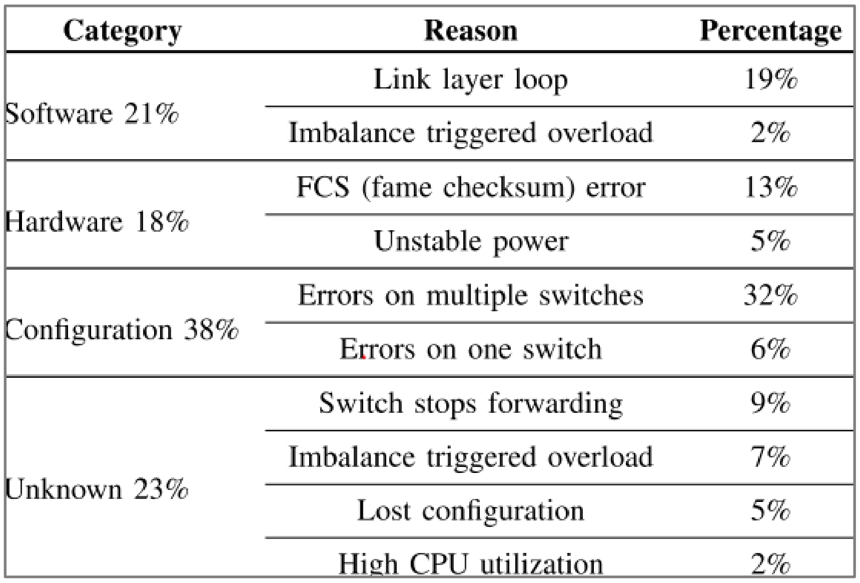
2.4 Overview DCN Performance Limiting Factors

In data center network applications and services are producing huge network traffic such as social networks, web portals searching, online banking, mobile application, weather forecasting, Internet of Things applications and so on. Handling such large amount of data is a difficult task for traditional DCN architectures and mechanisms and always leads to congestion. These results make DCN for poor service deliver. A typical DCN performance can be characterized and limited using many features such as architecture, congestion, bandwidth, reliability, throughput, latency and cost which have a great role in influencing overall system performance(Balakiruthiga & Deepalakshmi, 2019).

2.4.1 Technological Limitation in Legacy Network Devices

In traditional network approach has architecture that contains forwarding, controlling and management planes existing in the same device designed to provides business oriented proprietary requirements of typical industry. There is high configuration error in this type of network. By doing so, the purchased devices need training and support fee to be properly operated. Modification is limited as per industry prebuilt operation commands hence creativity is restricted. There is no open and common standard between different vendors so that inter-operability of network devices this raises vendors' dependability issues(Xia et al., 2017).

Table 1: Error Classification in Traditional DCN (Xia et al., 2017)



Category	Reason	Percentage
Software 21%	Link layer loop	19%
	Imbalance triggered overload	2%
Hardware 18%	FCS (fame checksum) error	13%
	Unstable power	5%
Configuration 38%	Errors on multiple switches	32%
	Errors on one switch	6%
Unknown 23%	Switch stops forwarding	9%
	Imbalance triggered overload	7%
	Lost configuration	5%
	High CPU utilization	2%

2.4.2 Architectural Limitations of Traditional DCN Infrastructure

Traditional DCN infrastructure is built based on a three-layer. This model has layers namely core layer switches which connect to distribution layer switches or aggregation switches. It, turn connect to access layer switches mostly located at the top of a rack (ToR). In this model the traffic between two nodes in the same rack Layer 2 of the network, is sent with low latency, which is good news. But the problem is that it is expensive and not deterministic the traffic moves to the aggregation layer and center core in Layer 3 traffic, it needs to leave the rack and reach the aggregation tier of switches before being routed, even back to the same rack. In traditional DCN once the model is put in place, change is difficult, visibility into traffic is limited, and debugging is a challenge, QoS, traffic prioritization, and packet or traffic capture for regulations or debugging are all challenges or impossible (Wang et al., 2014).

2.4.3 Network Congestion

A large number of packets may suddenly arrive on specific router, the arrival intensity of new packets goes beyond the router's transmission capacity, then router is overloaded could give rise to congestion. That means congestion occurs when load on the network, the number of packets to be sent to the network is greater than the capacity of the network the number of packets that a network can handle and so that this causes network performance issues traffic cannot traverse the network or availability issues , packet loss occurs due to packets fail to reach their destination which results TCP packet retransmissions that is bad bandwidth utilization make the latency problem on applications and network service (Hafeez et al., 2017)(Hao et al., 2019)(Zakia & Ben Yedder, 2017)(Huang & Dong, 2020).

2.4.4 Network Latency

In a network environment latency is a time delay between the system being observed or the time that data travels in the network. It can be expressed as one-way latency or as roundtrip latency. One-way latency, also known as delay, simply means the time it takes for data to travel from the transmitting node to the receiving node. Roundtrip latency, also known as Round Trip Time (RTT), measures the time data travels from transmitting node to the receiver plus the time that it takes for the transmitting node to get a response (acknowledgement) from the receiving node. When studying application performance, the most commonly used form of latency is the RTT. To address end-to-end service delivery in DCNs in communication paths in the Internet often traverse multiple system components operated by different network service organizations(Xue et al., 2015)(He et al., 2016).

2.4.5 Network Availability

Obviously; DCN is used as a network infrastructure for carrying, transmitting, storing and processing data. The connectivity and performance of application demands placed on the network. This can be done using network availability the amount of time a network is fully operational and measured as a percentage. It can be monitored to ensure the service consistently kept running for end-users. It can be tested using the ping program which sends an Internet Control Message Protocol (ICMP) echo request packet to the destination host. Maintaining a certain level of network availability can help organizations with disaster planning, recognizing when issues arise and providing users with a specified standard such as SLA or OLA (Nguyen et al., 2019).

2.4.6 Network Packet Loss

In network environment, data is sent or received across the network in small units called packets. This applies to everything we do on the internet, from browsing websites, emailing, uploading or downloading images or files, streaming, voice and video chatting and communication. When one or more of these packets is interrupted in its journey or fail to reach their destination it is named as packet loss. It causes the network congestion which occurs when network traffic hits its maximum limit, packets are discarded. Additionally, deficient infrastructure or traditional network devices, quality of the physical medium and reliable protocol like TCP in which retransmission makes the latency problem. Increased network latency and packets losses can affect substantially application performance(D. A. Popescu & Moore, 2021)(He et al., 2016).

2.4.7 Network Throughput

Throughput defined as the rate of successful data transfer in the network. That it is a sum of three different parameters: network capacity or bandwidth, latency and packet loss. Capacity means the maximum amount of information that can be transferred between two network nodes. The Low latency and high throughput are the most critical QoS performance requirements of DCN. For web applications continue to thrive, data center must remain effective by addressing these challenges(Xue et al., 2015)(Hafeez et al., 2017).

2.4.8 Delay

Latency is a measure of delay which is described as the time it takes for some data to get to its destination across the network usually measured as a round trip time which is the time taken for information to get to its destination and back again. It significantly affects application responsiveness. It results poor QoS when it is abundantly experienced in typical network. (Xue et al., 2015)

2.4.9 Network Bandwidth Utilization

A network bandwidth is a measurement indicating the maximum capacity of a wired or wireless communications link to transmit data over a network connection in a given amount of time. It is important to a variety of network applications. It can be measured using throughput, and packet pair in traditional DCN. However, these methods have poor accuracy, scalability, lack of robustness, poor agility in adapting to bandwidth changes, lack of flexibility in deployment(Akter et al., 2020).

2.4.10 Network Jitter

Jitter in computer network is the difference between a successful Voice over Internet Protocol (VoIP) call and a disastrous one. As information is transported across the Internet it usually sent at regular intervals and takes a set amount of time. Jitter is caused when there is a time delay in the sending of these data packets over network connection which is often caused by network congestion, and sometimes route changes. Essentially, the jitter can negatively impact the video and audio quality when making a call or conference call. So that a good congestion control mechanism can results a better network throughput with constant latency with insignificant variation or jitter (Thombre, 2018).

2.5 Overview DCN Performance Improving Mechanisms

Nowadays data centers are growing in numbers and size, and their networks are expanding to carry such larger amounts of traffic coming from constantly varying number of users and customers. With the growth of data volumes and variety of Internet applications DCN should become an efficient and promising infrastructure to support data storage, and provide network services and applications. These require higher number of bandwidths as more devices are being connected. These applications and services often impose different resource demands on the underlying infrastructure. To accommodate these needs data center owners, always change their resource requirements in through network re-configuration must take place(Shekhawat et al., 2018).

Traditional DCN architectures lack the flexibility to effectively support critical services and applications, which results poor support of QoS, deployment, manageability, and even results security attacks. To overcome these issues research community has been focused on SDN to meet those demands through improving the DCN performance in different aspects of operations being with different industries. Different techniques such as network traffic splitting that make redundant network paths within a network; Load balancing that distributes client requests across several network devises. Traffic steering(Leivadeas et al., 2016), QoS that implies giving some traffic classes higher priority service than others, possibly to the extent of giving guarantees on latency, bandwidth, and packet-loss to specific data flows. So that a typical data center can contain various technologies in order to utilize its network infrastructure efficiently and have a better network performance(Shirmarz & Ghaffari, 2020)(Shekhawat et al., 2018)(Sherwin, n.d.).

2.5.1 DCN Virtualization

A network virtualization is a solution that is used to create multiple logical networks by partitioning available resources and share them among different users. This enables to prioritize traffic that might be shared among different external networks through guarantee bandwidth sharing, multi-patching features. It enables network functions usually run on hardware to be delivered as software and managed as a single entity. The earliest implementations of network virtualization is Virtual Local Area Networks (VLAN) which make logically different single switch to support multiple networks on the same hardware infrastructure, such as multiple Internet Protocol (IP) addresses on the same switch for scaling network traffic and security purpose(Balodis et al., 2013)(Hu et al., 2014)(Sharma* & Tyagi, 2019).

It is data forwarding technology that increases the speed and controls the flow of network traffic based on the shortest path and deliver services through Internet where the actual hardware is managed and run by question, often with the help of a third-party. These technologies may have comparative advantages and disadvantages which including performance, usage and adoption, troubleshooting due to lack of expertise, privacy and security challenges (Shirmarz & Ghaffari, 2020)(Bitar et al., 2013)(Keti & Askar, 2015).

2.5.2 DCN Load-balancing

Distributing network traffic across multiple network devices accommodates massive data transmissions which improve application responsiveness. This can be achieved through load-balancing process which can handle millions of requests per second after it receives a connection, it selects a target from the target group using routing algorithm and forwards the request without modifying. It loads network traffic TCP/IP application protocols and applications efficient network infrastructures utilization in an economy of scale suitable to have higher application throughput and network availability to the end user and the overall network resiliency. A load balancer is used to distribute network or application traffic across a number of network entities (Ponciano & Anani, 2014).

2.5.3 Qos Implementation in DCN

In DCN addressing massive network traffic being carried from source to destination at low latency can increase the rate of successful data transfer, throughput. This can be achieved by implementing the QoS mechanism which provides network services based on different service level. Depending on business needs detailed parameters such delay, jitter, packet loss and throughput should be allocated. This makes network devices to treat the different application's traffic based the given SLA to guarantee and provide perfect service capabilities (Xue et al., 2015) (He et al., 2016).

2.5.3.1 Best-Effort Model

This model threats network traffic equally such that does not give any guarantees. So that there is no configuration needed for classification and differentiation for different applications. It provides scalability and eases to handle but it has a lack of service guarantee and lack of service differentiation(Thombre, 2018).

2.5.3.2 Integrated Service Model (IntServ)

Integrated Service Model is used to achieve end-to-end QoS to fulfill the requirement for real-time applications. The network bandwidth is reserved for applications to guaranteeing bandwidth, delay, and packet loss using resource reservation protocol for signaling and reservation before application takes the start. Once the bandwidth is reserved for a certain application, it cannot be reassigned for another application which is difficult to implement in today's large networks due to too much operation overhead (Hu et al., 2014).

2.5.3.3 Differentiated Service Model (DiffServ)

This model comes to overcome the limitation of the IntServ model that does use the signaling protocols. It specifies and controls network traffic by class so that certain types of traffic get precedence using four well-known mechanisms. First it uses traffic classification process to divide the network traffic into different categories at customer edge. Secondly it colors incoming network traffic packet on the physical interface using traffic marking. Then allocate the QoS parameters such as bandwidth, delay, and reliability using Per-Hop Behavior. Finally adjust the traffic output rate and is used to restrict the total traffic that leaves a network using traffic shaping mechanism to provide a uniform packets transmission rate and ensure network stability. It is widely used in industry due to its' scalability(Bitar et al., 2013).

Most likely the root causes of network performance degradations issues in the aspects of IT system raised above are solved through adjusting the following main QoS parameters considered to attain reliability DCN services stated as below network performance metrics formula [15].

$$\text{Packet Loss } = \frac{\text{Packet Sent}-\text{Packet Received}}{Packet\ Sent} x100\% \tag{1}$$

$$\text{Delay } = \frac{\text{Packet Length(bit)}}{\text{link bandwidth }(\frac{\text{bit}}{\text{second}})} \tag{2}$$

$$\text{Throughput } = \frac{\sum\text{Sent dat(bit)}}{\text{time data delivery(second)}} \tag{3}$$

$$\text{Jitter } = \frac{\sum\text{variation delay}}{\sum\text{packet reeived}}\text{second} \tag{4}$$

Generally, this study aims to improve DCN service and application performance to have great throughput of big flows and reduce the latency of small and medium flows. Then for future deployment, a scalable, low latency, high-speed, and DCN which is highly required in large-scale institutions like NMI, which is our case study. The monitoring tool like Solarwinds network performance monitoring (NPM) tools used to collect application or services status based monitoring, flow monitoring, packet capture, bandwidth analysis, and network monitoring to overcome concerns such as network availability, bandwidth, packet loss, congestion, latency, and jitters that can affect the overall performance and raise customer dissatisfaction issues.

2.6 Overview of Modern DCN: SDN

In traditional network the tight coupling between the controlling plane and forwarding plane makes the design and management processes of the networks a complex task. To overcome this revolutionary networking paradigm, SDN is emerged as the new network framework to make networks wholly controlled through software applications which decouples the controller from forwarding plane which are centrally programmed through open interfaces. This adds intelligence through programmability to confront limitations and challenges of today's networking infrastructures. This features makes the SDN to configure multiple devices centrally, specifically, in DCN, in short time with reduced maintenance cost which is time taking in conventional network. Due to this programming capabilities it gets more interest by many researchers in simplifying the design and management(Mujib & Sari, 2020)(Wang et al., 2014)(Hu et al., 2014)(De Oliveira et al., 2014).

The SDN has low cost and structural complexity comparing with conventional network as it is used to design an efficient and cost-effective DCN with high availability and security. These capabilities let IT administrators manage traffic flow with greater detail from one central dashboard using user interface (UI). A typical SDN has three main layers in its architecture. Namely control plane, data plane and application. These layers communicate with each other with the east-west and north-south application programming interfaces (API) as depicted in Figure 2.1 below(Shirmarz & Ghaffari, 2020).

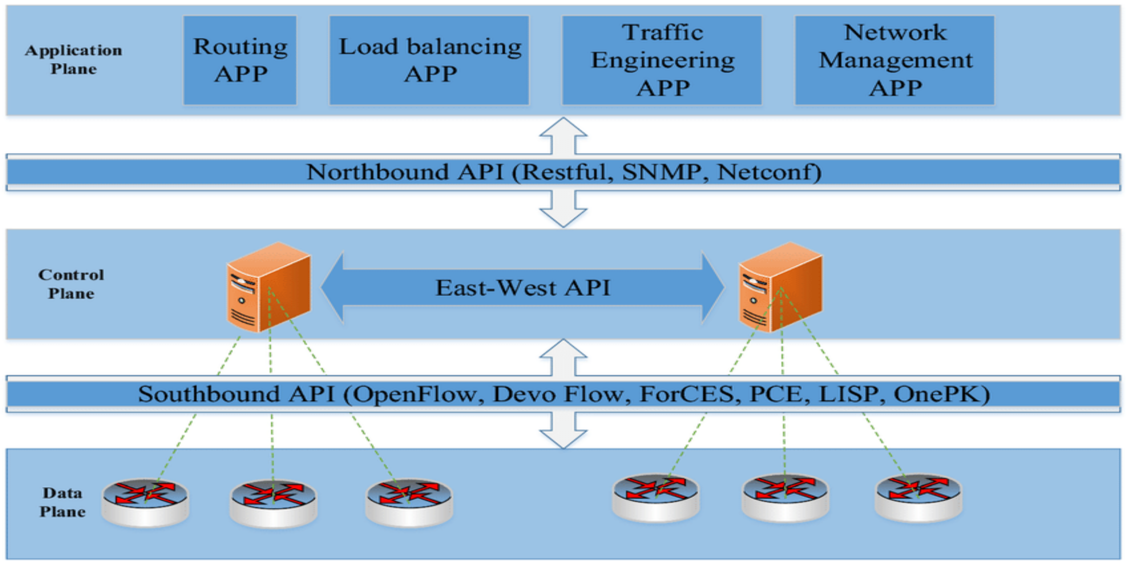


Figure 2. 1: SDN Architecture (Shirmarz & Ghaffari, 2020)

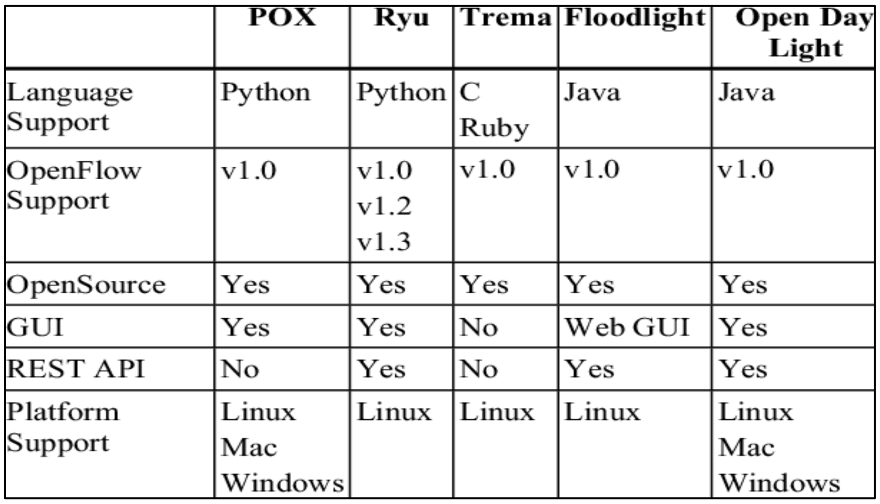
SDN provides two categories of API. The Southbound API which is used to achieve communication between the controlling and forwarding plane in order to easily supply and configure network elements and the Northbound API in order to provide various types of virtualized services for the controller.

2.6.1 Overview of SDN Controllers

An SDN Controller is a network entity that acts as brain of the SDN network. It uses southbound APIs for transmitting information to the specified switches/routers and northbound APIs are used to carry information between the controller and the applications and business logic. This enables network operators and engineers to program the network, which is not possible requirement in traditional networking. The controller has a graphical user interface (GUI) dashboard as it includes a web manager for displaying the network architecture and manage it(Kathiravelu, 2016).

Thus SDN, optimizes network performance by handling all communications between applications and devices to effectively manage and modify network flows to meet changing needs. Since network control plane is implemented in software, rather than firmware, administrators can centrally manage network traffic. There are many SDN controllers offered in different languages for programming preference as shown in Table 2.1 below(Huang & Dong, 2020).

Table 2.1- OpenFlow Controllers Comparisons(Huang & Dong, 2020)



	POX	Ryu	Trema	Floodlight	Open Day Light
Language Support	Python	Python	C Ruby	Java	Java
OpenFlow Support	v1.0	v1.0 v1.2 v1.3	v1.0	v1.0	v1.0
OpenSource	Yes	Yes	Yes	Yes	Yes
GUI	Yes	Yes	No	Web GUI	Yes
REST API	No	Yes	No	Yes	Yes
Platform Support	Linux Mac Windows	Linux	Linux	Linux	Linux Mac Windows

2.6.2 Open vSwitch

Open vSwitch (OVS) is a software implementation of a virtual multilayer network switch designed to enable effective network automation through programmatic extensions. It is the core element of many datacenter SDN deployments and the main use case is multi-tenant network virtualization. In some cases, it could be considered critical to many SDN deployments in data centers because it ties together all the VMs within a hypervisor instance on a server(Govindarajan et al., 2014).

2.6.3 Overview of OpenFlow Switch

An OpenFlow a tool that forwards packets in SDN environment. It consists of the three basic components. These includes OpenFlow Ports which packets will enter the switch and exit it through them. The Second component OpenFlow table which performs packet analyzing and forwarding. The third component is OpenFlow entries which are used to match and process packets according to their packet headers using channel interface to communicate the switch with the controller in such way that the switch receives the configuration from it as and depicted in Fig 2.3 below(Open Networking Foundation, 2015).

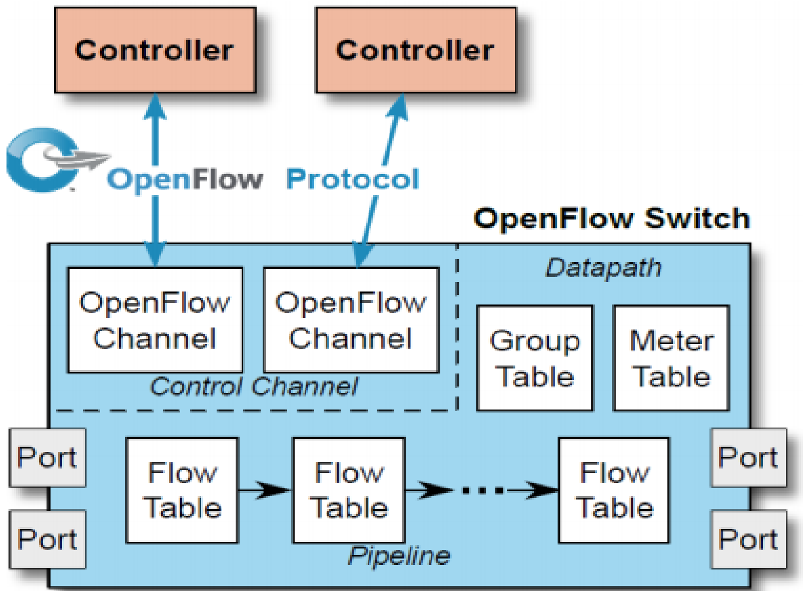


Figure 2.2: Components of OpenFlow Logical Switch (Open Networking Foundation, 2015)

2.6.4 Openflow Protocol

OpenFlow is a programmable network protocol mailny is used to manage and forward traffic in SDN based network environment. It defined by the Open Networking Foundation (ONF) to make communication between an OpenFlow switch and the controller in an SDN environment. It is also used to facilitate the programmability of the network through the configuration and control of data flows from a controller to OpenFlow switches (Kathiravelu, 2016).

2.6.5 Overview of Application Program Interface

Since the early dawn of networking, devices have been configured through virtual terminal through telnet or secure shell but the underlying rule still maintained that network is configured manually device-by-device by a human administrator. It’s obvious that this approach does not scale and is prone to human error, however it still remains the most prevalent method of network device configuration(Shirmarz & Ghaffari, 2020)(Govindarajan et al., 2014).

The SNMP is used to monitor and manage network devices using a strictly-defined data structures. Unfortunately, due to several architectural issues and poor vendor implementation, it has ended up being used mainly for basic monitoring tasks. The idea of programmatic management of network devices later resulted in the creation of NETCONF protocol which is still not widely used due to limited vendor support, however things may improve as the YANG-based network configuration gains greater adoption(Aziz et al., 2017).

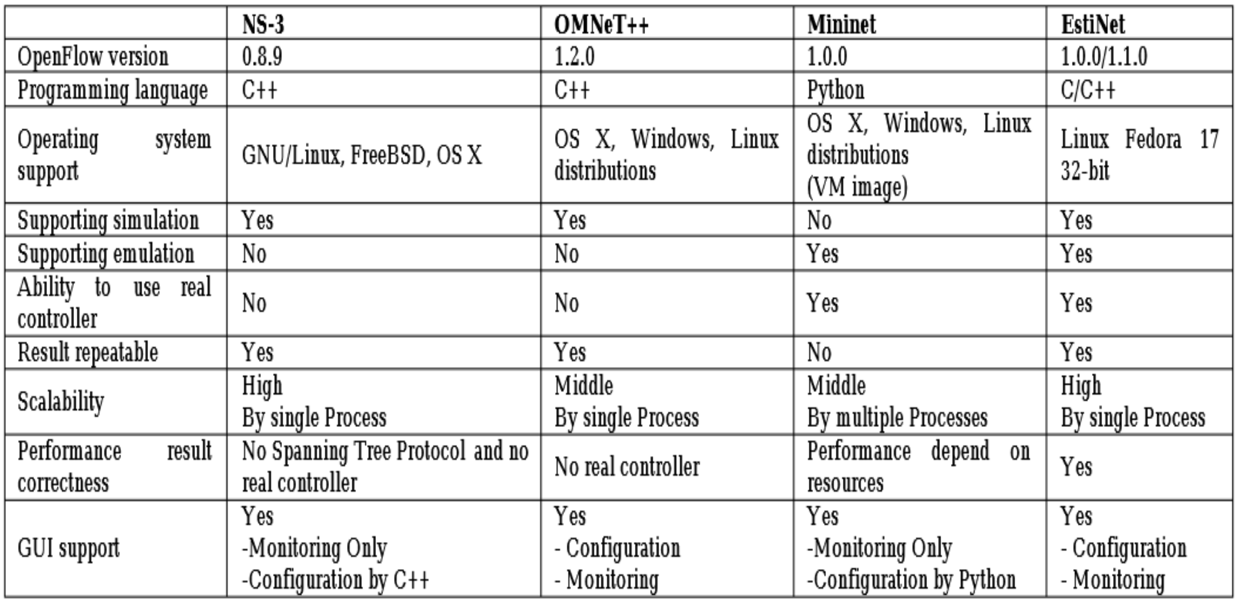
Finally, with the beginning of SDN, Representational State Transfer (REST) an application program interface (API) which is a set of routines, protocols, and programming tool with GUI components become a new de-facto standard for network provisioning. It is supported by most of the latest products of all the major vendors(Hafeez et al., 2017).

Postman is an application for testing APIs by sending request to the controller and getting the response back. It was used to send RESTCONF GET API to retrieve node inventory and topology as created by Mininet and seen by ODL controller (Valencic & Mateljan, 2019).

2.6.6 Overview of SDN Simulators

Network simulation simulator is software program that replicates the behavior of a real network. It demonstrates the behavior of a network and its components in abstract way. It is also methodology used to evaluate different network topologies without real world implementation is mainly used in different areas such as academic researchers, industrial development, to analyze, design, simulate and verify the performance of different network theories and hypotheses. Since establishing a network in a real time scenario is very difficult due to single test bed takes a large amount of time and cost. Thus a network simulation can generate certain parameters such as simulated throughput and simulated delay based on network design built in the simulator. Furthermore, analysts can study relationships between nodes, hosts and applications using simulations. So, this provides multiple design options before having to implement the outcome in real world(Li et al., 2014)(De Oliveira et al., 2014).

Table 2. 2-Comparison of Network Simulation Tools(Pupatwibul et al., 2015)



	NS-3	**OMNeT++**	**Mininet**	**EstiNet**
OpenFlow version	0.8.9	1.2.0	1.0.0	1.0.0/1.1.0
Programming language	C++	C++	Python	C/C++
Operating system support	GNU/Linux, FreeBSD, OS X	OS X, Windows, Linux distributions	OS X, Windows, Linux distributions (VM image)	Linux Fedora 17 32-bit
Supporting simulation	Yes	Yes	No	Yes
Supporting emulation	No	No	Yes	Yes
Ability to use real controller	No	No	Yes	Yes
Result repeatable	Yes	Yes	No	Yes
Scalability	High By single Process	Middle By single Process	Middle By multiple Processes	High By single Process
Performance result correctness	No Spanning Tree Protocol and no real controller	No real controller	Performance depend on resources	Yes
GUI support	Yes -Monitoring Only -Configuration by C++	Yes - Configuration - Monitoring	Yes -Monitoring Only -Configuration by Python	Yes - Configuration - Monitoring

2.6.7 Overview of Mininet

There are many major challenges in network simulation. These are scaling, evaluating the performance and easily migration to a real system with minimal changes for deployment. However, an emulation tools are used as testing the functions and evaluating the performances of SDN since it includes new programming languages, static analysis and debugging capability, and innovative tool. This also applies to a Mininet, in the sense that it is an emulation platform for networks, including hosts, switches, controllers, and network applications and for the functional testing of OpenFlow protocols. It was created in python and a Python API for user customization is provided on which hosts run standard Linux operating systems(De Oliveira et al., 2014).

Therefore, in this study, Mininet we used for observation purposes because it is a widely used experimentation tool in which easy to write Python scripts to set up and configure the emulation case which allows us to create custom topologies(Pupatwibul et al., 2015).

2.7 Related Work

Paper (Di. A. Popescu & Moore, 2018), presents that the Internet has grown at a very fast which result heavy traffic patterns and become unpredictable which can deteriorate the overall network performance, cause congestion. A network traffic monitoring and analysis are used to troubleshoot and resolve problems through constant monitoring of the network traffic and notifies the administrator whenever there is an outage. But in this paper rather showing and notifying the issues no improvement measure is proposed.

In paper(Keti & Askar, 2015) explains that Mininet and ODL were suitable tools to conduct scalability analysis and flow admission in a SDN that Mininet is freely available and has already built in OVS. It is also effortless because of easily building topology via drag and drop capability using Miniedit utility found in Mininet. The ODL will be used to improve overall network performance and scalability study. During this study, they also observed the effectiveness of Mininet especially on time and resources according to prototyping, deployment and sharing in DCN.

According to survey(Shirmarz & Ghaffari, 2020) SDN has shown a great performance improvement in data center and Cloud, Wireless, and WAN based on metrics such as delay, jitter, packet loss and energy. SDN offers a holistic view of the entire data center network to a logically centralized controller(Kathiravelu, 2016). It provides a large scalability and unified management for enterprise DCN requiring a differentiated QoS which makes it resolves incapability of the traditional DCN.

2.8 Research Gap

From the above related work, it is witnessed that fast evolution of Internet brings great challenges to network. This can be overcome through network traffic analysis as it is essential for businesses to identify and manage network applications and services depending on their purpose particularly to improve for network security, QoS enforcement, and trend analysis reason. Additionally has leveraged protocol to infer network latency and packet loss in small scale DCN and architecture change to SDN based large scale DCN which does not involve applications and network service identification prior to performance improvement(Shirmarz & Ghaffari, 2020) (Kathiravelu, 2016)(Keti & Askar, 2015)(Di. A. Popescu & Moore, 2018).

So that in this study, the researcher has only focused fill the above research gap by taking on NMI DCN performance improvement as a case study by proposing new platform and comparing both SDN and traditional topology on simulated environment

CHAPTER THREE: RESEARCH DESIGN AND METHODOLOGY

3.1 Case Description

In this study, business understanding is undertaken over the NMI DCN using methodologies and described in chapter one. An interview was conducted with the agency's ICT experts about the current status of the network performance problems and tried to access which Internet services are more widely used in the data center. The DCN infrastructure and the 'architecture is designed after physically visiting the environment as depicted on figure 3.1 below.

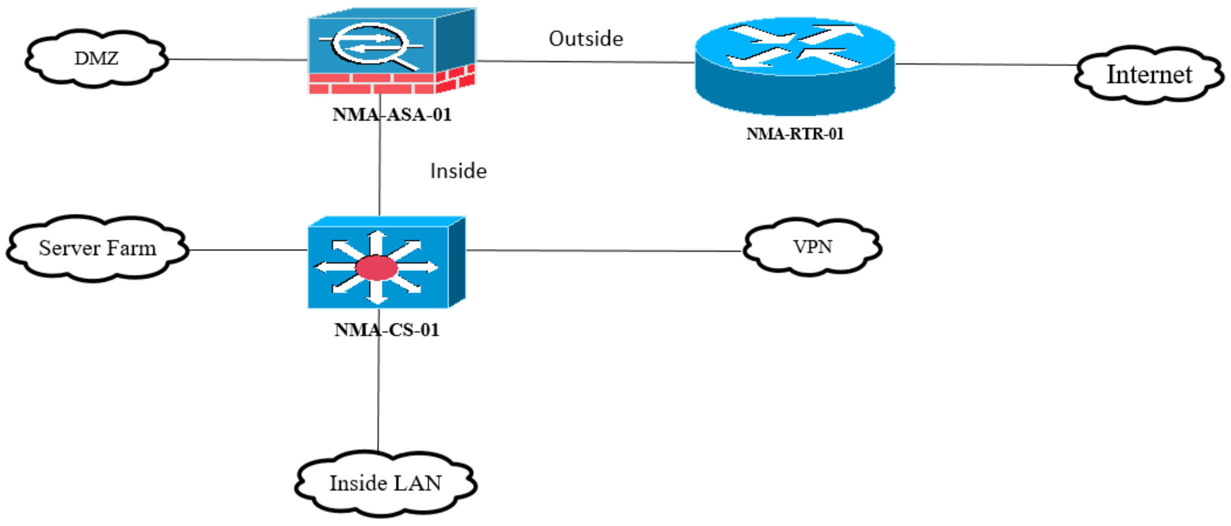


Figure 3. 1: Existing NMI DCN High Level Design

The network connectivity is composed of proprietary network devices such as firewall, routers and switches which were predominantly CISCO. 100Mbs Internet line is subscribed through fiber link and 40Mbps for Virtual Private Network (VPN) for exchange of data with the remaining thirteen branches throughout the country through our solely Internet Service Provider (ISP), Ethio-Telecom.

3.2 Methodology Flowchart

The main objective of this study is to improve network service performance, which is capable of determining the QoS metrics such as throughput, delay and loss based on the network services collected and findings of experimental analysis and the new proposed DCN architecture has also implemented as depicted in Fig. 3.2 below

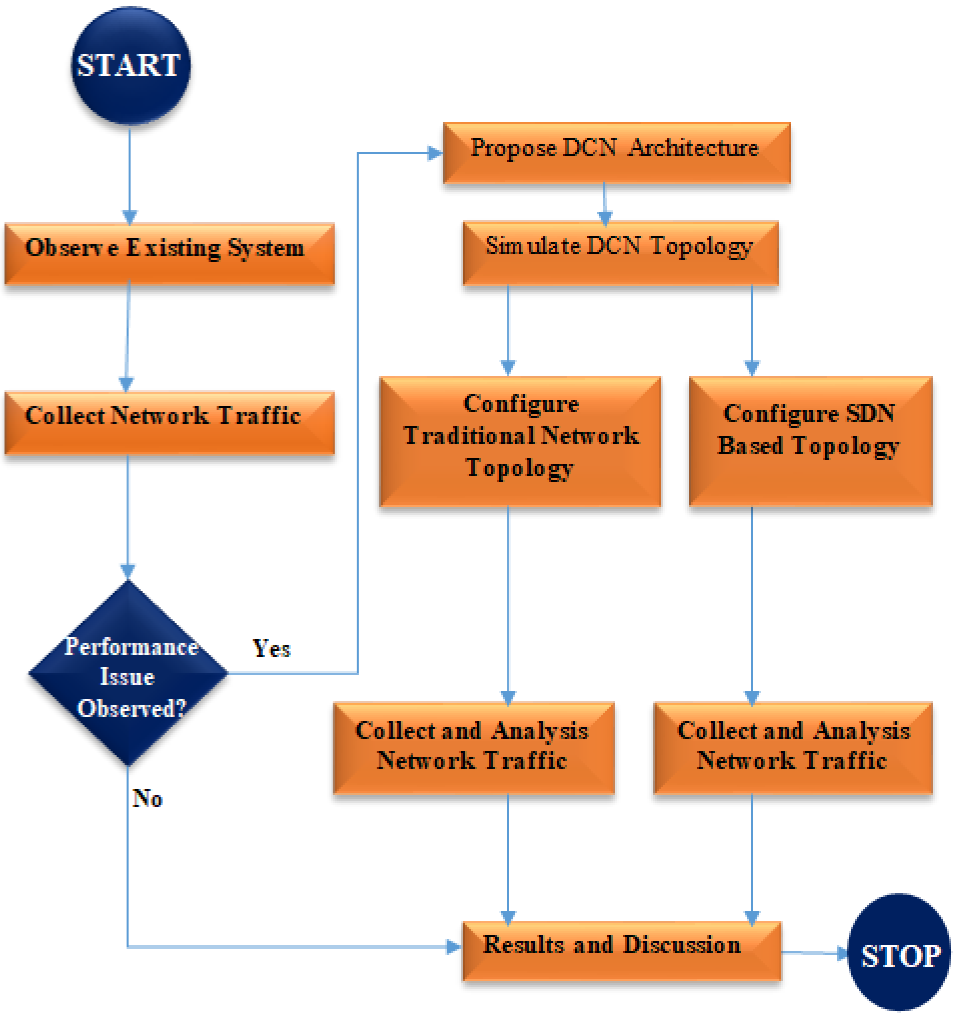

START
Observe Existing System
Collect Network Traffic
Performance Issue Observed?
Yes
No
Propose DCN Architecture
Simulate DCN Topology
Configure Traditional Network Topology
Configure SDN Based Topology
Collect and Analysis Network Traffic
Collect and Analysis Network Traffic
Results and Discussion
STOP

Figure 3. 2: Methodology flowchart for Improving DCN Performance Using SDN

3.3 Offline Data Collection from Existing Network

Two network protocols are used for collecting IP traffic information monitoring network traffic to analyze flow data. These are: NetFlow is a network protocol developed by Cisco, that tracks the number of packets sent, bytes sent, packet sizes and data sent from a flow exporter to a flow collector. Services and applications that serve as collectors are designed to receive data sent from exporters and provide data visualization and exploration to mirror all traffic.

Generally, NetFlow is used for network traffic measurement and designed to be embedded in any network devices. It provides continuous statistics on any protocol from Layer two to layer 7. Accordingly, all network traffic throughout can be accurately characterized and monitored. Thus, it used in congestion control, troubleshooting, security surveillance, network planning etc.

The researcher has used Solarwinds Network Performance Monitoring (NPM) and NetFlow Traffic Analyzer (NTA) trail version and Netflow v9 tools as data collection method from existing system. Analysis has performed to understand the existing network services performances.

3.4 Data Center Network Environment Setup and Tools Used

To achieve the objectives of this study, we do not need to buy hardware setup experimentation because of resource limitation. Instead, we simulate the experimental environment using two VMs; Mininet and ODL on the laptop running on Windows 10. Virtual box hypervisor is installed to run VMs. The HP laptop has Intel processor Core i7 generation 10 with 4 logical processors and runs at 2.9 GHZ, 16.0GB RAM specification. Then sample network services are collected from the simulated environment and analyzed using iperf tool. Finally, comparisons of traditional and SDN based DCN services are briefly discussed as shown in Fig. 3.3 below.

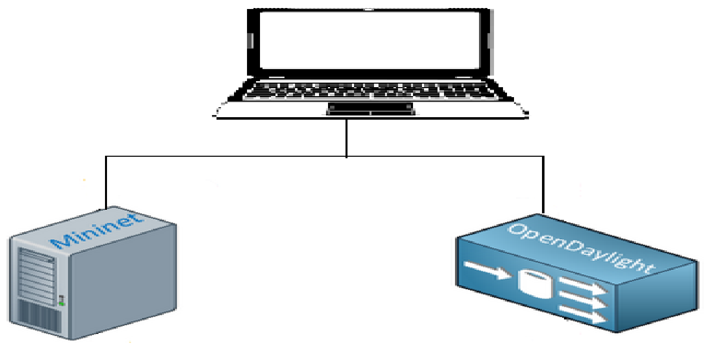


Figure 3. 3: Simulation Environment Setup

3.4.1 Overview of OpenDaylight Controller

OpenDayLight (ODL) is an open network platform which is a result of Open Source project led by the Linux Foundation. This is aimed to reduce development costs and eliminates propriety limitations for organizations not to commit to specific manufacturers that may block their development in the future. It has the following main components(Xiaohua & Canhui, 2020)(Aziz et al., 2017):

- Maven: to manage plugins and dependencies,
- Java: to develop applications and features,
- Open Service Gateway Interface (OSGi): to allows dynamically load bundles, packages and bind modules together for exchanging information.

- Karaf: is a container built on top of OSGi, used for packaging and installing applications.
- YANG: to model the application functionality and produce APIs.

3.4.2 Overview of Network Topology Using Miniedit

Since manually writing a Mininet custom topology using command-line interface (CLI) utility takes a lot longer, the Miniedit a GUI developed in python that allows to create custom network topologies. It is used as simple network editor and extends the use of Mininet to build, run and demonstrate network simulation. It displays a simple interface with a toolbar filled with common network elements that are needed for a network implementation. These are routers, switches, links, hosts and other devices to dragged and droped on the design area (Sharma* & Tyagi, 2019)(Keti & Askar, 2015). The main components in Miniedit are depicted as Fig 3.4 below.

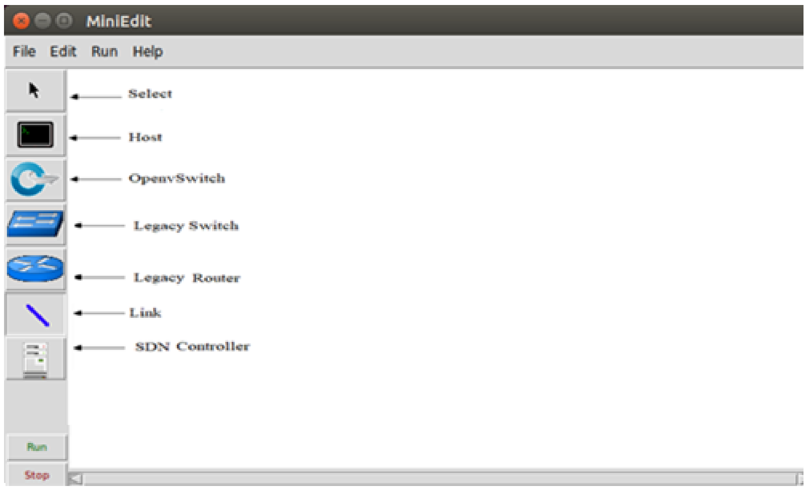


Figure 3. 4: MiniEdit GUI

The main buttons in this experimental setup are:

- Select: to select add or remove the devices to or from the topology.
- Host: to add new host to the topology
- Legacy switch: to add of a new legacy switch to the topology
- Legacy router: to add a new legacy router to the topology

- Link: to connect devices in the topology
- SDN Controller: to manage flows for network management and application performance
- Run: to start the emulation designing and configuring the topology
- Stop: stops the emulation.

3.4.3 Overview of Network Monitoring Tools

Network performance monitoring is an important concept in network management as it helps network operators to determine the behavior of a DCN and the status of its network components. It helps them to monitor QoS, and anomaly detection for quick decision making. Organizations need the performance of their network such as reliability and security issues using network monitoring tools to make a decision (Di. A. Popescu & Moore, 2018).

3.4.4 Overview Solarwinds Monitoring Tools

SolarWinds is a company that builds IT monitoring and management tools for System administrators and network engineers. It brings full visibility to all DCN devices, which go from reactive to proactive network management, configuration management and traffic intelligence to performance monitoring and topology mapping, readily see, understand, and resolve issues. Among those tools NTA allows to capture data from continuous streams of network traffic, and convert those raw numbers into easy-to-interpret charts and tables that quantify exactly how the corporate network is being used, by whom, and for what purpose. Network Performance Monitor (NPM) is a powerful and affordable network monitoring software enabling you to quickly detect, diagnose, and resolve network performance problems and outages These products are widely used in both the public and private sectors(Podolanko et al., 2014).

3.4.5 Overview of Iperf

An Iperf is a utility application used to create, allow to tune and measure TCP and UDP performance parameters and various characteristics such packet loss and delay is examined. It prints the periodic, intermediate jitter, and loss reports at specified intervals. It is being used by many researchers use Iperf network testing tool for SDN base DCN(Alraawi & Adam, 2021).

we downloaded mininet 2.2.3 from mininet.org then installed to build virtual network and The ODL Karaf-0.8.4 release software download *https://www.opendaylight.org* website and installed the controller on the Ubuntu 20.04.03 with the following IP information. The first VM is mininet with IP192.168.46.46/24 and second VM is ODL with IP 192.168.56.56/24.

Table 3: 1- Simulation Environment Setup

VM	**anagement IP**	**Purpose**
Mininet	192.168.46.46	Mininet Simulator
ODL	192.168.56.56	OpenDayLight Controller

The Python script is used in Mininet to create custom topologies containing ODL controller OpenFlow vSwitches and linux hosts. We Installed the Java Runtime Environment (JRE) and set JAVA_HOME using the following commands in the Ubuntu terminal.

```
sudo apt-get update && sudo apt-get install default-jre-headless
```

After download we unzip and start it. Then, we start Apache Karaf application that contains different modules or features need to be installed, started, stopped, updated, and uninstalled without requiring a reboot. By default, ODL has limited features. So that we need to install necessary modules to get a GUI using the following command after Karaf boot up.

Feature: install odl-restconf odl-l2switch-switch odl-mdsal –apidocs odl –dlux –core odl –dluxapps –nodes odl –dluxapps –topology odl –dluxapps –yangapps –yangui odl –dluxapps –yangvisualizer odl –dluxaps-yangman.

The ‘dluxapps’ is used to access the controller through web to configure network from the through Hypertext Transfer Protocol (HTTP). The ‘mdsal-apidocs’ allows to access the Yang API and the ‘l2switch-switch’ provides similar functionality to an Ethernet switch. Additionally, RESTCONF describes mapping specification of YANG to a RESTful interface.

3.4.6 Controller Access and Management

After complete installation of ODL in the VM, we can access it through the web application using any web browser by typing URL: *http:192.168.56.56: 8181/index.html* as appear in Fig.3.5 below.

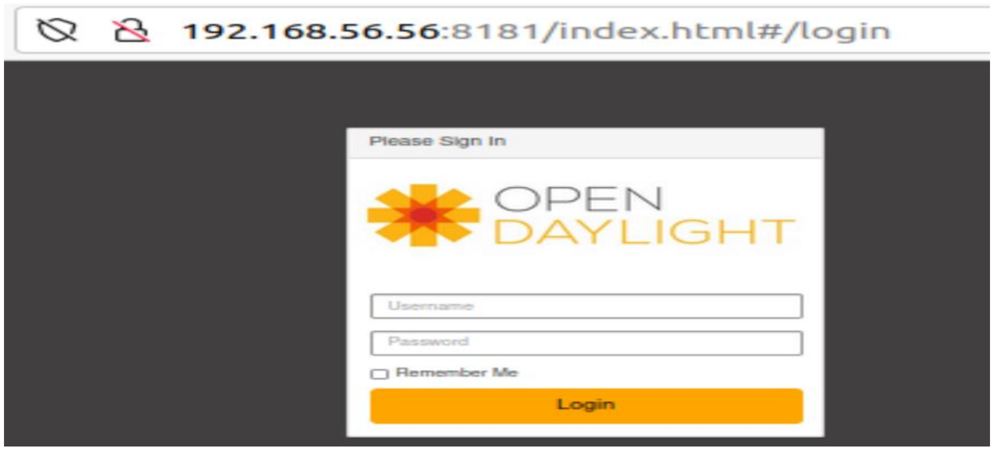


Figure 3. 5: ODL GUI Page

The default login credential ‘admin’ is used both user and password to display the following main working environment.

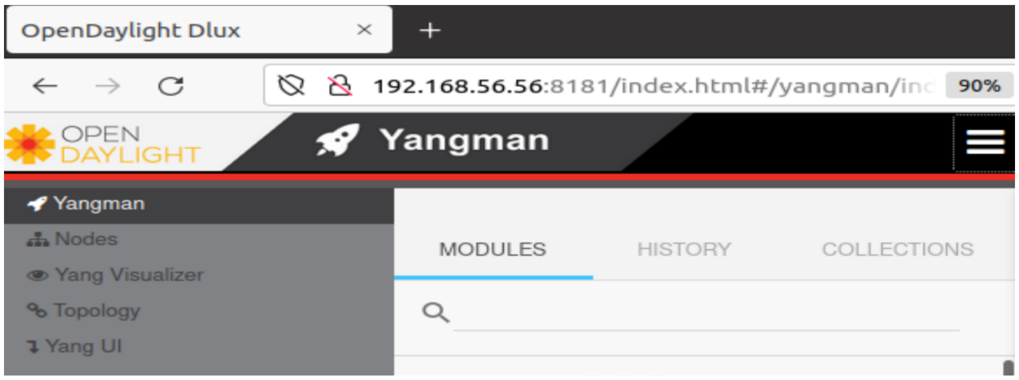


Figure 3. 6: ODL Main Working Environment

The 'Nodes' is used to track information of every network device connected to the controller. Where 'Topology' section is used to overview all network connections. The Yangman API is data modeling structure for legacy switches. Yang UI is a graphic client to configure and sent requests to the ODL local database (Medved et al., 2014).

3.5 Proposed DCN Design and Implementation Scenarios

The design contains a combination of qualitative, explorative, descriptive and quantitative requisites to gain a richer understanding of studies and better findings. The purpose of the quantitative nature is to get a deeper understanding of the DCN service and application performance improvement to be showed through extensive simulation work on emulation software, Mininet. Accordingly, study two DCN, two DCN scenarios, traditional legacy-based and SDN-based, with a similar pattern in terms of number of network devices and configurations, were implemented to compare and observe traffic loss, delay, and transmission throughput. Each network has designed as a fat tree topology.

3.5.1 Traditional DCN Topology Design

To compare a traditional and SDN-based DCN architecture, two almost similar topologies are considered. In traditional topology is designed without a controller which has seven switches in it. The first three switches are used as the distribution layer where the last four switches are used as access switches. By connecting three host machines, each one where any of the SDN-oriented protocols are implemented in this design.

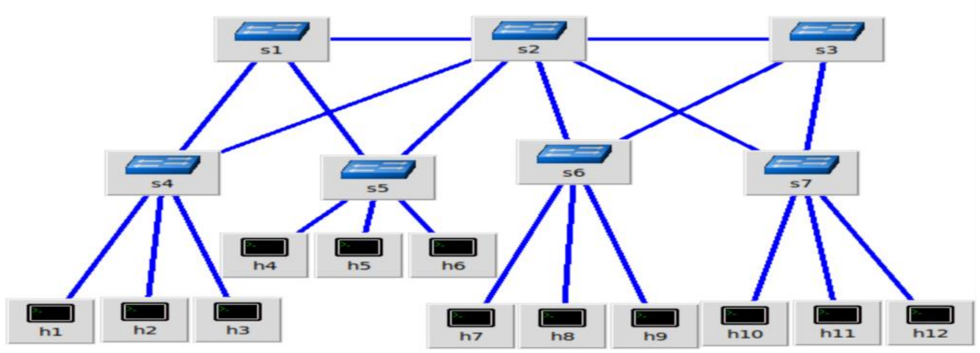


Figure 3. 7: Proposed Legacy DCN Design in Miniedit

Since there is no centralized control of the whole network, each switch has to be configured manually. This is one of the challenging issues in traditional networks. This is because the software that runs on these kinds of network switches is most likely proprietary devices such as Cisco Nexus switches or Juniper, which lack open source practicable alternatives.

3.5.2 Improved SDN Based DCN Design

n the second test, an SDN-based DCN was created in Mininet using python script. This design consists of seven OVS, twelve hosts and one ODL controller, as shown in Fig. 3.8. The controller is responsible for configuring the switches to forward the traffic as a layer two switch. The controller allows forwarding of traffic between all of the switches connected to it. Additionally, the controller is used to decouple the control plane from all the switches and acts as the network orchestrator and core of the network. It will be implemented by connecting it to all the switches in the design, which means three switches are used as distribution layer of the network and four switches will act as access points, on which three host machines are connected.

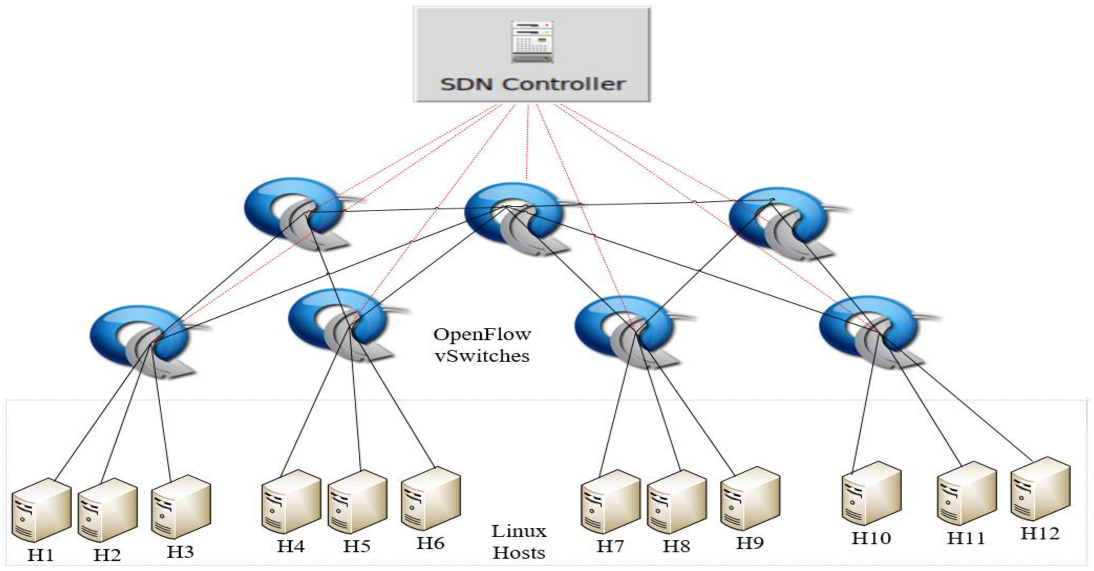


Figure 3. 8: Proposed SDN Based DCN HLD

In SDN based topology, ODL controller connects directly to all switches in the DCN network. All switches use OVS switching and the OpenFlow protocol for routing. The Linux operating system-based hosts are directly connected to the access switch. In order to perform the simulation, the design should be translated into a Python script that will be executed in Mininet and its script file should be modified at the controller with the remote controller IP 192.168.56.56 with TCP port 6633.

3.5.3 Connecting Mininet with ODL Controller

Once the design is completed, we start to simulate the proposed network for testing by starting both the controller and Mininet services and linking them to test the SDN network. We use the ODL controller for monitoring and management where the Mininet to create the network topology. The real-time traffic is collected by using ‘tcpdump’ utility in the form of ‘pcap’ files and the flow features are generated using the ‘iperf’.

After the creation of the network devices, the controller should detect the directly connected OVS switches. Initially, the controller does not recognize those elements connected to it because of no activity in the network. The elements will be easily shown by executing ping command which start traffic flows in the Mininet and the ODL would have a map of network elements.

CHAPTER FOUR: RESULTS AND DISCUSSIONS

4.1 Network Traffic Analysis

Internet traffic flow is collected from existing NMI edge Internet router from incoming and outgoing interfaces which are previously configured to send traffic to Solarwinds NPM and NTA monitoring tool. The network traffic services such as web browsing email services, social media and VoIP service coming into and to outgoing to the DCN are captured before perimeter firewall. These network traffic statistics has collected using as offline dataset for one month in the time interval from July 1, 2021 up to July 31, 2021 and the statistics

4.2 Existing Network Traffic Performance Analysis and Interpretation

As new technologies and applications are being emerged, the NMI System may be impacted and therefore the network should be monitored to ensure the better performance of the network. When network QoS parameters including delay, packet loss, and throughput more significant. When performance problems arise overall network services and applications are affected to compare those network access operations previously accustomed. From the infrastructure perspective, it is important to characterize the overall traffic patterns on the network using key metrics such as packet loss, delay, bandwidth and availability to analyze when evaluating network performance. This will provide insight into which flows are most congested over time and could become potential problem areas.

In order to analyze the NMI existing DCN link, Internet traffic flow is collected from existing edge Internet router from incoming and outgoing interfaces which are previously configured to send traffic to Solarwinds NPM and NTA monitoring tool. The network user applications and services such as web browsing email services, social media and VoIP services are coming into and to outgoing to the DCN captured before perimeter firewall. These network traffic statistics has collected using as offline dataset for one month in the time interval from July 1, 2021 up to July 31, 2021 and the statistics, charts and tables are presented in the following section. The analysis is done for network service performance metrics such as packet latency (delay) and throughput at the selected node. The 30 days of network traffic time in to four weeks. Week1 is from July 1 to July 7, 2021, Week2 is from July 8 to 15, 2021, Week3 is from July 16- 23, 2021 and Week4 is from June 24 to 31, 2021.

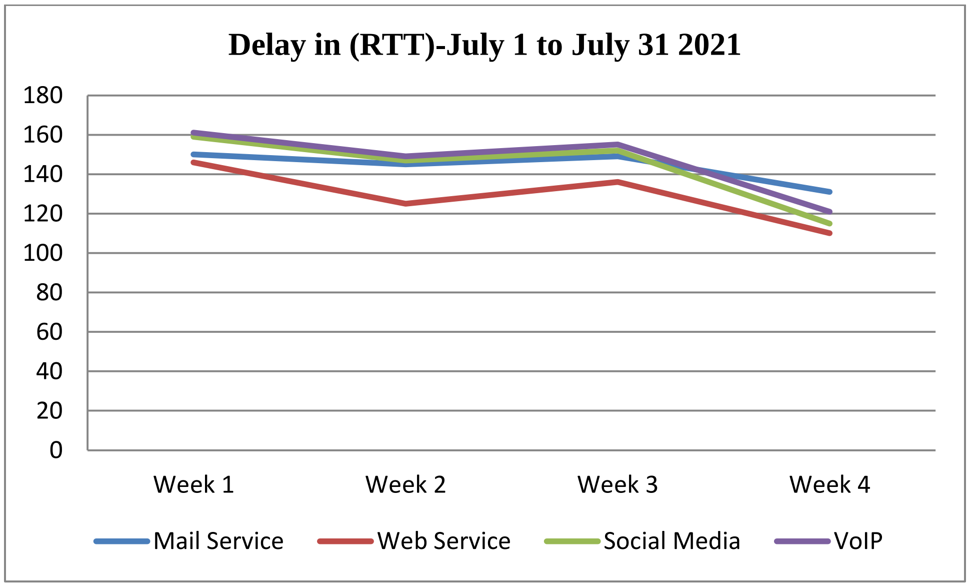


Figure 4. 1: Delay in (RTT)-July 1 to July 31 2021

Average response time for Mail service is 144ms av/erage response time for Web service is 129ms for social media application is 143ms and for VoIP applications is 147ms. The cumulative response time for these services and applications is 141ms, which needs a relative improvement.

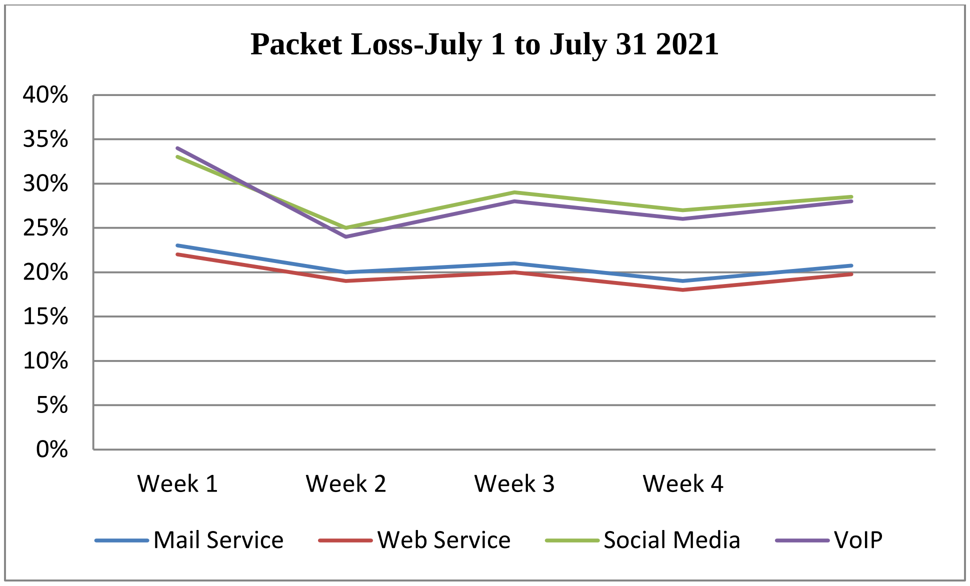


Figure 4. 2: Packet Loss-July 1 to July 31 2021

Packet loss occurs when data is transmitting over computer network, one or more packets may fail to reach their destinations. In other words, Packet loss is the number of packets that fail to reach the destination. Packet loss is the ratio of the number of packets lost to the total number of packets transmitted. In order to measure and evaluate customer or user satisfactions interacting with Internet services the available network bandwidth must be managed efficiently to ensure a stable network throughput to be delivered to the end use.

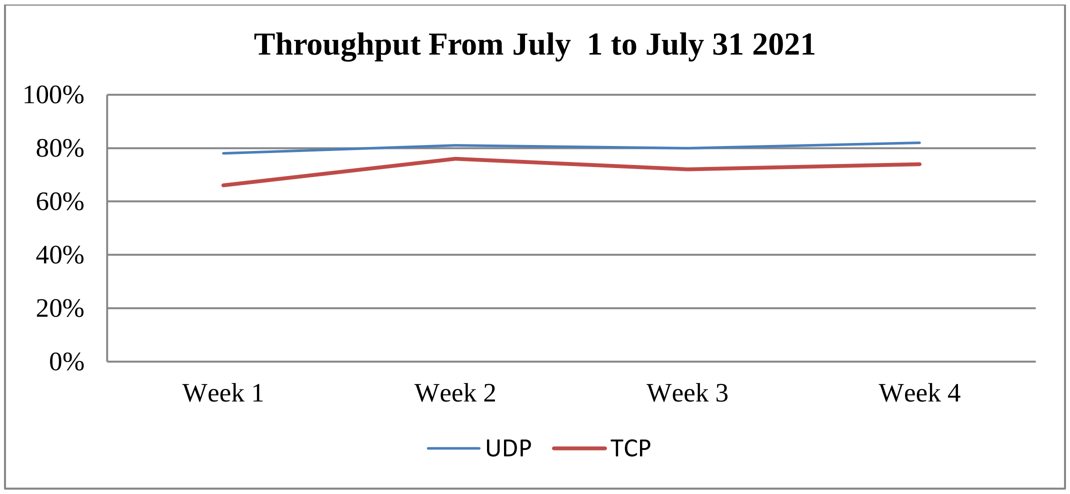


Figure 4. 3: Throughput from July 1 to July 31 2021

4.3 Throughput Comparison Graph in Mininet

Once the flows were added, the TCP test between hosts was successful using default UDP window size of 85.3 KBytes and the host using the default window size of 391 Kbyte. It is possiblee to transfer 896 MBytes at a rate of 751 Mbits/sec from within linux hosts. With UDP 1.32 GBytes was transferred at a rate of 1.13 Gbits/sec as seen in the TCP Iperf results to ensure reliability of the communication between the switch and the ODL controller.

Traffic generation and throughput measurement was conducted by means of Iperf tool that can generate TCP and UDP packets in order to measure the throughput of a network. Mininet provides this tool to evaluate the amount of traffic bandwidth transferred between network elements in a simulated network. The throughput test will consist of different iperf experiments between two hosts in both network for both, the tests are h1 between h3, h1 between h6 and h1 between h12 for both networks. The traffic flow made between the switch and the controller was captured using Wireshark. At the beginning, the OpenFlow Channel messages between the switch and the controller are observed. The ODL requested the identity and basic capabilities of the switch.

Accordingly, performance of both traditional and SDN based DCN is analyzed the throughput is an important parameter which gives approximately overall performance of a network. We have observed that the throughput over time by generating traffic requests per seconds. As the time increases throughput will also increase in SDN than the traditional network.

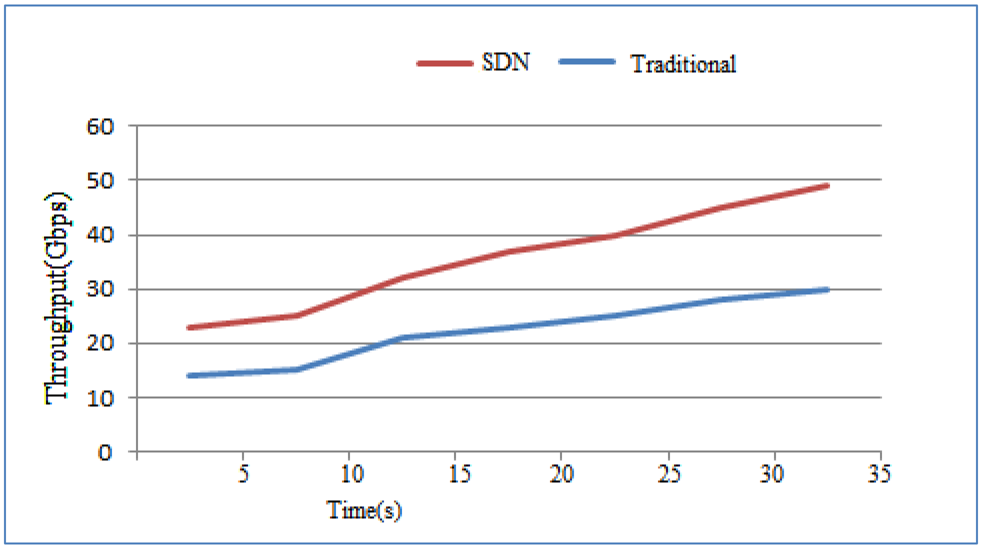


Figure 4. 4: SDN Based and Traditional DCN Throughput

From the above chart SDN based DCN performance has almost two times better in terms of TCP and UDP throughput as a number packet increases compared to the traditional network architecture. Such that the SDN supports better allocation of bandwidth makes DCN. From this we can generalize that using SDN in data center can improve network services and application performance as expected. The results that are shown in the previous figure proves that in all cases the SDN network performs two times better in terms of throughput as a number packet increases compared to the traditional network architecture. Such that the SDN supports better allocation of bandwidth makes DCN.

4.4 Delay Comparison of Both Legacy and SDN Based DCN

A time delay is measured from the source to the destination or round-trip delay time which is plus the latency from the destination back to the source which be measured using ping service without any packet processing. In this study, the hosts (h1, h3, h6 and h12) have been selected from both networks to achieve the delay test. It is observed the SDN network has lower delay than traditional DCN. As the ping increments, delay is decrement for both networks. But the traditional DCN get more suffer than the SDN based DCN architecture because of SDN network's route frames are stored in the memory where as in a traditional network each switch takes time to process MAC learning and update an internal database.

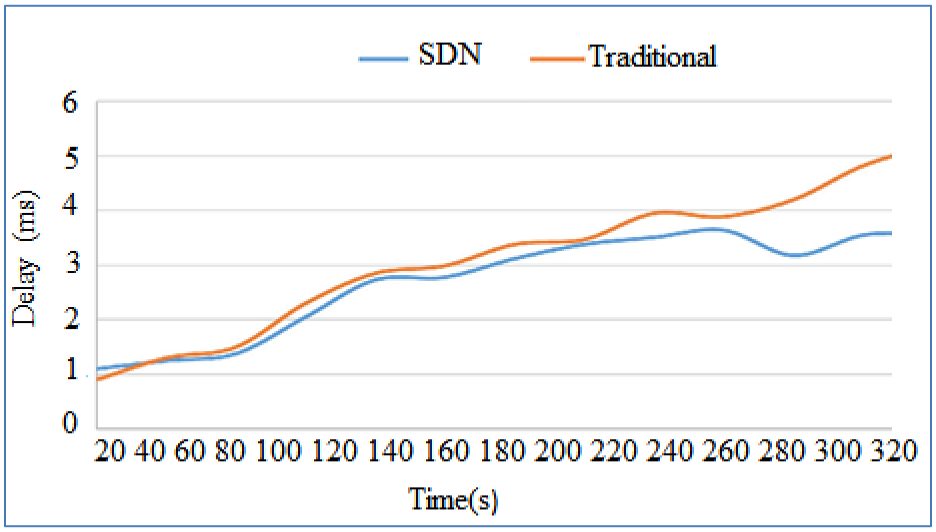


Figure 4. 5: Delay Comparison for both Traditional and SDN Based DCN

From the above line chart, we can observe that Traditional DCN has Cumulative Delay (Td) randomly taken at time 80,160,240,320 is 3.35ms as below calculated:

$$\mathrm{TD} = \frac{(1.4+3+4+5)\ \mathrm{ms}}{4} \qquad (1)$$

Where SDN based DCN Cumulative Delay (Md) has a cumulative delay of 2.73ms.

$$\mathrm{Md} = \frac{(1.2+2.8+3.4+3.5)\ \mathrm{ms}}{4} \qquad (2)$$

To get the time difference between two data center we subtract SDN based Md from Traditional DCN cumulative delay Td which is 2.73 from 3.35ms we get 0.62ms. From this result we can conclude that SDN based DCN has improvement of around 22.71% become faster than traditional DCN.

Hence from this result we can conclude that using existing Internet subscription in use or in production environment in Fig. 4.1: Delay in (RTT)-July 1 to July 31 2021 above average response time for Mail, Web, social media and VoIP services and application is 144ms, 129ms, 143ms and 147ms respectively. The cumulative response time for these services and applications is 141ms, which needs a relative improvement. Accordingly using the below formula:

$$\mathrm{Rtn} = \mathrm{Rto} - \frac{Rto\ \text{Existing Average Response Time in ms}}{100\%} x22.71\% \tag{3}$$

Where *Rto* is existing average response time and *Rtn* is new average response time. Such that using SDN based DCN network, we can achieve average response time for Mail service 111.3ms, Web service is 99.7ms for Social media application 110.5ms and for VoIP applications 113.62ms. The cumulative response time for SDN based DCN if Internet services and applications is 108.78ms, which has 32.22ms relative improvement for all network traffic being transmitted in DCN.

4.5 Summary of Finding

In order to compare and demonstrate the efficiency in implementation network design and traffics generation is applied and measured in both traditional and SDN based DCN scenarios. Different SLA parameter such as packet loss, delay, and throughput has observed in existing on both existing and the two simulated DCN environment. From the test result we can conclude that using the proposed new architecture, Software Defined Networking (SDN) with existing Internet subscription, it is possible to achieve average response time for Mail service 111.3ms, Web service is 99.7ms for Social media application 110.5ms and for VoIP applications 113.62ms.

Such that using SDN based DCN the cumulative response time for Internet services and applications is for SDN based DCN if Internet services and applications is 108.78ms, which has 32.22ms relative improvement for all network traffic being transmitted in DCN.As a result, deploying SDN provides the best performance for demanding services in DCN in a fast and cost-effective manner, with more features than the traditional non-controller-based network. These results prove that the SDN is the future networking architecture to implement in large datacenters as it introduces new improvements to the current business needs.

CHAPTER FIVE: CONCLUSION AND RECOMMENDATION

5.1 CONCLUSION

Newly emerged network services and applications made a huge demand for network performance and a large amount of traffic is being generated from this very fast rate growing of network traffic has made traditional network technologies to reach their limit of capabilities. These can produce network congestion in this kind DCN node because of carrying more data than its capacity and experience QoS degradation in connectivity and response speed throughout a given network and reduces the performance of network applications and services. This makes network engineers and operators to manually classify and provide QoS for each application which is difficult to the analysis and diagnosis to maintain consistent SLA or OLA.

So that DCN traffic identification and working on its improvement is so critical to be stay globally connected which is the key technology for almost all organizations to quickly find, share information, communicate with people around the world, and manage their finances. So that data center owners need better service delivery mechanism to accommodate an increasing network applications and services demands.

According this study provides a significant tested takeaway on how to identify network service and applications in a typical DCN, NMI for better network management and improve its performance. This makes to identify useful patterns and input information to propose and compare DCN architectural change to improve network service performance to have better QoS. As observed from the result, traditional DCN architectures are not well-suitable, scalable and flexible where the newly proposed SDN based architecture has better network performance to handle network traffics.

5.2 RECOMMENDATION

In this study some issues addressed to improve the network performance. However; there are still challenging and critical issues that are left for researchers to handle and manage massive network traffic coming to NMI DCN. As this work is focused traffic classification on existing network, comparing both existing conventional and proposed SDN network architecture and designed topology. There are a series of tasks that would have been interesting to perform which are beyond the scope of this thesis. When users from branches or the Internet attempt to access a list of backend servers such as web servers, application servers, and database servers, and the traffic should be load balanced to improve DCN services and applications by distributing clients' access into the servers using the SDN system as a controller, with security as an important direction for future research.

REFERENCES

Alraawi, A. A. M., & Adam, S. A. N. (2021). Performance Evaluation of Controller Based SDN Network over Non-controller Based Network in Data Center Network. *Proceedings of: 2020 International Conference on Computer, Control, Electrical, and Electronics Engineering, ICCCEEE 2020*, 7–10. https://doi.org/10.1109/ICCCEEE49695.2021.9429642

Alrokayan, M., Vahid Dastjerdi, A., & Buyya, R. (2015). SLA-aware provisioning and scheduling of cloud resources for big data analytics. *2014 IEEE International Conference on Cloud Computing in Emerging Markets, CCEM 2014*. https://doi.org/10.1109/CCEM.2014.7015497

Aziz, M., Fazely, H. A., Landi, G., Gallico, D., Christodoulopoulos, K., & Wieder, P. (2017). SDN-enabled application-aware networking for data center networks. *2016 IEEE International Conference on Electronics, Circuits and Systems, ICECS 2016*, 372–375. https://doi.org/10.1109/ICECS.2016.7841210

Balakiruthiga, B., & Deepalakshmi, P. (2019). A Simple Congestion Avoidance mechanism for OpenDaylight (ODL) - Multipath TCP (MPTCP) Network structure in Software Defined Data Center (SDDC). *2019 5th International Conference on Advanced Computing and Communication Systems, ICACCS 2019*, *1*, 886–893. https://doi.org/10.1109/ICACCS.2019.8728522

Balodis, R., Opmane, I., Mohammad, N., Benltayef, A., Ahmed, A., Uddin, M., Abubakar, A., Shah, A., Adeleke, I., Bari, M. F., Boutaba, R., Esteves, R., Granville, L. Z., Podlesny, M., Rabbani, M. G., Zhang, Q., Zhani, M. F., Birke, R., Podzimek, A., … Smirni, E. (2013). IFIP AICT 387 - History of Data Centre Development. *Journal of Power Technologies*, *94*(2), 4–9.

Bitar, N., Gringeri, S., & Tiejun, J. X. (2013). Technologies and protocols for data center and cloud networking. *IEEE Communications Magazine*, *51*(9), 24–31. https://doi.org/10.1109/MCOM.2013.6588646

Chang, H. T., & Wang, S. Y. (2015). Using SDN technology to mitigate congestion in the OpenStack data center network. *IEEE International Conference on Communications*, *2015-Septe*, 401–406. https://doi.org/10.1109/ICC.2015.7248354

De Oliveira, R. L. S., Schweitzer, C. M., Shinoda, A. A., & Prete, L. R. (2014). Using Mininet for emulation and prototyping Software-Defined Networks. *2014 IEEE Colombian Conference on Communications and Computing, COLCOM 2014 - Conference Proceedings*. https://doi.org/10.1109/ColComCon.2014.6860404

Govindarajan, K., Meng, K. C., Ong, H., Tat, W. M., Sivanand, S., & Leong, L. S. (2014). Realizing the Quality of Service (QoS) in Software-Defined Networking (SDN) based Cloud infrastructure. *2014 2nd International Conference on Information and Communication Technology, ICoICT 2014*, 505–510. https://doi.org/10.1109/ICoICT.2014.6914113

Hafeez, T., Ahmed, N., Ahmed, B., & Malik, A. W. (2017). Detection and Mitigation of Congestion in SDN Enabled Data Center Networks: A Survey. *IEEE Access*, *6*(XX), 1730–1740. https://doi.org/10.1109/ACCESS.2017.2780122

Han, Y., Seo, S. S., Li, J., Hyun, J., Yoo, J. H., & Hong, J. W. K. (2014). Software defined networking-based traffic engineering for data center networks. *APNOMS 2014 - 16th Asia-Pacific Network Operations and Management Symposium*. https://doi.org/10.1109/APNOMS.2014.6996601

Hao, J., Shi, Y., Sun, H., Sheng, M., & Li, J. (2019). Rerouting based congestion control in data center networks. *2019 IEEE International Conference on Communications Workshops, ICC Workshops 2019 - Proceedings*. https://doi.org/10.1109/ICCW.2019.8757147

He, Y., Sun, Y., Yang, Y., Li, H., & Wu, X. (2016). End-To-End Latency Bottleneck Analysis for Multi-class Traffic in Data Center Networks. *Proceedings - 2015 6th International Conference on Intelligent Systems Design and Engineering Applications, ISDEA 2015*, 366–371. https://doi.org/10.1109/ISDEA.2015.98

Hu, F., Hao, Q., & Bao, K. (2014). A survey on software-defined network and OpenFlow: From concept to implementation. *IEEE Communications Surveys and Tutorials*, *16*(4), 2181–2206. https://doi.org/10.1109/COMST.2014.2326417

Huang, B., & Dong, S. (2020). An Enhanced Scheduling Framework for Elephant Flows in SDN-Based Data Center Networks. *Proceedings - IEEE Symposium on Computers and Communications*, *2020-July*. https://doi.org/10.1109/ISCC50000.2020.9219688

Ignaciuk, P., & Bartoszewicz, A. (2013). Congestion control in data transmission networks: Historical perspective. In *Communications and Control Engineering* (Issue 9781447141464). https://doi.org/10.1007/978-1-4471-4147-1_2

Kathiravelu, P. (2016). Software-defined networking-based enhancements to data quality and QoS in multi-tenanted data center clouds. *Proceedings - 2016 IEEE International Conference on Cloud Engineering Workshops, IC2EW 2016*, 201–203. https://doi.org/10.1109/IC2EW.2016.19

Keti, F., & Askar, S. (2015). Emulation of Software Defined Networks Using Mininet in Different Simulation Environments. *Proceedings - International Conference on Intelligent Systems, Modelling and Simulation, ISMS*, *2015-Octob*, 205–210. https://doi.org/10.1109/ISMS.2015.46

Leivadeas, A., Falkner, M., Lambadaris, I., & Kesidis, G. (2016). Dynamic traffic steering of multi-tenant virtualized network functions in SDN enabled data centers. *IEEE International Workshop on Computer Aided Modeling and Design of Communication Links and Networks, CAMAD*, 65–70. https://doi.org/10.1109/CAMAD.2016.7790332

Li, D., Shang, Y., & Chen, C. (2014). Software defined green data center network with exclusive routing. *Proceedings - IEEE INFOCOM*, *61170291*, 1743–1751. https://doi.org/10.1109/INFOCOM.2014.6848112

Medved, J., Varga, R., Tkacik, A., & Gray, K. (2014). OpenDaylight: Towards a model-driven SDN controller architecture. *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014, WoWMoM 2014*. https://doi.org/10.1109/WoWMoM.2014.6918985

Mujib, M., & Sari, R. F. (2020). Performance Evaluation of Data Center Network with Network Micro-segmentation. *ICITEE 2020 - Proceedings of the 12th International Conference on Information Technology and Electrical Engineering*, 27–32. https://doi.org/10.1109/ICITEE49829.2020.9271749

Nguyen, T. A., Min, D., Choi, E., & Tran, T. D. (2019). Reliability and availability evaluation for cloud data center networks using hierarchical models. *IEEE Access*, *7*(c), 9273–9313. https://doi.org/10.1109/ACCESS.2019.2891282

Open Networking Foundation. (2015). OpenFlow Switch Specification (Version 1.5.1).

Current, *0*, 1–36. https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.5.1.pdf

Podolanko, J., Datta, S., & Das, S. K. (2014). Performance analysis of real-time traffic over 802.11n Wireless Local Area Networks: An experimental study. *IWCMC 2014 - 10th International Wireless Communications and Mobile Computing Conference*, 453–457. https://doi.org/10.1109/IWCMC.2014.6906399

Ponciano, J. P., & Anani, N. (2014). Load balancing in modern network infrastructures - A simulation model. *2014 9th International Symposium on Communication Systems, Networks and Digital Signal Processing, CSNDSP 2014*, 841–846. https://doi.org/10.1109/CSNDSP.2014.6923944

Popescu, D. A., & Moore, A. W. (2021). Measuring Network Conditions in Data Centers Using the Precision Time Protocol. *IEEE Transactions on Network and Service Management*, *18*(3), 3753–3770. https://doi.org/10.1109/TNSM.2021.3081536

Popescu, Di. A., & Moore, A. W. (2018). A First Look at Data Center Network Condition Through the Eyes of PTPmesh. *TMA 2018 - Proceedings of the 2nd Network Traffic Measurement and Analysis Conference*. https://doi.org/10.23919/TMA.2018.8506493

Pupatwibul, P., Banjar, A., AL Sabbagh, A., & Braun, R. (2015). A Comparative Review: Accurate OpenFlow Simulation Tools for Prototyping. *Journal of Networks*, *10*(5). https://doi.org/10.4304/jnw.10.5.322-327

Sharma*, P. K., & Tyagi, D. S. S. (2019). Improving Security through Software Defined Networking (SDN): AN SDN based Model. *International Journal of Recent Technology and Engineering (IJRTE)*, *8*(4), 295–300. https://doi.org/10.35940/ijrte.d6814.118419

Shekhawat, V. S., Gautam, A., & Thakrar, A. (2018). Datacenter Workload Classification and Characterization: An Empirical Approach. *2018 13th International Conference on Industrial and Information Systems, ICIIS 2018 - Proceedings*, 1–7. https://doi.org/10.1109/ICIINFS.2018.8721402

Sherwin, J. (n.d.). *Software-Defined Networking for Data Centre Network Management: A Survey*. 1–9.

shirmarz, A., & Ghaffari, A. (2020). An Autonomic Software Defined Network (SDN)

Architecture With Performance Improvement Considering. *Journal of Information Systems and Telecommunication (JIST)*, *8*(30), 121–129. https://doi.org/10.29252/jist.8.30.121

Shirmarz, A., & Ghaffari, A. (2020). Performance issues and solutions in SDN-based data center: a survey. In *Journal of Supercomputing* (Vol. 76, Issue 10). Springer US. https://doi.org/10.1007/s11227-020-03180-7

Sugeng, W., Istiyanto, J. E., Mustofa, K., & Ashari, A. (2015). The Impact of QoS Changes towards Network Performance. *International Jurnal of Computer Networks and Communications Security*, *3*(2), 48–53. http://www.ijcncs.org/published/volume3/issue2/p5_3-2.pdf

Thombre, S. (2018). Network Jitter Analysis with varying TCP for Internet Communications. *2018 3rd International Conference for Convergence in Technology, I2CT 2018*, 1–7. https://doi.org/10.1109/I2CT.2018.8529816

Valencic, D., & Mateljan, V. (2019). Implementation of NETCONF protocol. *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2019 - Proceedings*, 421–430. https://doi.org/10.23919/MIPRO.2019.8756925

Wang, T., Su, Z., Xia, Y., & Hamdi, M. (2014). Rethinking the data center networking: Architecture, network protocols, and resource sharing. *IEEE Access*, *2*, 1481–1496. https://doi.org/10.1109/ACCESS.2014.2383439

Xia, W., Zhao, P., Wen, Y., & Xie, H. (2017). A Survey on Data Center Networking (DCN): Infrastructure and Operations. *IEEE Communications Surveys and Tutorials*, *19*(1), 640–656. https://doi.org/10.1109/COMST.2016.2626784

Xiaohua, Y., & Canhui, H. (2020). Design and Implementation of OpenDayLight Manager Application. *Proceedings - 2020 13th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics, CISP-BMEI 2020*, 977–982. https://doi.org/10.1109/CISP-BMEI51763.2020.9263553

Xue, L., Chiu, C. H., Kumar, S., Kondikoppa, P., & Park, S. J. (2015). FaLL: A fair and low latency queuing scheme for data center networks. *2015 International Conference on Computing, Networking and Communications, ICNC 2015*, 771–777. https://doi.org/10.1109/ICCNC.2015.7069444

Yao, F., Wu, J., Venkataramani, G., & Subramaniam, S. (2014). A comparative analysis of data center network architectures. *2014 IEEE International Conference on Communications, ICC 2014*, 3106–3111. https://doi.org/10.1109/ICC.2014.6883798

Zakia, U., & Ben Yedder, H. (2017). Dynamic load balancing in SDN-based data center networks. *2017 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference, IEMCON 2017*, 242–247. https://doi.org/10.1109/IEMCON.2017.8117206

APPENDIXES

I. Traditional DCN Architecture Topology Python Script

In this architecture the network elements are configured by using the Python scripts and designed to be implemented on Mininet. In this custom network topology; network elements such as switches and hosts connected and configured to be tested. As a result, the controller no longer required. Such that the following will be typed in Mininet to begin the simulation.

sudo python 2.7 Traditional_DCN _Topology.py

```
#!usr/bin/python2.7
from subprocess import call
from mininet.net import Mininet
from mininet.node import Controller, RemoteController, OVSController
from mininet.node import CPULimitedHost, Host, Node
from mininet.node import OVSKernelSwitch, UserSwitch
from mininet.cli import CLI
from mininet.log import setLogLevel, info
from mininet.link import TCLink, Intf
def datacenterNetwork():
 net = Mininet(topo=None, build=False, ipBase='10.0.0.0/8')
 info('*** Adding switches\n')
 s1 = net.addSwitch('s1', cls=UserSwitch)
 s2 = net.addSwitch('s2', cls=UserSwitch)
 s3 = net.addSwitch('s3', cls=UserSwitch)
 s4 = net.addSwitch('s4', cls=UserSwitch)
 s5 = net.addSwitch('s5', cls=UserSwitch)
 s6 = net.addSwitch('s6', cls=UserSwitch)
 s7 = net.addSwitch('s7', cls=UserSwitch)
  info('*** Adding hosts\n')
 h1 = net.addHost('h1', cls=Host, ip='10.0.0.1', defaultRoute=None)
 h2 = net.addHost('h2', cls=Host, ip='10.0.0.2', defaultRoute=None)
 h3 = net.addHost('h3', cls=Host, ip='10.0.0.3', defaultRoute=None)
 h4 = net.addHost('h4', cls=Host, ip='10.0.0.4', defaultRoute=None)
 h5 = net.addHost('h5', cls=Host, ip='10.0.0.5', defaultRoute=None)
 h6 = net.addHost('h6', cls=Host, ip='10.0.0.6', defaultRoute=None)
 h7 = net.addHost('h7', cls=Host, ip='10.0.0.7', defaultRoute=None)
 h8 = net.addHost('h8', cls=Host, ip='10.0.0.8', defaultRoute=None)
 h9 = net.addHost('h9', cls=Host, ip='10.0.0.9', defaultRoute=None)
 h10 = net.addHost('h10', cls=Host, ip='10.0.0.10', defaultRoute=None)
 h11 = net.addHost('h11', cls=Host, ip='10.0.0.11', defaultRoute=None)
 h12 = net.addHost('h12', cls=Host, ip='10.0.0.12', defaultRoute=None)
 net.addLink(s1, s2)
 net.addLink(s2, s3)
 net.addLink(s1, s4)
 net.addLink(s4, s2)
 net.addLink(s1, s5)
 net.addLink(s5, s2)
 net.addLink(s2, s6)
 net.addLink(s6, s3)
 net.addLink(s2, s7)
 net.addLink(s6, s3)
 net.addLink(s3, s7)
 net.addLink(h1, s4)
 net.addLink(h2, s4)
 net.addLink(h3, s4)
 net.addLink(h4, s5)
 net.addLink(h5, s5)
 net.addLink(h6, s5)
net.addLink(h7, s6)
 net.addLink(h8, s6)
 net.addLink(h9, s6)
net.addLink(h10, s7)
 net.addLink(h11, s7)
 net.addLink(h12, s7)
info('*** Starting network\n')
 net.build()
 info('*** Starting switches\n')
 net.get('s1').start()
 net.get('s2').start()
 net.get('s3').start()
 net.get('s4').start()
 net.get('s5').start()
 net.get('s6').start()
 net.get('s7').start()
info('*** Network Elements Configuration\n')
```

II. SDN Architecture Based DCN Topology Python Script

In this architecture; the network elements are declared and configured using the Python scripts and APIs are explicitly designed for Mininet. The custom network topology created and its entire routing of the entire network is done. The network elements such as switches and hosts are declared, connected and configured, and ready to be tested. In this architecture, there is SDN controller that should be running in the designed topology. The type of controller is set to remote controller that needs the instantiation. The following command is typed in Mininet to begin the simulation process:

sudo python2.7 SDN_Based_ DCN_Topology.py

```
#!usr/bin/python2.7
from subprocess import call
from mininet.net import Mininet
from mininet.node import Controller, RemoteController, OVSController
from mininet.node import CPULimitedHost, Host, Node
from mininet.node import OVSKernelSwitch, UserSwitch
from mininet.cli import CLI
from mininet.log import setLogLevel, info
from mininet.link import TCLink, Intf
def datacenterNetwork():
 net = Mininet(topo=None, build=False, ipBase='10.0.0.0/8')
 info('*** Adding OpenDayLight controller\n')
 c0 = net.addController(name='c0', controller=RemoteController,
ip='192.168.56.56', protocol='tcp', port=6633)
 info('*** Adding switches\n')
 s1 = net.addSwitch('s1', cls=OVSKernelSwitch)
 s2 = net.addSwitch('s2', cls=OVSKernelSwitch)
 s3 = net.addSwitch('s3', cls=OVSKernelSwitch)
 s4 = net.addSwitch('s4', cls=OVSKernelSwitch)
 s5 = net.addSwitch('s5', cls=OVSKernelSwitch)
 s6 = net.addSwitch('s6', cls=OVSKernelSwitch)
 s7 = net.addSwitch('s7', cls=OVSKernelSwitch)
 s8 = net.addSwitch('s8', cls=OVSKernelSwitch)
 s9 = net.addSwitch('s9', cls=OVSKernelSwitch)
 info('*** Adding hosts\n')
 h1 = net.addHost('h1', cls=Host, ip='10.0.0.1', defaultRoute=None)
 h2 = net.addHost('h2', cls=Host, ip='10.0.0.2', defaultRoute=None)
 h3 = net.addHost('h3', cls=Host, ip='10.0.0.3', defaultRoute=None)
 h4 = net.addHost('h4', cls=Host, ip='10.0.0.4', defaultRoute=None)
 h5 = net.addHost('h5', cls=Host, ip='10.0.0.5', defaultRoute=None)
 h6 = net.addHost('h6', cls=Host, ip='10.0.0.6', defaultRoute=None)
 h7 = net.addHost('h7', cls=Host, ip='10.0.0.7', defaultRoute=None)
 h8 = net.addHost('h8', cls=Host, ip='10.0.0.8', defaultRoute=None)
 h9 = net.addHost('h9', cls=Host, ip='10.0.0.9', defaultRoute=None)
 h10 = net.addHost('h10', cls=Host, ip='10.0.0.10', defaultRoute=None)
 h11 = net.addHost('h11', cls=Host, ip='10.0.0.11', defaultRoute=None)
 h12 = net.addHost('h12', cls=Host, ip='10.0.0.12', defaultRoute=None)
 info('*** Adding links\n')
net.addLink(s1, s2)
 net.addLink(s2, s3)
 net.addLink(s1, s4)
 net.addLink(s4, s2)
 net.addLink(s1, s5)
 net.addLink(s5, s2)
 net.addLink(s2, s6)
 net.addLink(s6, s3)
 net.addLink(s2, s7)
 net.addLink(s6, s3)
 net.addLink(s3, s7)
 net.addLink(h1, s4)
 net.addLink(h2, s4)
 net.addLink(h3, s4)
 net.addLink(h4, s5)
 net.addLink(h5, s5)
 net.addLink(h6, s5)
net.addLink(h7, s6)
 net.addLink(h8, s6)
 net.addLink(h9, s6)
net.addLink(h10, s7)
 net.addLink(h11, s7)
 net.addLink(h12, s7)

 info('*** Starting network\n')
 net.build()
 info('*** Starting Controllers\n')
 for controller in net.controllers:
 controller.start()
 info('*** Starting switches\n')
 net.get('s1').start([c0])
 net.get('s2').start([c0])
 net.get('s3').start([c0])
 net.get('s4').start([c0])
 net.get('s5').start([c0])
 net.get('s6').start([c0])
 net.get('s7').start([c0])
info('*** Network Element Configuration\n')
```