COMPARATIVE ANALYSIS OF THE PERFORMANCE OF OPEN SHORTEST PATH FIRST AND OPENFLOW ON QUALITY-OF-SERVICE METRICS IN A LARGE SIMULATED MOBILE CORE NETWORK

BY

ODHIAMBO JOSEPH NICHOLAS OMUMBO

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE IN INFORMATION TECHNOLOGY

SCHOOL OF COMPUTING AND INFORMATICS

MASENO UNIVERSITY

DECLARATION

I declare that this thesis report is my original work, and where there is work or other individuals' contributions, it has been duly acknowledged, and relevant citations are given. To the best of my knowledge, this thesis has not been previously presented to any other educational institution or forum for examination. No part of this thesis may be reproduced without the author or Maseno University's prior permission**.**

Odhiambo Joseph Nicholas Omumbo

MSC/CI/00057/2014

[illegible] Digitally signed by Nicholas Odhiambo
DN: cn=Nicholas Odhiambo, o=US-ARMY, ou=MRDA-K, email=nicholas.odhiambo@usamru-k.org, c=KE
Date: 2022.09.20 10:20:28 +03'00'

Signature: .... **Date**17-SEPTEMBER-2022..

APPROVAL

The undersigned certify they have read and hereby recommend for acceptance of Maseno University thesis entitled **"Comparative Analysis of the Performance of Open Shortest Path First and OpenFlow on Quality of Service Metrics in a Large Simulated Mobile Core Network."**

Dr. Titus Muhambe Mukisa (Ph.D.)

Signature: [illegible]... **Date** 19-SEPTEMBER-2022..

School of Computing and Informatics, Maseno University

Dr. Ratemo Makiya Cyprian (Ph.D.)

Signature: Dr Ratemo... **Date** 19-SEPTEMBER-2022..

School of Computing and Informatics, Maseno University

ACKNOWLEDGEMENT

Achievements are always teamwork and standing on the shoulder of giants. I would like to thank God for the gift of life to pursue this dream at Maseno University. To the supervisors, Dr. Titus Muhambe and Dr. Cyprian Ratemo, this journey would have come to naught without your guidance. Your words of wisdom, the school leadership in Dr. Okoth Sylvester, and Dr. Henry Okora truly shaped this work. I would also like to extend a special thanks to the OMNeT++ development team for providing me with useful insights on how to improve the simulation environment. To my sister, brothers, and mother, your regular encouragement to complete this endeavor is no mean feat.

DEDICATION

This work is dedicated to Caroline Braeuer and Jabel Maria, whose life was taken before their time. To my little Shani, you are the best gift I have ever had since you came into this world.

ABSTRACT

Mobile application systems are increasingly being defined using Internet Protocol (IP). Therefore, one of the challenging tasks that mobile IP phone service companies have to deal with as their networks expand is an upsurge of user equipment (UE) and routers, which places excessive demand on the routing infrastructure and results in degraded quality of service (QoS) values at the mobile core network. Studies show that most mobile IP phone subscriber companies struggle to meet acceptable QoS values due to their continued use of classical routing methods. Classical routing protocols such as Open Shortest Path First (OSPF) lose efficiency as the network size increases, which leads to poor QoS values. An attempt to explore emerging Software Defined Network (SDN) technologies to enhance QoS and promote a centralized controller-based routing method has emerged in the recent past. OpenFlow routing is an instance of SDN found to improve routing in wired and small wireless networks. This study's main objective was to compare the performance of OSPF and OpenFlow routing in a large mobile IP core network. Specific objectives were to compare the performance of OSPF and OpenFlow routing on jitter, packet delivery ratio (PDR), throughput, and end-to-end delay. This research adopted a mixed design methodology consisting of exploratory and experimental research design where Objective Modular Network Testbed in C++ (OMNeT++) network simulator was used with simuLTE add-on to model two side-by-side mobile IP networks core architecture running OSPF and OpenFlow protocols. Applications that present high demand for processing and routing used; interactive gaming, VoiP, audio streaming, and Internet Protocol Television (IPTV) were used to test routing efficiency. The setup test environment consisted of 1000 UEs, 80 OSPF routers, and 80 OpenFlow switches which are considered a large network. Notably, Jitter was improved by 10 milliseconds when OpenFlow routing was used compared to OSPF. This improvement was consistent across the network with the addition of more UEs and Routers. In instances where OSPF improved PDR, the value is less significant, with a standard deviation of 0.7 mbs. This study demonstrated that OpenFlow could improve routing efficiency in large simulated mobile IP core networks compared to the OSPF routing network by over 60 percent across the four QoS metrics considered in the experiment. The study also confirmed that the network size impacts the QoS parameters. This study further recommends testing OpenFlow in large mobile IP production networks.

ABBREVIATIONS

AIP	All IP Network
API	Application Programming Interface
CAK	Communication Authority Kenya
CAM	Content Addressable Memory
CAPEX	Capital Expenditure
CP	Control Plane
CSP	Communication Service Provider
DP	Data Plane
EPC	Evolved Packet Core
FIB	Forwarding Information Base
GSM	Global System for Mobile Communications
IGP	Internal Gateway Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ITU	International Telecommunication Union
LTE	Long Term Evolution
NHS	National Health Services
OMNeT++	Object Oriented Modular Network using C++
ONF	Open Networking Foundation
OPEX	Operating Expenditure
OS	Operating System
OSPF	Open Shortest Path First
PDR	Packet Data Ratio
QoE	Quality of Experience
QoS	Quality of Service
RAN	Radio Area Network
SDN	Software Defined Networks
SIM	Subscriber Identification Module
UE	User Equipment
VoiP	Voice over Internet Protocol

OPERATIONAL DEFINITIONS OF KEY TERMS

All IP Based Network: Network that uses IP as its main base for all operations

Classical Routing: techniques that apply known routing protocols such as OSPF, EIGRP, or static methods.

Communication Service Provider: Company that provides mobile IP services for rental fee

Control Plane: conducts highly intelligent functions in a network like generation of Internet Protocols (IP) routing tables, forwarding of packets, and convergence of network events

Internet Protocol: TCP/IP network layer model concerned with logical addressing of nodes in a network.

Large Network: IP network consisting of 1000 User Equipment (UE) and 50 OSPF or OpenFlow routers and switches.

Open Shortest Path First: Link-state routing protocol that routers utilize hello packets to create neighborship and calculate routing paths based on cost

OpenFlow: Standard Southbound interface designed for SDN that connects access level pieces of equipment and a centralized controller.

Quality of Experience: The measure of QoS from the end-user perception of the quality of services offered by the communication service provider network

Quality of Service: An attribute of networks that can be used to prioritize or reprioritize packets based on business or technical choices before they are passed on to the next processing interface

Software-Defined Network: New paradigm in networks design that proposes for centralized and programmable Control Plane (CP)

LIST OF TABLES

LIST OF FIGURES

TABLE OF CONTENTS

CHAPTER ONE

INTRODUCTION

The chapter introduces the main objective of this investigation. Section 1.0 gives the background of the study. Section 1.1 covered the problem statement. Section 1.2.1 to 1.2.3 provides the specific objectives of the study. In sections 1.3 to 1.5, the study covered Justification, Assumptions, and Scope. Finally, in section 1.6, there is a summary of the key issues in the chapter.

1.1 Background Information

Globally the demand for mobile IP network services has been on upward growth in the last decade (Costa-requena, Kantola, & Llorente, 2014; Grayson, Shatzkamer, & Wainner, 2009; ITU, 2009; Liu, Ding, & Tarkoma, 2013; World Bank, 2011). According to (Cisco, 2020a) report, 300 million mobile applications will be downloaded by 2023. This trend is projected to continue in subsequent years (Cisco, 2015b; ITU, 2017; Open Networking Foundation, 2013; Stallings, 2002). Figure 1.1 shows the cisco forecast of smartphone and IP services growth within the decade in billions of users.

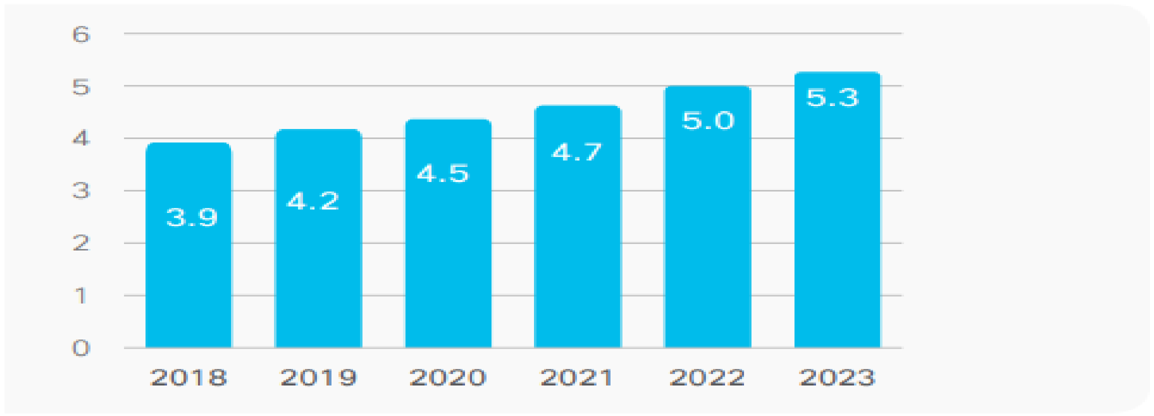


Figure 1. 1 Cisco projection of Mobile IP traffic (Cisco, 2020b)

Reports by (Cisco, 2015a; ITU, 2017) indicate that the worldwide mobile IP data services increased by 24.3 Exabytes in 2021. Mobile internet use also rose exponentially by 32 percent among Kenyans (CAK, 2018; Chepken, Blake, & Marsden, 2012; ITU, 2016, 2017). Figure 1.2 depicts the CAK projection of mobile IP growth in Kenya.

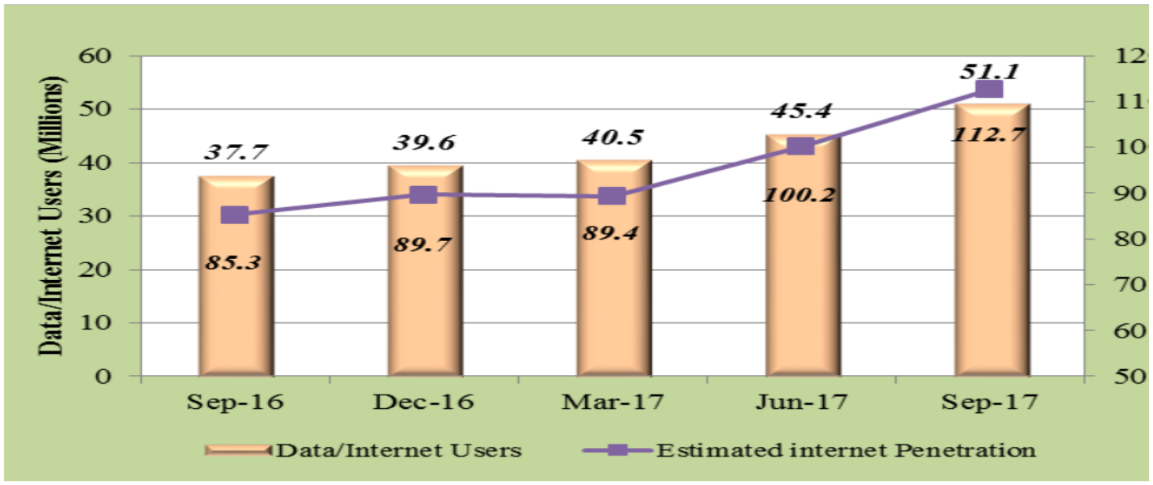
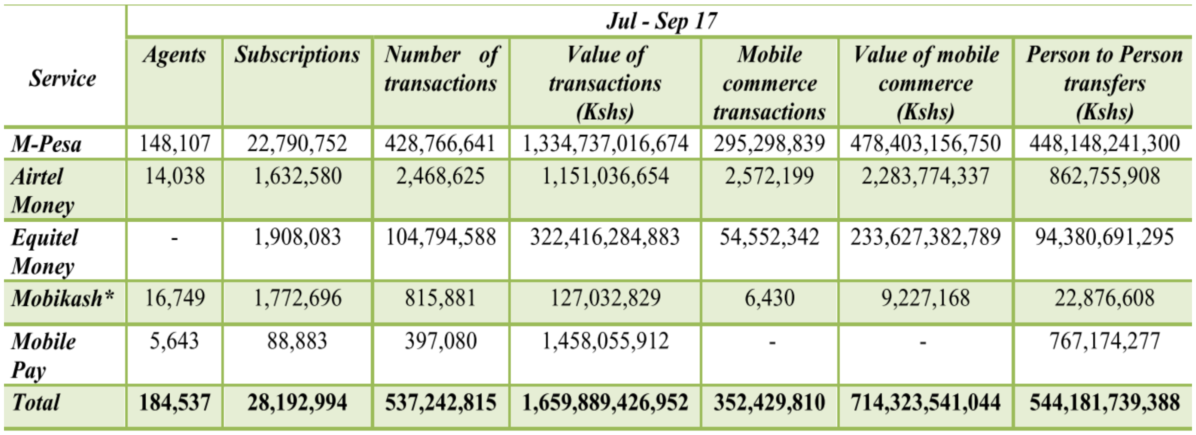


Figure 1. 2 Estimated Number of Mobile IP Internet Users in Kenya (CAK, 2019)

The gradual growth in mobile networks coverage and mobile phone ownership is both a business advantage and a Quality of Service (QoS) implementation challenge for mobile IP phone service providers as they try to meet recommended QoS of service metrics (Araniti et al., 2014; CAK, 2018; Cisco, 2014). Whereas an increasing number of subscribers means more revenue for the CSPs (Oloko, Anene, Kiara, & Kathambi, 2014), demand for more IP-based services, which in most cases are required simultaneously, places a considerable burden on the large mobile IP service provider infrastructure in terms of meeting the routing QoS demands (Araniti et al., 2014; Open Networking Foundation, 2012; Stallings, 2002). Figure 1.3 shows the CAK analysis of CSPs that deal with money transfers in Kenya, which confirmed that a large number of users is directly proportional to the revenue generated.



	Jul - Sep 17						
Service	*Agents*	*Subscriptions*	*Number of transactions*	*Value of transactions (Kshs)*	*Mobile commerce transactions*	*Value of mobile commerce (Kshs)*	*Person to Person transfers (Kshs)*
M-Pesa	148,107	22,790,752	428,766,641	1,334,737,016,674	295,298,839	478,403,156,750	448,148,241,300
Airtel Money	14,038	1,632,580	2,468,625	1,151,036,654	2,572,199	2,283,774,337	862,755,908
Equitel Money	-	1,908,083	104,794,588	322,416,284,883	54,552,342	233,627,382,789	94,380,691,295
*Mobikash**	16,749	1,772,696	815,881	127,032,829	6,430	9,227,168	22,876,608
Mobile Pay	5,643	88,883	397,080	1,458,055,912	-	-	767,174,277
Total	**184,537**	**28,192,994**	**537,242,815**	**1,659,889,426,952**	**352,429,810**	**714,323,541,044**	**544,181,739,388**

Figure 1. 3 Mobile money transfer service in numbers (CAK, 2019)

Inability to meet acceptable QoS values in packet routing and forwarding among CSP results in dropped calls and excessive IP call delay and could subject CSP to regulator penalties (Araniti et al., 2014; CAK, 2015, 2018; Szigeti, Hattingh, Barton, & Briley, 2013). Therefore, CSPs must address QoS issues to avoid the mishaps of poor QoS.

Popular services requested by users on mobile IP networks include interactive gaming services, live streaming of videos, chatting using secure applications such as WhatsApp, calling using audio over IP, voice over internet calls, and mobile IP money transfer services (Aker & Mbiti, 2010; Bisio et al., 2012; Cisco, 2020a; Cisconewsroom, 2013; Donovan, 2012; Omwansa & Sullivan, 2012). These services require a high Packet Data Ratio (PDR), increased throughput, low jitter, and low delay in a mobile IP core CSP network. According to (CAK, 2014), in a customer advisory on mobile IP performance where PDR, jitter, throughput, and delay are collectively referred to as technical QoS values, CAK pointed out that these are essential metrics that determine how optimum a mobile network service is operating. Furthermore, CAK advised end-users to be keen on these measurements to ensure they acquire value from their subscription to particular CSP companies. Without a proper mechanism for addressing QoS values in the routing process within the mobile IP provider core network, such scenarios can result in poor QoS values (CAK, 2014).

Like in any other typical network, routing is a key function to enable users to reach other networks which are in remote subnets (Costa-requena et al., 2014; Li, Mao, & Rexford, 2012; Moy, 1999; Odom, 2017, 2019; Tanenbaum, 2013; Wallace, 2014). The inability of mobile IP service provider companies to meet the best QoS values as required by regulators such as CAK in Kenya has been attributed to the continuous use of traditional routing approaches in the IP network core (Araniti et al., 2014; CAK, 2018; Clare, 2020; Li et al., 2012; Open Networking Foundation, 2013; Yap et al., 2010).

In the 1990s, efforts to simplify network operations management were not straightforward (Azodolmolky, 2013; Feamster, Rexford, & Zegura, 2013). According to (Feamster et al., 2013), many researchers were frustrated by the rigidity of network development's standardization process and the inability to program and automate network operations. Standards take several years before they are published in Request for Comments (RFC) (Casado & Foster, 2013; Cassado, 2015; Feamster, Balakrishnan, Rexford, Shaikh, & van der Merwe, 2004; Feamster et al., 2013). With the shortcomings of routing in traditional networks, the urge to offer new agile solutions brought forward the Software-Defined Network (SDN).

Mobile IP applications have revolutionized the healthcare, banking, and security sectors immensely (Grayson et al., 2009). Several mobile IP applications today help users to pay bills, detect disease prevalence in an area, stream online TV programs, chat securely using IP messages and check the weather forecast from the comfort of handheld smartphones (CitizenTV, 2021; Ecobank, 2018; NHS, 2020; Rosenfeld, Sina, Sarne, Avidov, & Kraus, 2018; Safaricom, 2018; WhatsApp, 2017).

Throughput, Jitter, PDR, and delay affect routing quality in a mobile IP provider network, especially in a large mobile IP core network. Large mobile IP networks offer a different disposition from earlier years concerning attaining high routing standards in the core network (Araniti et al., 2014; Li, Mao, & Rexford, 2016; Stalling, 2009; Stallings, 2002). While the expansion of the mobile IP network is rapid and demand for more IP-based services continues to grow exponentially, routing in large mobile service providers is reportedly continuing to deploy traditional routing methods (Rakuten, 2019a, 2019b).

Findings by (Araniti et al., 2014; Feamster et al., 2004; Li et al., 2016; Liu et al., 2013; Miller, 2002a; Mueck et al., 2017; Rakuten, 2019a) showed that continuing to use classical routing methods in large CSP core networks does not improve the overall routing processes that are required of a large all IP based network (AIP) core network.

1.1 Statement of the Problem

Mobile IP network applications are increasingly being defined using IP, resulting in a large core network comprising several routers and switches. Classical routing cannot effectively address the routing QoS values desired of a large CSP core network. Classical routing generates high jitter, low PDR, low throughput, and high delay resulting in an overall loss of efficiency in the core network. Several regulators and provider reports reveal that QoS among mobile service providers has failed to meet the contractual standards in successive years. This can be attributed to the continued use of classical routing approaches such as OSPF by the mobile subscriber companies in the core network. It is also evident that increased UEs and routers negatively impact the QoS values in the core network. Techniques such as Network-Based Access Recognition (NBAR) or hybrid classical protocols such as EIGRP require a lengthy training curve for network QoS engineers. They are also designed to reside in one network plane, with no mechanism to address agile business requirements.

1.2.1 General Objective

To comparatively analyze the performance of OSPF and OpenFlow on Quality of Service metrics in a large simulated mobile core network.

1.2.2 Specific Objectives

i. To compare the performance of OSPF and OpenFlow routing on jitter on a large simulated mobile IP core network.
ii. To compare the performance of OSPF and OpenFlow routing on packet data ratio on a large simulated mobile IP core network.
iii. To compare the performance of OSPF and OpenFlow on throughput routing on a large simulated mobile IP core network.
iv. To compare the performance of OSPF and OpenFlow on end-to-end delay routing on a large simulated mobile IP core network.

1.2.3 Research Questions

i. What performance levels will OSPF and OpenFlow routing have on jitter in a large simulated mobile IP core network?
ii. What performance levels will OSPF and OpenFlow routing have on PDR in a large simulated mobile IP core network?
iii. What performance levels will OSPF and OpenFlow routing have on throughput in a large simulated mobile IP core network?
iv. What performance levels will OSPF and OpenFlow routing have on end-to-end delay in a large simulated mobile IP core network?

1.3 Justification of the Study

More access systems in mobile networks are continuously being defined using IP protocols which implies that more IP traffic is being witnessed in the core network (Grayson et al., 2009; Rosenfeld et al., 2018). Degradation caused by unacceptable routing QoS values in an already expansive mobile IP network could result in loss of revenue and unhappy end-users (CAK, 2015, 2016;

Cisconewsroom, 2013; Li et al., 2012; Star, 2014). Therefore, this study aimed to provide important insight into the performance of OpenFlow and OSPF routing on QoS, in the large CSP network core and provide feedback on how the two protocols performed in large mobile IP core networks. In addition, the routing engineers would adopt the recommended routing architecture in mobile IP core networks to improve routing speeds.

1.4 Assumptions and Limitations of the Study

While selecting OMNeT++ and simuLTE to test the performance of OpenFlow and OSPF in a large CSP network, control plane are not included as part of the simulator are provided as part of the simulator setup. This omission is done to improve the simulation speeds and processor utilization. The number of UEs and routers is also limited to 1000 UEs and 80 routers for the open-source version (Andras, 2017; Nardini, Antonio, Rudolf, Varga, & Meszero, 2021). However, the absence of the control plane protocols did not hinder the research from exhaustively testing routing performance in a large mobile IP core network because the UEs can share resources such as bandwidth using OMNeT++ inbuilt algorithm.

1.5 Scope of the Study

The research focused its experiment on IPv4 due to its widespread use and implementation in end-user devices (Graziani, 2013; Odom, 2016, 2017).

This study was conducted in an emulator OMNeT++ and LTE simuLTE add-on. Objective Modular Network Testbed in C++ is an open-source emulator to test and design mobile IP networks.

This study limited its evaluation to technical IP QoS metrics: PDR, jitter, end-to-end delay, and throughput. These QoS measurements are important factors that can influence user experience, which is the most important factor in mobile IP service (CAK, 2014; Janevski, Jankovic, & Markus, 2013; Szigeti et al., 2013).

1.6 Summary

This chapter provided the background information to the research. The mobile data industry was observed to be on the upward curve, with growth of subscribers at 6 percent globally and 32.12 percent locally. The study also concluded that with an increase in mobile IP service subscribers, the demands for the current mobile IP network required new strategies for routing in the IP network core. The use of traditional routing approaches was still present in most mobile networks. Further review showed that classical routing protocols are not optimal in the core network setup. This research also outlined the objectives and the study questions that needed to be addressed by experimenting with the research objectives. The importance of mobile IP in several facets of the end-user justified the need for this research.

Finally, the study stated the scope and limitations and provided an overview of the experimental simulation tool OMNeT++.

CHAPTER TWO

LITERATURE REVIEW

This chapter reviews the literature relevant to the study. In section 2.1, an overview of routing architectures and QoS is illustrated. In section 2.2, an explanation of network planes is done, followed by studies done on specific (QoS) metrics in sections 2.3 to 2.7. Finally, the knowledge gap that this study aims to address is covered in section 2.8.

2.1 Overview of Routing and Quality of Service in the Core Network

Routing is a process that primarily allows devices in different mobile IP subnets to communicate. Routing is normally performed by a router or multi-layer switch. Routing has two main approaches to achieve its main goal; either locate routes through static configuration or dynamically through a routing algorithm (Odom, 2017, 2019). In Figure 2.1, two IP subnets, 168.10.0.0 and 168.20.0.0, are connected by routers A and B. The two routers have the role of locating subnets in different IP subnets to move data between two or more subnets.

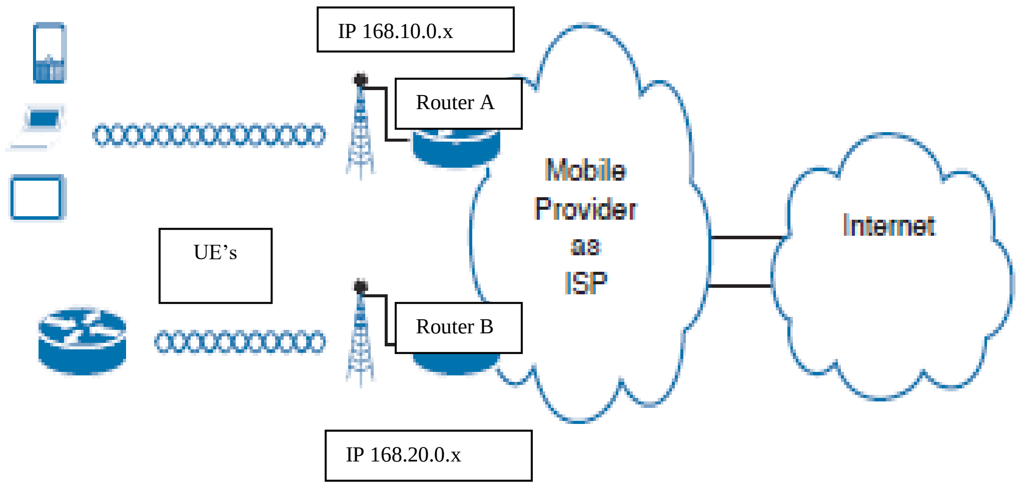


Figure 2. 1 Mobile Internet Protocol Network with Subnets (Author, 2020)

Large CSPs have several routers and switches in dispersed zones in their core networks. Therefore, subnets are usually set up to segment the network into small units requiring communication through routing.

2.1.1 Legacy versus SDN Routing Protocols Architecture: An overview

OSPF is a link-state protocol used in many network domains (Shaikh, Isett, Greenberg, Roughan, & Gottlieb, 2002). Link states are the category of IGP and dynamic routing protocols that primarily use hello messages to converge and maintain network health. While there are several link-state protocols, this study selected OSPF due to its ready availability in the test environment, wide use, and open source license. OSPF uses a distributed approach where every device attempts to describe the network accurately and sends periodic hello messages to inform the adjacent routers about the network state (Moy, 1999). Typically in OSPF, the CP and DP reside in one hardware box (Odom, 2019).

OpenFlow is a new SDN-based forwarding protocol that uses a centralized controller mode to abstract the central CP that sends flow entries to OpenFlow switches and the DP. This abstraction ensures that the switches concentrate on the DP to optimize network bandwidth and vision (Jain et al., 2014).

With the advent of new routing approaches, fresher parameters to compare routing functionality have been devised (Casado & Foster, 2013; Feamster et al., 2004; Zhang & Yan, 2015). The new comparison methods focus mainly on the architecture of legacy and modern SDN-based protocols. As observed by (Zhang & Yan, 2015) and (Feamster et al., 2004) the respective architectures contribute to how effective one type of routing protocol operates compared to another, which uses a different architecture.

2.1.2 Centralization versus Distributed

In SDN architectures, a centralized controller is deployed to provide the CP plane logic in the network. Centralization of routing allows for the separation of CP and DP. When the CP and DP are separated, the SDN-based network has a domain-wide view of the entire network. This method has been observed to route data packets faster and save bandwidth due to the absence of excessive control information shared in the network.

Distributed networks devolve the routing logic across all the routers in the network. The routers rely on shared hello packets to maintain, add and remove routes as may be required. Studies by (Miller, 2002b) and (Feamster et al., 2004) show that distributed routing design is prone to errors and especially when the network increases in size.

2.1.3 Simplicity versus Complexity

SDN routed networks simplify sharing and forwarding network state information using flow entries. Flow entries consist of three main actions applied to the data packets; match, forward or drop. The controller has a real-time engine identified as Link Discovery Manager that discovers switches in the network and populates the CAM table with flow entries with special regard for QoS. This approach has been observed to reduce communication between controllers and switches by over 20 percent, improving QoS metrics such as PDR (Zhang & Yan, 2015).

In distributed routing architecture like OSPF, the networks start by exchanging network information using hello packets before they can become neighbors and share route information. Thereafter, the routers will keep sending information on the network to check the link status and availability of neighboring devices. As observed earlier, the sending of additional hello packets in the network uses bandwidth to the detriment of QoS metrics such as PDR and Jitter.

2.1.4 Expanded versus Limited forwarding headers.

In a typical legacy routing format, packets are primarily forwarded using source and destination IP. The routing process also depends on the routing information base (RIB) and forwarding information base (FIB). FIB and RIB are distributed tables containing the memory of routes known to the routers and their neighbors. However, traffic forwarding flow tables are used in OpenFlow networks rather than just the source and destination addresses. Each flow entry has three actions: forward, redirect, and drop. Each flow table in the switch is composed of flow entries made with counters, and match fields and instructions are set to apply to traffic when they meet forwarding conditions found in the data headers (Hewlett Packard, 2013; Tourrilhes, Sharma, Banerjee, & Pettit, 2014).

2.1.5 Performance comparison of the network architectures

The network design review demonstrated that the routing protocol architecture contributed to the effectiveness of one particular approach compared to another.

2.2 Overview of Data, Control, and Management planes

According to (Odom, 2017), planes refer to a specific task carried out by the network component in the network communication architecture. There are three main planes in networks: Data, Control, and Management (Azodolmolky, 2013; Feamster et al., 2004; Hewlett Packard, 2013; Odom, 2017; Tourrilhes et al., 2014). Data plane refers to a networking device's task to forward a message. For instance, data plane protocols encapsulate data with the correct data headers and ports to ensure the correct delivery to the right host and application program. Figure 2.2 (Author, 2020) illustrates the data encapsulation process for bidirectional traffic.

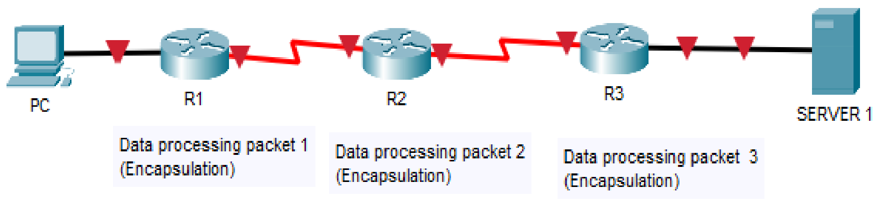
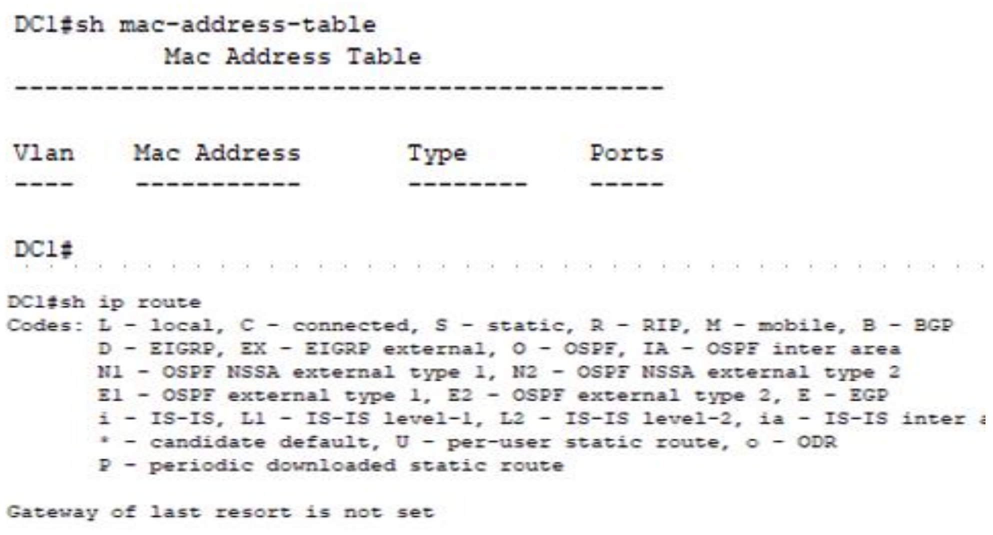

PC
R1
R2
R3
SERVER 1
Data processing packet 1 (Encapsulation)
Data processing packet 2 (Encapsulation)
Data processing packet 3 (Encapsulation)

Figure 2. 2 Traffic encapsulation (Author, 2020)

Control plane, on the other hand, refers to the task in networking devices that ensures data is delivered to the next destination address (Odom, 2017; Stalling, 2009). For instance, routing protocols such as OSPF, a distributed protocol, ensure that routers have the right routes added to the forwarding information base (FIB) to deliver the packet to the next destination listed in the DP encapsulation process. Figure 2.3 shows the CP and DP bundled in one hardware plane with listed MAC address and control plane protocols.

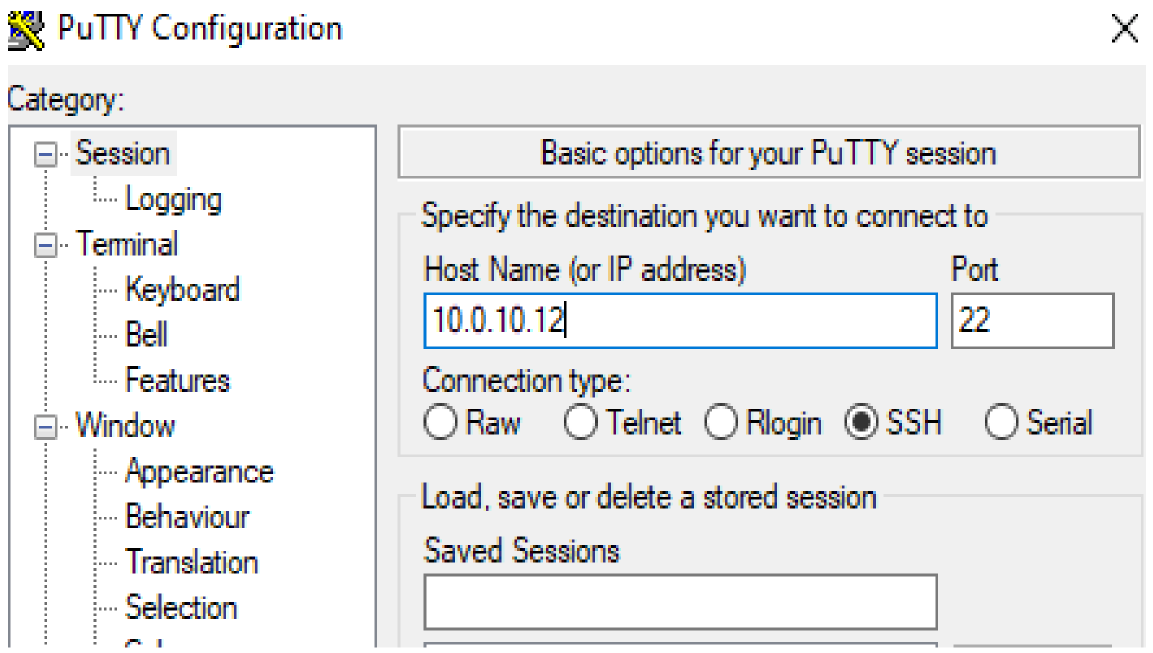


```
DC1#sh mac-address-table
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----

DC1#
```

```
DC1#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter a
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set
```

Figure 2. 3 Control plane between routers in a single hardware bundle (Author, 2020)

According to (Stalling, 2009) and (Odom, 2017) the management plane performs the overhead tasks that affect the behavior of the data and control plane. It provides the ability for network engineers to configure devices to influence the behavior of the network devices. It uses well-known protocols such as telnet and secure shell to access network devices via a console or remote connection. Figure 2.4 shows the management plane configuration of a router located at 10.0.10.12.



Host Name (or IP address): 10.0.10.12
Port: 22
Connection type: Raw Telnet Rlogin [✓ SSH] Serial

Figure 2. 4 Figure 2.4 Management plane tool (Author, 2020)

2.2.1 Plane organization in legacy and SDN protocols

In legacy routing protocols CP and DP are bundled in one hardware box. Figure 2.5 illustrates the three main planes in classical routing network architecture. Routing in legacy protocols involves two main parameters; source and destination address (Hewlett Packard, 2013; Odom, 2019)

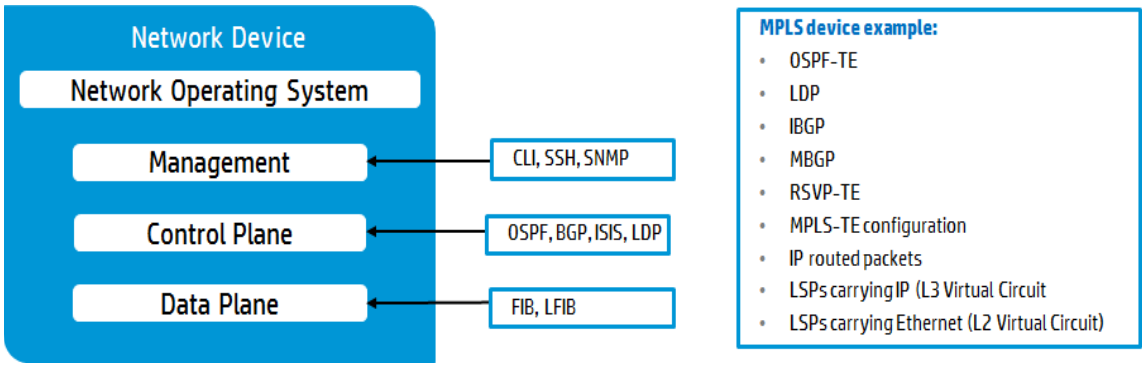


Figure 2. 5 Classical Router plane organization (Hewlett Packard, 2013)

On the other hand, OpenFlow routing separates CP and DP. The data plane resides on an OpenFlow switch, router, or firewall that performs forwarding functions. The controller's domain view of the network performs the intelligent decision of creating neighborship and calculating the best path for a packet to reach a destination. OpenFlow switch and controller interact through four main messages, packet-receive, send-packet-out, modify forwarding table, and get-statistics (Hewlett Packard, 2013; ONF, 2018b). Figure 2.6 (Author, 2020) shows a controller-based network with a centralized controller.

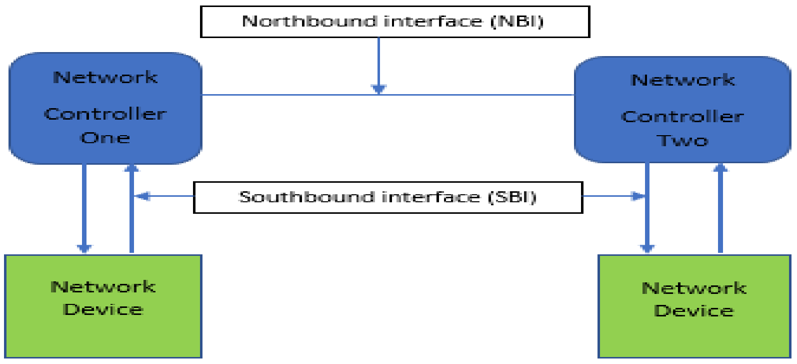


Figure 2. 6 Controller-based network architecture (Author, 2020)

2.2.3 Performance comparison between legacy and SDN-based plane architecture

OpenFlow routing uses several criteria to match flow entries for IP traffic. Table 2.1 shows a summary of criteria used by OpenFlow to match traffic paths. Column 1 under Rule criterion shows the fields that can be matched in OpenFlow protocol; column 2 shows actions that the controller takes on the packet before its delivered or forwarded to the next egress interface. Column 3 shows counters that can be recorded on the packet processing process that occurs on the controller device.

Table 2. 1 OpenFlow Headers

Rule criterion	**Action**	**Statistics**
Switch port VLAN ID VLAN PCP MAC Source MAC destination Ether type IP source IP Destination IP ToS IP Protocol L4 Source L4 Destination	Forward to one or more packets Drop Encapsulate and Forward Send to normal processing pipeline Modify Fields	Packets Byte

Source: (Author, 2020)

Consequently, multiple actions can be carried on packets through granular inspection of packet headers. Packets are forwarded based on several criteria such as ingress port, VLAN ID, Layer 3 source and destination, and Layer 4 source and destination. This has been shown to vastly improve the packet forwarding process (Hewlett Packard, 2013; Odom, 2017; Tourrilhes et al., 2014). The extended forwarding mechanism in OpenFlow is achieved by separating CP and DP.

In legacy routing, scholars (Araniti et al., 2014; Feamster et al., 2004; Hewlett Packard, 2013; Miller, 2002a; Odom, 2017; Shaikh et al., 2002; Tourrilhes et al., 2014) and (Sundaran, 2002) agree that bundling the CP and DP in on hardware is prone to routing errors and lack of efficiency in QoS values. In combining the CP and DP, there is less room for programming in classical routing approaches. Programming of networks opens room for more subtle criteria in defining routes and routing. In addition, the distributed nature of routing does not always guarantee accurate information in forwarding packets (Hewlett Packard, 2013; Sundaran, 2002; Tourrilhes et al., 2014). From the literature review, this study showed that combining DP and CP reduced the routing efficiency in networks that require less jitter and high throughput of data, such as large mobile core network IP networks. Further, the review demonstrated that legacy routing protocols are not well suited to address the challenges that face modern CSP core networks. Newer approaches such as OpenFlow needed to be investigated for performance.

2.3 Quality of Service

2.3.1 Introduction to Quality of Service

There are various definitions of QoS. According to (Grayson et al., 2009), QoS is a way of implementing unmanaged fairness in a network. (Cisco, 2012) defines QoS as a measure of transmission quality and network service availability. An ITU manual (ETSI, 2003) and (Janevski et al., 2013) define QoS as the ability to segment network traffic or differentiate between traffic types for the network to treat certain traffic differently from others. These definitions agree on the need to manage network traffic using QoS tools to improve user experience in most networks. This study, therefore, defines routing QoS as the management of network traffic using inbuilt software tools to enhance user experience in the network.

Quality of service constitutes a vital part of any network infrastructure, affecting routing in equal measure (Grayson et al., 2009). While the mobile IP network core and RAN have expanded exponentially (Cisco, 2016a; ITU, 2009), classical routing methods have been largely unable to address effectively new challenges posed by large mobile IP core networks like simultaneous access and bandwidth-intensive applications such as live streaming (Gude et al., 2008; Miller, 2002a; Sundaran, 2002). In addition (Miller, 2002a) and (Sundaran, 2002) agree that classical routing is suboptimal, and its connections on hosts are unreliable, especially as the network size expands.

According to (Szigeti et al., 2013), the technical QoS values against which network performance is measured are summarized in Table 2.2. These QoS values, which are industry-approved, formed the baseline on which the experimental results were used to answer the research questions and objectives.

Table 2. 2 Recommended Quality of Service Values

Application	**Delay**	**Jitter**	**PDR**	**Throughput**
Interactive gaming	<=150ms	<=30ms	>=95%	384kbps
VIDEO	<=150ms	<=30ms	>=95%	384kbps
Voice	<=150ms	<=30ms	>=95%	384kbps
IPTV	<=150ms	<=30ms	>=95%	384kbps
DNS	200-400ms	30-50ms	0.1%	384kbps

Source (Szigeti et al., 2013)

2.3.2 Jitter with OpenFlow and OSPF

According to (Odom, 2017), Jitter is a variation in a one-way packet transmission between two or more nodes connected in a network. According to (Cisco, 2005), Jitter is the difference in the end-to-end delay between packets sent over a single communication medium simultaneously. On the other hand, (Spirent, 2007) defines jitter as the absolute value of the difference between two consecutive packets sent between two nodes and attached to some data stream. In view of the above definitions, this study adopted Jitter as the difference in bidirectional delay by two or more packet streams sent over a communication media. Figure 2.7 (Tanenbaum, 2013) shows jitter and its impact on bidirectional communication.

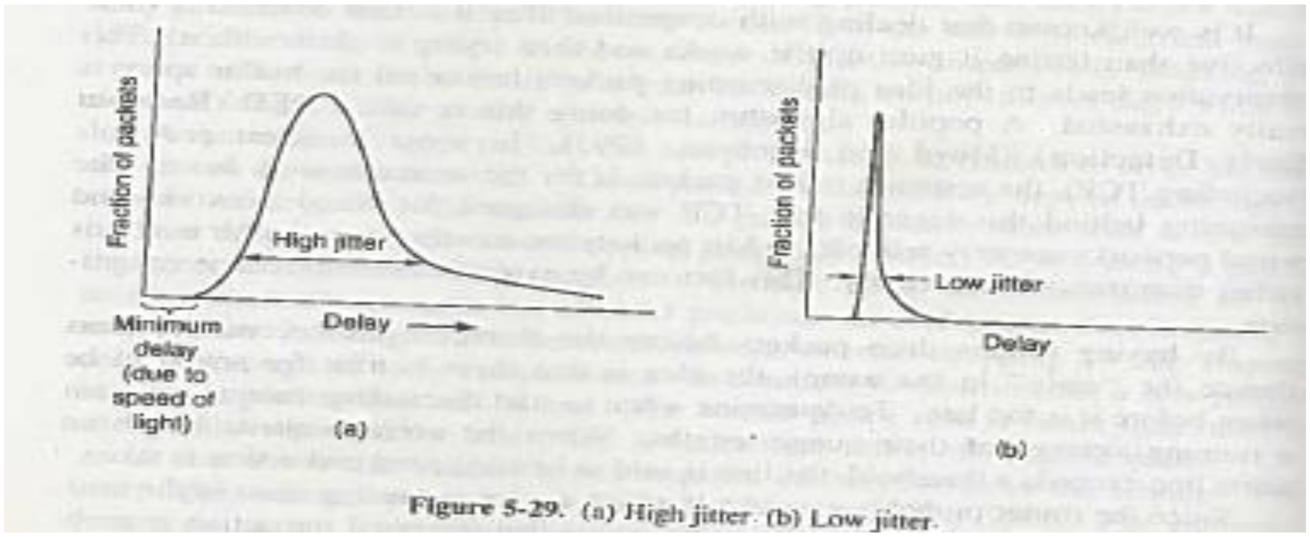


Figure 2. 7 Graph on Jitter (Tanenbaum, 2013)

According to (Poretsky, Perser, Eramilli, & Khurana, 2006), in a request for comments (RFC) report, Jitter forms an important factor in any communication media due to its ability to influence the end-user experience in a network. In explaining, Jitter (Tanenbaum, 2013) stated that if one packet takes 20 milliseconds (msec) and another 40 milliseconds (msec) to arrive, that will give an uneven quality to the data stream. Excessive jitter can degrade the value of voice and data packets to unacceptable quality levels (Spirent, 2007).

(Poretsky et al., 2006), developed the formula for computing jitter is as follows

D (i) –D (i-1)) where D = Forwarding Delay, and i = order of packets arrival…. equation 2…x

Routing in a large mobile IP network poses a different challenge than in small wireless and wired networks. Studies by (Araniti et al., 2014; Azodolmolky, 2013; Li et al., 2016; Miller, 2002b; Odom, 2017) showed that classical routing could not adequately tackle challenges presented by (QoS) requirements. A study by (Araniti et al., 2014) demonstrated that jitter in small wireless and wired network could be improved by adopting robust routing approaches such as OpenFlow

According to (Araniti et al., 2014), OpenFlow routing in a small mobile network was carried out in a small wireless network using a sample of 30 UEs and a single eNodeB with the aim of testing for OpenFlow routing and its impact on QoS values. In addition, the results showed that OpenFlow routing improved jitter levels in the small wireless network in the overall routing process compared to the traditional routing approach based on OSPF. According to (Cisco, 2010), in its technology implementation known as Cisco performance routing CPr, reports that its deployment enhances routing performance in the network by implementing master and border controller routers. Master controller has a global view network, and the border routers forward packets to the master router, thus decreasing jitter levels due to its readily available forwarding information base (FIB). Hewlett Packard implemented OpenFlow in its suite of switches (Hewlett Packard, 2013), and according to (Tourrilhes et al., 2014), in the implementation of OpenFlow, HP switching is informed by the fact that classical routing cannot address the levels of jitter in its current network requirements. In a study by (Jain et al., 2014), while experimenting with partial OpenFlow deployment for Google corporation, they reported that OpenFlow introduced considerable savings in its data centers and improved routing by 20 percent compared to classical approaches used originally. In addition, they cited two key benefits from the report: efficient operation of the WAN and better bandwidth management.

In several studies by (Feamster et al., 2004; Jain et al., 2014; ONF, 2018a; Yap et al., 2010; Zhang & Yan, 2015), they reported that OpenFlow was able to improve or lower jitter levels in their networks by up to 20 percent or more compared to legacy routing protocols.

In the study by (Jain et al., 2014) from the Google OpenFlow experiment, they concluded that while the results showed promise using SDN routing, its findings could not generalize QoS performance to all forms of WANs and core networks. Also, the study by (Araniti et al., 2014) stated that its experiment was done in a small wireless network over a short period. Therefore, it cannot represent how jitter would behave in a large mobile IP core network. While CPr routing supports the separation of control and data planes, it is a proprietary approach. Furthermore, this research did not find any reports on how CPr could impact jitter in large mobile IP network core (Cisco, 2010). In several studies (Feamster et al., 2004; Jain et al., 2014; ONF, 2018a; Yap et al., 2010; Zhang & Yan, 2015), there was no evidence of jitter evaluation in large mobile IP core networks.

The study underlined that OpenFlow performs better than legacy routing approaches such as OSPF in handling Jitter in small wireless and wired networks. However, the review did not find any evidence of studies that compared Jitter in OpenFlow and OSPF in the large mobile core network.

2.3.4 Packet Data Ratio with OpenFlow and OSPF

According to (Araniti et al., 2014) (PDR) is the ratio of data packets received by the destination divided by source packets. Similarly, the study by (Balasubramanian, 2006) defined (PDR) as the ratio amount in the percentage of packets delivered from sending to receiving device. Packet Data Ratio is the ratio of data packets received by the destinations to those generated by the source (Rohal, Dahiya, & Dahiya, 2013). In view of the similarities in definitions (Araniti et al., 2014; Balasubramanian, 2006; Rohal et al., 2013), this study defined PDR as the ratio of packets received over sent packets from the source node. Formula 2.2 below (Rohal et al., 2013) provides a mathematical calculation of PDR as

$$PDR = S1 \div S2 \ \ldots \ \textit{equation 2.2}$$

In equation 2.2, S1 is the total number of data packets received by each destination, and S2 is the total of data packets generated by each source.

Less PDR value in mobile service provider core network results in loss of communication and may cause garbled audio over IP or VoiP calls (Araniti et al., 2014; Cisco, 2005; Spirent, 2007). According to studies by (Araniti et al., 2014; Cisco, 2005; Odom, 2017), all agreed that (PDR) should be kept at maximum value. As stated by (Odom, 2017) and (Tanenbaum, 2013), they all contend that other factors like external interference, lack of standards, and faulty cabling can result in data loss, negatively affecting PDR. However, the main reason for undesirable PDR levels was the deployment of classical routing approaches in mobile IP provider networks (Azodolmolky, 2013; Cisco, 2005; Li et al., 2012; Odom, 2017). Therefore, studies by (Araniti et al., 2014; D 'souza, Sundharan, Lokanath, & Mittal, 2016; Feamster et al., 2004, 2013) suggested the need for new routing architectures to address problems associated with minimal PDR in traditional routed networks in mobile CSP core networks. OpenFlow has shown improvement in QoS values when deployed in wired and small wireless networks (Araniti et al., 2014; Jain et al., 2014; Zhang & Yan, 2015). Figure 2. 9 illustrates how SDN stabilized PDR in a small simulated wireless network.

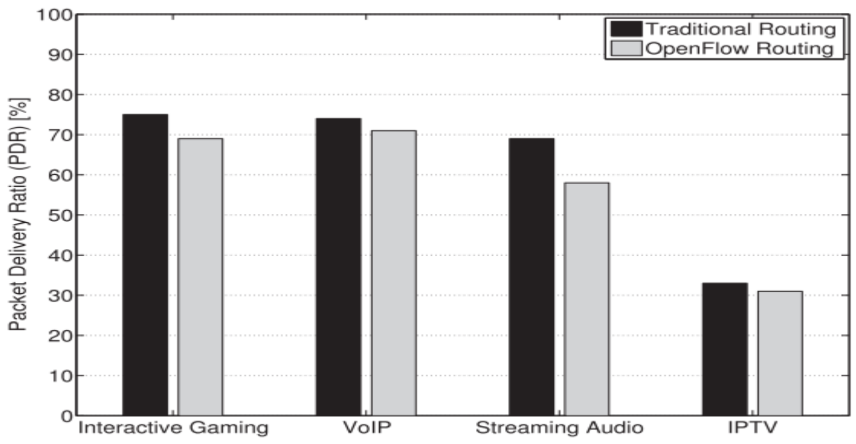


Figure 2. 8 PDR comparison of OpenFlow and OSPF network (Araniti et al., 2014)

In a study by (Araniti et al., 2014), OpenFlow routing in the wireless network used a sample of 30 UEs and one AP to investigate the impact of OpenFlow routing on a small wireless network PDR values improved by over 10 percent when compared to traditional routing approaches.

In its pioneer research at B4, Google experimented partially with OpenFlow in its data center and observed an improvement of up to 20 percent PDR compared to the legacy routed network in its backhaul network (Google, 2010; Jain et al., 2014).

Juniper's technical guide on OpenFlow (Juniper, 2014; Networks, 2015) also stated that the classical approach to routing makes networks rigid, static, and averse to changes. According to (Networks, 2015), OpenFlow routed networks are open and agile and improve PDR values. Similar to several other leading network vendors like HP, Juniper has incorporated OpenFlow in a series of its switches (Networks, 2015)

In the study by (Jain et al., 2014) from the Google OpenFlow experiment, they concluded that while the findings indicate SDN improved PDR, its findings could not be used as a one-stop solution for PDR performance in all forms of WANs and core networks. The study by (Araniti et al., 2014) stated that its experiment was done in a small wireless network. It cannot, therefore, represent how PDR would be impacted in a large core CSP network. While CPr routing supports the separation of control and data planes, it is a proprietary approach. Furthermore, this research did not find any reports on how CPr could impact jitter in large mobile IP network core (Cisco, 2010). In several studies (Feamster et al., 2004; Jain et al., 2014; ONF, 2018a; Yap et al., 2010;

Zhang & Yan, 2015), there was no evidence of how PDR performed in large mobile IP core networks.

This review showed that OpenFlow performs better than legacy routing approaches such as OSPF in handling PDR. However, the review did not find any evidence of studies that examined the comparative performance of PDR in OpenFlow and OSPF in large mobile core networks.

2.3.5 End-to-End delay with OpenFlow and OSPF

Scholars concur on the definition of delay as the time taken for a data packet to travel from one point to another (Araniti et al., 2014; Balasubramanian, 2006; Odom, 2017; Rohal et al., 2013). According to (Odom, 2017) and (Grayson et al., 2009), the delay is the time ms packets take to travel on bidirectional communication. Studies by (Balasubramanian, 2006; Cisco, 2005, 2012; Odom, 2017; Rohal et al., 2013) all agreed that packet delay constitutes time taken to arrive from one destination to another either as one way or a two-way round trip. This study agrees that there was no significant difference in how scholars defined end-to-end delay from the following definitions. Therefore, this research defines end-to-end delay as the time it takes for data packets to travel from source to destination and back. Formula 2.3 showed the computation for the end-to-end delay in a network, explained below.

Average delay =S/N…. equation 2.3 (Balasubramanian, 2006)

In formula 2.3, S is the sum of time spent to deliver packets for each destination, while N is the number of data packets arriving at all destination nodes.

End-to-end delays exceeding 150 ms are QoS metrics that can negatively impact network traffic and affect end-user experience (CAK, 2014; Janevski et al., 2013; Szigeti et al., 2013). Excessive delay results in data loss, which also results in incomplete communication. It’s not uncommon to read about mobile core network services grounding to a halt because of technical hitches like delays (CAK, 2014, 2015, 2016, 2018).

According to (Poretsky et al., 2006) and (Janevski et al., 2013), end-to-end delay constitutes an important element in core networks due to its ability to influence the end-user experience in a network. Routing in a large mobile IP core network poses a different challenge than in small wireless and wired networks. Studies by (Araniti et al., 2014; Azodolmolky, 2013; Li et al., 2016; Miller, 2002b; Odom, 2017) showed that classical routing could not adequately tackle challenges presented by (QoS) requirements. Studies by (Azodolmolky, 2013; D ’souza et al., 2016; Hu, Hao, & Bao, 2014; Jain et al., 2014) demonstrated an end-to-end delay in small wireless and wired networks could be improved by adopting robust routing protocols such as OpenFlow.

According to (Araniti et al., 2014), OpenFlow routing in a small mobile network was carried out in a small wireless network using a sample of 30 UEs and one eNodeB with the aim of testing for OpenFlow and OSPF routing and its impact on QoS values. In addition, the results showed that in the overall routing process, OpenFlow routing improved end-to-end delay by 20 percent in the small wireless network compared to the traditional routing approach based on OSPF (Araniti et al., 2014).

In another study (Zhang & Yan, 2015), in comparing the convergence performance of SDN and legacy protocol routing QoS, SDN improved end-to-end delay by 14 percent from a network of 120 routers. Figure 2.10 shows the improvement of end-to-end delay, especially as network routers increase.

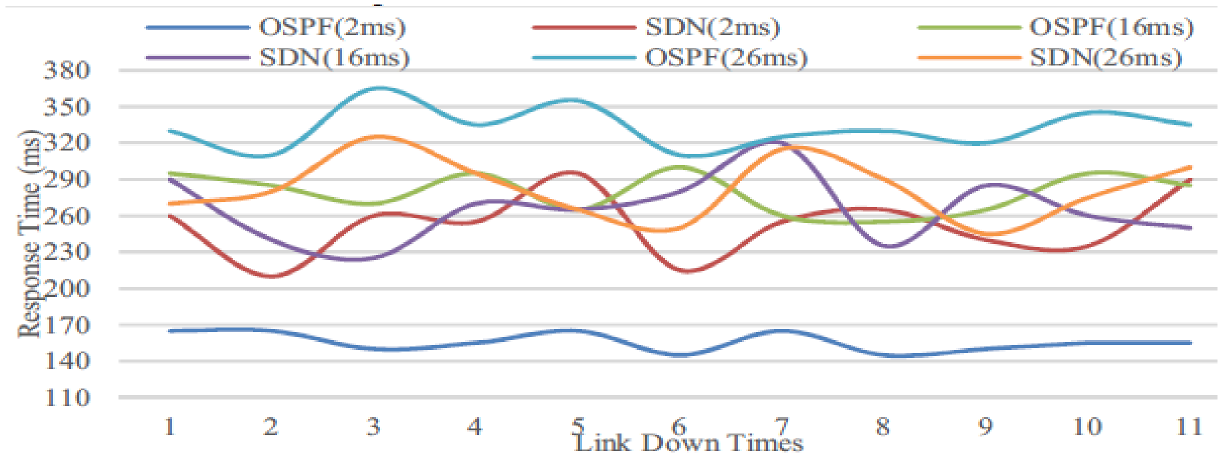


Figure 2. 9 Performance evaluation of convergence in SDN and OSPF (Zhang & Yan, 2015)

According to (Cisco, 2010), its proprietary technology implementation known as Cisco performance routing CPr, reports that its deployment enhances routing performance in the network by implementing master and border controller routers. Master controller has a global view network, and the border routers forward packets to the master router, thus decreasing end-to-end delay levels due to its readily available forwarding information base (FIB).

Hewlett Packard implemented OpenFlow in its suite of switches (Hewlett Packard, 2013) (Tourrilhes et al., 2014), which showed HP OpenFlow implementation was informed by the fact that classical routing cannot address the levels of end-to-end delay in its current network requirements.

In a study by (Jain et al., 2014), while experimenting with partial OpenFlow deployment for Google corporation, they reported that OpenFlow introduced considerable savings in its data centers and improved routing by 20 percent compared to classical approaches used originally. In addition, they cited two key benefits from the report: efficient operation of the WAN and better bandwidth management.

In several studies (Feamster et al., 2004; Jain et al., 2014; ONF, 2018a; Yap et al., 2010; Zhang & Yan, 2015), they reported that OpenFlow was able to improve or lower end to end delays levels in their networks by up to 20 percent or more compared to legacy routing protocols.

In the study by (Jain et al., 2014) from the Google OpenFlow experiment, they concluded that while the results showed promise using SDN routing, its findings could not generalize end-to-end delay performance in all forms of WANs and core networks. Also, the study by (Araniti et al., 2014) stated that its experiment was done in a small wireless network. Therefore, it cannot represent how end-to-end delay would behave in a large mobile IP core network. While CPr routing supports the separation of control and data planes, it is a proprietary approach. Furthermore, this research did not find any reports on how CPr could impact the end-to-end delay in large mobile IP network core (Cisco, 2010). In several studies (Feamster et al., 2004; Jain et al., 2014; ONF, 2018a; Yap et al., 2010; Zhang & Yan, 2015), this study did not find any evidence of an end-to-end delay evaluation in large mobile IP core network.

This review showed that OpenFlow has improved performance compared to legacy routing approaches such as OSPF in end-to-end handling delay. However, the review did not find any evidence of studies that examined the comparative performance of end-to-end delay in OpenFlow and OSPF in the large mobile core network.

2.3.6 Throughput with OpenFlow and OSPF

According to (Balasubramanian, 2006), throughput is defined as the rate at which packets pass through a network medium. Similarly, (Odom, 2017) describes throughput as the speed of a link in bits per second. According to (Rohal et al., 2013), throughput is the total number of packets delivered over the total. The formula for Throughput is computed as follows

$$throughput = N/1000 \ldots \text{ equation 2.4}$$

In equation 2, N refers to the number of bits received in all destinations. With respect to definitions by (Balasubramanian, 2006; Odom, 2017; Rohal et al., 2013), this study defines throughput is defined as the amount of data in mbs that passes through a network link.

Maximum throughput is an important goal of any QoS strategy in network routing. Higher throughput indicates better service and is likely to promote a desirable end-user experience in a network. As stated by (Balasubramanian, 2006), throughput in mobile networks determines the speed at which data or call is delivered to a customer. Several approaches achieve maximum throughput in a network flow (Odom, 2017).

Configuration approaches such as traffic policing and shaping are some of the known methods adopted by most vendors to achieve higher throughput (Feamster et al., 2013; Shahbaz & Feamster, 2015). Such approaches are akin to legacy routing methods like OSPF.

The throughput of less than 318 mbs can negatively impact network traffic and affect end-user experience (Businessdaily, 2021; CAK, 2014; Janevski et al., 2013; Star, 2014; Szigeti et al., 2013). For instance, less throughput can result in a loss of revenue for CSP companies and sometimes attract penalties from regulatory bodies due to unacceptable end-user experience (CAK, 2015; Star, 2014).

According to (Poretsky et al., 2006) and (Janevski et al., 2013), throughput is an important measurement in core networks due to its ability to influence the end-user experience in a network and the value invested. Routing in a large mobile IP core network poses a different challenge than in small wireless and wired networks. Studies by (Araniti et al., 2014; Azodolmolky, 2013; Li et al., 2016; Miller, 2002b; Odom, 2017) showed that classical routing could not adequately tackle challenges presented by (QoS) requirements. Studies by (Azodolmolky, 2013; D ’souza et al., 2016; Hu et al., 2014; Jain et al., 2014) demonstrated that throughput in small wireless and wired network could be improved by adopting robust routing approaches such as OpenFlow.

The research by (Araniti et al., 2014) evaluated OpenFlow and OSPF protocols in a small mobile network in a small wireless network using 30 UEs and one eNodeB. Their findings showed that OpenFlow improved Throughput by 20 percent compared to OSPF routing.

In a study by (Jain et al., 2014), while experimenting with OpenFlow deployment for Google corporation, they reported that OpenFlow introduced considerable savings in its data centers and improved routing by 20 percent compared to classical approaches used originally. In addition, they cited two key benefits from the report: efficient WAN operation and better Throughput.

To improve throughput for CSP companies, they continue investing large sums of money to increase their physical network coverage and data-carrying capacity (CAK, 2018; Nation, 2014; Njanja Annie, 2017).

In several studies (Feamster et al., 2004; Jain et al., 2014; ONF, 2018a; Yap et al., 2010; Zhang & Yan, 2015), they reported that OpenFlow was able to improve throughput levels in their networks by up to 20 percent or more compared to legacy routing protocols.

Configuration approaches give network engineers more control over the network data paths in the mobile core network. However, the benefits of automation and open system, such as in OpenFlow, are lost. The configuration also requires many years of experience and presents a steep learning curve for aspiring network engineers (Casado & Foster, 2013; Feamster et al., 2004, 2013). The study by (Araniti et al., 2014) admits that its experiment was done in a small wireless network and cannot indicate how throughput would behave in a large mobile IP core network. In the study by (Jain et al., 2014) from the Google OpenFlow experiment, they concluded that while the results showed promise using SDN routing, its findings could not generalize throughput performance in all forms of WANs and core networks. Investing in expanding the network infrastructure by CSPs increases the carrier's capacity. However, such an approach increases the CAPEX and does not eliminate the bottleneck of legacy routing architectures. In several studies (Feamster et al., 2004; Jain et al., 2014; ONF, 2018a; Yap et al., 2010; Zhang & Yan, 2015), there was no evidence of comparative performance analysis of OSPF and OpenFlow on throughput in large mobile IP core network.

This review showed that OpenFlow improved throughput performance by 14 percent compared to OSPF. However, the review did not find any evidence of studies that examined the comparative performance of OSPF and OpenFlow routing on throughput large mobile IP network core.

2.8 Knowledge Gap

Based on the above literature review, mobile applications are rapidly being deployed on IP-based protocols. The study also noted that more companies still use classical routing approaches such as OSPF and MPLS in the core network. However, classical routing cannot effectively address the new demand for large mobile IP network applications. It was interesting to note that classical routing environments which bundle CP and DP have been found not to provide a solution to faster routing in large mobile IP core networks. On the other hand, OpenFlow separates CP and DP and has demonstrated promising results in wired and small wireless networks. However, OpenFlow showed desirable results in small wireless and wired networks; it had not been tested in large mobile IP core networks.

2.9 Summary

This chapter reviewed several documents relevant to the study subject and presented an outline of the same. The research showed an overview of routing protocols and how they accomplish their task. Two routing methods were reviewed in detail, namely, OSPF and OpenFlow. The plane architecture and placement affect how one protocol could gain an advantage over another. The study also examined the network architecture of OSPF and OpenFlow and showed that OpenFlow, through its routing approach, performs 50 percent more efficiently than OSPF. The study also showed that CSPs that don’t address challenges of routing QoS routing face regulator penalties and loss of revenue. The study showed the need to address the routing gap in large mobile IP core networks due to the continued use of classical routing approaches.

CHAPTER THREE

RESEARCH METHODOLOGY

This chapter outlines the process and tools used in the study to achieve the research objectives. Section 3.1 describes the conceptual framework, and section 3.2 research design. Sections 3.3 to 3.6 illustrate the research tools, simulation environment approach, and ethical considerations.

3.1 Conceptual Framework

According to (Zhang & Yan, 2015) and (Cassado, 2015), the efficiency of routing protocols between legacy and SDN architectures is influenced using the following characteristics; 1) Centralization versus Distributed, 2) Simplicity versus Complexity, and 3) Expanded versus Limited forwarding headers. Figures 3.1 shows a controller-based network built on the core principle of SDN centralization. Figure 3.2 illustrated a legacy network with distributed routing logic and bundled CP and DP across all routers.

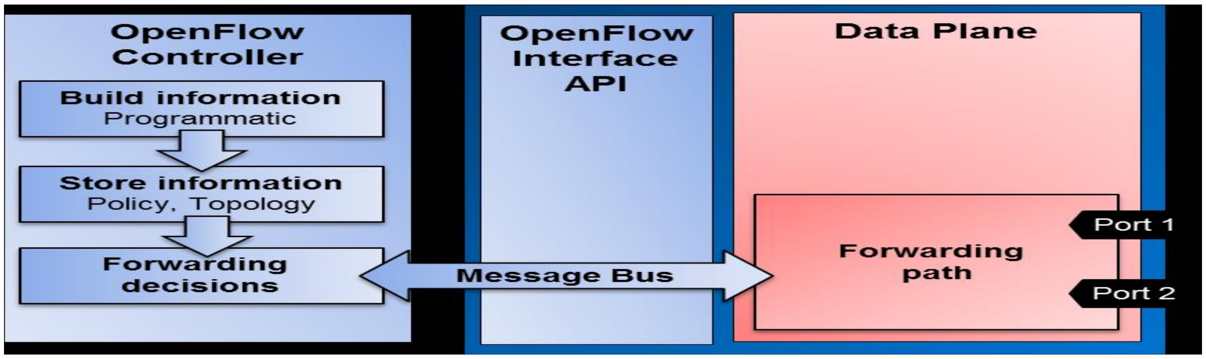


Figure 3. 1 SDN-based network framework (Author, 2020)

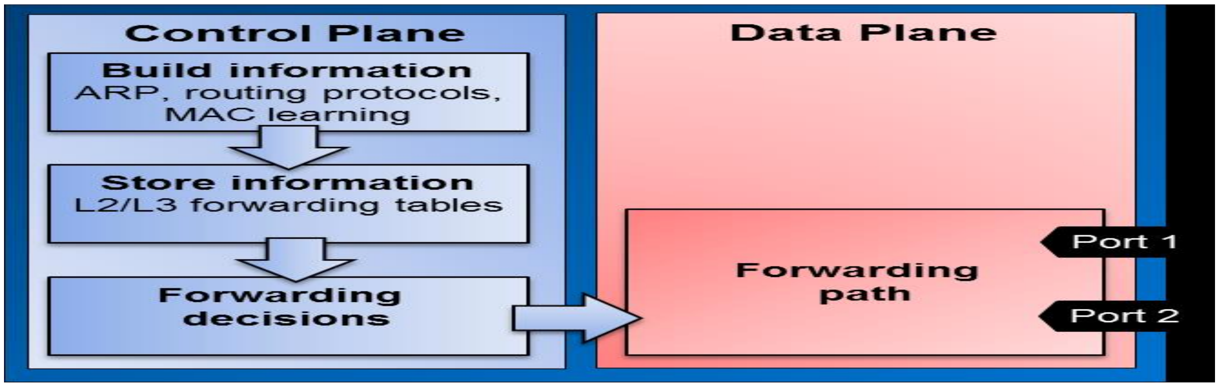


Figure 3. 2 Legacy routed network framework (Author, 2020)

3.2 Research Design

A mixed design method consisting of exploratory and experimental approaches was adopted for this study. The choice for this hybrid method was influenced by two main reasons 1) to test two routing approaches, OSPF and OpenFlow, and the relationship with QoS metrics, and 2) to discover knowledge areas that have not been tested in the large mobile IP core network. Also, other network design components incorporated in this research included UEs, Routers, and End users' applications.

3.3 Research Tools and Materials

The following tools were used to test the objectives in this research, as outlined earlier in chapter 1.

3.3.1 Objective Modular Network Testbed in C++

While there are several emulator software for simulating networks (Werhle, Klaus;Mesut & Left, 2010), this research selected Objective Modular Network Testbed in C++ and simuLTE, both open-source emulators. OMNeT++ and SimuLTE were used to simulate LTE mobile IP core networks to achieve the objectives of this study. Other useful properties that have been observed in OMNeT++ and simuLTE, which were considered by this study for inclusion, are the ability to be extendable to accommodate new features and the availability of SDN platforms such as OpenFlow

Figure 3.3 illustrates OMNeT ++ launch window.

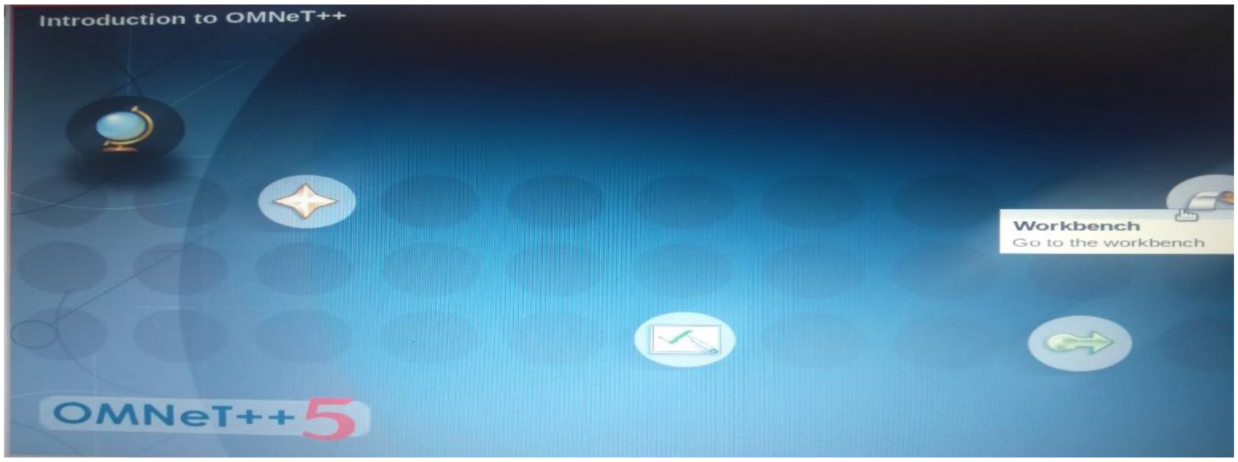


Figure 3. 3 OMNeT ++ workspace launch window (Author, 2020)

3.3.2 Open Shortest Path First

Open Shortest Path First (OSPF) is an open-source routing protocol that uses the Dijkstra algorithm and performs routing tasks by dynamically locating neighboring routers using hello packets. OSPF also does forwarding of traffic using source and destination. Open shortest path first is installed as an add-on package in OMNeT++ inbuilt routers. This makes OSPF preferable when testing of classical routed networks is needed. Figure 3.4 shows the partial OSPF router topology of the core network designed in the experimental setup.

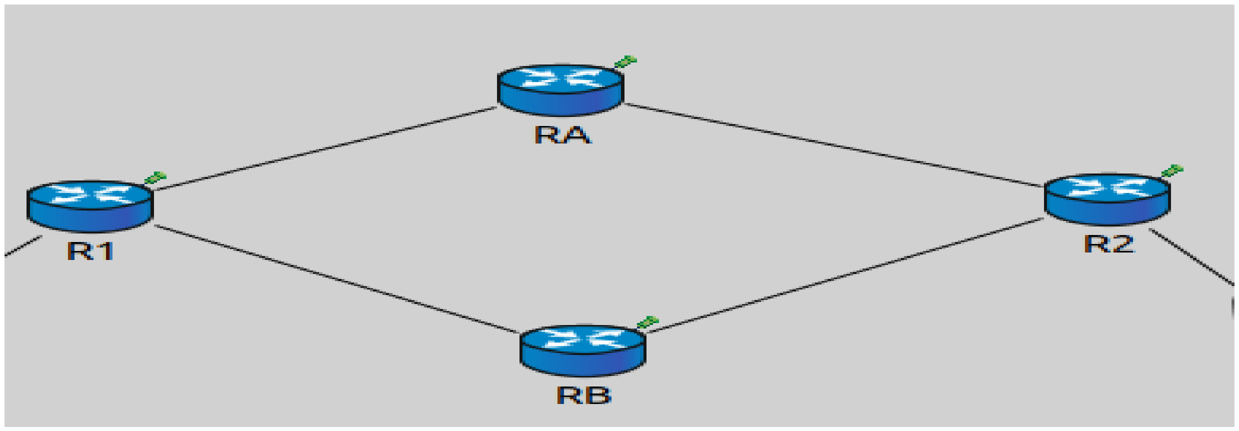


Figure 3. 4 OSPF routers in OMNeT ++ (Author, 2020)

3.3.3 OpenFlow Controller

OpenFlow uses the centralized controller approach to push flow entries to OpenFlow switches. Figure 3.5 shows the OpenFlow controller in the study setup. The OpenFlow controller used in this research is inbuilt in OMNeT ++ and designed using the C++ programming language.

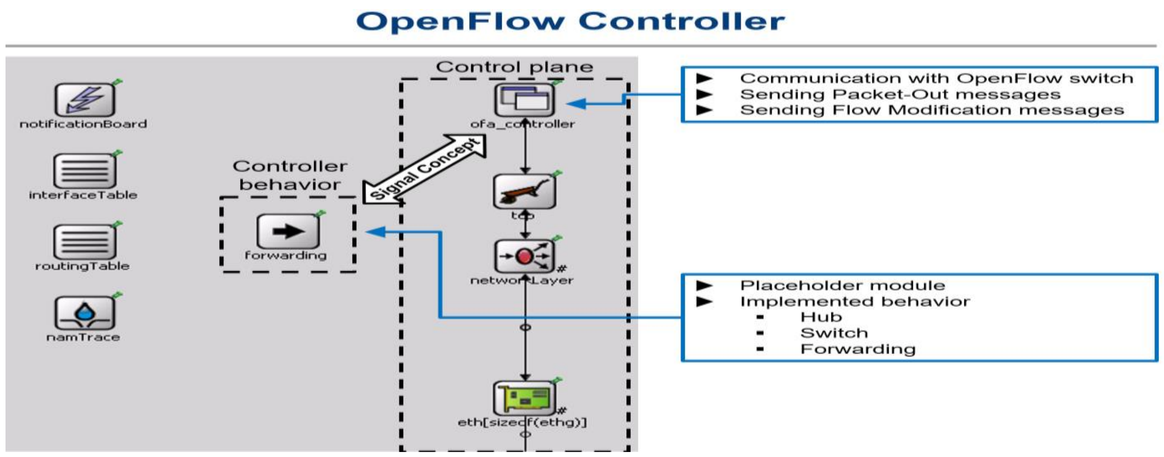
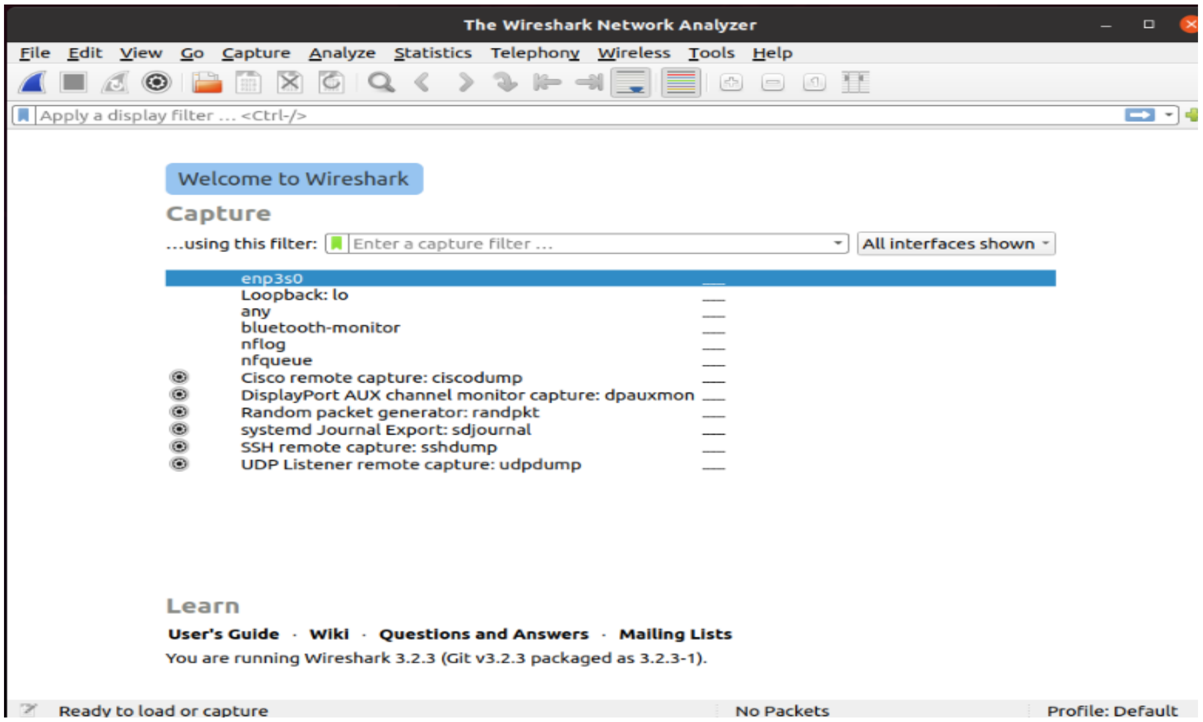


Figure 3. 5 OpenFlow controller (Author, 2020)

3.3.4 Wireshark

Wireshark is open-source software that captures data packets in a network to understand data transport characteristics between two or more nodes (Lamping, Ulf; Sharpe, Richard and Warnicke, 2018). Wireshark was used to analyze traffic and the path taken by IP traffic when traversing the CSP network between UEs, servers, and routers. Figure 3.6 illustrates the Wireshark launch and setup page.


The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Welcome to Wireshark
Capture
…using this filter:
Enter a capture filter …
All interfaces shown
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists

enp3s0
Loopback: lo
any
bluetooth-monitor
nflog
nfqueue
Cisco remote capture: ciscodump
DisplayPort AUX channel monitor capture: dpauxmon
Random packet generator: randpkt
systemd Journal Export: sdjournal
SSH remote capture: sshdump
UDP Listener remote capture: udpdump

You are running Wireshark 3.2.3 (Git v3.2.3 packaged as 3.2.3-1).

Ready to load or capture | No Packets | Profile: Default

Figure 3. 6 Wireshark launch page (Author, 2020)

3.3.5 IP Address space

IPv4 private address block from class A was used to identify devices in the CSP network uniquely. Class A address can provide enough address for a network of over 1000 user IP addresses. Table 3.1 summarizes the actual address design.

Table 3. 1 IP Address Scheme

Subnet	IP Network	Address Range	Broadcast
0	10.0.0.0	10.0.0.1-10.15.255.254	10.15.255.255
1	10.16.0.0	10.16.0.1-10.31.255.254	10.31.255.255
2	10.32.0.0	10.32.0.1-10.47.255.254	10.47.255.255
3	10.48.0.0	10.48.0.1-10.63.255.254	10.63.255.255
4	10.64.0.0	10.64.0.1-10.79.255.254	10.79.255.255

5	10.80.0.0	10.80.0.1-10.95.255.254	10.95.255.255
6	10.96.0.0	10.96.0.1-10.111.255.254	10.111.255.255
7	10.112.0.0	10.112.0.1-10.144.255.254	10.144.255.255
8	10.128.0.0	10.128.0.1-10.143.255.254	10.143.255.255
9	10.144.0.0	10.144.0.1-10.159.255.254	10.159.255.255

Source (Author, 2020)

In Table 3, IP networks from different subnets were subdivided by OSPF routers for classical networks or OpenFlow controllers for the SDN architecture. The IP Address range is the set of IP addresses that can be assigned to nodes, and broadcast is the system address that can communicate between devices and is not assignable to any device. The UEs acquired IP addresses via Dynamic Host Configuration Protocol (DHCP). The DHCP tool automatically assigning IP addresses to UEs is Network Configurator. The network configurator reads the IP plan from an XML file in the network setup before issuing it to UEs, router interfaces, etc.

Each subnet will have 1,048,574 IP addresses. The formula for calculating usable IP addresses and subnets was as per equation 3.1

2^n-2 $2^{20}-2= 1,048,574$… equation 3.1 where n is the number of bits borrowed from the host part of the IP address

3.3.6 Summary of simulation technologies and network standards

Table 3.2 lists the testing parameters and numbers that were used for this experiment

Table 3. 2 Simulation parameters

Parameter	Technology and UEs	Application Ports
Wireless standard	LTE	N/A
Frequency Channel	LTE channel control	N/A
Application 1	Interactive gaming	80
Application 2	Audio streaming	5000
Application 3	VoiP	3389
Application 4	IPTV	4000

Simulation run time in minutes	60,80,100,80,120	N/A
Number of UEs	200,400,600,800,1000	N/A
Number of Routers/OpenFlow	20,40,60,80	N/A

The LTE technology in simuLTE and its associated modules, such as the eNodeB and Frequency channel provided in this study, was designed to mirror the real production RAN closely. The four most used applications in the CSP networks and inbuilt in simuLTE network were chosen as shown in table 3.2. According to (Odom, 2017), large networks are classified as having 50 routers or more. Because the main objective of this study was to test the performance of OSPF and OpenFlow routing in a large mobile IP core network, this research was able to deploy up to 80 routers at peak maximum. The choice of the number of routers ensured that the simulated CSP network could generate enough data for analysis to fulfill this study's objectives.

3.4 Design and Setup of the Simulation Environment

3.4.1 Simulation process overview

The simulation process is illustrated in Figure 3.7 for OSPF and OpenFlow protocols.

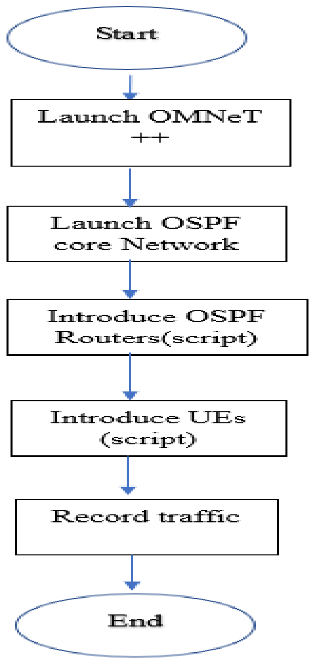


Figure 3. 7 OSPF testing process (Author, 2020)

3.4.2 Experiment Setup and Design

This study created two LTE-like environments in the OMNeT++ simulator and OpenFlow and simuLTE add-ons. The comparison network was constructed using OSPFv2, which is compatible with IPv4. The simuLTE and OpenFlow plugins enabled LTE and OpenFlow systems, respectively. The simulation experiment was set up on Ubuntu 18.04 operating system with OMNeT++ 5.5.1 and INET 3.6, simuLTE 0.9.1, and OpenFlow 1.3.4. Figure 3.9 shows the OMNeT++ IDE for orchestrating the simulation.

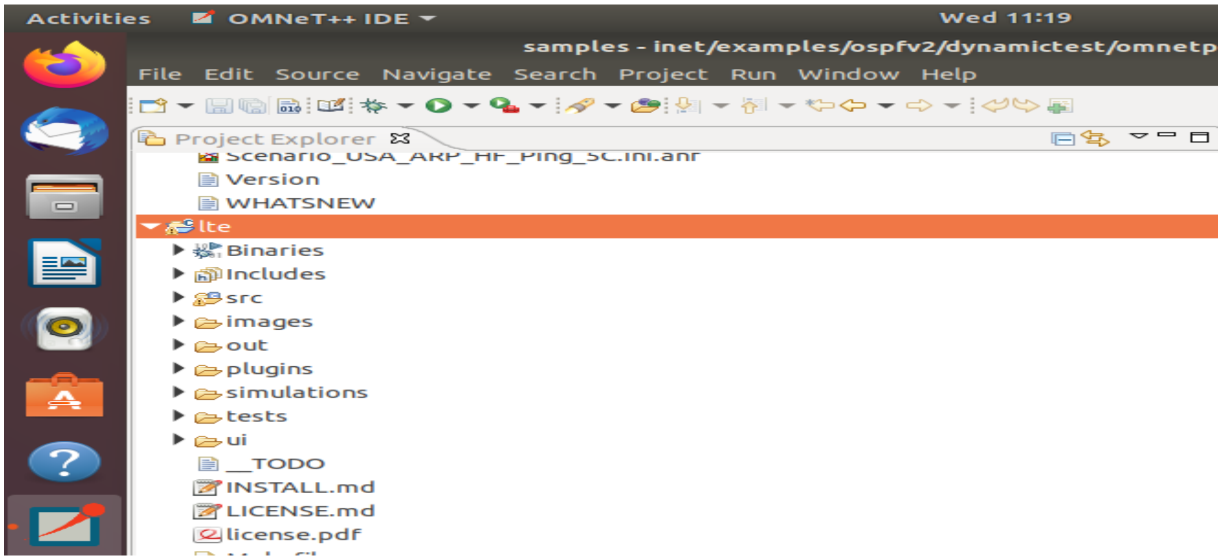


Figure 3. 8 OMNeT++ Application launch page (Author, 2020)

Two test network architecture was created based on the classical network running OSPF in the core and the SDN network running OpenFlow. Figure 3.10 illustrates the OSPF network, and Figure 3.11 controller-based network setup running OpenFlow.

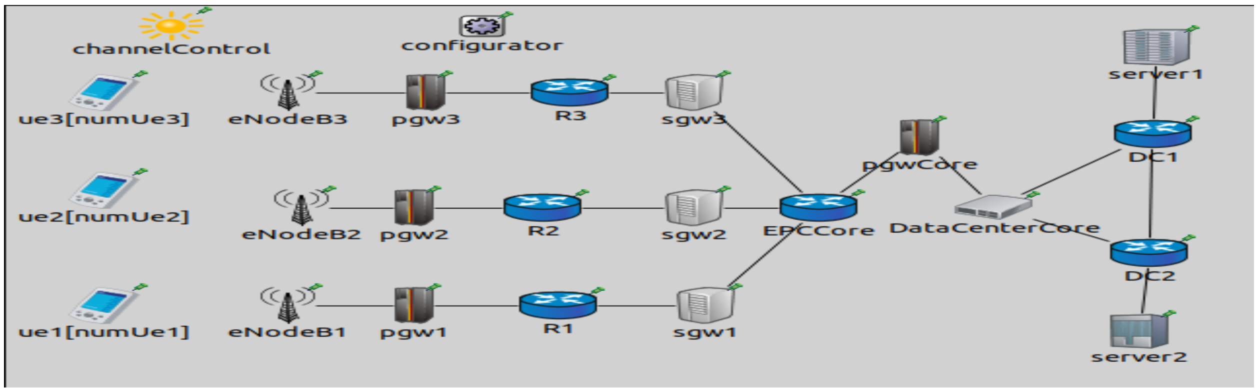


Figure 3. 9 OSPF core network (Author, 2020)

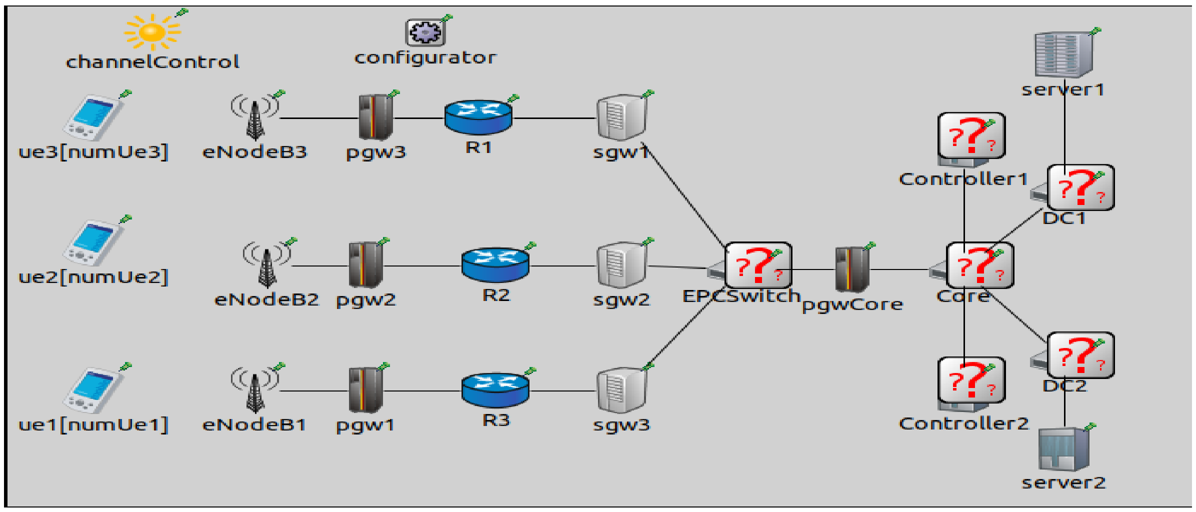


Figure 3. 10 OpenFlow core network (Author, 2020)

The modules deployed in the OSPF core network were; UEs, packet gateway filter (PGW), and application servers hosted on the internet. The OpenFlow core network UEs, PGW, HyperFlow, OpenFlow controller, and OpenFlow switches were added to the core network. The specific modules used and their data structures are briefly outlined in the list (1) to (3).

1. User Equipment

UE emulates a smartphone or any mobile device used in the CSP network. It supports user datagram (UDP) and transport control protocols (TCP). Figure 3.12 illustrates UEs structure running UDP and TCP.

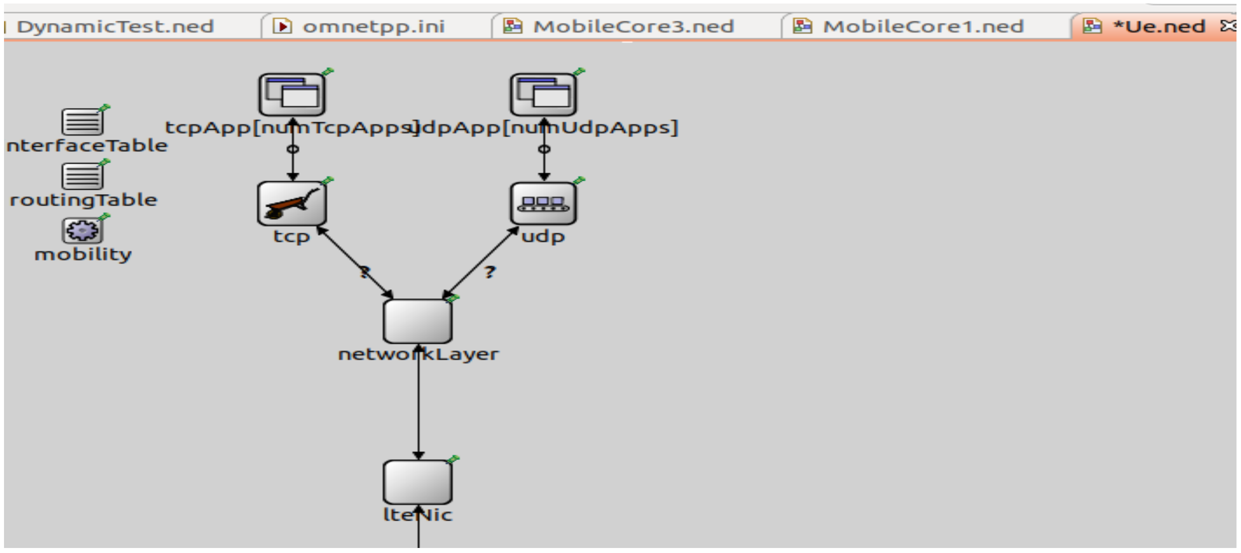


Figure 3. 11 UE with TCP and UDP protocols (Author, 2020)

2. OSPF Router

Routers in the OMNeT++ framework are configured with OSPF protocol and connected to the data center. Routers build the network topology using the Dijkstra algorithm until all the routers achieve convergence. Figures 3.13 shows the OSPF router's internal structure.

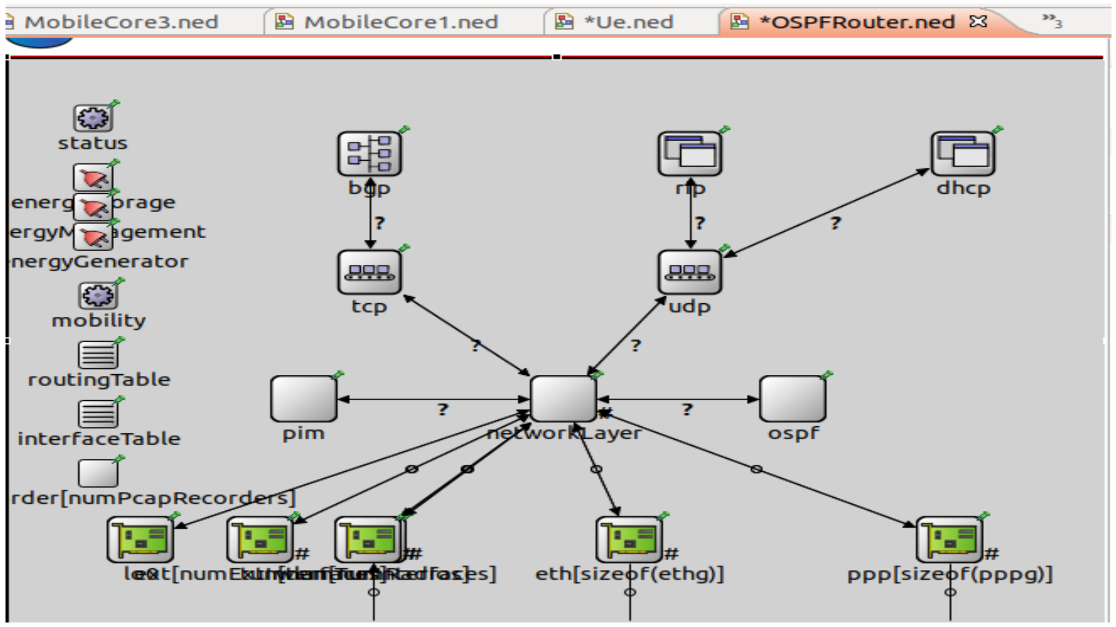


Figure 3. 12 OSPF Router Internal Structure (Author, 2020)

3. OpenFlow controller

The controller is a POX-based OpenFlow protocol and is readily available as an add-on in OMNeT++. Figure 3.14 shows the structure of the OpenFlow controller.

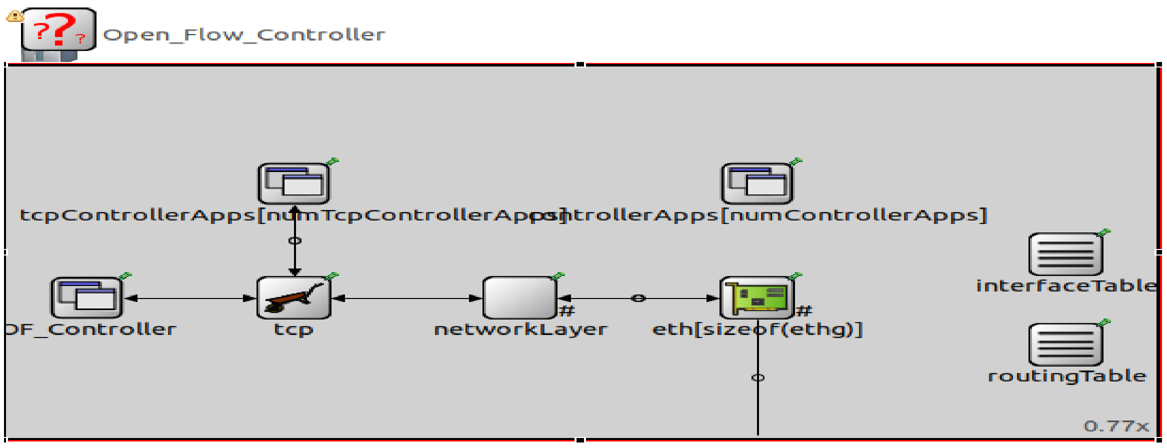


Figure 3. 13 OpenFlow Controller (Author, 2020)

4. OpenFlow switch

Figure 3.15 shows the OpenFlow switch with modified DP and CP planes to accept flows from the OpenFlow controller.

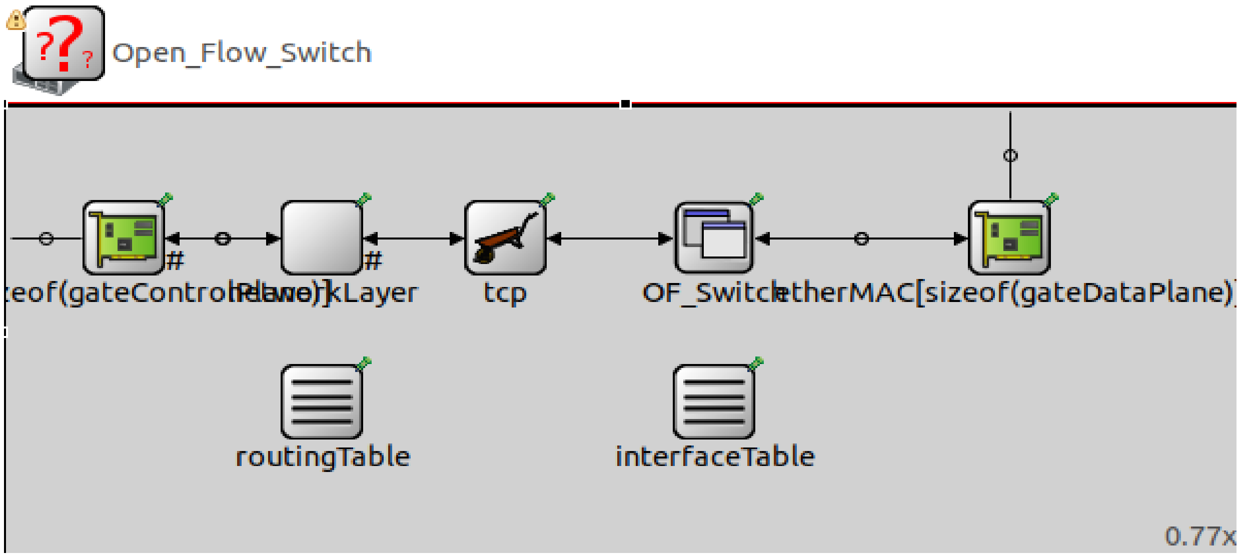
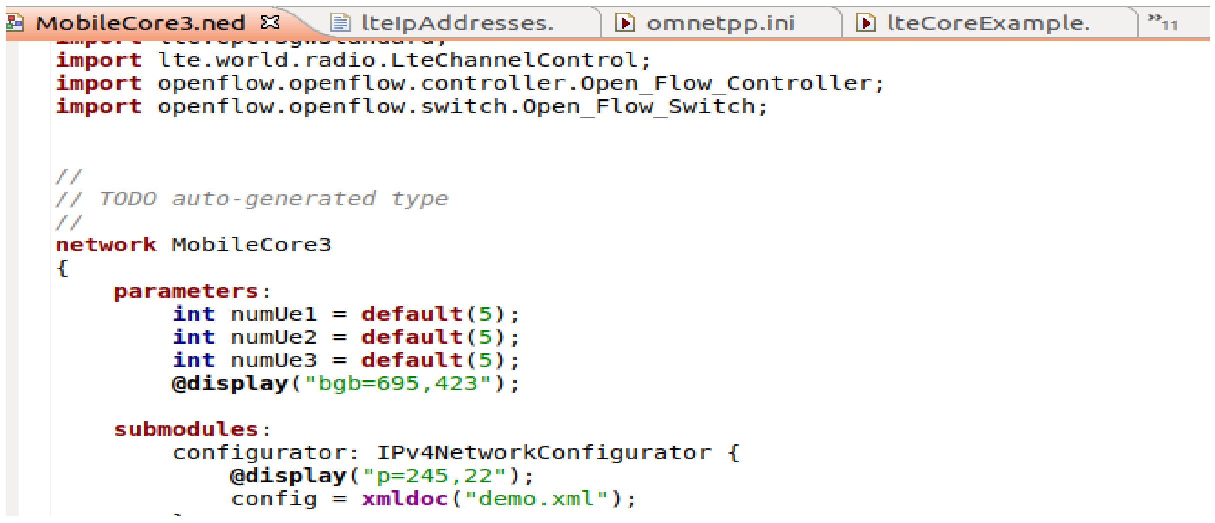


Figure 3. 14 OpenFlow switch data structure (Author, 2020)

5. Network configurator

The IP configurator is shown in figure 3.16.



```
import lte.world.radio.LteChannelControl;
import openflow.openflow.controller.Open_Flow_Controller;
import openflow.openflow.switch.Open_Flow_Switch;


//
// TODO auto-generated type
//
network MobileCore3
{
    parameters:
        int numUe1 = default(5);
        int numUe2 = default(5);
        int numUe3 = default(5);
        @display("bgb=695,423");

    submodules:
        configurator: IPv4NetworkConfigurator {
            @display("p=245,22");
            config = xmldoc("demo.xml");
```

Figure 3. 15 Omnetpp.ini IP configurator XML script (Author, 2020)

3.4.3 OSPF Network Operation

When the UEs come on, they are connected to the network via eNodeB RAN. Figure 3.17 shows the initial process of UEs relating to the RAN before they can start sending information into the network.

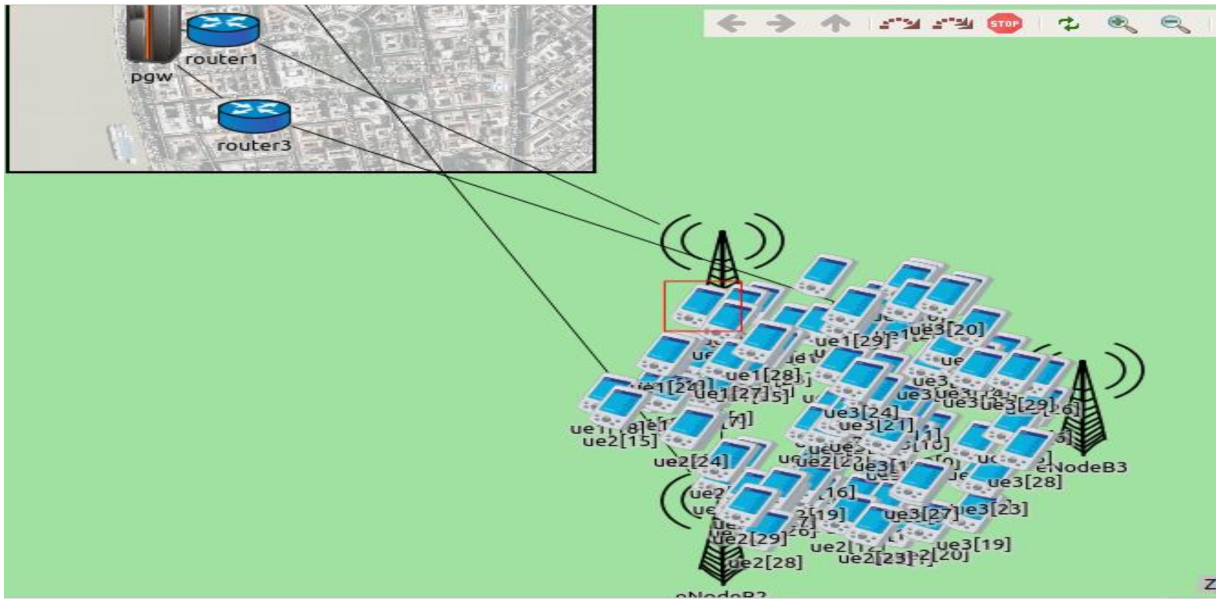
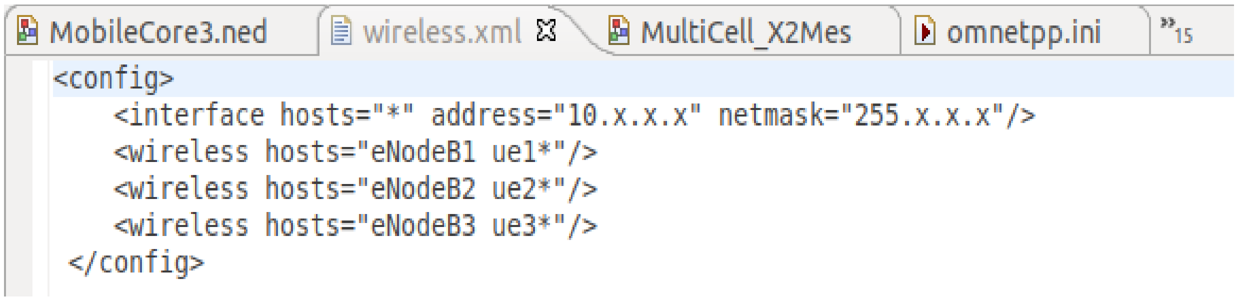


Figure 3. 16 Partial capture of UEs connecting to the RAN in the OSPF network (Author, 2020)

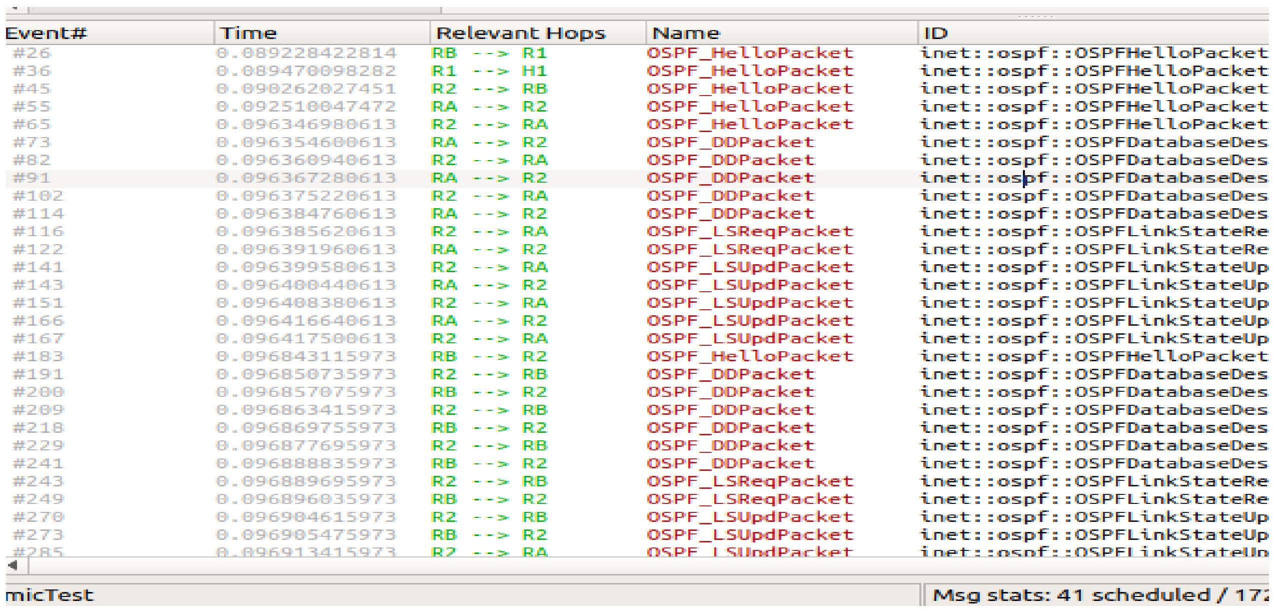
The eNodeB is also placed in different networks from the omnetpp.ini file, and this ensures that there is no overlapping of frequency used among the UEs. Figure 3.18 illustrates the ometpp.ini configuration used to assign UEs to respective eNodeB.



```
<config>
    <interface hosts="*" address="10.x.x.x" netmask="255.x.x.x"/>
    <wireless hosts="eNodeB1 ue1*"/>
    <wireless hosts="eNodeB2 ue2*"/>
    <wireless hosts="eNodeB3 ue3*"/>
 </config>
```

Figure 3. 17 eNodeB Network assignment script (Author, 2020)

The OSPF routers initialized communication through the five main steps, namely 1) Hello discovery, 2) Sharing of link details, 3) Exchange of LSA information, 4) and 5) Neighborship and sharing of routing of traffic. This OSPF process indicates that the CP and DP are bundled into the same hardware as this study highlighted earlier in the conceptual framework. Figure 3.19 (Author, 2020) shows the OSPF router discovery and neighborship process, which leads to network convergence.



```
Event#      Time               Relevant Hops   Name                 ID
#26         0.089228422814     RB  --> R1      OSPF_HelloPacket     inet::ospf::OSPFHelloPacket
#36         0.089470098282     R1  --> H1      OSPF_HelloPacket     inet::ospf::OSPFHelloPacket
#45         0.090262027451     R2  --> RB      OSPF_HelloPacket     inet::ospf::OSPFHelloPacket
#55         0.092510047472     RA  --> R2      OSPF_HelloPacket     inet::ospf::OSPFHelloPacket
#65         0.096346980613     R2  --> RA      OSPF_HelloPacket     inet::ospf::OSPFHelloPacket
#73         0.096354600613     RA  --> R2      OSPF_DDPacket        inet::ospf::OSPFDatabaseDes
#82         0.096360940613     R2  --> RA      OSPF_DDPacket        inet::ospf::OSPFDatabaseDes
#91         0.096367280613     RA  --> R2      OSPF_DDPacket        inet::ospf::OSPFDatabaseDes
#102        0.096375220613     R2  --> RA      OSPF_DDPacket        inet::ospf::OSPFDatabaseDes
#114        0.096384760613     RA  --> R2      OSPF_DDPacket        inet::ospf::OSPFDatabaseDes
#116        0.096385620613     R2  --> RA      OSPF_LSReqPacket     inet::ospf::OSPFLinkStateRe
#122        0.096391960613     RA  --> R2      OSPF_LSReqPacket     inet::ospf::OSPFLinkStateRe
#141        0.096399580613     R2  --> RA      OSPF_LSUpdPacket     inet::ospf::OSPFLinkStateUp
#143        0.096400440613     RA  --> R2      OSPF_LSUpdPacket     inet::ospf::OSPFLinkStateUp
#151        0.096408380613     R2  --> RA      OSPF_LSUpdPacket     inet::ospf::OSPFLinkStateUp
#166        0.096416640613     RA  --> R2      OSPF_LSUpdPacket     inet::ospf::OSPFLinkStateUp
#167        0.096417500613     R2  --> RA      OSPF_LSUpdPacket     inet::ospf::OSPFLinkStateUp
#183        0.096843115973     RB  --> R2      OSPF_HelloPacket     inet::ospf::OSPFHelloPacket
#191        0.096850735973     R2  --> RB      OSPF_DDPacket        inet::ospf::OSPFDatabaseDes
#200        0.096857075973     RB  --> R2      OSPF_DDPacket        inet::ospf::OSPFDatabaseDes
#209        0.096863415973     R2  --> RB      OSPF_DDPacket        inet::ospf::OSPFDatabaseDes
#218        0.096869755973     RB  --> R2      OSPF_DDPacket        inet::ospf::OSPFDatabaseDes
#229        0.096877695973     R2  --> RB      OSPF_DDPacket        inet::ospf::OSPFDatabaseDes
#241        0.096888835973     RB  --> R2      OSPF_DDPacket        inet::ospf::OSPFDatabaseDes
#243        0.096889695973     R2  --> RB      OSPF_LSReqPacket     inet::ospf::OSPFLinkStateRe
#249        0.096896035973     RB  --> R2      OSPF_LSReqPacket     inet::ospf::OSPFLinkStateRe
#270        0.096904615973     R2  --> RB      OSPF_LSUpdPacket     inet::ospf::OSPFLinkStateUp
#273        0.096905475973     RB  --> R2      OSPF_LSUpdPacket     inet::ospf::OSPFLinkStateUp
#285        0.096913415973     R2  --> RA      OSPF_LSUpdPacket     inet::ospf::OSPFLinkStateUp
micTest                                                    Msg stats: 41 scheduled / 17
```

Figure 3. 18 OSPF router discovery and convergence process (Author, 2020)

Once the CSP network has converged, the UEs send IP packets with the destination address of internet servers hosting the requested application, just as it happens in a real network. The sending of packets IP selection is balanced through a round-robin algorithm inbuilt in OMNeT++ (OMNeT++, 2019). The IP data packets are distinguished using port numbers summarized in table 3.3. The port numbers ensure that the right application is received on the correct port from requesting device.

Table 3. 3 Applications and port numbers on the server

Application	Port number
Interactive gaming	80
Audio streaming	5000
VoiP	3389
IPTV	4000

Source (Author, 2020)

When the packet arrives at the CSP core network, the OSPF router encapsulates the IP packet and sends it to the next-hop address until it reaches the application server. The IP header and service requested are sent back to the UE using the randomly generated port number source IP received in the server request as it happens in a real network. After the bidirectional exchange of information between UEs and servers is completed, QoS metrics are recorded continuously using the OMNeT++ IDE. The data was then exported using the Panda tool for further analysis.

3.4.4 OpenFlow Network Operation

OpenFlow uses a controller-based approach, which implies a centralized controller with the CP function (Feamster et al., 2013; Klein & Jarschel, 2013; Salih, Cosmas, & Zhang, 2015). The DP function is transferred to the switches, which rely on the controller's flow entries pushed to the CAM tables. Figure 3.20 shows the initial connection between the UEs and RAN.

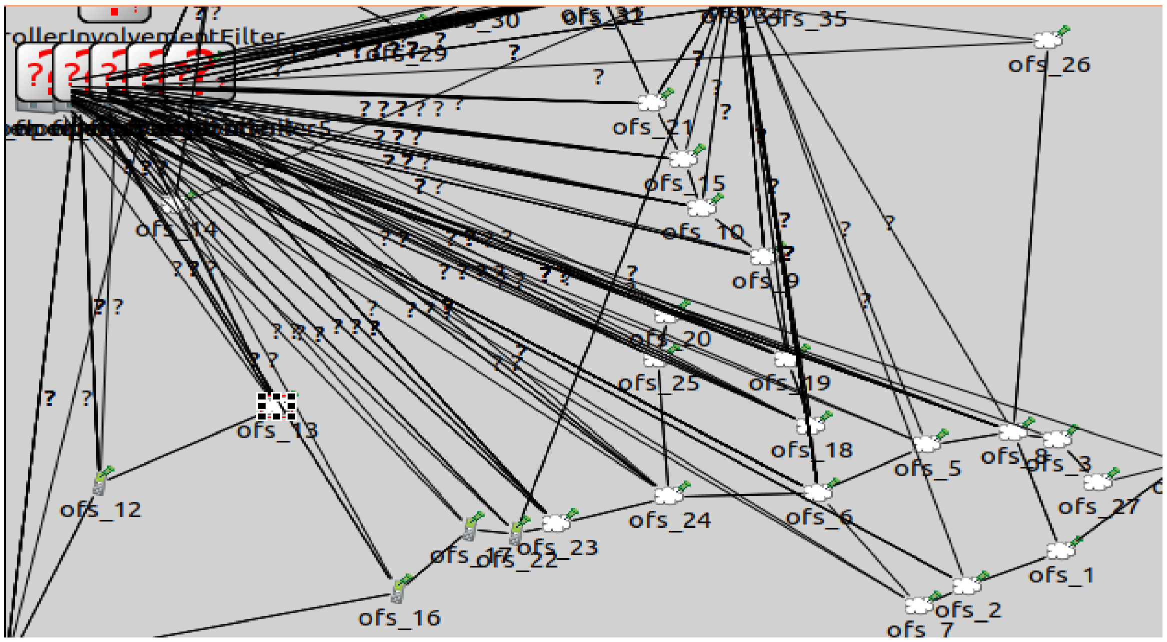
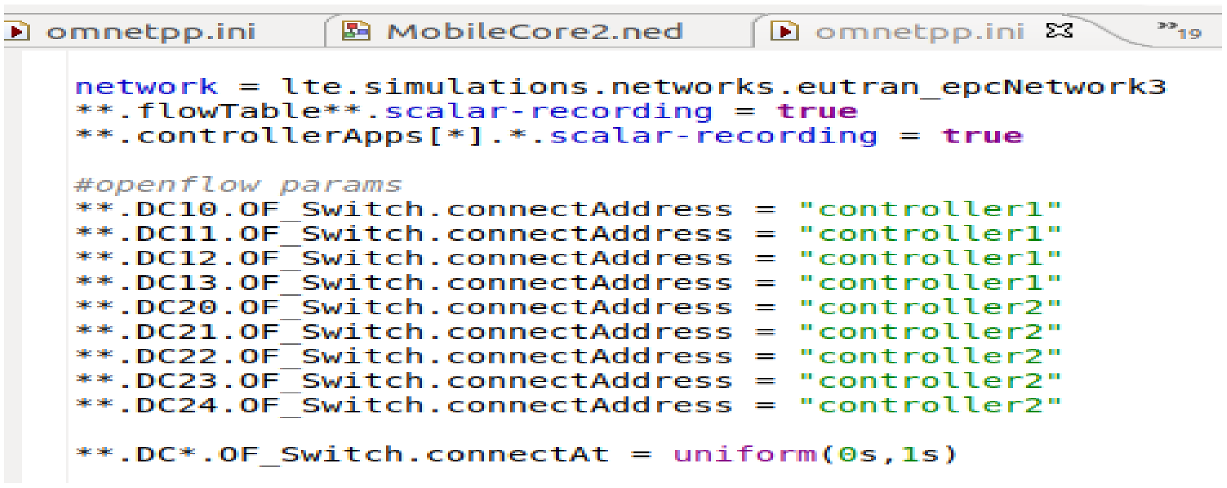


Figure 3. 19 UEs connecting to RAN in OpenFlow network (Author, 2020)

At the onset of the network operation, OpenFlow switches sent packets received to the centralized OpenFlow controller. The controller then pushed the flow entries to the switches, as summarized in figure 3.21. Subsequent packets are forwarded using flow information and simple queries in the switch CAM tables without continuous reference to the controller.



```
network = lte.simulations.networks.eutran_epcNetwork3
**.flowTable**.scalar-recording = true
**.controllerApps[*].*.scalar-recording = true

#openflow params
**.DC10.OF_Switch.connectAddress = "controller1"
**.DC11.OF_Switch.connectAddress = "controller1"
**.DC12.OF_Switch.connectAddress = "controller1"
**.DC13.OF_Switch.connectAddress = "controller1"
**.DC20.OF_Switch.connectAddress = "controller2"
**.DC21.OF_Switch.connectAddress = "controller2"
**.DC22.OF_Switch.connectAddress = "controller2"
**.DC23.OF_Switch.connectAddress = "controller2"
**.DC24.OF_Switch.connectAddress = "controller2"

**.DC*.OF_Switch.connectAt = uniform(0s,1s)
```

Figure 3. 20 Controller and switch association script (Author, 2020)

3.5 Population, Sample, and Sampling Procedure

The research focused on routers and switches running OSPF and OpenFlow protocols. The simulation environment selected for this study was OMNeT ++ due to its scalability and open design nature. Purposive sampling was used to integrate the entire 1000 UEs and 80 routers permissible in the simulation environment. The uniformity largely influenced the choice for this sampling method in the UEs, routers, switches, and the simulation environment. In addition, this represented a large mobile IP network. Applications used include video, audio streaming, IPTV, and VoiP, commonly deployed in mobile IP core networks.

3.6 Data Capture and Analysis

Data was captured in OMNeT++ in units of ms for Jitter and End to End delay. Throughput and PDR were recorded in mbs. Data was aggregated in groups of UEs, routers, and OpenFlow switches, as shown in table 3.3. The data was then copied into excel sheets for further analysis and descriptive statistics like graphs. With the help of the Panda tool, specific data for jitter, throughput, end-to-end delay, and PDR was extracted from the raw data in excel files. The CVS files from OMNeT ++ IDE were copied into Wireshark, and the data was analyzed for path tracing by UEs while traversing the network.

Data from the simulator was collated, and computation of mean values was done for QoS metrics. The mean values were summarized in graphs, as discussed later in chapter 4. To ascertain whether there was any significant difference in the QoS metrics collected, a stdev test was done on the values and the results recorded.

3.7 Ethical Considerations

The simulation was mainly carried out using OMNET++, simuLTE, OpenFlow, and OSPF, fully licensed under GPL (Andras, 2016; Klein & Jarschel, 2013; Nardini et al., 2021).

In places the study referred to other publications, full reference and acknowledgment have been done to recognize the work of the original owners of the cited publications. Sections of this study were presented at the OMNeT++ summit by the researcher (OMNeT++, 2020). An extension of this work was also published in the researcher's peer-reviewed journal mecspress (IJNICS, 2021). The approval letter to research in Kenya is also shown in Appendix 1.

3.8 Summary

This chapter discussed a framework, showing the two approaches used in OSPF and OpenFlow networks and the subsequent experimental setup. The experimental approach was explained in detail. The data components and structures and their application in the experiment were also outlined. Finally, an overview of the ethical considerations undertaken to ensure this study's integrity is presented.

CHAPTER FOUR

RESULTS AND DISCUSSIONS

This chapter presents the findings that were generated from the simulation campaign. Section 4.1.0 to 4.5.4.1.3 presents an overview of architectural design in routing protocols and how they impact routing efficiency, the results, and discussions from the simulation setup. Section 4.6 summarizes the results and discussions.

4.1 Overview of architectural design in Routing algorithms and Efficiency

As stated in chapters 2 and 3, the efficiency of routing algorithms are evaluated based on the following characteristics 1) centralized versus distributed architecture, 2) simplicity versus complexity, 3) expanded forwarding criteria versus limited (Hewlett Packard, 2013; Odom, 2017; Tourrilhes et al., 2014; Zhang & Yan, 2015). These key factors influence the efficiency of any specific routing approach.

4.2 Performance of OpenFlow and OSPF routing on Jitter in large mobile IP core network.

4.2.1 Jitter and Routers

In the first objective, the amount of jitter generated was tested using OpenFlow and OSPF routers added into the core network, as stated in chapter 3. The results in figure 4.1 show that OpenFlow routed traffic with an average jitter value of 25 ms for the first 20 to 60 routers and 31 ms for a similar number of OSPF routers. When 20 more routers were added to attain 80, jitter levels increased to an average of 35 ms for OpenFlow and 40 ms for the OSPF protocol. This increase in jitter values is expected when traffic travels over a longer path in the core network.

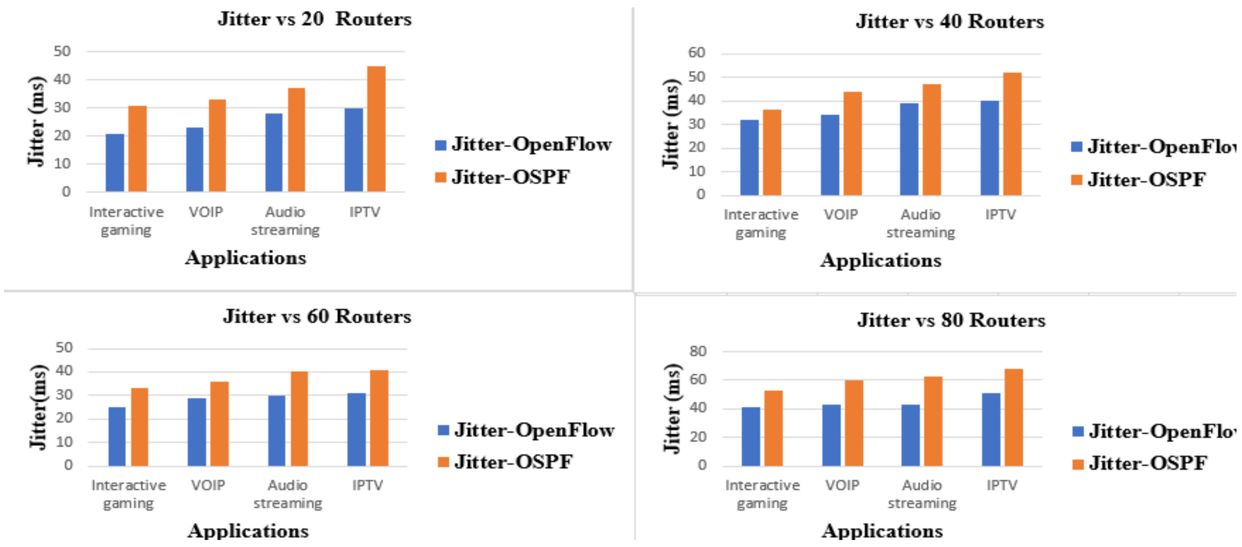


Figure 4. 1 Jitter vs Routers (Author, 2020)

4.2.2 Jitter and User Equipment

Figure 4.2, jitter was tested against UEs in increasing order until the experimental objective of 1000 UEs was achieved. For the first 200 UEs that sent bidirectional traffic on the network, OpenFlow registered average jitter values of 21 ms while OSPF was 31ms.

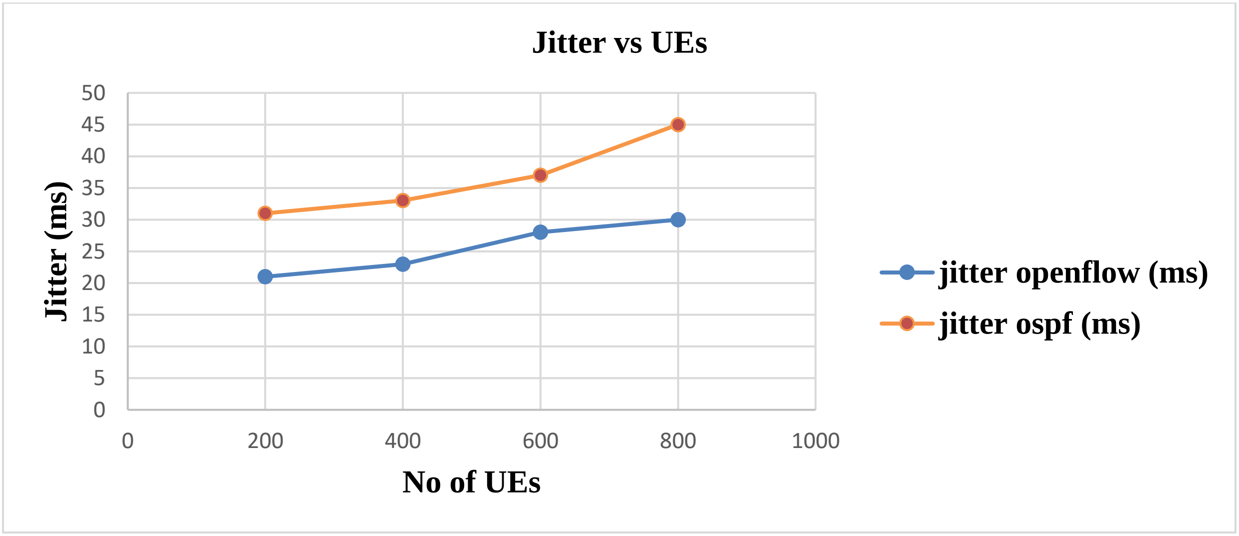


Figure 4. 2 Jitter vs UES (Author, 2020)

As more UEs were added to the network, jitter values generated for the two protocols increased. According to figure 4.3, a trend is established; whereas more UEs are added to the core network, the jitter levels in the network also increase. For instance, with 800 UEs, OSPF jitter increased to an average of 40 ms while OpenFlow jitter was 30 ms.

4.2.3 Jitter and Application

In figure 4.3, jitter results from traffic generated by UEs are shown. Applications generate different jitter values, as seen in figure 4.3. Generally, IPTV has higher jitter at 45 ms and 30 ms for OSPF and OpenFlow, respectively. Interactive gaming has less jitter for OpenFlow at 20 ms and 31ms for OSPF when testing 1000 UEs in the core network. Internet Protocol for Television traffic registered an average jitter of 30 ms for OpenFlow and 45 ms for OSPF for the same number of UEs.

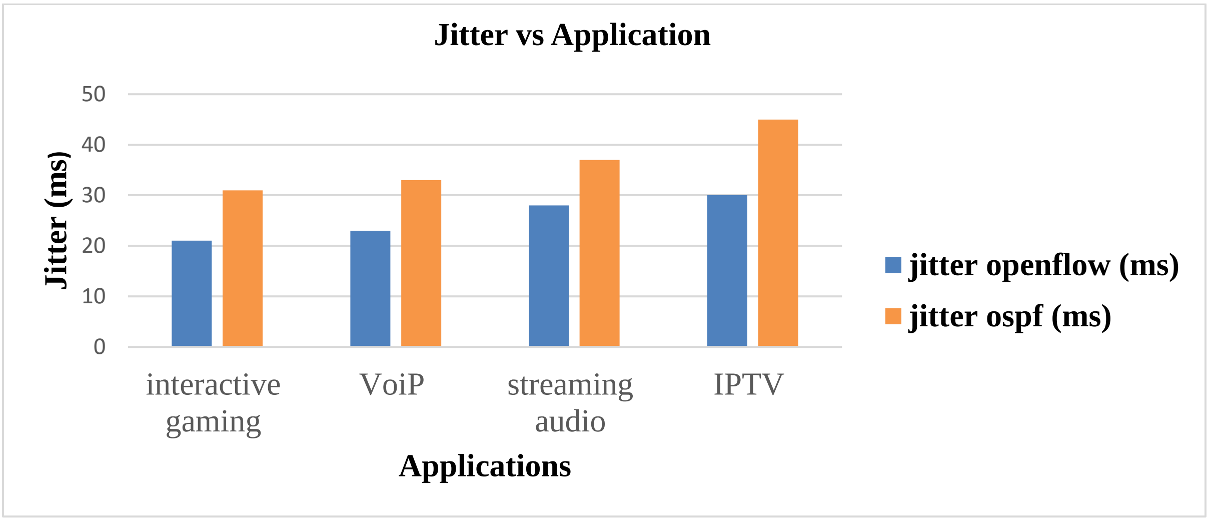


Figure 4. 3 Jitter vs Application (Author, 2020)

From the results in figure 4.3, a similar trend in section 4.2 shows that as more UEs are added to the network, jitter values increase for both OSPF and OpenFlow.

4.2.4 Performance Analysis of Jitter versus OpenFlow and OSPF results

4.2.4.1 Specific discussions on Routers, UE, and Applications performance

According to (Szigeti et al., 2013), the optimum jitter value for bidirectional communication is 30 ms or less. As stated earlier in chapters 1 and 2, less jitter guarantees less loss of information and improved clarity on the communicating nodes (CAK, 2014; Janevski et al., 2013).

In figures 4.1 to 4.3, jitter values for a large number of routers, UEs, and applications show that OpenFlow generates less jitter at an average of 29 ms compared to OSPF 38 .6 ms. Further analysis to establish the significance of the difference in jitter values for the three dependent variables, UE, routers, and application against OpenFlow and OSPF, showed a std of 8.30 ms, indicating a significant difference in the jitter values. Therefore, this study confirms that the OpenFlow protocol generates less jitter in large networks compared to OSPF. Figure 4.4 provides a summary of network components versus routing protocols in use.

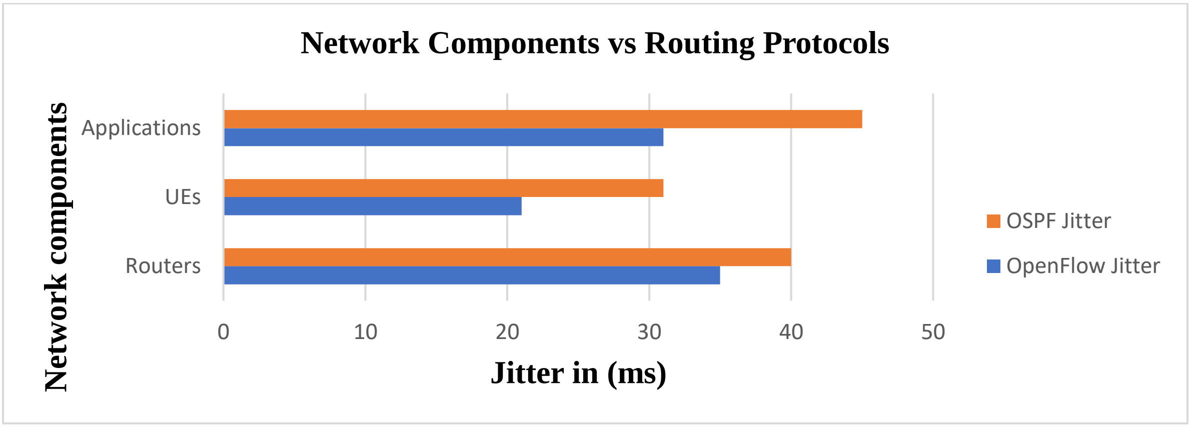
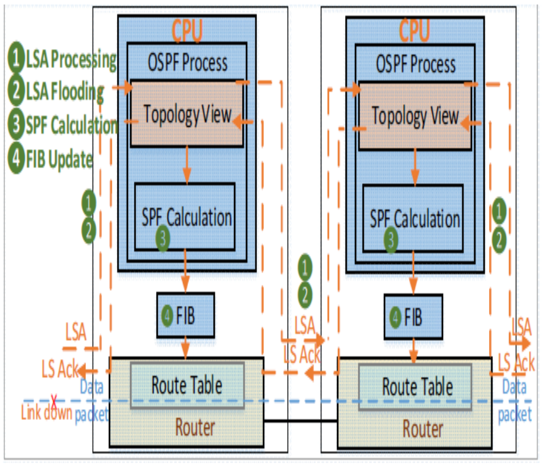


Figure 4. 4 Network components in a large network and routing protocols (Author, 2020)

The research can further deduce that the architecture of the routing algorithm stated in section 4.1 impacts jitter levels in OSPF and OpenFlow.

4.2.4.2 Centralized versus distributed architecture

Each packet must be queried after routing has converged before being forwarded to the next-hop destination based on the distributed architecture in the OSPF network. This process is known as encapsulation, which implies removing and adding headers in the packets for delivery to the next destination. With the expansion of the network size, as expected, the OSPF algorithm routing table also expands, resulting in increased jitter in the forwarding of IP traffic. Figures 4.5 and 4.6 (Author, 2020) illustrate the routing table population process of OSPF and OpenFlow controller flow entries, a conceptual design from this experimental study. OpenFlow has fewer steps to converge at four compared to OSPF's eight stages.



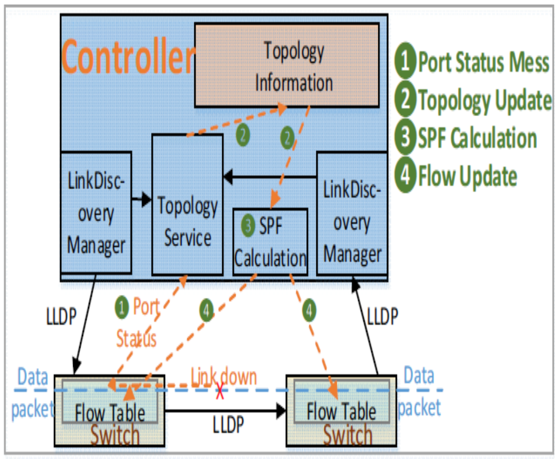


Figure 4. 5 OSPF convergence (Author, 2020) **Figure 4. 6 OpenFlow convergence** (Author, 2020)

OpenFlow, at the onset of the network operation, the data packets received by the switch are forwarded to the centralized controller. Then the controller pushes flow entries to the switches, as summarized in figure 4.6. Subsequent packets are now forwarded using simple query flow tables in the switches. This OpenFlow behavior removes the need for switches to periodically reference the controller except when a new change is detected in the network paths. This limiting or removal of excessive control packets being flooded in the network periodically is a feature that has been noted to improve routing efficiency in the EIGRP algorithm (Cisco, 2016b; Doyle, 2016; Savage, Moore, Slice, Paluch, & White, 2017).

4.2.4.3 Simplicity versus Complexity

Similar to other link-state routing algorithms, OSPF uses different types of OSPF packets to inform adjacent devices on the sequence of events such as loss of connection, availability of new routes, or announcement of new link status (Doyle, 2016; Moy, 1999). Table 4.1 summarizes the OSPF control packets used to create, detect and sustain neighborship between routers.

Table 4. 1 OSPF control packets

Types Packet	Function
Hello	Neighbor discovery
Database Description	Verify LSDB between routers
Link state request	Request specific link details
Link state Update	Send specific link-state updates
Link State Acknowledgment	Implement reliability in OSPF

The exchange of link-state packets in OSPF can improve robustness and remove the single point of failure (SPOF). However, studies by (Doyle, 2016; Feamster et al., 2004; Shaikh et al., 2002; Zhang & Yan, 2015) show that the extensive OSPF packets add to network complexity, which impacts QoS in overall network performance. OpenFlow addresses complexity problems through simple flow entries pushed out to forwarding devices such as routers and firewalls (Braun & Menth, 2014; Casado & Foster, 2013; Cassado, 2015; Feamster et al., 2013; Moy, 1999). According to (Feamster et al., 2004) and (Miller, 2002a), the additional role of OSPF routers in performing route computation and sharing of links state packets makes the legacy approach prone to inconsistencies, especially in a large routing domain. Instead of several link-state packets, OpenFlow uses flow entries to simplify the forwarding process (Hewlett Packard, 2013). A detailed analysis summarized in figure 4.4 of the study experiment logs shows that the complexity of OSPF affects routing speed as it consumes extra bandwidth and slows down the packet forwarding process. OpenFlow uses less bandwidth after the initial exchange of flow entries between the controller and switches.

4.2.4.4 Expanded versus Limited forwarding headers

In OSPF routing, packets are primarily forwarded using source IP and destination IP. The routing process also depends on the routing information base (RIB) and forwarding information base (FIB), distributed tables containing the memory of routes known to the routers and learned from adjacent routers. This action reduces routing efficiency due to the underutilization of modern routers, like high processing speeds and increased memory (Odom, 2017). Limiting path choice in mesh or star network to destination IP implies that routers are restricted to specific IP paths even if there are less congested routes. With the centralization of routing in OpenFlow, the controller has central logic derived from the network's domain view. The domain-wide view ensures fewer routing errors and the best path for traffic routing in the core network.

In OpenFlow, traffic forwarding flow tables are used rather than RIB, FIB, or mac addresses. Each flow entry has three actions: forward, redirect, and drop. Details of these actions were fully explored in chapter 3. Each flow table in the switch is composed of flow entries made with counters, match fields, and instructions set to apply to traffic when they meet conditions from the sending device. According to (ONF, 2018b), expanded forwarding criteria improve routing efficiency in OpenFlow by 60 percent. A Google experiment at Google labs (Jain et al., 2014) reported that deploying OpenFlow in part of their network with a special focus on expanded forwarding criteria improved traffic utilization by 95 percent. Increased traffic utilization implies that there is less jitter generated in bidirectional traffic.

4.3.0 Performance of OpenFlow and OSPF routing on PDR in a large mobile IP network.

4.3.1 PDR and Routers

In the second objective, the amount of PDR gained was tested using OpenFlow and OSPF routers added into the core network, as stated in chapter 3. The results in figure 4.8 illustrate that OpenFlow routed traffic with an average PDR value of 118 mbs for the first 20 routers and 128 mbs for a similar number of OSPF routers. When 20 more routers were added to attain 40 routers, PDR levels were reduced for both protocols, with OpenFlow gaining 92.75 mbs and OSPF having 84.25 mbs, less than OpenFlow PDR. When the 80 routers were added to the network, the average PDR for OpenFlow was 75.5 mbs, while OSPF gained a PDR value of 71.25 mbs. As the network size increased, there was reduced PDR for both protocols.

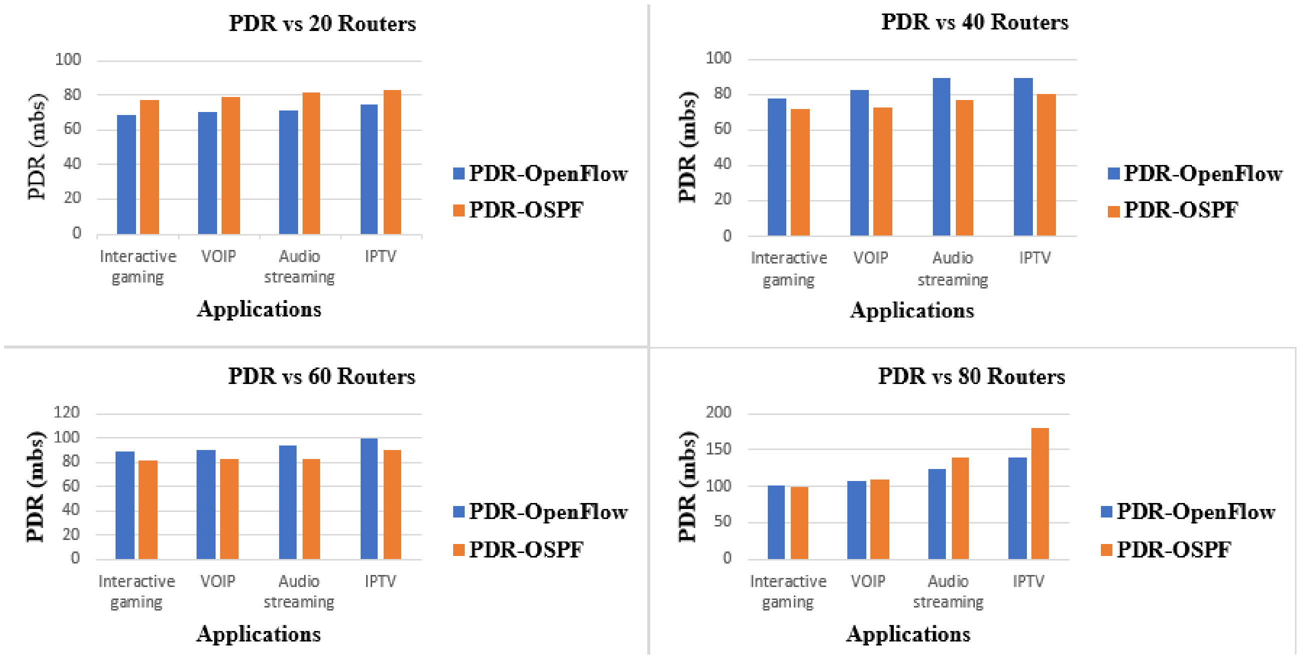


Figure 4. 7 PDR and Routers (Author, 2020)

4.3.2 Packet Data Ratio and User Equipment

Figure 4.9, jitter was tested against UEs in increasing numbers until the experimental objective of 1000 UEs was achieved. For the first 200 UEs that sent information on the network, OpenFlow gained a PDR value of 140 mbs while OSPF was 180 mbs. When 400 more UEs are added to the network, the PDR values are reduced again for both protocols at 123 mbs for OpenFlow and 140 for OSPF. When the network became large at 1000 UEs, the PDR values for OpenFlow and OSPF were the same for both protocols at 102 mbs.

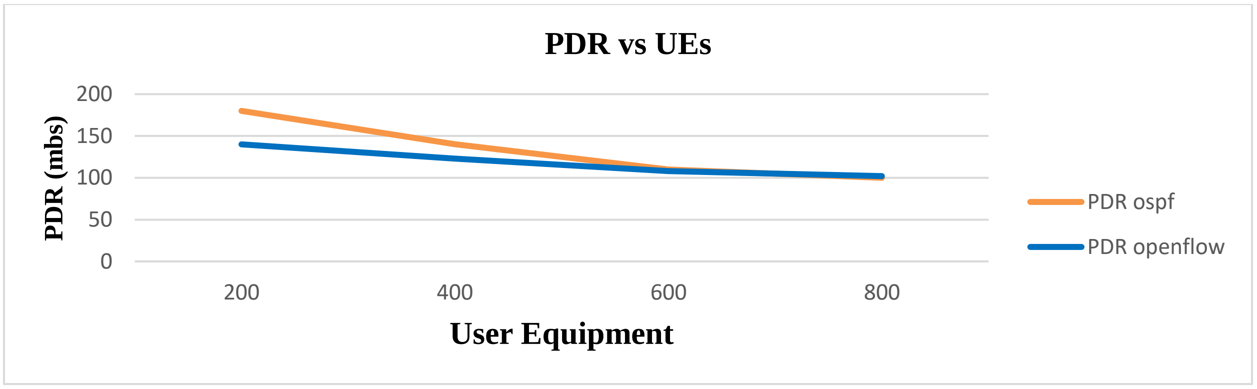


Figure 4. 8 Packet data ratio versus User Equipment (Author, 2020)

4.3.4 Packet Data Ratio and Application

Figure 4.10 shows PDR results from bidirectional traffic generated by UEs. As seen in earlier tests between UEs and routers, applications gain different PDR values. Interactive gaming gained PDR values of 180 mbs for OSPF while OpenFlow was 140 mbs. The next application test, VOIP, gained a PDR of 140 mbs for OSPF while OpenFlow was 123 mbs. Internet TV gained PDR values of 100 mbs for OSPF, while OpenFlow's PDR value was 102.

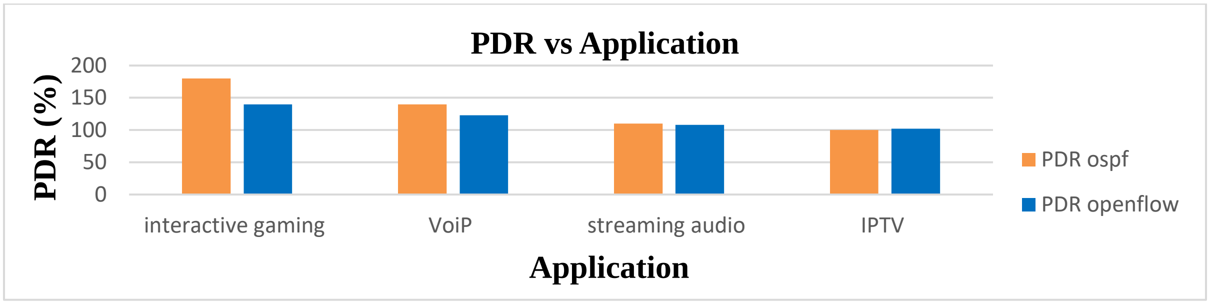


Figure 4. 9 Packet data Ratio versus Application (Author, 2020)

4.3.5 Performance Analysis of Packet Data Ratio versus OpenFlow and OSPF

4.3.5.1 Specific discussions on Routers, UE, and Applications performance

According to (Szigeti et al., 2013), the desired PDR value for bidirectional communication is 95 percent or more of traffic delivered. As stated in chapters 1 and 2, more PDR guarantees desired quality of experience and clarity on the communicating nodes (CAK, 2014; Janevski et al., 2013).

In figures 4.8 to 4.10, PDR values for a large number of routers, UEs, and applications show that OSPF gains more PDR when the network has fewer routers and UEs. However, as the network size increases, the PDR values for both protocols stabilize and almost appear identical. Further analysis of the network components, namely; routers, UEs, and applications, shows that the OSPF PDR values were 118.26 percent, while OpenFlow gained a PDR value of 109 percent. This study can deduce that the number of devices in the core network affects the PDR values. The PDR values reduce as more devices and end-users connect to the network. A closer analysis was conducted to establish the significance of the difference in PDR values for the network components against OpenFlow and OSPF. A std of 0.19 percent was derived, indicating no significant difference between PDR in the OSPF and OpenFlow algorithms. This finding is similar to a study by (Araniti et al., 2014), where they reported that OpenFlow still improved PDR, especially as more nodes were added to the network. Therefore, this research can deduce that PDR in OSPF is optimum for the first few nodes compared to OpenFlow. However, as the network size increases, there is no difference in the PDR values. This study can further deduce that the architecture of the routing algorithm mentioned in section 4.1 impacts the PDR levels of routing using OSPF and OpenFlow.

4.3.5.2 Centralized versus distributed architecture

Based on the distributed architecture, each packet must be inspected in the OSPF network before being forwarded to the next-hop destination after routing has converged. With the expansion of the network size, as expected, the OSPF algorithm routing table also expands, resulting in less PDR in OpenFlow compared to OSPF in forwarding IP traffic. Figure 4.5 and 4.6, seen earlier in section 4.2.4.1.2, illustrates the routing table population process of OSPF and OpenFlow. OpenFlow has fewer steps to converge at four compared to OSPF's eight stages. OpenFlow takes less time to converge at an average of 10 seconds less than OSPF during the population of routing tables and sharing of flow entries. When the network size is small at between 200 to 600 UEs, OSPF has a PDR of 180 mbs, compared to OpenFlow 140 mbs. As more UEs are added, PDR is improved in OpenFlow with the same values as OSPF, with both protocols registering PDR values of 102 mbs at the peak of network performance. As seen in chapter 3, PDR is calculated as bytes sent over bytes received. Therefore, this study can deduce that while the network is small in size, below 80 routers and less than 1000 UEs, OSPF can gain more PDR at an average of 118.25 mbs compared to OpenFlow 109 mbs. However, as the network increases in size and delay tendencies start to operate in the OSPF network, there is no significant difference in PDR values for the two routing protocols.

4.3.5.3 Simplicity versus Complexity

Similar to other link-state routing algorithms, OSPF uses different types of OSPF packets to inform adjacent devices on the sequence of events such as loss of connection, availability of new routes, or announcement of new link status (Doyle, 2016; Moy, 1999). Table 4.1, shown earlier, illustrates the OSPF control packets used to create, detect and sustain neighborship between routers, derived from the experiment debug logs of this research. While the exchange of link-state packets in OSPF can improve robustness and remove the single point of failure (SPOF), studies by (Doyle, 2016; Feamster et al., 2004; Shaikh et al., 2002; Zhang & Yan, 2015) show that the extensive OSPF packets add to network complexity which in turn impact QoS in overall network performance. OpenFlow addresses complexity problems through simple flow entries pushed out to forwarding devices such as routers and firewalls (Braun & Menth, 2014; Casado & Foster, 2013; Cassado, 2015; Feamster et al., 2013; Moy, 1999). According to (Feamster et al., 2004) and (Miller, 2002a), the additional role of OSPF routers in performing route computation and sharing of links state packets makes the legacy approach prone to inconsistencies, especially in a large routing domain. Instead of several link-state packets, OpenFlow uses flow entries to simplify the forwarding process (Hewlett Packard, 2013). A detailed analysis shown in Figure 4.4 of the study experiment logs shows that the complexity of OSPF affects routing speed as it consumes extra bandwidth and slows down the packet forwarding process. OpenFlow uses less bandwidth after the initial exchange of flow entries between the controller and switches, improving the PDR values, especially as the network increases.

4.3.5.4 Expanded versus Limited forwarding headers

In OSPF routing, packets are primarily forwarded using source IP and destination IP. The routing process depends on the routing information base (RIB) and forwarding information base (FIB). RIB and FIB are distributed tables that contain the memory of routes known to the routers and learned from adjacent routers. This approach in relying on a distributed table mechanism reduces routing efficiency due to the underutilization of modern routers like high processing speeds and increased memory (Odom, 2017). Limiting path choice in mesh or star network to destination IP implies that routers are restricted to specific IP paths even if there are less congested traffic paths. With the centralization of routing in OpenFlow, the controller has central logic derived from the domain view of the network. The domain view ensures fewer errors in routing and the best choice of routes for traffic, which ensures that there is more PDR for the end-user traffic.

In OpenFlow, traffic forwarding flow tables are used rather than RIB, FIB, or mac addresses. Figure 4.7, also shown earlier, illustrates a typical OpenFlow switch and controller operation in forwarding traffic in the network. Each flow entry has three actions: forward, redirect, and drop. How OpenFlow forwarding operates was explored in chapter 3. Each flow table in the switch comprises flow entries with counters, match fields, and instructions set to apply to traffic when they meet specified conditions. According to (ONF, 2018b), expanded forwarding criteria improve routing efficiency in OpenFlow by 60 percent. A Google experiment at Google labs (Jain et al., 2014) reported that deploying OpenFlow in part of their network with a special focus on expanded forwarding criteria improved traffic utilization by 95 percent. Increased traffic utilization implies that there is more PDR generated in bidirectional traffic. Figures 4.1 to 4.3 show a similar utilization of traffic, which increases PDR and thus improves the quality of user experience in the large mobile network.

4.4 Performance of OpenFlow and OSPF routing on Throughput in large Mobile IPv4 network

4.4.1 Throughput and Routers

In the third objective, throughput gained was tested using OpenFlow and OSPF. Routers were added into the core network, as stated in chapter 3, gradually. The results in figure 4.7 indicate that OpenFlow routed traffic with an average throughput value of 713 mbs for the first 20 routers and 660 mbs for a similar number of OSPF routers. When 20 more routers were added to attain 40 routers, throughput values decreased to an average of 680 mbs for OpenFlow and 640 mbs for the OSPF protocol. With the addition of 60 more routers, the throughput dropped significantly for both protocols, with OpenFlow gaining 390 mbs while OSPF gained a throughput of 300 mbs.

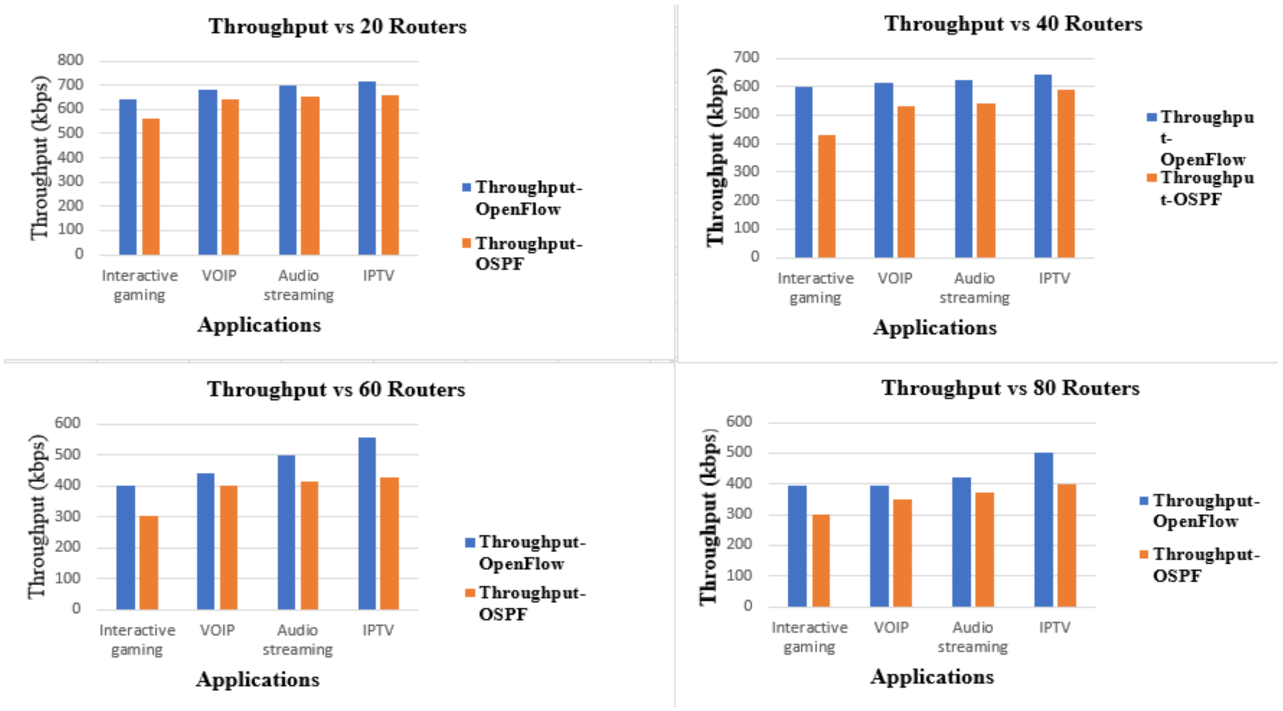


Figure 4. 10 Throughput vs Routers (Author, 2020)

4.4.2 Throughput and User Equipment

In figure 4.9, throughput was tested against UEs in increasing order until the experimental design of 1000 UEs was achieved. For the first 200 UEs that sent bidirectional traffic to the network, OpenFlow registered average throughput values of 713 mbs while OSPF gained a throughput of 660 mbs. When 400 UEs were added to the network, the throughput values generated for the two protocols were reduced, with OpenFlow gaining 700 ms while OSPF gained a throughput of 653 mbs. At 800 UEs, the throughput for OpenFlow was 643 mbs, and OSPF gained a throughput of 563 mbs.

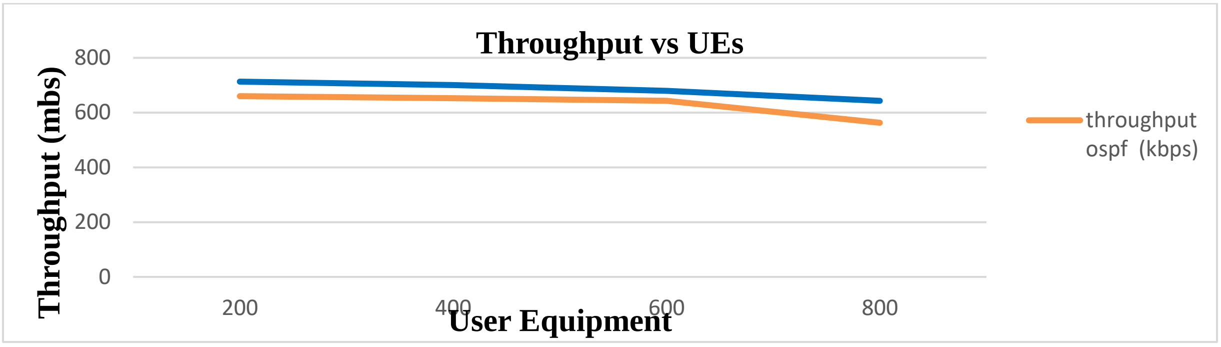


Figure 4. 11 Throughput vs UEs (Author, 2020)

4.4.3 Throughput and Applications

Figure 4.9 illustrates throughput results from bidirectional application traffic generated by UEs. Interactive gaming gained a throughput of an average of 660 mbs and 713 mbs for OSPF and OpenFlow, respectively. Voice over the Internet gained 653 mbs for OSPF compared to 700 mbs for OpenFlow. In streaming audio, the throughput for OSPF was 653 mbs, while OpenFlow achieved a throughput of 680 mbs. Finally, in testing for IPTV, the throughput for OSPF was 563 mbs, while OpenFlow gained a throughput of 643 mbs.

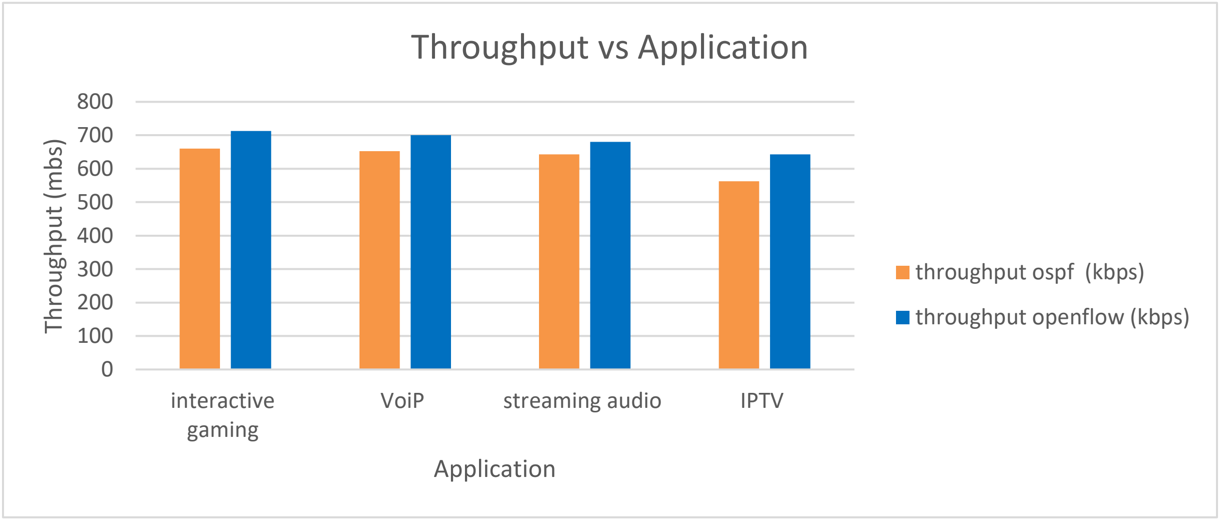


Figure 4. 12 Throughput vs. Application (Author, 2020)

4.4.4 Performance Analysis of Throughput versus OpenFlow and OSPF

4.4.4.1 Specific discussions on Routers, UE, and Applications performance

In the standards document for network QoS (Szigeti et al., 2013), the optimum throughout value for bidirectional communication is 318 mbs or more. As defined earlier in chapters 1 and 2, more throughput implies less loss of information and clarity on the communicating nodes (CAK, 2014; Janevski et al., 2013). In figures 4.7 to 4.9, this study observed that OpenFlow gained more throughput at an average of 637.229 mbs than OSPF at 568.29 mbs. To establish a significant difference between throughput values generated from the experiment, an std deviation test was performed on the throughput values for the network components and the two study routing protocols. The std of 81 mbs derived showed a significant difference between the throughput generated OSPF and OpenFlow. This study deduced that OpenFlow has high throughput compared to OSPF.

Further analysis was done to ascertain the impact of the network's size on the generated throughput values. With the addition of more routers and UEs, the throughput values reduced, indicating large network behavior. Therefore, according to the study objectives, this research also showed that the test environment was a large mobile IP core network. This study can further deduce that the architecture of the routing algorithm mentioned in section 4.1 impacted the throughput levels of routing in OSPF and OpenFlow.

4.4.4.1.1 Centralized versus distributed architectures

Each packet must be queried after routing has converged before being forwarded to the next-hop destination based on the distributed architecture in the OSPF network. As expected, with the expansion of the network size, the OSPF algorithm routing table also expands, resulting in increased times for IP forwarding. This increase in control traffic affects the throughput of end devices in the network. Chapter 3 shows OpenFlow has fewer steps to converge at four than OSPF's eight steps. Few steps imply more end-user throughput than OSPF, which uses extra link-state packets to maintain the network operation.

As observed earlier in chapter 3, OpenFlow, the data packets received by the switch are forwarded to the centralized controller at the onset of the network operation. The controller pushes flow entries to the switches, as summarized in figure 4.6. Subsequent packets are now forwarded using simple query flow tables in the switches. This OpenFlow operation removes the need for switches to continuously reference the controller except when a new change is detected in the network.

4.4.4.1.2 Simplicity versus Complexity

The OSPF protocol uses different types of link-state packets to inform adjacent devices on the sequence of events such as loss of connection, availability of new routes, the cost to certain networks, or announcement of new interfaces that have come up (Doyle, 2016; Moy, 1999). Table 4.1, shown earlier, describes the OSPF control packets used to create, detect and sustain neighborship between routers, which have been derived from the experiment logs of this research.

As observed earlier in chapters 3 and 4, link-state packets in OSPF can improve robustness and remove a single point of failure (SPOF). However, such behavior can add to network complexity, impacting QoS in overall network operation. OpenFlow addresses complexity problems through simple flow entries pushed out to forwarding devices such as routers and firewalls (Braun & Menth, 2014; Casado & Foster, 2013; Cassado, 2015; Feamster et al., 2013; Moy, 1999).

According to (Feamster et al., 2004) and (Miller, 2002a), the additional role of OSPF routers in performing route computation and sharing of links state packets makes the legacy approach prone to inconsistencies, especially in large network domains. Instead of several state packets, OpenFlow uses simple flow entries to simplify the forwarding process of traffic and a dedicated, intelligent discovery engine to maintain the network from a centralized and abstracted view (Hewlett Packard, 2013). A detailed analysis shown earlier and summarized in figure 4.4 of the study experiment logs shows that the complexity of OSPF affects routing speed as it consumes extra bandwidth and uses additional bandwidth intended for control data, which turn affects the throughput.

4.4.4.1.3 Expanded versus Limited forwarding headers.

In OSPF routing, packets are primarily forwarded using source IP and destination IP. As observed earlier in chapters 3 and 4, the routing process also depends on the routing information base (RIB) and forwarding information base (FIB. Using two parameters, source, and destination IP, reduces routing efficiency due to the less application of high routing and memory speeds (Odom, 2017) found in modern routers and switches. Limiting path choice in mesh or star network to destination IP implies that routers are restricted to specific IP paths even if there are less congested routes.

With the centralization of routing in OpenFlow, the controller has central logic derived from the domain view of the network. The domain view ensures fewer errors in routing and the best choice of routes for traffic. More throughput is gained when packets are routed through less congested routes, which improves the mobile user experience.

Each flow entry has three actions: forward, redirect, and drop. Details of these actions are covered in chapter 3. Each flow table in the switch comprises flow entries with counters, match fields, and instructions set to apply to traffic when they meet certain conditions. According to (ONF, 2018b) and (Jain et al., 2014), expanded forwarding criteria in OpenFlow improved throughput values by 60 percent. This study can therefore deduce that increased traffic utilization improved throughput gained in bidirectional traffic communication.

4.5 Performance of OpenFlow and OSPF routing on End to end delay in large mobile IPv4 network

4.5.1 End to end to delay and Routers

In the fourth objective, end-to-end delay for bidirectional traffic was tested using OpenFlow and OSPF as per the study design in chapter 3. Figure 4.11 showed OpenFlow routed traffic with an average end-to-end delay of 123 ms for the first 20, while OSPF registered a delay of 160.74 ms. With the addition of 20 more routers, the delay values for OpenFlow and OSPF were 122.5 ms and 170.5 ms, respectively. There was an addition of 20 more routers which increased the number of routers to 60, and the delay generated was 132 ms for OpenFlow while OSPF registered values of 194.5 ms. Finally, 20 more routers were added to create a large network of 80 routers, generating delay values of 165.5 ms for OpenFlow and 215.25 ms for OSPF.

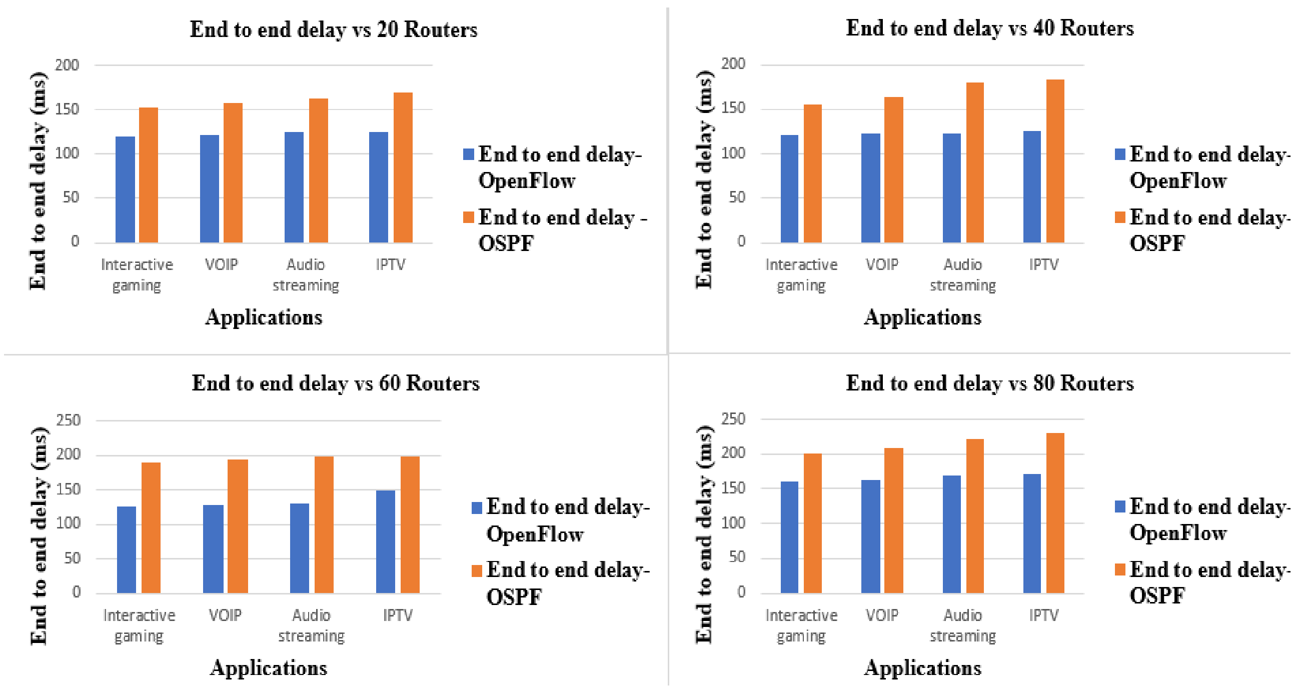


Figure 4. 13 End-to-end delay vs routers (Author, 2020)

4.5.2 End to End delay and User Equipment

Figure 4.12, end-to-end delay was tested against UEs in increasing order as stated in the study design in chapter 3. For the first 200 UEs that sent bidirectional traffic to the network, OpenFlow registered average end-to-end delay values of 120 ms while OSPF was 153 Ms. When 400 UEs were added to the network, the end-to-end delay increased for OpenFlow and OSPF at 122ms and 157 ms, respectively. When 200 more UEs were added to the network to make 600, OpenFlow registered a delay of 125 ms while OSPF generated 163 Ms. When the experiment objective of 1000 UEs was reached in the network, the end-to-end delay for OpenFlow was 130 ms, while OSPF registered the highest delay of 195 Ms.

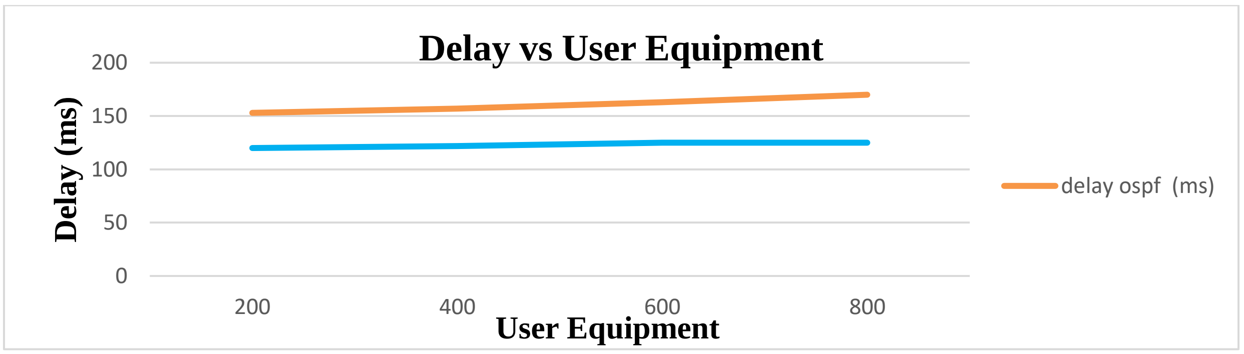


Figure 4. 14 End to End Delay vs. UEs (Author, 2020)

4.5.3 End-to-end delay and Application

Figure 4.13 shows a complete end-to-end delay resulting from UEs' two-way application traffic. Applications generate different end-to-end delay values, as observed in this study. Interactive gaming had an end-to-end delay of 120 ms and 153 ms for OSPF and OpenFlow, respectively. In the next application tested, VoiP generated an end-to-end delay of 157 ms for OpenFlow and 159 ms for OSPF. Audio traffic registered a delay value of 151 ms for OpenFlow, while OSPF delay values were 153 Ms. In the fourth application tested, IPTV generated a delay of 120 ms for OpenFlow while OSPF delay was 149 Ms.

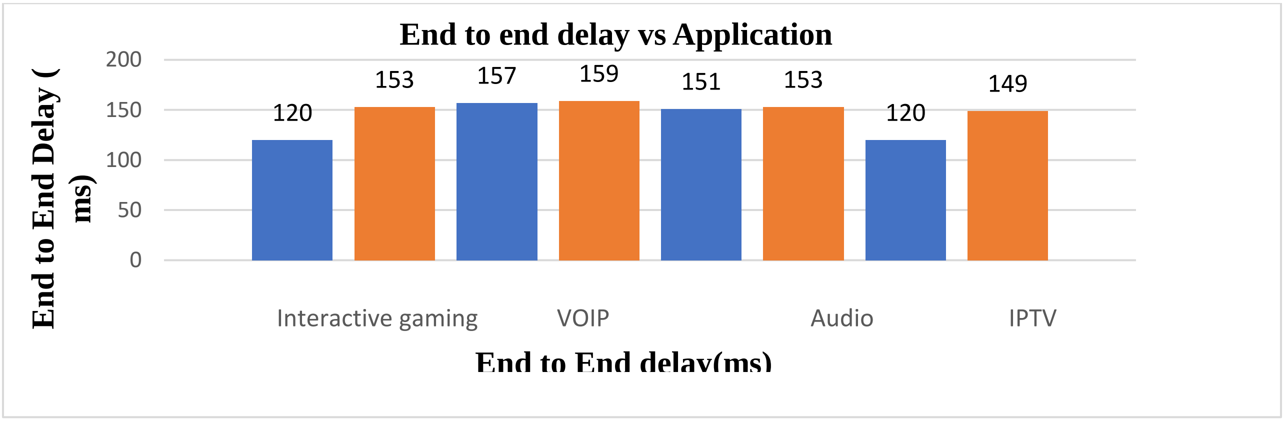


Figure 4. 15 End to End Delay vs Applications (Author, 2020)

4.5.4 Performance Analysis of End-to-End delay versus OpenFlow and OSPF

4.5.4.1 Specific discussions on Routers, UE, and Applications performance

According to (Szigeti et al., 2013), the optimum end-to-end delay value for bidirectional communication is 150 ms or less. According to (CAK, 2014) and (Janevski et al., 2013), less end-to-end delay guarantees a minimal loss of information and clarity on the communicating nodes.

In figures 4.11 to 4.13 end to end delay values for a large number of routers, UEs and applications show that OpenFlow generates a less end-to-end delay of 132.38 ms compared to OSPF 168.178 Ms. A deeper analysis to establish the significance of the difference in the end-to-end delay values for the three network components, UE, routers, and application against OpenFlow and OSPF, showed a std of 22.75 Ms. This finding is an indication that there was a significant difference in the end-to-end delay values associated with the OpenFlow and OSPF. This study deduced that OpenFlow has less end-to-end delay compared to OSPF. Further analysis was done to ascertain the impact of the network's size in relation to end-to-end delay experienced in the network. With more routers, applications, and UEs, the end-to-end delay values increased for both study protocols, showing a large network behavior. This study thus observed that the architecture of the routing algorithm impacted the end-to-end delay in the mobile IP core network.

4.5.4.1.2 Centralized versus distributed architecture

The OSPF network uses a distributed architecture approach, also discussed in chapters 1 and 2 in this research. After routing has converged, each packet must be inspected before being forwarded to the next-hop destination. With the expansion of the network size, as expected, the OSPF algorithm routing table also expands, resulting in a high end-to-end delay in forwarding IP traffic. OpenFlow has fewer steps to converge at four compared to OSPF, with eight stages repeated over timed periods in the network. This increased sending of hello packets creates competition for data and control packets in the OSPF network, resulting in an OSPF delay that deviates from the end-to-end baseline delay of 150 ms or less.

OpenFlow, at the onset of the network operation, the data packets received by the switch are forwarded to the centralized controller. Then the controller pushes flow entries to the switches, as summarized in figure 4.6. Subsequent packets are now forwarded using simple query flow tables in the switches. The reactionary behavior of OpenFlow removes the need for switches to periodically reference the controller except when a new change is discovered in the network. This approach has been observed to reduce the end-to-end delay in mobile IP network core.

4.5.4.1.3 Simplicity versus Complexity

Similar to other link-state routing algorithms, OSPF uses different types of OSPF packets to inform neighboring devices about the sequence of events such as loss of connection, availability of new routes, or discovery of new interfaces (Doyle, 2016; Moy, 1999). Table 4.1 also shows the OSPF control packets used to create, detect, and sustain neighbor routers.

As seen in chapter 4, exchanging link packets can add to robustness and remove the single point of failure (SPOF) in OSPF. However, studies by (Doyle, 2016; Feamster et al., 2004; Shaikh et al., 2002; Zhang & Yan, 2015) show that the increased OSPF packets contribute to complexity, which impacts QoS in overall network performance. OpenFlow addresses complexity problems through simple flow entries pushed out to forwarding devices such as routers and firewalls (Braun & Menth, 2014; Casado & Foster, 2013; Cassado, 2015; Feamster et al., 2013; Moy, 1999).

According to (Feamster et al., 2004) and (Miller, 2002a), the additional role of OSPF routers in performing route computation and sharing of links state packets makes the legacy approach prone to inconsistencies, especially in a large routing domain. Instead of several link-state packets, OpenFlow uses flow entries to simplify the forwarding process (Hewlett Packard, 2013). A detailed analysis summarized in figure 4.4 of the study experiment logs shows that the complexity of OSPF affects routing speed as it consumes extra bandwidth and slows down the packet forwarding process. OpenFlow uses less bandwidth after the initial exchange of flow entries between the controller and switches.

4.5.4.1.3 Expanded versus Limited forwarding headers

In OSPF routing, packets are primarily forwarded using source IP and destination IP. This approach reduces routing efficiency due to the disuse of modern characteristics in routers like high processing speeds and increased memory (Odom, 2017). Limiting path choice in mesh or star network to destination IP implies that routers are restricted to specific IP paths even if there are less congested routes. With the centralization of routing in OpenFlow, the controller has central logic derived from the domain view of the network. The domain view ensures fewer errors in routing and the best choice of routes for traffic. In OpenFlow, traffic forwarding flow tables are used rather than RIB, FIB, or mac addresses.

Each flow entry has three actions: forward, redirect, and drop. Flow operations in OpenFlow were highlighted in chapter 3. According to (ONF, 2018b), expanded forwarding criteria improve OpenFlow end-to-end delay routing efficiency by 60 percent. A Google experiment at Google labs (Jain et al., 2014) reported that deploying OpenFlow in part of their network with a special focus on expanded forwarding criteria improved traffic utilization by 95 percent. Increased traffic utilization implies less end-to-end delay generated in two-way traffic flow. This study confirmed more traffic utilization in the OpenFlow network by over 60 percent, which reduced the end-to-end delay in the mobile IP network.

4.6 Summary

In chapter 4, this study reported the results, analyzed the values' trends, and performed significance tests. The research also discussed the findings and deductions based on the results from the simulation network. The chapter also looked at the architectural differences between the two protocols, OpenFlow and OSPF, and how those differences contribute to efficiency in QoS measures.

CHAPTER FIVE

CONCLUSIONS AND RECOMMENDATIONS

This chapter provides the conclusion and recommendations of the research done. Section 5.1 covers the conclusions, while section 5.2 gives the recommendations for future research.

5.1 Conclusions

This study, in its introduction, observed that there was evidence that CSP service networks are still using legacy routing protocols such as OSPF in the core network, despite the existence of new 5G ready routing protocols like OpenFlow. The study also demonstrated from an extensive literature search that new mobile IP applications are increasingly being developed at 70 percent more than the USSD type.

A literature review showed a gap in information available on OSPF and OpenFlow's QoS performance on a large mobile IP core CSP network. Four objectives were formulated to compare the performance of OSPF and OpenFlow on jitter, PDR, throughput, and end-to-end delay in a large mobile IP core network.

Two simulated CSP mobile core networks were developed running OpenFlow and OSPF protocols. Routers, UEs, and OpenFlow switches were gradually added to the network until the specified numbers in the experimental setup were achieved. The UEs sent different types of traffic to internet servers, and metrics were recorded on the OMNeT++ IDE and Wireshark. The results were aggregated using the IDE tools and exported to an excel spreadsheet. Panda tool was used to extract metrics such as jitter and PDR.

The study results showed that across the research-specific objectives of testing for the performance of jitter, PDR, throughput, and end-to-end delay in a large CSP core network, OpenFlow improved routing QoS by 60 percent compared to OSPF. In instances where there was better performance in PDR values from the OSPF protocol routed network, further analysis showed no significant difference in the OpenFlow and OSPF with std of 0.19 mbs. The study confirmed that centralization, simplicity, and expanded matching criteria improved QoS levels and enhanced routing quality in a large IP core network when comparing performance between OSPF and OpenFlow protocols. The results further revealed that OpenFlow improved routing by 60 percent of all the QoS metrics compared to OSPF.

5.2 Recommendations

The author recommends that further investigation be conducted in a real mobile IP core network. Such future research will provide an opportunity to establish if OpenFlow's benefits on the simulated network can be extended to the production network. Furthermore, a production network will provide the right platform to test the Quality of Experience (QoE), which varies from user to user.

REFERENCES

Aker, J. C., & Mbiti, I. M. (2010). Mobile Phones and Economic Development in Africa Working Paper 211 June 2010 Mobile Phones and Economic Development in Africa. *Center for Global Development*, (June 2010), 1–44.

Andras, V. (2016). OMNeT++. In *OMNeT++* (Vol. 5485353-1E). https://doi.org/10.1007/SpringerReference_27988

Andras, V. (2017). A Quick Overview Of The OMNeT ++ IDE. In *Omnet++*.

Araniti, G., Cosmas, J., Iera, A., Molinaro, A., Morabito, R., & Orsino, A. (2014). OpenFlow over wireless networks Performance analysis. *2014 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting*, 1–5. https://doi.org/10.1109/BMSB.2014.6873559

Author. (2020). *Research*. Maseno University.

Azodolmolky, S. (2013). *Software Defined Networking with OpenFlow*.

Balasubramanian, D. (2006). QoS in Cellular Networks. *IEEE Communications Surveys & Tutorials*, *1*(1), 1–24.

Bisio, I., Delfino, A., Luzzati, G., Lavagetto, F., Marchese, M., Fr, C., & Valla, M. (2012). Opportunistic Estimation of Television Audience Through Smartphones. *IEEE Communications Surveys & Tutorials*.

Braun, W., & Menth, M. (2014). Software-Defined Networking Using OpenFlow: Protocols, Applications and Architectural Design Choices. *Future Internet*, *6*(2), 302–336. https://doi.org/10.3390/fi6020302

Businessdaily. (2021). CA puts Telkom, Airtel on notice over call quality - Business Daily. Retrieved March 1, 2021, from BusinessDaily website: https://www.businessdailyafrica.com/bd/economy/ca-puts-telkom-airtel-on-notice-over-call-quality-3275124

CAK. (2014). *Quality of Service Brochure*. Nairobi.

CAK. (2015). Mobile operators Fail to Meet Quality of Service targets for the third year running. Retrieved January 6, 2018, from http://www.ca.go.ke/index.php/what-we-do/94-news/382-mobile-operators-fail-to-meet-quality-of-service-targets-in-three-consecutive-years

CAK. (2016). *Quarterly Sector Statistics Report Second Quarter for the Financial Year 2015 / 2016* (Vol. 2016).

CAK. (2018). *First Quarter Sector Statistics Report For The Financial Year 2017 -2018* (Vol. 2018). Nairobi.

CAK. (2019). *CAK*. Retrieved from Communication Authority Kenya

Casado, M., & Foster, N. (2013). Abstractions for Software-Defined Networks. *Stanford University*, *1*(1), 1–8.

Cassado, M. (2015). *Martin Casado Interview*. Retrieved from www.youtube.com

Chepken, C., Blake, E., & Marsden, G. (2012). Telecommuting In The Developing World : A Case Of The Day-Labour Market. Capetown University.

Cisco. (2005). *Measuring Delay , Jitter , and Packet Loss with Cisco IOS SAA and RTTMON*. https://doi.org/24121

Cisco. (2010). Cisco Performance Routing. In *Cisco*.

Cisco. (2012). *Quality of Service Design Overview*.

Cisco. (2014). *Quality of Service Overview What Is Quality of Service ? Who Could Benefit from Using Cisco IOS QoS ?* San Francisco, CA.

Cisco. (2015a). Cisco Visual Networking Index : Global Mobile Data Traffic Forecast Update , 2010 – 2015. *Cisco Blog*, *1*(2), 42. Retrieved from http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.html

Cisco. (2015b). Cisco Visual Networking Index: Forecast and Methodology, 2015-2020.

Forecast and Methodology, 22. https://doi.org/1465272001663118

Cisco. (2016a). Cisco Visual Networking Index Predicts Near-Tripling of IP Traffic by 2020. *Newsroom.Cisco.Com*, 17–20.

Cisco. (2016b). *Enhanced Interior Gateway Routing Protocol Wide Metrics Last updated*.

Cisco. (2020a). Cisco annual internet report. In *cisco*. Retrieved from https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.pdf

Cisco. (2020b). *Cisco Annual Internet Report*.

Cisconewsroom. (2013). Mobile Transformation: What could 4G do to Africa? | The Network. Retrieved January 21, 2015, from Cisco blog website: http://newsroom.cisco.com/feature-content?type=webcontent&articleId=1266951

CitizenTV. (2021). Viusasa. Retrieved May 31, 2021, from https://viusasa.com/

Clare, D. (2020). Verizon is delaying the shutdown of its 3G wireless network - CNN. Retrieved January 11, 2021, from CNN website: https://edition.cnn.com/2021/01/07/tech/verizon-3g-shutdown-paused/index.html

Costa-requena, J., Kantola, R., & Llorente, J. (2014). Software Defined 5G Mobile Backhaul. *1st International Conference on 5G for Ubiquitous Connectivity (5GU)*, 258–263.

D ’souza, D., Sundharan, K. P., Lokanath, S., & Mittal, V. (2016). Improving QoS in a Software-Defined Network. *Capstone*, 1–9.

Donovan, K. P. (2012). Mobile Money, More Freedom? The Impact of M-PESA’ s Network Power on Development as Freedom. *International Journal of Communication*, *6*, 2647–2669.

Doyle, J. (2016). CCIE Professional Development: Routing TCP/IP. In *Cisco Press*. Retrieved from http://portal.acm.org/citation.cfm?id=521370

Ecobank. (2018). *Ecobank Mobile App Benefits*.

ETSI. (2003). *Satellite Earth Stations and Systems (SES). Broadband Satellite Multimedia IP. IP Interworking over Satellite; Performance, Availability and Quality of Service - ETSI Technical Report, TR 102 157 V1.1.1*. *1*, 1–83.

Feamster, N., Balakrishnan, H., Rexford, J., Shaikh, A., & van der Merwe, J. (2004). The case for separating routing from routers. *Proceedings of the ACM SIGCOMM Workshop on Future Directions in Network Architecture - FDNA '04*, 5. https://doi.org/10.1145/1016707.1016709

Feamster, N., Rexford, J., & Zegura, E. (2013). The Road to SDN : An Intellectual History of Programmable Networks. *ACM SIGMOD Conference*, 1–13.

Google. (2010). *OpenFlow at Google*. *1*, 1–37. Retrieved from www.google.com

Grayson, M., Shatzkamer, K., & Wainner, S. (2009). *IP design for Mobile Networks*. cisco press.

Graziani, R. (2013). IPv6 Fundamentals: A Straightforward Approach to Understanding IPv6. In *IPv6 Fundamentals*.

Gude, N., Koponen, T., Pettit, J., Pfaff, B., Casado, M., McKeown, N., & Shenker, S. (2008). NOX: towards an operating system for networks. *SIGCOMM Computer Communication Review*, *38*(3), 1–6. https://doi.org/10.1145/1384609.1384625

Hewlett Packard. (2013). *HP OpenFlow Protocol Overview*.

Hu, F., Hao, Q., & Bao, K. (2014). A Survey on Software Defined Networking (SDN) and OpenFlow From Concept to Implementation. *IEEE Communications Surveys & Tutorials*, *16*(c), 1–1. https://doi.org/10.1109/COMST.2014.2326417

IJNICS. (2021). International Journal of Computer Network and Information Security(IJCNIS). Retrieved July 26, 2021, from http://www.mecs-press.org/ijcnis/v13n3.html

ITU. (2009). The world in 2009: A decade of ICT growth driven by mobile technologies. In *ICT facts and figures 2009*. Geneva.

ITU. (2016). Internet users by region and country, 2010-2016. Retrieved August 4, 2018, from ITU website: https://www.itu.int/en/ITU-D/Statistics/Pages/stat/treemap.aspx

ITU. (2017). *Facts and 2017 figures*. 8.

Jain, S., Kumar, A., Mandal, S., Ong, J., Poutievski, L., Singh, A., … Vahdat, A. (2014). B4 : Experience with a Globally-Deployed Software Defined WAN. *IEEE Communications Magazine*.

Janevski, T., Jankovic, M., & Markus, Sc. (2013). Quality Of Service Quality Regulation Manual. In *ITU*.

Klein, D., & Jarschel, M. (2013). *An OpenFlow Extension for the OMNeT++ INET Framework*. https://doi.org/10.4108/simutools.2013.251722

Lamping, Ulf; Sharpe, Richard and Warnicke, E. (2018). *Wireshark User ' s Guide*.

Li, L. E., Mao, Z. M., & Rexford, J. (2012). Toward Software-Defined Cellular Networks. *2012 European Workshop on Software Defined Networking*, 7–12. https://doi.org/10.1109/EWSDN.2012.28

Li, L. E., Mao, Z. M., & Rexford, J. (2016). CellSDN : Software-Defined Cellular Networks. *IEEE Communications Magazine*, 1–6.

Liu, Y., Ding, A. Y., & Tarkoma, S. (2013). *Software-Defined Networking in Mobile Access Networks*.

Miller, G. (2002a). Problems with Traditional Routing. *MIT*, 1–11.

Miller, G. (2002b). *Why traditional routing is not always the best*.

Moy, J. (1999). *Anatomy of Routing Protocol* (First). Massachusetts: Addison Wesley.

Mueck, M., Ambrosini, F., Cadzow, S., Choi, S., Ivanov, V., Pagnozzi, M., & Siaud, I. (2017). *Software Radio Reconfiguration : A highly efficient and modular software reconfiguration approach for mobile devices*.

Nardini, G., Antonio, V., Rudolf, H., Varga, A., & Meszero, L. (2021). SimuLTE - LTE System Level Simulation Model and Simulator for INET & OMNeT++. Retrieved February 24, 2021, from Github website: https://github.com/inet-framework/simulte

Nation. (2014). Safaricom, Airtel splash Sh8bn for yu. Retrieved February 5, 2015, from Nation Media website: http://mobile.nation.co.ke/news/Safaricom--Airtel-splash-Sh8bn-for-yu/-/1950946/2227344/-/format/xhtml/-/4k8wu4/-/index.html

Networks, J. (2015). Junos ® OS OpenFlow Feature Guide. *Juniper*, 1–204.

NHS. (2020). NHS COVID-19 tracking app. Retrieved June 7, 2021, from https://www.nhs.uk/apps-library/nhs-covid-19/

Njanja Annie. (2017). Airtel cuts call rates to rival networks in market fight - Daily Nation. Retrieved January 12, 2018, from NMG website: http://www.nation.co.ke/business/Airtel-cuts-rates-rival-networks-market-fight/996-3868776-k5xf9pz/index.html

Odom, W. (2016). *ICND 100-105*.

Odom, W. (2017). *ICND 200-105*.

Odom, W. (2019). *CCNA 200-301 , Volume 1 Official Cert Guide*. Retrieved from www.ciscopress.com

Oloko, M., Anene, E. B., Kiara, P. G., & Kathambi, I. (2014). *Empirical Analysis on Determinants of Non-Profitability Margins : A Case of Telkom Kenya in the Telecommunications Industry*. *4*(3), 1–4.

OMNeT++. (2019). *OMNeT++ Version 5.5*. OMNeT++.

OMNeT++. (2020). 7th OMNeT++ Community Summit 2020. Retrieved July 26, 2021, from OMNeT++ website: https://summit.omnetpp.org/archive/2020/

Omwansa, T., & Sullivan, N. (2012). *Money, Real Quick*. Nairobi.

ONF. (2018a). M-CORD Open Source Reference Solution for 5G Mobile Wireless Networks. Retrieved January 10, 2018, from ONF website: https://www.opennetworking.org/solutions/m-cord/

ONF. (2018b). Software-Defined Networking (SDN) Definition - Open Networking Foundation. Retrieved January 10, 2018, from ONF website: https://www.opennetworking.org/sdn-definition/

Open Networking Foundation. (2012). *Software-Defined Networking : The New Norm for Networks*.

Open Networking Foundation. (2013). Wireless Networks. In *Open Networking Foundation*.

Poretsky, S., Perser, J., Eramilli, S., & Khurana, S. (2006). *Terminology for Benchmarking Network-layer Traffic Control Mechanisms*.

Rakuten. (2019a). Evolving to software-defined mobile networks. In *Manufacturing Engineer* (Vol. 73). Retrieved from http://www.scopus.com/inward/record.url?eid=2-s2.0-0028735143&partnerID=40&md5=2e8d205ad2c5de2ad68b9e7c9346ced5

Rakuten. (2019b). *Reimagining the End-to-End Mobile Network in the 5G Era*.

Rohal, P., Dahiya, R., & Dahiya, P. (2013). Study and Analysis of Throughput , Delay and Packet Delivery Ratio in MANET for Topology Based Routing Protocols (AODV , DSR and DSDV). *International Journal for Advance Research in Engineering and Technology, 1*, 1–5.

Rosenfeld, A., Sina, S., Sarne, D., Avidov, O., & Kraus, S. (2018). A Study of WhatsApp Usage Patterns and Prediction Models without Message Content ∗. *ArXiv*, 1–24.

Safaricom. (2018). MTIBA. Retrieved October 27, 2018, from Safaricom website: https://www.safaricom.co.ke/about/innovation/social-innovation/m-tiba

Salih, M. A., Cosmas, J., & Zhang, Y. (2015). OpenFlow 1 . 3 Extension for OMNeT ++. *015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, 1–8. https://doi.org/10.1109/CIT/IUCC/DASC/PICOM.2015.246

Savage, Moore, S., Slice, D., Paluch, P., & White, R. (2017). Cisco's Enhanced Interior Gateway Routing Protocol (EIGRP). In *RFC* (Vol. 53).

Shahbaz, M., & Feamster, N. (2015). The case for an intermediate representation for programmable data planes. *Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research - SOSR '15*, 1–6. https://doi.org/10.1145/2774993.2775000

Shaikh, A., Isett, C., Greenberg, A., Roughan, M., & Gottlieb, J. (2002). A case study of OSPF behavior in a large enterprise network. *Proceedings of the Second ACM SIGCOMM Workshop on Internet Measurment Workshop - IMW '02*, 217. https://doi.org/10.1145/637235.637236

Spirent. (2007). Measuring Jitter Accurately. In *Spirent Communications*.

Stalling, W. (2009). Introduction to SDN. *The Internet Protocol Journal*, *16*(1), 1–40. Retrieved from http://aim.bmj.com/content/27/1/1.short

Stallings, W. (2002). *Wireless Communications and Networks*.

Star. (2014). Safaricom hit by four-hour network outage | The Star. Retrieved January 22, 2015, from http://www.the-star.co.ke/news/article-161738/safaricom-hit-four-hour-network-outage

Sundaran, R. (2002). *Internet Routing Problems*.

Szigeti, T., Hattingh, C., Barton, R., & Briley, K. (2013). *End-to-End QoS network design*.

Tanenbaum, A. S. (2013). Computer Networks. In M. Horton (Ed.), *Pearson Education* (5th ed.). https://doi.org/10.1016/j.comnet.2008.04.002

Tourrilhes, J., Sharma, P., Banerjee, S., & Pettit, J. (2014). The Evolution of SDN and OpenFlow : A Standards Perspective. *HP*, 15.

Wallace, K. (2014). *CCNP Routing and Switching ROUTE 300-101 Offical Cert Guide*.

WhatsApp. (2017). *WhatsApp Encryption Overview*. 1–12. Retrieved from https://www.whatsapp.com/security/WhatsApp-Security-Whitepaper.pdf

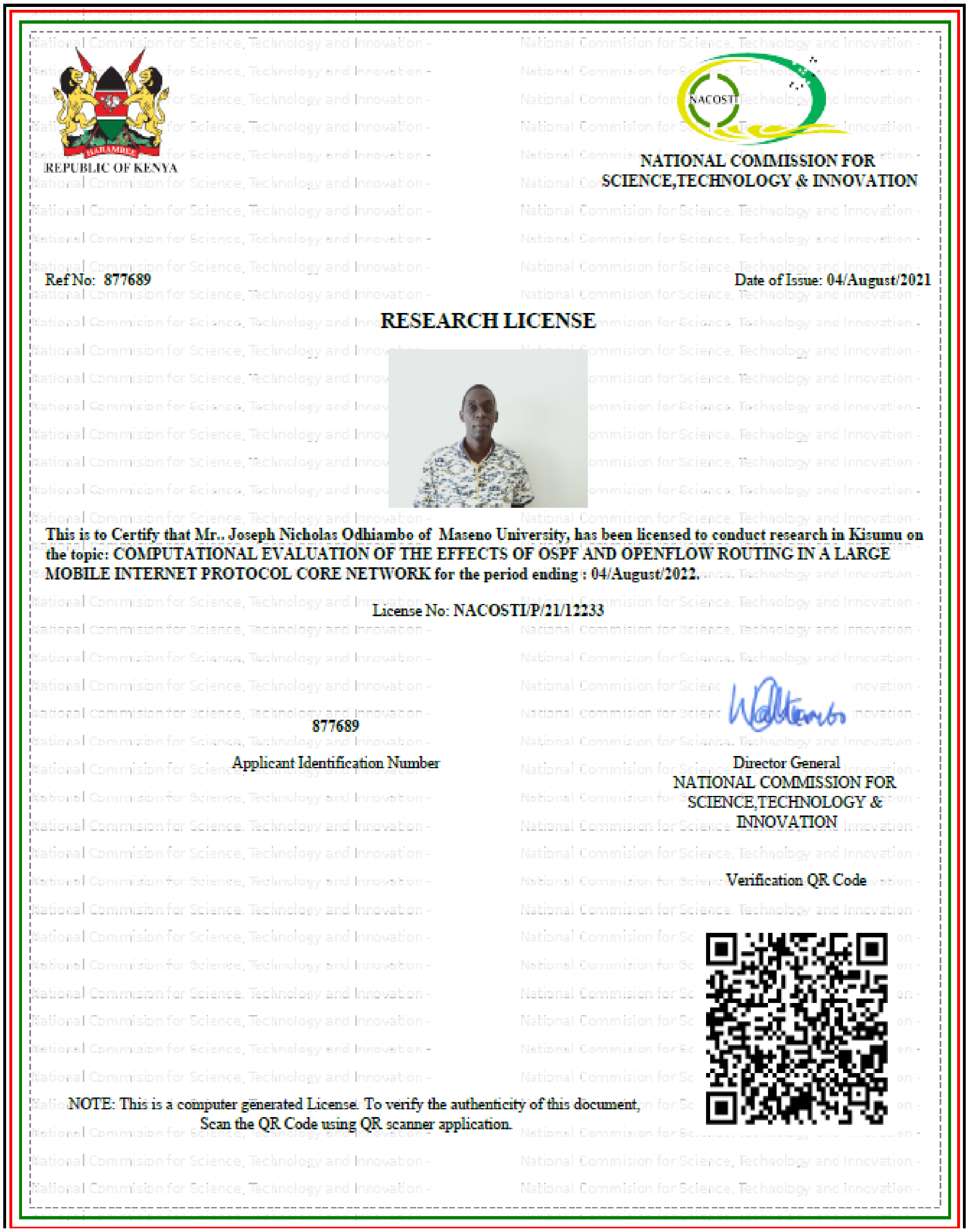
World Bank. (2011, June 24). Yes, Africa Can (P. Chuhan-Pole & M. Angwafo, Eds.). https://doi.org/10.1596/978-0-8213-8745-0

Yap, K.-K., Sherwood, R., Kobayashi, M., Huang, T.-Y., Chan, M., Handigol, N., … Parulkar, G. (2010). Blueprint for introducing innovation into wireless mobile networks. *Proceedings of the Second ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures - VISA '10*, 25. https://doi.org/10.1145/1851399.1851404

Zhang, H., & Yan, J. (2015). Performance of SDN Routing in Comparison with Legacy Routing Protocols. *Proceedings - 2015 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, CyberC 2015*, 491–494. https://doi.org/10.1109/CyberC.2015.30

APPENDIX

Appendix 1 Approval letter from NACOSTI



REPUBLIC OF KENYA

NATIONAL COMMISSION FOR SCIENCE,TECHNOLOGY & INNOVATION

Ref No: 877689

Date of Issue: 04/August/2021

RESEARCH LICENSE

This is to Certify that Mr.. Joseph Nicholas Odhiambo of Maseno University, has been licensed to conduct research in Kisumu on the topic: COMPUTATIONAL EVALUATION OF THE EFFECTS OF OSPF AND OPENFLOW ROUTING IN A LARGE MOBILE INTERNET PROTOCOL CORE NETWORK for the period ending : 04/August/2022.

License No: NACOSTI/P/21/12233

877689

Applicant Identification Number

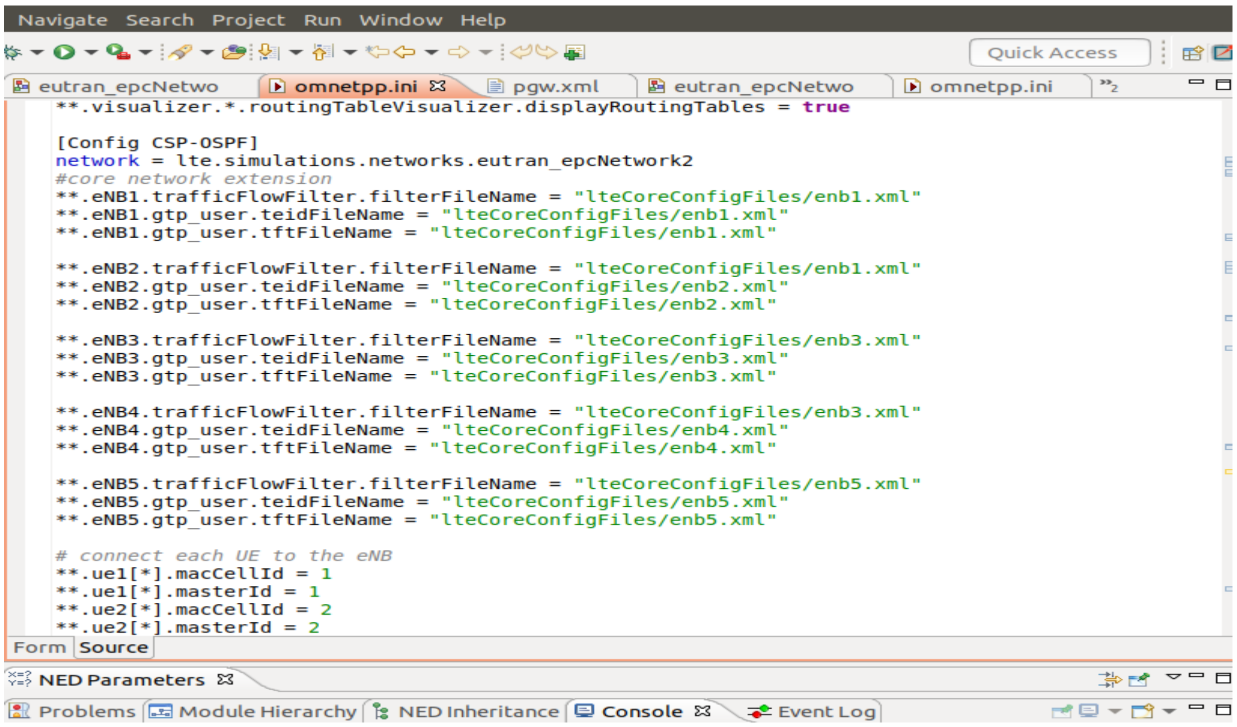
[illegible]

Director General
NATIONAL COMMISSION FOR SCIENCE,TECHNOLOGY & INNOVATION

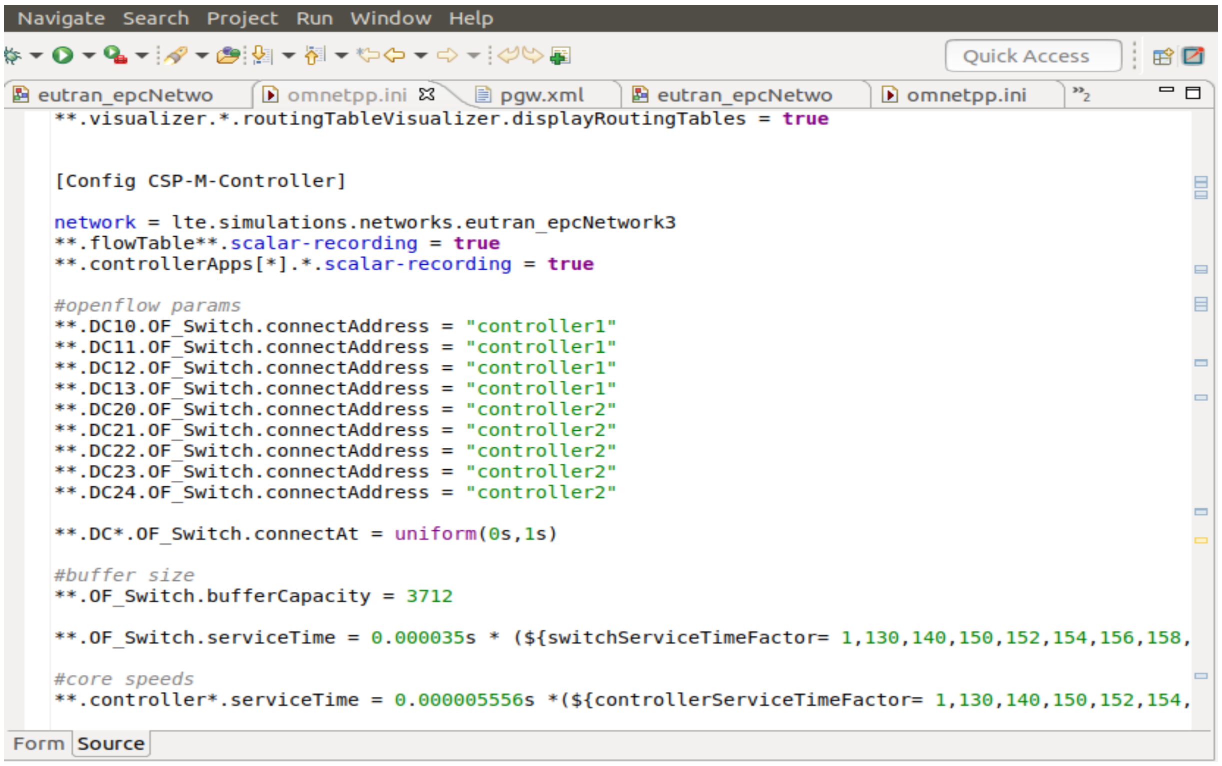
Verification QR Code

NOTE: This is a computer generated License. To verify the authenticity of this document, Scan the QR Code using QR scanner application.

Appendix 2 Ometpp.ini files for OSPF and OpenFlow networks


Navigate Search Project Run Window Help
Quick Access
eutran_epcNetwo
omnetpp.ini
pgw.xml
eutran_epcNetwo
omnetpp.ini
Form Source
NED Parameters
Problems Module Hierarchy NED Inheritance Console Event Log

```
**.visualizer.*.routingTableVisualizer.displayRoutingTables = true

[Config CSP-OSPF]
network = lte.simulations.networks.eutran_epcNetwork2
#core network extension
**.eNB1.trafficFlowFilter.filterFileName = "lteCoreConfigFiles/enb1.xml"
**.eNB1.gtp_user.teidFileName = "lteCoreConfigFiles/enb1.xml"
**.eNB1.gtp_user.tftFileName = "lteCoreConfigFiles/enb1.xml"

**.eNB2.trafficFlowFilter.filterFileName = "lteCoreConfigFiles/enb1.xml"
**.eNB2.gtp_user.teidFileName = "lteCoreConfigFiles/enb2.xml"
**.eNB2.gtp_user.tftFileName = "lteCoreConfigFiles/enb2.xml"

**.eNB3.trafficFlowFilter.filterFileName = "lteCoreConfigFiles/enb3.xml"
**.eNB3.gtp_user.teidFileName = "lteCoreConfigFiles/enb3.xml"
**.eNB3.gtp_user.tftFileName = "lteCoreConfigFiles/enb3.xml"

**.eNB4.trafficFlowFilter.filterFileName = "lteCoreConfigFiles/enb3.xml"
**.eNB4.gtp_user.teidFileName = "lteCoreConfigFiles/enb4.xml"
**.eNB4.gtp_user.tftFileName = "lteCoreConfigFiles/enb4.xml"

**.eNB5.trafficFlowFilter.filterFileName = "lteCoreConfigFiles/enb5.xml"
**.eNB5.gtp_user.teidFileName = "lteCoreConfigFiles/enb5.xml"
**.eNB5.gtp_user.tftFileName = "lteCoreConfigFiles/enb5.xml"

# connect each UE to the eNB
**.ue1[*].macCellId = 1
**.ue1[*].masterId = 1
**.ue2[*].macCellId = 2
**.ue2[*].masterId = 2
```

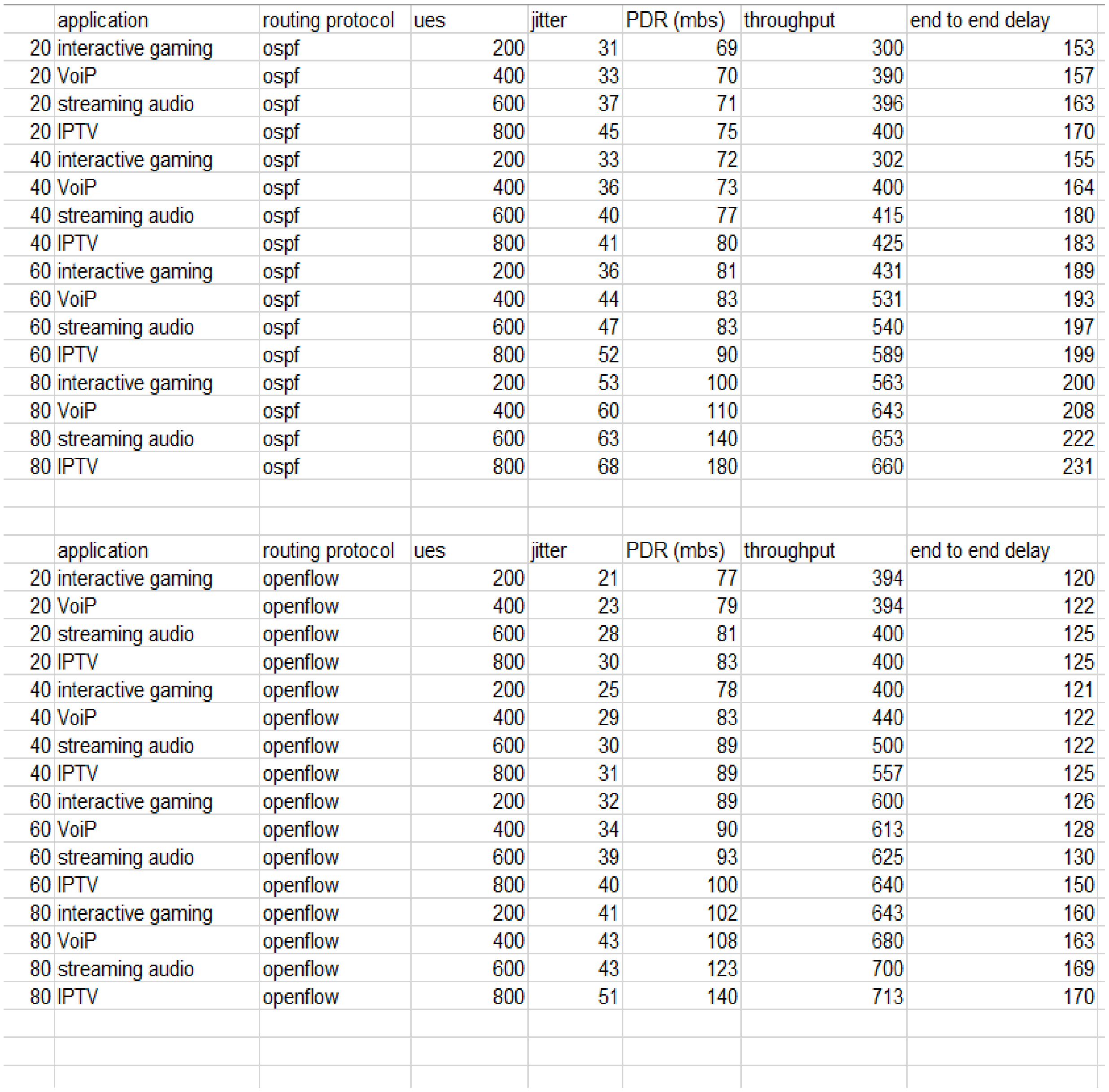


```
**.visualizer.*.routingTableVisualizer.displayRoutingTables = true


[Config CSP-M-Controller]

network = lte.simulations.networks.eutran_epcNetwork3
**.flowTable**.scalar-recording = true
**.controllerApps[*].*.scalar-recording = true

#openflow params
**.DC10.OF_Switch.connectAddress = "controller1"
**.DC11.OF_Switch.connectAddress = "controller1"
**.DC12.OF_Switch.connectAddress = "controller1"
**.DC13.OF_Switch.connectAddress = "controller1"
**.DC20.OF_Switch.connectAddress = "controller2"
**.DC21.OF_Switch.connectAddress = "controller2"
**.DC22.OF_Switch.connectAddress = "controller2"
**.DC23.OF_Switch.connectAddress = "controller2"
**.DC24.OF_Switch.connectAddress = "controller2"

**.DC*.OF_Switch.connectAt = uniform(0s,1s)

#buffer size
**.OF_Switch.bufferCapacity = 3712

**.OF_Switch.serviceTime = 0.000035s * (${switchServiceTimeFactor= 1,130,140,150,152,154,156,158,

#core speeds
**.controller*.serviceTime = 0.000005556s *(${controllerServiceTimeFactor= 1,130,140,150,152,154,
```

Appendix 3 Code for creating a mobile network

```
[Config MobiCoreNetwork]

network = lte. simulations. networks. MobiCoreNetwork

Codes used to create a cell tower and associate UEs
*. numUe = ${numUEs=200,400,600,800,1000}
#============= Amount of applications ================
*. ue [*]. numUdpApps = 1
*. server. NumUdpApps = ${numUEs}
# connect each UE to the eNB
**.ue[*]. macCellId = 1
**. ue[*]. masterId = 1

#============= Positioning and mobility ============
*. eNodeB. Mobility.initFromDisplayString = false
*. eNodeB.mobility. initialX = 300m
*. eNodeB.mobility. initialY = 300m
*. ue [*]. mobility. constraintAreaMaxX = 600m
*. ue[*]. mobility. constraintAreaMaxY = 600m
*. ue [*]. mobility. constraintAreaMinX = 0m
*. ue [*]. mobility. constraintAreaMinY = 0m
*. ue [*]. mobility. initFromDisplayString = false
*. ue [*]. mobility. InitialX = uniform(0m,600m)
*. ue [*]. mobility. InitialY = uniform(0m,600m)
*. ue [*]. mobility. Speed = 1mps
*. ue [*]. mobilityType = "LinearMobility"
#------------------------------------#
```

Appendix 3 Results tables in averages



	application	routing protocol	ues	jitter	PDR (mbs)	throughput	end to end delay
20	interactive gaming	ospf	200	31	69	300	153
20	VoiP	ospf	400	33	70	390	157
20	streaming audio	ospf	600	37	71	396	163
20	IPTV	ospf	800	45	75	400	170
40	interactive gaming	ospf	200	33	72	302	155
40	VoiP	ospf	400	36	73	400	164
40	streaming audio	ospf	600	40	77	415	180
40	IPTV	ospf	800	41	80	425	183
60	interactive gaming	ospf	200	36	81	431	189
60	VoiP	ospf	400	44	83	531	193
60	streaming audio	ospf	600	47	83	540	197
60	IPTV	ospf	800	52	90	589	199
80	interactive gaming	ospf	200	53	100	563	200
80	VoiP	ospf	400	60	110	643	208
80	streaming audio	ospf	600	63	140	653	222
80	IPTV	ospf	800	68	180	660	231

	application	routing protocol	ues	jitter	PDR (mbs)	throughput	end to end delay
20	interactive gaming	openflow	200	21	77	394	120
20	VoiP	openflow	400	23	79	394	122
20	streaming audio	openflow	600	28	81	400	125
20	IPTV	openflow	800	30	83	400	125
40	interactive gaming	openflow	200	25	78	400	121
40	VoiP	openflow	400	29	83	440	122
40	streaming audio	openflow	600	30	89	500	122
40	IPTV	openflow	800	31	89	557	125
60	interactive gaming	openflow	200	32	89	600	126
60	VoiP	openflow	400	34	90	613	128
60	streaming audio	openflow	600	39	93	625	130
60	IPTV	openflow	800	40	100	640	150
80	interactive gaming	openflow	200	41	102	643	160
80	VoiP	openflow	400	43	108	680	163
80	streaming audio	openflow	600	43	123	700	169
80	IPTV	openflow	800	51	140	713	170