Republic of Iraq

Ministry of Higher Education and Scientific Research

University of Babylon

College of Information Technology

Department of Information Networks

SLICING SCHEME WITH TRAFFIC-AWARE QoS ENSURED FOR SDN/NFV-5G NETWORKS

A Thesis

Submitted to the Council of the College of Information Technology, the University of Babylon in Partial Fulfilment of the Requirements for the Master of Science in Information Technology – Information Networks

By

Adian Rasmi Hasan Bashir

Supervised by

Asst. Prof. Dr. Firas Sabah Salih Hadi

2022 A.D. **1444 A.H.**

﴿إِنْ أُرِيدُ إِلَّا الْإِصْلَاحَ مَا اسْتَطَعْتُ ۚ وَمَا تَوْفِيقِي إِلَّا بِاللَّهِ ۚ عَلَيْهِ تَوَكَّلْتُ وَإِلَيْهِ أُنِيبُ﴾

صدق الله العلي العظيم

سورة هود

الآية88

Supervisor Certification

I certify that the dissertation entitled “Slicing scheme with traffic-aware QoS ensured for SDN/NFV-5G networks” was prepared under my supervision at the department of Information Networks / College of Information Technology/ University of Babylon as partial fulfilment of the requirements of the degree of Master in Information Technology

Signature:

Supervisor Name: Dr. Firas Sabah Al-Turaihi

Title: Asst. Professor

Date: / /2022

The Head of the Department Certification

In view of the available recommendations, I forward the thesis entitled “Slicing scheme with traffic-aware QoS ensured for SDN/NFV-5G networks” for debate by the examination committee.

Signature:

Name: Prof. Dr. Saad Talib Hasson

Title: Professor

Head of Information Networks

Date: / /2022

Abstract

The upcoming Fifth Generation (5G) mobile network brings along a huge challenge regarding Quality of Service (QoS) that the current technologies cannot meet. The present study works towards increasing the QoS within fair Service Level Agreement (SLA) constraints, due to the difficulty in achieving QoS within SDN/NFV- enabled 5G networks. Using Network Slicing (NS) technology and the Dijkstra algorithm were adopted to optimize the uses of network resource allocating and the QoS values. The network has been tested before and after running the Dijkstra algorithm and network slicing, with the main focus on QoS parameters like bandwidth, delay, jitter, latency, and packet loss among nodes within the network.

For the test bed, the Mininet simulation platform is used in addition to the custom network topology, nodes, and other details whose management is controlled by the RYU controller during the SDN experiment. Finally, iPerf is used to evaluate the network performance.

The resulting data indicates that a significant increase has been observed with regard to the network performance after the algorithm and network slicing were run., The network bandwidth was increased and reduced delay and jitter.

The network utilization has been improved and the traffic is controlled and managed using Network Slicing because each slice has its own set of requirements to meet the demands of various use cases. The technology efficiently allocates network resources for maximum cost-efficiency.

Acknowledgments

(In the name of Allah, the most gracious, the most merciful)

Foremost, I thank Allah (SWT) And I dedicate this thesis to the God Almighty and the Prophet Muhammed (P.B.U.H) and Imam Mahdi (May God hasten his reappearance).

Secondly, I wish to express my gratitude to my supervisor, ***Dr. Firas Sabah Al-Turaihi****, who initiated scientific research, for his support in the course of this study, and his patience, assistance, and motivation.*

I would like to sincerely be grateful to my husband for his unwavering support, motivation, and patience in this phase. And my son (Abdullah)

To My late parents, May Allah grants their souls al-Jannat ul Firdaus.

Finally, not least of all, I owe so much to my family for their unconditional love and confidence. As I cannot thank any of you by name because it will take a lifetime but would like to let you know that without your prayers and blessings; and heartfelt love and support. I could never have finished this thesis. So, thank you all for that.

Author's Declaration

This thesis has been submitted to the University of Babylon to fulfil the Master of Information Technology-Information Networks degree requirements. It has not been submitted to any other university for the same degree. The work described here is all mine, except for expert and summary summaries with properly cited sources.

Signature:

Name: **Adian Rasmi Al-Khafaji**

Date: **/ / 2022**

List of publications and participations in student conferences

1. **Published in"1st. Babylon International Conference on Information Technology and Science 2021 (BICITS 2021)- Babil- IRAQ".**
   - R. Alkhafaji and F. S. Al-Turaihi, "Slicing scheme with traffic-aware QoS ensured for SDN/NFV-5G networks," 2021 1st Babylon International Conference on Information Technology and Science (BICITS), 2021, pp. 327-331, doi: 10.1109/BICITS51482.2021.9509901.
2. **Accepted in "8th International Conference on Contemporary Information Technology and Mathematics (ICCITM)"**
   - Traffic-aware QoS guaranteed SDN/NFV-5G Network with Multi-Layer Network Slicing and Resource Allocation.

List of Content

List of Figures

List of Tables

Abbreviations

5G	Fifth Generation mobile network
CAPEX	Capital Expenditures
CLI	Command-Line Interface
eMBB	Enhanced Mobile Broadband
EMS	Element Management System
FDD	Frequency Division Duplex
Gbps	Gigabits Per Second
GNMN	Next-Generation Mobile Network
GSM	Global System for Mobile
HD	High-Definition
ICMP	Internet Control Message Protocol
IP	Internet Protocol
Kbps	Kilobits Per Second
LTE	Long-Term Evolution
MAC	Media Access Control
MANO	Management and Orchestration
Mbps	Megabits Per Second
MMS	Multimedia Messages
mMTC	Massive machine-type communication
NFV	Network Function Virtualization
NFVI	Network Function Virtualization Infrastructure
NS	Network Slicing
OF	OpenFlow
OPEX	Operational Expenditure
OS	Operating System
OSPF	Open Shortest Path First
OSS/BSS	Operational Support System / Business Support System
PPP	Public-Private Partnership
QoE	Quality of Experience

QoS	Quality of Service
RAN	Radio Access Network
SCTP	Stream Control Transmission Protocol
SDN	Software-Defined Networks
SLA	Service Level Agreement
SMS	Short Message Services
TCL	Tool Command Language
TCP	Transmission Control Protocol
TDD	Time Division Duplex
UDP	User Datagram Protocol
uRLLC	Ultra-reliable low-latency communications
VIM	Virtualized Infrastructure Manager
VLAN	Virtual Local Area Network
VM	Virtual Machine
VNF	Virtual Network Functions
VNFM	VNF Manager
VTN	Virtual Tenant Networks
VX LAN	Virtual Extensible Local Area Networks
Wifi	Wireless Fidelity.
Wi-Max	Wireless Inter-Operability for Microwave Access.

Chapter One

Introduction

Chapter 1 Introduction

1.1 General Overview

The huge amount of data in Fifth Generation (5G) networks makes network administration and operation more complicated than before [1]. This creates the necessity of redesigning the network model to discover a solution that allows managers to perform management activities without having to set up the network's devices manually, Network administration needs to be done while eliminating the onerous effort of manually inspecting and validating routing and Quality of Service (QoS) parameters in each network node. With such a wide range of potential applications, new technologies like Software Defined Networking (SDN) and Network Function Virtualization (NFV) are being brought to 5G to help accomplish a wider range of services and applications [2].

The SDN can provide a controller-centred network administration mode by decoupling the control and data planes, thereby allowing network managers to configure their networks remotely and flexibly. On the other hand, NFV is a type of abstracting technique for virtualizing network resources, in a similar way to the abstraction of storage and computing resources within the cloud. In this instance, numerous independent virtual networks can share the physical network nodes and connections, allowing them to function on top of the shared physical infrastructure at the same time.

The integration of SDN and NFV into 5G mobile communication provides unique benefits, including flexibility in service deployment, cost reductions, improved service quality, and efficiency in approaching different scenarios. Network Slicing is one of the fundamental 5G features that enable different services, organizations, and business verticals to be embedded into one network that meets its resources to their business and technological requirements. [3] According to ITU-T there are many 5G services, that are

grouped into three primary sets: enhanced Mobile Broad-Band (eMBB), massive Machine-Type Communications (mMTC), and Ultra-Reliable and Low-Latency Communications (URLLC) [4]. Varying 5G services demand different levels of QoS in terms of latency (jitter), bandwidth, delay, and packet loss.

1.2 Research Challenges

When observing the relevant literary works, it is noticed that QoS achievements in SDN/NFV enable 5G networks to remain a major challenge due to the following issues:

1. Traffic density variations from one slice to another, leading to load imbalances.
2. Load imbalance between network elements leads to the degradation of QoS.
3. The lack of traffic priority knowledge results in insufficient QoS values.

1.3 Aim and Objectives

This thesis aims to achieve the required level of quality of services (QoS) for designing Slicing Scheme with Traffic-Aware QoS Ensured For SDN/NFV 5G Networks.

- **The following thesis objectives are to be achieved:**

1. Using the Dijkstra algorithm to provide a higher bandwidth, and reduce jitter, latency, and delay.
2. Using Network Slicing for controlling and managing traffic.
3. A comparison of the results before and after the proposed work will be conducted.

1.4 Thesis Structure

This thesis is structured in the following way:

1. Chapter Two the Literature Review, which demonstrates various studies in the study field and some associated existing techniques and concepts about the present 5G wireless network development. A full overview of the architecture of SDN and NFV is presented. Furthermore, the current popular state of the art in network slicing is discussed.

2. Chapter Three describes the reliable environment and the proposed architectural design for implementation.

3. Chapter Four describes the performance results after implementation. Furthermore, the outcomes are analysed, compared the results before and after proposed work, and discussed.

4. Chapter Five states the conclusions that have been obtained after completing the work and discussing it. It also includes several recommendations for improving the system in the future.

Chapter Two

Related Works

And

State of Art

Chapter 2 Related works and State of Art

Overview

In this chapter will introduce related works background of 5G technologies SDN-NFV and Network slicing

2.1 Related works

Network virtualization is the foundation of the idea of network slicing. By sharing a single substrate network with numerous heterogeneous and service-specific virtual networks, network virtualization enables flexible and dynamic network administration. Software-defined networking (SDN) and networking function virtualization (NFV) will implement network slicing [5].

1. Foukas et al. [6] described a network slicing architecture with five key parts: the data layer, the multi-domain network operating system layer, the service layer, the administration layer, and the control layer.
2. Costanzo et al. [7] proposed a slicing control prototype in the cloud RAN, which is considered a reference architecture for 5G networks. Prototypes use the FlexRAN SDN controller and the Open-Air Interface platform.
3. To enable resource sharing and autonomous resource orchestration in a flexible 5G architecture, Simon et al. [8] recommend that SDN and NFV be included in the design of the building to allow the coexistence a range of services, and rapid development of new services. A similar concept to network slicing is resource slicing. Service called "Slice" (SlaaS)is a service that allows users to configure resources on-demand.
4. Nikaein et al., [9] Novel segment-based 5G architecture that uses NFV, SDN, and cloud computing has been proposed. They developed the components required to enable network slicing and showed a proof-of-concept. This work's network store technique can make it possible to slice a 5G network dynamically. With the use of SDN, NFV, and network slicing, the authors of the 5G NORMA project aim to create an

end-to-end, multi-user, multi-service 5G mobile network architecture. The implementation of 5G network splitting is also being actively investigated by mobile operators, device makers, and open-source communities. The schemas offer practical instructions for developing flexible and durable RAN segments.

5. Afolabi et al. [10] presented a detailed study that explains the key concepts for network slicing, illustrates use cases, and identifies the techniques that enable resource sharing and identify the main obstacles to their implementation. This article also highlights the importance of enabling technology, including as SDN and NFV, in developing the network slicing concept for telecom networks.
6. Ibarra-Lancheros et al [11] evaluated the network slicing technique's quality of service standards for 5G networks built on SDN. By distributing network slices to user profiles on particular topologies, the open-source controlled Floodlight allots bandwidth.
7. By utilizing SDN capabilities, Kumar et al. propose a special structure for reducing overheads and managing such real-time network flows. NFV allows dynamic services to be built and customized for specific situations, as well as assigning flexible features to each slice [12] .
8. Shuqi et al, have presented a network virtualization (NFV) and centralized control (SDN) scheme based on 5G NS technology to investigate the pre-slices network scheme [13] .
9. Li et al. discussed how SDN and NFV can dynamically allocate virtual network resources like network bandwidth, server processing power, and network element processing power to create core network slices for specific service requirements [14].
10. A management and orchestration framework that integrates SDN/NFV and is based on VTN was presented by Nikaein et al. for dynamically deploying Virtual Tenant Networks (VTN) [15] .

11. Richart et al. examined SDN and NFV for network slicing and resource slicing in virtual wireless networks [16].
12. However, despite the fact that Group et al. and Kim et al. described 5G network slicing in terms of SDN/NFV, these studies are limited in at least one of the following ways. (1) only provide a cursory review of 5G network slicing standardization efforts, (2) they neglect to highlight current research initiatives, current state-of-the-art efforts, and challenges, and neither provide specific research recommendations on how SDN, NFV, and Cloud/edge computing are used to accelerate and realize the benefits of 5G network slicing. As well, (3) they do not cover important aspects of SDN and NFV for 5G network slicing, such as different architectural strategies, implementations, and deployment methods[17],[18] .

State of Art

2.2 Software Defined Networking (SDN)

Conventional networks have a variety of hardware components, mainly routers, switches, and firewalls. These devices provide the hardware functionality through which data is transferred and the software components that monitor the data flow through the hardware. Such networks perform similarly to closed structures with limited control over their nodes. The key impediment to the evolution of existing networks is the fact that the user plane and control plane are coupled. This ensures that each node in the network takes its own forwarding decisions. As a result, designing new applications and functionalities in network elements becomes more complicated, as all changes in applications and/or functionalities must be configured to all concerned infrastructure elements.

In terms of function, Software-Defined Networking (SDN) allows network access programmatically, thereby enabling automatic control and orchestration techniques through the implementation of configuration policies across various entities such as a router, switch, or server; and the decoupling of the applications which execute such operations from the network devices' operating systems. In other terms, by supplying the network with common software control, this new paradigm decouples the control and data planes, that is, the physical infrastructure from the logic control. Because of this decoupling, direct virtualization is possible, which makes the introduction of new protocols and management simple and relevant [19].

SDN is also an architecture that enables logically centralized knowledge and distributed control, with regulations that dictate forwarding rules centralized and real forwarding rule processing distributed across multiple platforms. Application policy computation (e.g., QoS, access control lists, and tunnel creation) occurs locally in real-time in this model, while policy quality, protection, and monitoring are handled centrally, after which they are shifted to switching/routing nodes. Such a process allows for greater network stability, power, and scalability, as well as the use of models, variables, various user databases, and policies [20].

2.2.1 Software Defined Network Architecture

The SDN architecture is divided into the following components [21], as shown in Figure (2.1):

1. **Application layer:** In this layer, various programs are running and communicating with the controller layer via the Northbound interface to keep the controller up to date on network demands.

2. **Controller layer:** This logical entity's function is to translate requests from the application layer to the underlying data plane and to prepare a network description for the application layer using statistics and events.
3. **Forwarding layer:** These logical components, which are special switches, can be configured via the network's controller layer, exposing visibility and immense influence over their advertised forwarding and data processing capacities. They interact with the controller layer through the Southbound interface [22].

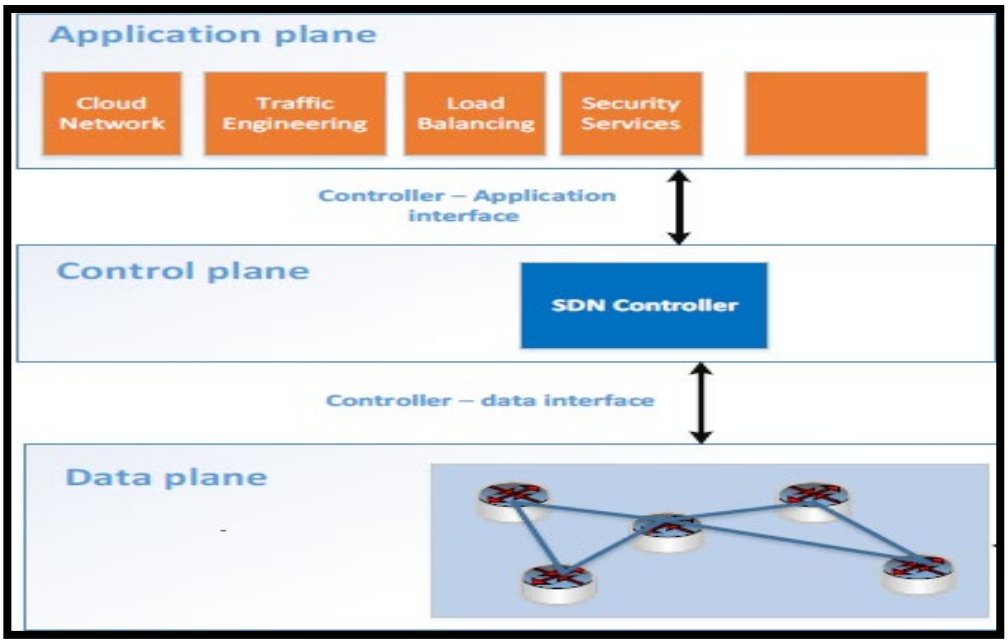


Figure 2.1 SDN Architecture [23]

2.2.2 SDN Controller

The SDN Controller is the location where network management takes place. It is the network's "brain" that is responsible for establishing packet processing rules, followed by the development of the entire network switching strategy. In other words, the SDN controller serves as a strategic point for managing the flow control between the switches/router's underneath (through the southbound APIs) as well as the applications and business logic above (through the northbound APIs) for implementing intelligent networks. SDN controllers encourage automatic network administration and make it easy to

implement and maintain enterprise systems by removing the control plane from network hardware and operating it as software instead [24].

An SDN controller platform is usually made up of pluggable modules which could handle various network tasks. Such fundamental activities could involve inventorying the network equipment and its capacities, and collecting network traffic. Extensions that improve flexibility and enable more sophisticated features may also be inserted, like running algorithms for the performance of analytics and the orchestration of new rules across the network. As packets require specialized and complicated handling, they can be managed by the SDN controller, which can make the appropriate decision. There are many types of controllers, as summarized in the table 2.0.1 below. As for the work presented in this thesis, the Ryu controller has been adopted.

Table 2.1 Types of SDN Controllers [12]

Types of Controllers	Programming Language	Platform Support	Southbound Interface	Northbound Interface	Open stack Support
ONOS	Java	Linux, MAC OS, And Windows	OF1.0, 1.3, NETCO NF	REST API	N
Open-Day-Light	Java	Linux, MAC OS, And Windows	OF1.0, 1.3, 1.4, NET- CONF/ YANG, OVSDB, PCEP, BGP/LS, LISP, SNMP	REST API	Y
NOX	C++	Most Supported On Linux	OF 1.0	REST API	N
POX	Python	Linux, MAC OS, And Windows	OF 1.0	REST API	N
RYU	Python	Most Supported On Linux	OF 1.0, 1.2, 1.3, 1.4, NO NF, OF- CONFIG	REST For Southbound	Y
Beacon	Java	Linux, MAC OS, And Windows	OF 1.0	REST API	N
Maestro	Java	Linux, MAC OS, And Windows	OF 1.0	REST API	N
Flood-Light	Java	Linux, MAC OS, And Windows	OF 1.0, 1.3	REST API	N
Iris	Java	Linux, MAC OS, And Windows	OF 1.0, 1.3, OVSDB	REST API	N
MUL	C	Most Supported On Linux	OF 1.4, 1.3, 1.0, OVSDB, OF- CONFIG	REST API	Y
Runs	C++	Most Supported On Linux	OF 1.3	REST API	N

A. RYU Controller

Ryu is widely known as an open-source software application built on a component. It is fully implemented and supported by NTT labs in Python. Ryu, like other SDN controller systems, offers software components that have well-defined APIs to which developers have access to enable them to build new applications for controlling and managing networks. A major benefit of Ryu is the fact that several southbound protocols for system management, including OpenFlow, Network Configuration Protocol (NETCONF), OF-Config (OF), and other protocols, are supported [25]. This component-based approach enables enterprises to tailor deployments to their requirements; engineers can rapidly and efficiently change existing modules or create their own for ensuring the satisfaction of the evolving application demands through the underlying network.

B. Architecture of Ryu

Ryu, like every other SDN controller, can build and send OpenFlow messages, listen for asynchronous events like removed flow, and parse and manage incoming packets. The Ryu Controller framework's design is depicted in Figure (2.2) below:

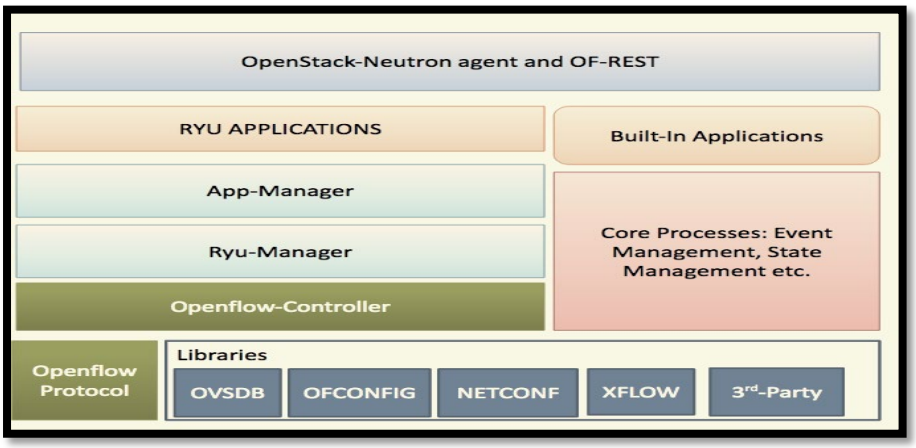


Figure 2.2 Ryu Architecture [26]

C. The OpenFlow protocol

OpenFlow is one approach to the implementation of SDN. It specifies a coordination protocol among controllers and the multiple packets forwarding components of the network infrastructure, representing the control and data planes respectively. The majority of modern switches and routers have a kind of hardware flow table which allows the processing of packets at line speed. In addition, OpenFlow is built on the idea of flow tables, whereby every table entry is identified through a match with a related operation. The architecture of the match fields, as well as the potential actions, is determined by a list of standard packet headers and actions supported by most switches. By doing so, the OpenFlow designers aimed to lower the threshold to switch vendors adopting the protocol. Figure (2.3) depicts the OpenFlow protocol's supported match fields.

In Port	VLAN ID	Ethernet			IP			TCP	
		SA	DA	Type	SA	DA	Proto	Src	Dst

Figure 2.3 OpenFlow packet header fields supported [22].

As an OpenFlow-enabled switch receives a packet, it aims to compare the packet's headers to the flow table entries included. In case it fits none of the previous entries, the packet information is sent to a dispatcher, who decides what to do about it. Whenever the controller decides on a suitable response for the packet, the action is sent back to the switch as a directive. Furthermore, the controller may modify the flow table in the switch through new entries which match the packet headers for the next packets in the flow not to be forwarded up to the controller. This allows the switch to perform hardware line-rate packet processing. The overall network traffic management approach is therefore applied within the controller as software algorithms. SDN and

OpenFlow allow the unified control of several switches, whereby the management interface remains consistent for all switches within the network which follow the protocol, regardless of the vendors[27]. There are many OpenFlow versions as shown in Figure (2.4) below.

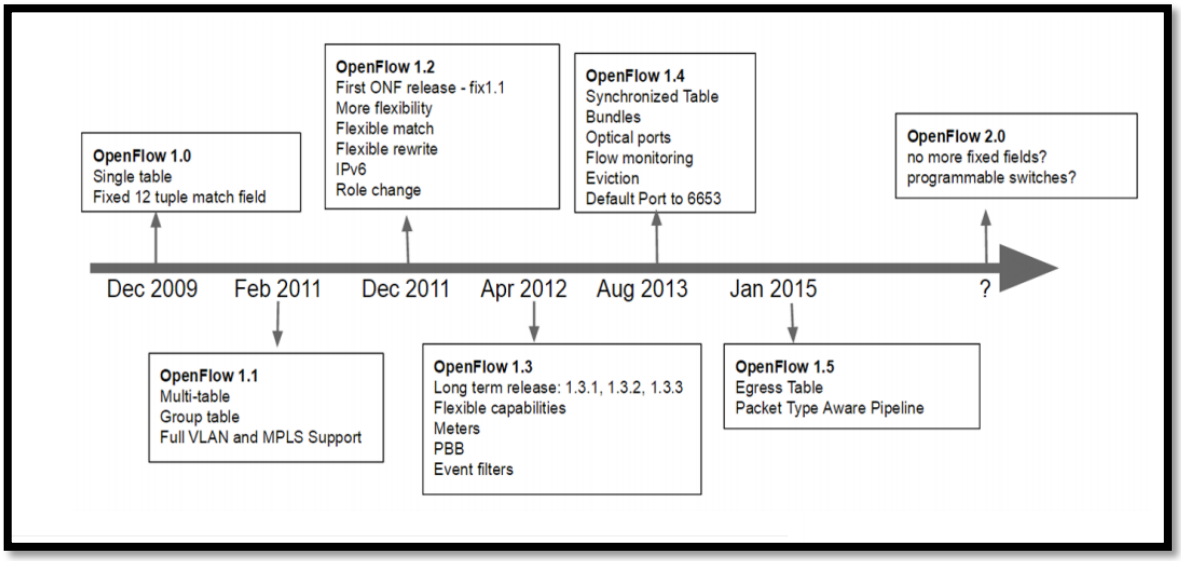


Figure 2.4 OpenFlow Versions [27]

2.2.3 SDN Advantages

Software-Defined Networking can alter the way network engineers and developers develop and run networks to satisfy customer requirements. Networks have been open standards, non-proprietary, and simple to program and maintain since the implementation of SDN. SDN would give corporations and carriers increased opportunities for their networks, allowing them to customize and refine their networks and lower total network costs[28]. Some of the most important SDN advantages are mentioned below:

1- **Centralization:** Centralization can be described as a network that can be programmed centrally. It enables network administrators to set up, monitor, protect, and manage network resources using SDN apps. Furthermore, network intelligence is concentrated (logically) within the SDN controller, thereby maintaining a comprehensive view of the network.

2- **Network management Simplicity:** With SDN, the network can be interpreted and handled as a single node, abstracting complex default network management functions into relatively simple interfaces.

3- **Rapid service deployment:** new features and software can be launched in hours rather than days.

4- **Automated configuration: Manually** configured functions such as VLAN assignment and QoS configuration can be configured automatically.

5- **Open standards-based and vendor-independent:** Given the capability offered through SDN controllers rather than vendor-specific devices and protocols, the network architecture and execution are simplified while still using SDN-based orchestration and management mechanisms to automatically install, customize, and upgrade devices across the existing network.

6- **A higher level of innovation:** SDN could drive progress by abstracting the network, allowing services to be built on top of it without vendor lock-in, and using generic and standard interfaces.

7- **Improved network security and reliability:** Whenever SDN controllers have complete control and influence over the network, they can implement policies such as access control, traffic engineering, QoS, and security.

8- **Network Virtualization:** As storage and server virtualization have become more widely implemented, networks will also benefit from SDN being virtualized.

9- **Reduced operational expense:** By using SDN, the expenses will be driven into the networking software. As a result, hardware manufacturers will concentrate on lowering the cost of physical

equipment. Furthermore, due to various technological implementations, the devices may be modified/upgraded via software rather than replaced with new ones. This lowers CAPEX while further simplifying management, resulting in a decrease in OPEX.

2.3 Network Function Virtualization (NFV)

Generating new services in existing networks is a significant task when these platforms are vertically integrated, proprietary, and mostly implemented in hardware (along with the corresponding software). As a result, they are difficult to update and maintain, and they are limited in terms of mobility. As a result, changes to both hardware and software are required to identify new services, resulting in an improvement in CAPEX and OPEX. Network innovation offers the ability to reduce such problems by introducing newly emerging innovations that allow us to have more scalable and less manufacturer-dependent networks that use open standards while still decreasing CAPEX and OPEX [29].

As a workaround, Network Function Virtualization (NFV) will address the aforementioned issues. NFV is a modern IT virtualization platform that capitalizes on developments in dynamic cloud infrastructure and SDN. NFV aims to move network functions away from specialized hardware and tends toward program patterns that operate on a general-purpose virtualized network. End users can scale out efficiently to follow evolving traffic and service use levels by moving intelligence and workloads into applications. Furthermore, NFV enables the use of various management points in networks through the use of programmed applications running on various platforms [30]. As a result, standardization and compatibility of these management points with multi-

vendor solutions seem to be applicable. Based on what has been mentioned so far, the following are typical characteristics of virtualized networks:

1- **The Dynamic Provisioning of Services:** Network operators can automatically scale out NFV output on demand by providing specific granularity control, based on the network's current state.

2- **Hardware and Software Decoupling:** The isolation between hardware and software creates the possibility of separate evolution for each section.

3- **Flexible Deployment of Network Functions:** NFV is capable of dynamically implementing network functionality on a collection of hardware components that can perform several tasks at varying times in different data centres.

2.3.1 Network Function Virtualization Architecture

As shown in Figure 2.5 below, the NFV architecture consists of the following sections:

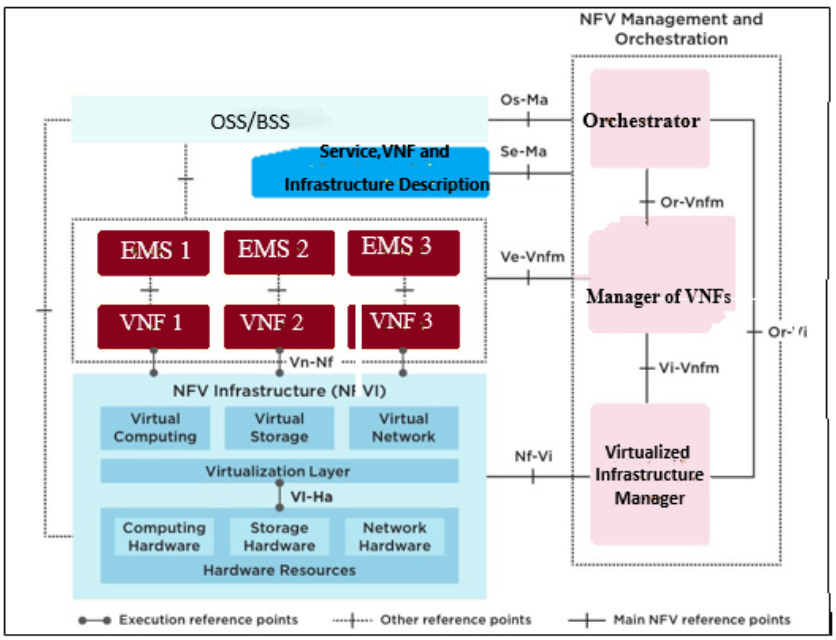


Figure 2.5 Architecture of NFV [30]

1- **VNF (Virtualized Network Functions)**: NFV is implemented through the virtualization of network function components such as Virtualized Network Functions (VNF) [31]. It is worth noting that even though only one sub-function of a network entity is virtualized, it is always referred to as a VNF. A VNF generates a network feature, and when many VNFs are combined, a virtualized network section is implemented. Virtualization is possible for three types of devices:

   - Network feature equipment (routers, switches, Access Points, etc.).
   - Network-connected storage (file servers and databases).
   - Network-connected IT computers (firewalls, network device management systems, etc.).

2- **EMS (Element Management System):** This is the VNF element management (EM) model. The EM is in charge of the VNF functional management, such as Faults, Configurations, Accountings, Performances, and Security Management. The VNFs can be managed by the EM using proprietary interfaces.

3- **Manager of VNFs (VNFM):** A VNF Manager, as the definition indicates, handles a VNF or a group of VNFs. It manages VNF instances during their life cycle. The VNF life cycle maintenance includes configuring, managing, and decommissioning VNFs. It is necessary to clarify that the EM is in charge of managing functional components, whereas the VNFM is in charge of managing virtual components [32].

4- **Network Function Virtualization Infrastructure (NFVI):** The environment in which VNFs run is referred to as Network Function Virtualization Infrastructure (NFVI). Both the physical and computational infrastructures, as well as the virtualization layer, are all part of NFVI.

5- **Virtualized Infrastructure Manager (VIM):** The administration entity for NFVI is VIM. It is in charge of monitoring and maintaining the NFVI computing, network, and storage services within the infrastructure domain of an operator. Furthermore, it is in charge of gathering performance metrics and events.

6- **Orchestrator of NFV:** It creates, manages, and deletes VNF network facilities. If there are several VNFs, the orchestrator will enable the construction of a service that spans several VNFs. Furthermore, the NFV Orchestrator is in charge of the global resource control for NFVI services.

7- **Operational Support System/Business Support System (OSS/BSS):** In NFV architecture, OSS/BSS corresponds to an operator's OSS/BSS. The OSS is in charge of network administration, error management, configuration management, and operation management, while BSS is responsible for customer service, inventory management, and order management. Using common interfaces, an operator's existing OSS/BSS can be combined with the NFV Management and Orchestration.

2.3.2 The Advantages of NFV

There are several advantages related to the application of NFV. The most prominent ones are described below.

1- **Lower CAPEX:** The most obvious advantage of NFV is that it reduces the need to buy expensive specialized networking hardware. Instead, companies can fulfil their basic requirements by using commodity servers. These servers typically cost a fraction of the price of expensive appliances, and in most situations, the network may be extended/modified easily by connecting additional VMs to the infrastructure's current hardware. Furthermore, since companies need

stable and uninterrupted connectivity for their networks, they incorporate continuity into their networks if a certain piece of hardware fails by switching resources to other hardware components when required [33].

2- **Reduced OPEX:** NFV can reduce network maintenance time by centralizing network management (especially when used in combination with SDN). Furthermore, NFV implementations can be more cost-effective, which results in lower utility costs.

3- **Greater flexibility:** Since costly equipment for different functions is no longer needed, it is much easier for companies to deploy new functions. NFV also makes it faster and less expensive for enterprises to experiment with emerging network feature technology.

4- **Enhanced scalability:** NFV enables companies to expand their networks optimally. They would no longer need to purchase expensive network hardware for multiple network functions. It means that adding additional network bandwidth is much more affordable because it demands much less time and effort on behalf of the network administrators.

5- **Improved security:** Adding new protection technologies to the network becomes much less costly in an NFV environment, making it easy to keep up with cybercrime.

6- **Enhanced Customer Experience:** Another beneficial aspect of the implementation of NFV considers the end-users. In the event of a hardware breakdown that might otherwise result in outages or slow results, NFV makes it simple to migrate workloads to other hardware without the need to buy new costly equipment. Furthermore, the reduced costs and increased scalability make it easier to meet rising demand. As

a result, telecom carriers may have service level agreements (SLAs), and end-users do not need to be concerned about disruption because the network performance remains high.

7- **Faster Provisioning:** One of the main reasons attracting service providers to migrate to NFV is the opportunity to rapidly introduce additional features to networks. NFV allows service providers to provision new network capabilities on existing networks while leveraging extra VMs to handle the increased workload.

8- **Independence of manufactured Hardware:** Since various universal standards make it easier to have connectivity between various network components, the virtualized nature of NFV allows organizations to have interoperability rather than sticking to any proprietary standards.

2.4. Wireless Mobile Communication

The initial formulation and growth of wireless mobile communication technology began during the eighties of the previous century[34]. Throughout the past decades, cellular mobile innovations underwent several years of technological growth and development. In 4G, the quality of service (QoS) and security tend to be heavily advertised, although the cost per bit is poor. In contrast to earlier network generations, 4G has several drawbacks, including higher power usage (battery consumption) and relatively higher infrastructure expenses used to deploy it. The new generation will be 5G. The main goal is to provide full wireless connectivity with virtually no restrictions. Since six billion users have access to smartphones, the various generations of cellular technology can be dissected. Furthermore, modern wireless networking networks are unwavering in their determination to meet the growing demands of users. In 5G technology, the rates of data calls have been easier as compared to earlier generations because of the excellent and extremely scalable coverage

quality. It has bandwidth control and increased significance and reliability in addition to its declining cost. Such an evolution does not only reflect the strong demand of people globally to collaborate and engage with one another, in addition, to having access to knowledge, but also the enormous steps that creativity has taken to meet the need [35]. With the exponential increase in demand for smart mobiles, all IP-based 4G Long-Term Evolution (LTE) networks grew to be a part of daily life. This led to the emergence of a new set of user-oriented mobile multi-media technologies such as video conferences, live videos, e-healthcare, and online gaming. Moreover, while the industry is preparing for the initial launch of 5G networking for commercial purposes, many seem to be concerned about security threats and dangers. 5G technologies reinforce a large number of mobile devices, which leads to a huge expansion in capacity and the creation of a next-generation threat environment which would inexorably add 5G security problems. Such new technologies meet customer needs while also opening up new market opportunities for remote operators to increase their profits.

The Fifth Generation

The 5G technology is a combination of the aforementioned advantages of cell phones, including high-speed dialling, music recording, cloud data access, and instant high-definition (HD) uploading. The 5G demands the designation of newer radio bands over 20 GHz. 5G networks are now designed to support newer applications like IoT services, and lifeline connectivity through natural disasters [36]. It is designed as an unprecedented device to broadcast huge amounts of information at gigabits per second (Gbps), thereby enabling media news streams and HD TV programs. Even though 4G has only established a short while ago, it tends to be rather insufficient in coping with the different requirements concerning dense networks and expanded bandwidth considerations like the widespread usage of smartphones, considering the data

speeds, coverage, battery lifespan, and IoT. This is not considered to be a fault in 4G, as the mobile revolution had not begun yet at the time the 4G standards and solutions were selected. More recent applications are constantly being created. However, the main aim of 5G is to overcome the existing shortcomings of 4G. It aims towards designing the architecture to satisfy potential needs regarding data rate, speeds, coverage, and battery lifespan to provide a cost-efficient network which could easily undergo scaling [20].

- **5G Technology Advantages**

- Faster data transmission speeds than previous generations
- Abundant memories
- Fast dialling speeds
- HD quality impressions
- More appealing and reliable
- Peak paces for uploads and downloads
- Diagnostics through the internet
- Access speed of up to 25 Mbps
- High-quality services to avoid mistakes
- Bi-directional Broadband

- **5G Issues and Challenges**

There are several major obstacles faced when determining the success of potential networks through the use of affordable innovations which are currently under research and development:

- **The number of supporting equipment:** Several old devices would not be compatible with 5G, so they would all need to be replaced - a costly process.
- **The volume of data**: With the advancement of technology, data volumes of each network are also increasing every year, and this trend is expected to continue. Because many applications have high resolution

video calls, live streaming, downloading, etc., each network needs to support a huge volume of data.

- **The exclusion of technology**. A 5G network implementation also involves a lack of immediate accessibility for average pockets, as well as a delay in its implementation due to a lack of enabling technologies.
- **Upload speeds**: Mobile phone users can enjoy high download speeds with 5G technologies. The 5G network causes phones to produce more heat, so the battery technology on mobile phones needs to improve while using a 5G connection as well.

With the preparation for the initial launch of 5G for commercial purposes, many seem to be concerned about the security threats and challenges to be met. 5G networks will have a large number of mobile users, and significantly increased latency, which will build a hazardous environment for the following generations, thereby undoubtedly posing new security problems [36].

2.5 Network Slicing

Network slicing can be defined as the process of sub-dividing a network into logically isolated sub-networks. The core strategies used here are NFV and SDN, and the resulting network slices will cater to specific connectivity requirements. A network slice, for example, can help in catering lower latency communication with a stability guarantee and may seem to be implemented as a single network resulting in the abstraction from the underlying network technologies. Figure 2.6 depicts the principle of network slicing, in which the physical networks are split into two slices labelled slice 1 and slice 2. The three generic sorts of networks imagined in 5G wireless networks, namely eMBB, URLLC, and mMTC, will be linked in Industry 4.0 [37]. Because of the variations in the requirements of each provider, it is impossible to optimize the network at the same time to adhere to all three types of networks. This problem can be solved by network slicing, in which separate slices are generated for each service, after which the resources are optimized to ensure the optimal

functioning of the service. The URLLC slice, for example, can be used to provide crucial control orders to a computer in case of emergencies, whereas the eMBB slice could be employed for supporting connectivity among mobile robots and the internet for firmware upgrades. Several problems can be found during network slicing deployment in Industry 4.0 environments. Firstly, developing a communication of ultra-reliability and low-latency can be rather problematic due to the difficulty of correctly modelling queuing delays. Secondly, Industry 4.0 networks are of heterogeneity and include several legacy protocols, which in turn complicate end-to-end research [38].

Through network slicing, the network perspectives undergo significant transformations via the abstraction, isolation, orchestration, softwarization, and separation of logical network elements from the physical networking resources beneath it, thereby enhancing the specifications, principles, and capability. Network slicing can be supported by creating a bunch of network resources management plane. They are connected to both the physical and virtual network, and repairs functions where applicable. This leads to the instantiation of the whole network along with its service functions assigned to the slices. As for slice operations, the management planes govern the network resources totally, along with the network and service function assigned to each of the slices. Next, they are reconfigured to be more suitable to the physical property requirements, so as to produce end-to-end services. Ingress routes are especially designed to guarantee that the suitable traffic moves to the slice of relevance [39].

The slice institution can be either business driven or technology driven. The first type involves slices that support various differences and repair features in business cases. The second type, however, are slices formed by a number of resources (physical lor virtual) grouped together to function as subnetworks or clouds. Those slices accommodate service elements and

network functions, either physically or virtually, altogether with network segments like access, core, and edge enterprise network [29].

The operators make use of slicing in networks to alter completely differing services to permit allocating and unleashing network resources consistent with the context and competition policies for the operator. This kind of approach to slice networking will be able to reduce the operation costs. Besides, this process creates potential softwarizing and programmable results, as it enables innovative ideas to be worked out. The latter are needed for completing the services that are offered [40]. Network slicing can also be realized and managed by means of the network softwarizing techniques.

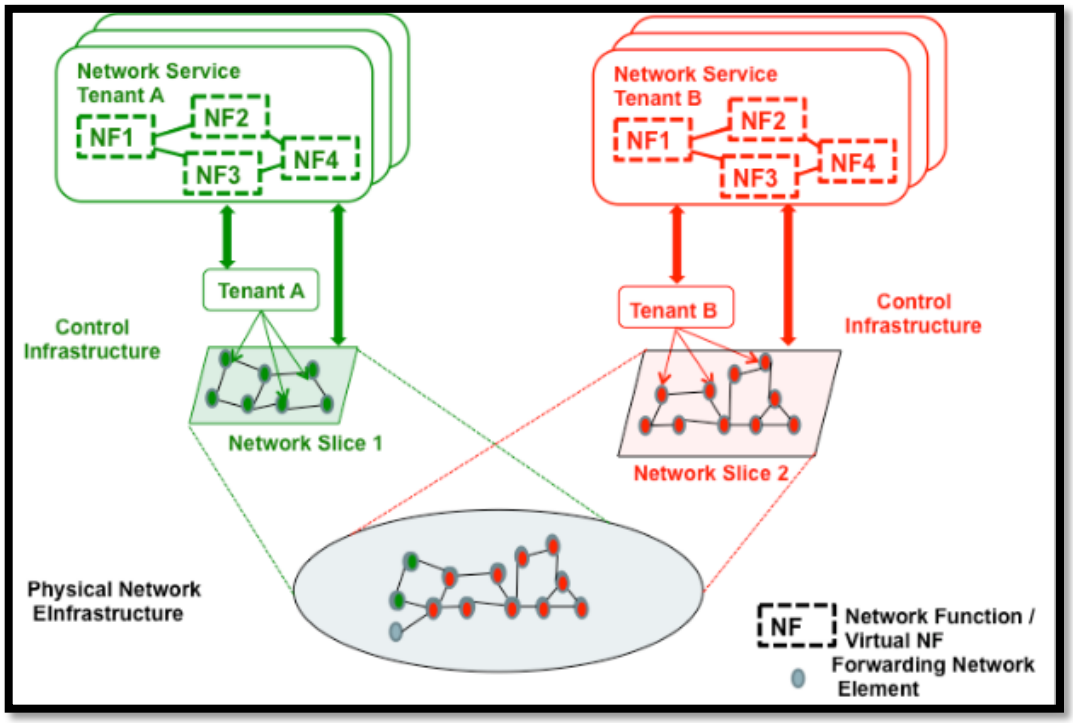


Figure 2.6 Network Slicing [40]

2.5.1 Network Slicing Architecture

The public-private partnership (PPP) technological vision of 5G infrastructures suggests dividing the network structure into 5 layers: service, infrastructure, orchestration, business function, and network function layers. The alliance's next-generation mobile network (GNMN) suggested the division

into 3 levels: business applications, business enabling, and infrastructure resources. Figure 2.7 illustrates the basic 5G network slice structure. [41] This frame has three layers, namely the layer of service, network function, and the layer of infrastructure. It comprises a management and orchestration (MANO) entity in addition to these three layers. The MANO translates service models and uses cases into parts. The network architecture layer addresses both the central and radio access networks' physical network infrastructure. Otherwise, the infrastructure layer would still monitor and allocate services to the slices of infrastructure. The network function layer encapsulation the activities needed for the configuration of network functions and the life cycle maintenance, apart from the infrastructure layer. The functions of this network are then chained to have a complete service [42].

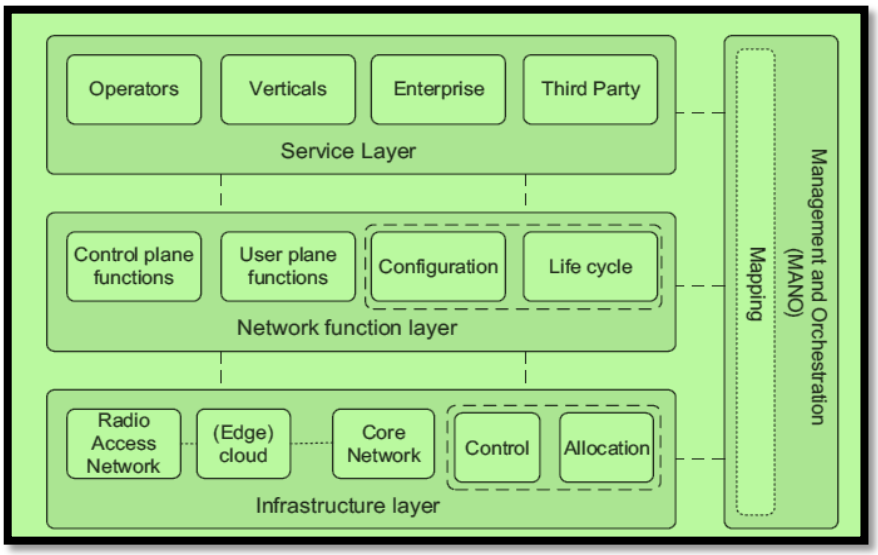


Figure 2.7 Network Slicing Architecture [41]

2.5.2 Life-cycle management and network slicing characteristics

Through network slicing, operators can create networks with a logical division simultaneously, in order to produce enhanced services in various markets cases. Such situations necessitate a wide range of criteria regarding the service features, customized networks, and virtual networks practically (data, control, and managing planes), resources of network performance,

isolations, physical properties, and QoS concerns. Network slices are formed using the required network functions and resources. The latter are selected out of a larger set of resources and (virtual) network functions for the original service or purpose intended [43].

- **The reference framework in NS can be divided into two levels, as follows:**

- The first level is that of the network slice life-cycle management. Int involves the set of states of purposeful activities through which network slices go. They include creating, operating, and deleting the slices
- The second level is that of the network slice instances, such as the activities of network slices, as illustrated in Figure (2.8).

Each of the network slice functions initiated are put into a map based on the appropriate framework levels. This includes the establishing and maintaining functions [44].

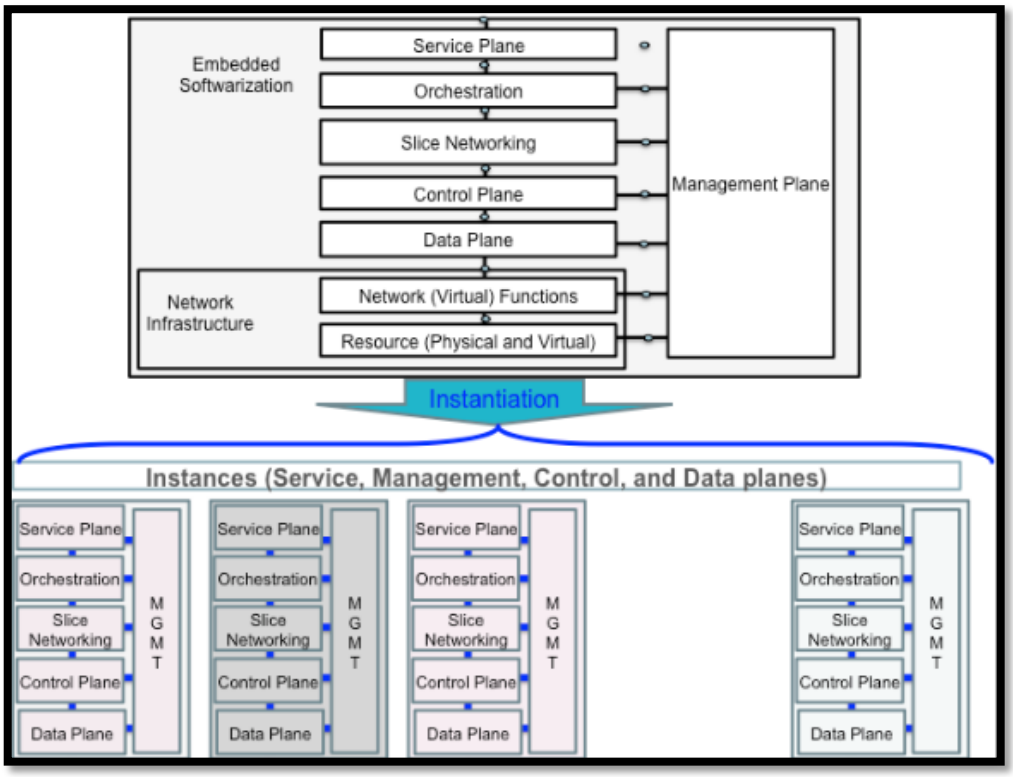


Figure 2.8 Lifecycle of Network Slicing [45]

To develop and use network slice services and activities, it is obvious that all of the life-cycle managing features of network slicing solutions are determined by certain aspects that are to be taken into consideration. The main idea in the structure is the concept of governance tenets. It is authorization with a logic centralization which governs all network slices within a certain domain [45].

- The separation tenet involves the independency of slices, which are isolated to a suitable degree.
- As for the capability exposure tenet, it enables the slices to expose information to third parties about the services that the slices offer. This could be concerning connection, information mobility, and autonomy. The information exposure takes place employing specific interfaces/APIs, yet within the restrictions imposed by operators.

The following points are expected in the quest of solving the aforementioned tenets while maintaining the appropriate qualities within the framework of 5G networks [37]:

2.5.3 Capabilities for Network Slicing:

There are several aspects that network slices are capable of, as follows:

- It ensures that each of the data plane, control plane, management plane, and service plane are isolated. The enablers within the slices provide safety, security, efficiency, and multi-tenancy.
- It provides methods that enable a variety of service requirements in Network Slicing, such as assurance of Service end-to-end QoS inside the slices.
- Network slicing provides recursion, which refers to the segmentation methods that enable a slicing hierarchy with parent-child connections.
- It provides the methods and strategies for managing the balance between how flexible and how efficient the slices are.

- The network resources and functions are optimized using strategies for selecting network resources and functions automatically.
- As for monitoring, a single or multi-domain context is used to monitor the network slicing status, behaviour, and interconnection.
- The network slicing capability is exposed via APIs in terms of slice design and interaction, as well as the slice programmability and control [46].

2.5.4 Operation of network slices

- Slice management has a varying range of applications, such as radio network, wire access, core, transport, and edge networks. The operations include creating, activating/deactivating, and protecting the network, in addition to ensuring that the slice model remains flexible, extendible, safe, and scalable.
- The slices could be managed and operated autonomically. This includes slice protocols for self-configuring, self-composing, self-monitoring, self-optimizing, and self-elastic functions of slices.
- The slices are stitched or composed using providing effective techniques for that purpose. These techniques may take a vertical orientation (using service, management, and control planes), a horizontal orientation (through access, core, and edge segments) or a combination of both orientations.
- Slice orchestration is found to be across network elements.
- The network slices are service mapped dynamically and automatically.
- The aforementioned procedures and operations are integrated using effective enablers and strategies[47].

2.6 The Dijkstra Algorithm

The Dijkstra algorithm takes its name from the Dutch researcher Edsger Dijkstra who made the algorithm public in 1959 [48]. This algorithm supports the graphs that are accustomed to resolving obstacles using the single-source

shortest paths. In addition to being used in routing algorithms, it is also used as a subroutine in other graph algorithms. An algorithm finds the shortest path between a given source vertex (node) and every other vertex in a graph with the lowest cost vertex. When the shortest path has been determined from a single vertex to a single destination vertex, the algorithm can be stopped. For example, Dijkstra's algorithm can determine the shortest route between two cities if they share a direct road and the vertices of the graph represent cities. As a result, the shortest path first is widely used in network routing protocols, most notably IS-IS and OSPF. The Dijkstra algorithm has several necessary applications used in daily life [49]. The Google Maps application also makes use of this algorithm in detecting the shortest path between two points. Such algorithms take into account the right general resolution to ensure the improvement of problematic aspects. However, it has been found that this algorithm might have lower performance than alternative solutions in certain situations. An essentially beneficial aspect of this algorithm is the fact that the unwanted nodes need not be visited after reaching the supposed destination nodes.

2.7 Mininet

Mininet can be defined as a network emulator which enables the simulation of massive networks using one system. More than one end-host, switch, router, and connection are used to run one Linux kernel. It makes use of lightweight virtualizing features for displaying one machine appear as a complete network, using identical kernels, systems, and user codes. There are several advantages when using Mininet, as follows:

1. Mininet is a free and open-source project.
2. Custom topologies may be constructed.
3. Mininet executes actual applications.
4. The forwarding of packets could be adjusted.

Unlike the simulator, Mininet tends to execute a genuine conde without modification when connecting to a real network readily. This type of code includes applications, OS kernels, and control planes (both OpenFlow controller and Open vSwitch codes) [48].

2.8 iPerf

iPerf is one of the popular network test tools for determining the bandwidth performance and network quality of Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) [49]. The user may do multiple tests on the network's bandwidth availability, latency, jitter, and data loss by changing different settings with regards to timings, buffer, and protocol (TCP, UDP, SCTP with IPv4 and IPv6). iPerf is open-source software that operates on different platforms like Linux, UNIX, and Windows. The Iperf tool aids in network performance measurement by generating client and server functionality for both source and destination nodes. Figure (2.9) illustrates this process.

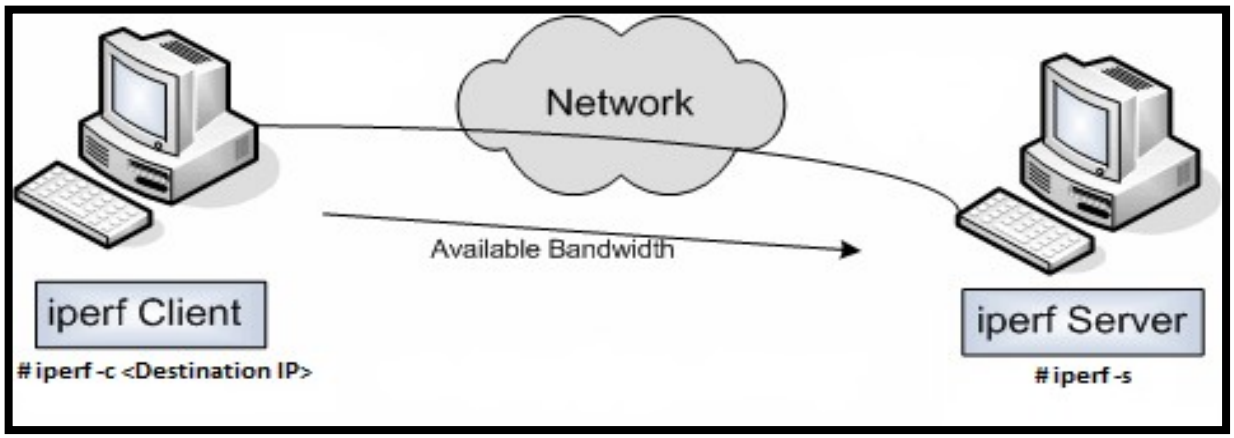


Figure 2.9 iPerf Bandwidth measurement.

Chapter Summary

In this chapter, we summarize the efforts made on this front by presenting different research papers related to resource allocation, quality of services (QoS), and Network Slicing in 5G. We begin by discussing SDN and NFV architectures and then wireless networks with the generations, after that, we explained network slicing.

Chapter Three

System Model

Chapter 3. System Model

3.1 Introduction

This chapter describes the Dijkstra algorithm and network slicing. This algorithm has been chosen for its ability to minimize traffic routing via vulnerable connections while ensuring that the QoS network constraints are satisfied. It also covers the elements and software tools utilized in setting up the test bed in this study.

3.2 Overview of Implementation

In this study, a Linux test-bed was built by utilizing the Mininet software for emulating each of the networks and open-source RYU as an SDN controller, The Python programming language has been adopted for designing the topology and building the short-path algorithm program. iPerf is used for assessing network efficiency. NFV used virtualization via installing a VirtualBox (hypervisor) and installing software that mentions above inside it, the design steps are represented in the diagram below (Figure 3.1).

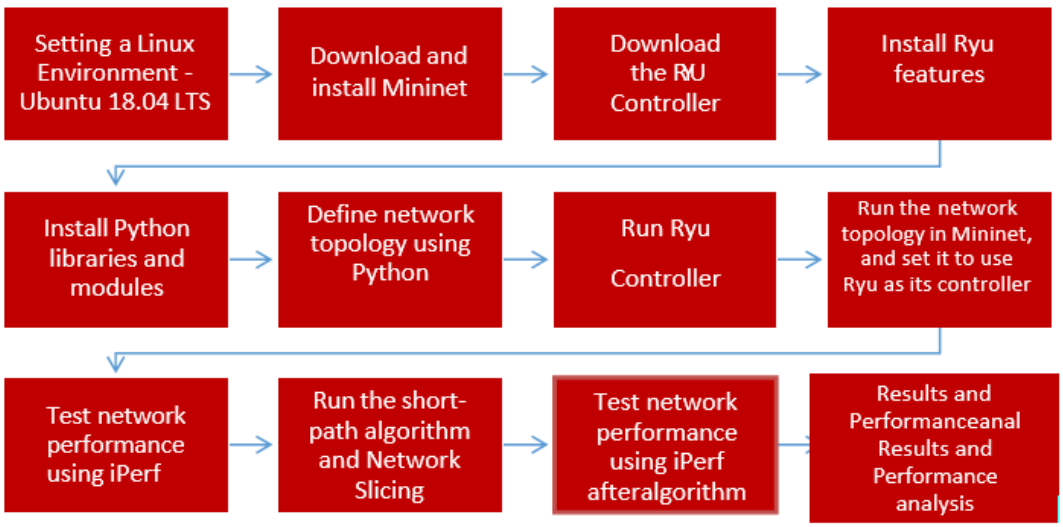


Figure 3.1 Design Steps

3.3 The Proposed Architecture

This work aims to design a framework in which network topology achieves the quality of service (QoS) and minimizes traffic routing by finding the shortest path between sources and destinations using Dijkstra algorithm and network slicing.

Figure 3.2 illustrates the proposed architecture, which involves the separation of the framework into different layers of the network. The parts of each layer represent network components that determine data routing partially. The Mininet simulation environment is installed in the bottom layer (forwarding layer), which contains the network appliances such as 5G devices, routers, and switches. Thus, it will perform the process of network creation. The forwarding layer is responsible for routing network packets. The second layer is the Network Control Layer, where the Ryu Controller is installed, and it is on this layer that routing control is performed (Dijkstra algorithm) as well as the network slicing, as mentioned in the previous chapter. The last layer is the application layer, which handles network functions like firewalls, load balancers, QoS, and routing. The application layer is responsible for communicating with the controller layer through the Northbound interface. APIs or REST protocols are used to implement the Northbound interface. The southbound interface is the place where the network functions are communicated from the controller layer towards the forwarding plane. The OpenFlow protocol version 1.3 is here adopted in the communication between controlling and forwarding layers.

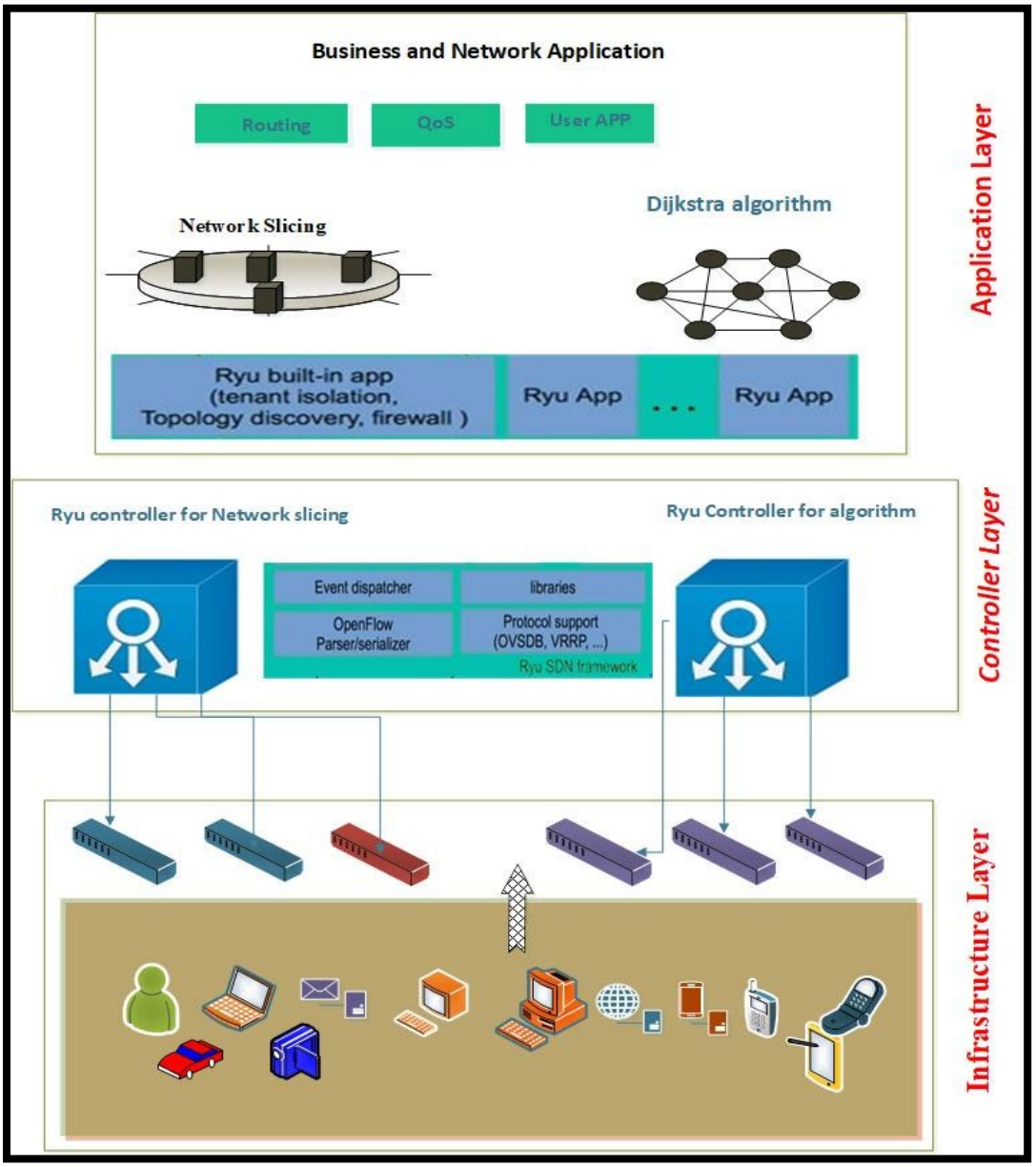
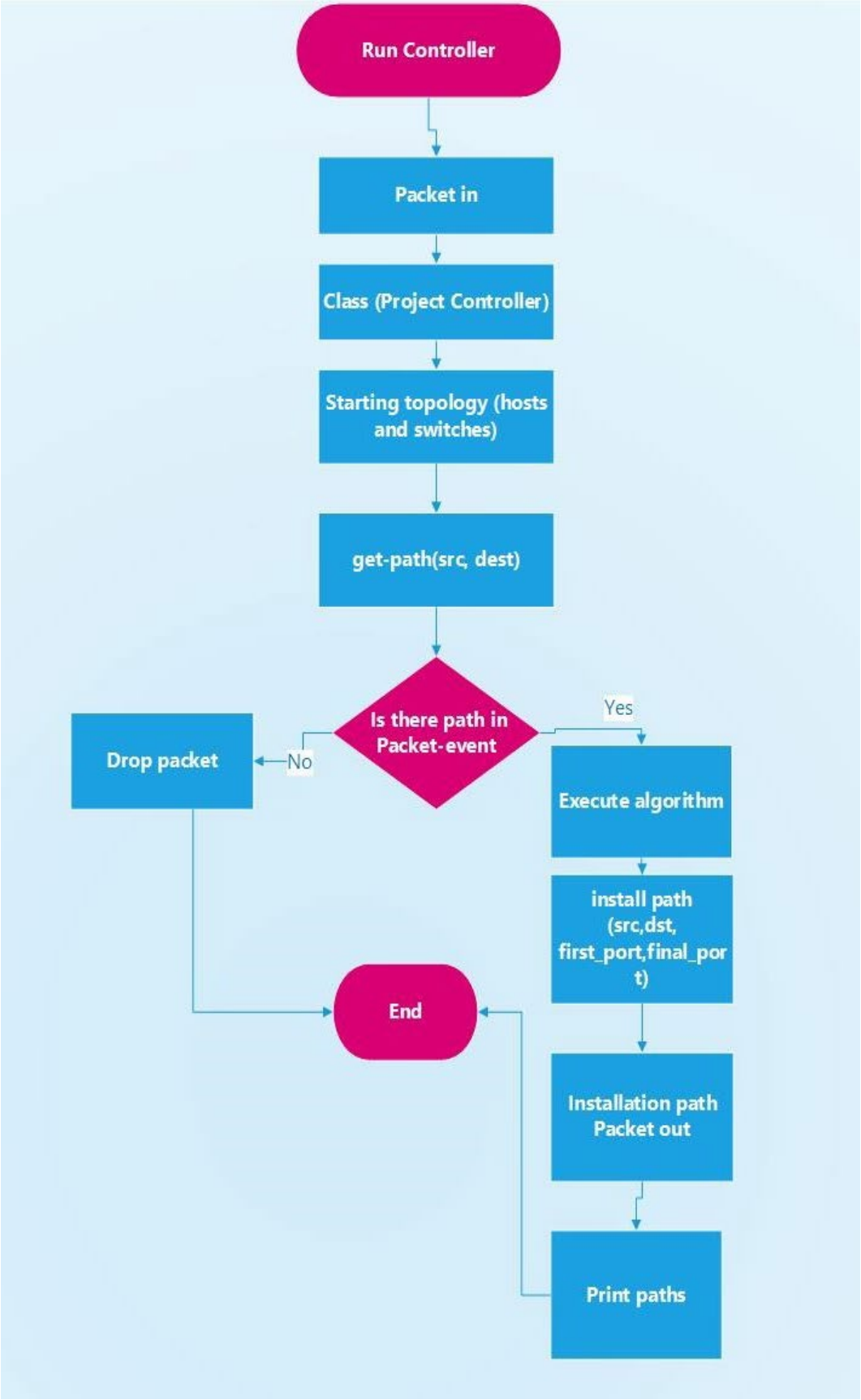


Figure 3.2 Proposed architecture

3.4 The Algorithm Description

The initial phase is when running the Ryu controller to gather operational information about the topology and devices involved, as well as aspects like the IP address, MAC address, port, connection, and so on. The next stage is to use Dijkstra's algorithm in finding the shortest path, as shown in Flowchart (3.1). The Dijkstra algorithm is a path-finding algorithm used for determining the optimum path between two nodes. G = (V, E) is a topological graph theory where V represents vertices/nodes and E represents the edge or connection linking every node. E (u, v) can be written for each edge, whereby u represents the node source and v is the node destination [50]. To find the destination node, the Dijkstra algorithm visits all neighbouring nodes starting with the ones that have the least edge cost. Next, all these nodes are referred to as visited nodes, whereby the edge (u, v) through which it passes is referred to as edge relaxation [51]. After that, paths are constructed between the source (u) and destination (v) node whenever the edge (u, v) is relaxed. The obtained value is an aggregation of that path costs. When all nodes have been visited, the search algorithm's procedure will be complete. Algorithm 3.1 presents the Dijkstra algorithm procedure, as shown below.



Flowchart 3.1 the algorithm work

```
Algorithm 1 the shortest path algorithms
-------------------------------------------------
1.      def min_distance(dist, T):
2.      min = float('Inf')  #float('inf') is used for setting a variable with an infinitely large value
3.      node = 0
4.      for v in T:
5.      if dist[v] < min:
6.      min = dist[v]
7.      node = v
8.      return node
9.      def get_path(src, dst, first_port, final_port):
10.     # Dijkstra's algorithm
11.     print ("get_path is called, src=%s dst=%s first_port=%s final_port=%s" % (src, dst, first_port,
final_port))
12.     dist[src] = 0             # Distance from source to source
13.     T = set(switches)   # Start off with the source node
14.     while len(T) > 0:    # The main loop
15.     u = min_distance(dist, T)   # vertex in Q with smallest distance in dist[] and has not been visited
Source node in first case
16.     T.remove(u)    #remove u from Q or remove from unexplored
17.     for v in switches:
18.     if adjacency[u][v] != None:
19.     # print v
20.     w = 1
#Choose a vertex w, such that distance[w] is minimum and visited[w] is 0. Mark as visited[w]
 #Recalculate the shortest distance of remaining vertices from the source.
#Only, the vertices not marked as 1 in array visited[ ] should be considered for recalculation of distance.
i.e. for each vertex v
21.     if distance[u] + w < dist[v]:
22.     dist[v] = dist[u] + w
23.     previous[v] = u // set/update edgs
24.     r = []
25.     p = dst
26.     r.append(p)
27.     q = previous[p]
28.     while q is not None:
29.     if q == src:
30.     r.append(q)
31.     break
32.     p = q
33.     r.append(p)
34.     q = previous[p]
35.     r.reverse()
36.     if src == dst:
37.     # We build the shortest path and display it
38.     path = [src]
39.     else:
40.     path = r
41.     # Now add the ports
42.     r = []
43.     in_port = first_port
44.     for s1, s2 in zip(path[:-1], path[1:]):
45.     out_port = adjacency[s1][s2]
46.     r.append((s1, in_port, out_port))
47.     in_port = adjacency[s2][s1]
48.     r.append((dst, in_port, final_port))
49.     return r
```

Algorithm 3.1 Dijkstra algorithm procedures [54].

3.5 Network Slicing

There are several types of Slicing such as topology slicing and servicing slicing in thesis, it used service slicing. Service slicing consist from two slice one of slice for Gaming slice and other one for another service. The topology of slicing will be part from the network. The slice and associated traffic flow can be defined as in following:

- **Allocation of traffic flows to slices:** Match rules must be specified before traffic packets can be assigned to flows. Following that, actions are applied to the categorized traffic packets. As a result, first, several match rules are built for classifying incoming packets and differentiating between video and other traffic. Figure (3.3) below presents how this pseudocode can be implemented. The packets from s2, s5, and s3 are routed towards the outport linked with the destinations, at end switches s1, s4, and s6. All packets sent to s2 and s3 are checked to see if they are UDP packets with a destination port of 9998, indicating Game transmission. Game traffic seems to be categorized by gaming slices; whereby the slice number in the instance coding is 1. Others, such as UDPs having different destination ports, TCPs, and Internet Control Message Protocols (ICMPs), are classified as alternative traffic. This additional traffic is classified as non-gaming, or slice no. 2 within the sample coding. All forms of verification help to create matches. The outport may be found by inspecting the flow according to the definition of the step before it. As actions, all switches need to deal with incoming packets. In light of the assignment of traffic to appropriate slices, the packet flood flows towards the out port on switches s2 and s3 in the centre.

```
def _packet_in_handler(self, ev):
    save datapath of packet
    get in_port of packet
    get dpid of switch
   if switch is end switch:
      if packet from middle switch:
         send packet to the out_port of its destination

      elif UDP packet and destination port 9998:
           slice_numer = 1
           look up out_port
           actions
           match

      elif UDP packet and destination port not 9998:
          slice_numer = 2
          look up out_port
          actions
          match

       elif TCP packet:
           slice_numer = 2
            look up out_port
            actions
            match

       elif ICMP packet:
           slice_numer = 2
            look up out_port
            actions
            match

   if switch is middle switch:
      flood port
      actions
      match
  add_flow
  send_package
```

Figure 3.3 Network Slicing Steps [55]

Chapter Summary

In this chapter, a network- slicing architectural solution for 5G networks has been presented. Using the Dijkstra algorithm for improving the quality of service (QoS). Then, the proposed solution for network slicing by creating a virtual port and allocation of resources to each slice.

Chapter Four
Evaluation and Validation

Chapter4 Evaluation and Validation

4.1 Introduction

This chapter presents the results obtained after implementing the shortest path algorithm and network slicing. A comparison was made between the results after and before the proposed work

4.2 Building the platform

In this project, the Dijkstra algorithm and Network slicing are performed in a virtual environment with all the devices being emulated. The actual hardware used to experiment consisted of a Hp laptop with Intel(R) Core i7-4600U, CPU, 2.70 GHz, 16.0 GB RAM, and 256 GB SDD. The hypervisor is Virtual Box as (NFV) is used to virtualize the environment for loading a Mininet image. A 64bit image is used to create a VM in Virtual Box with 4GB RAM and a 20GB HDD. In the Virtual Box, Ubuntu 20.04 is installed for creating the operating environment for the simulation. As explained in chapter two.

4.3 Network Topology

Network topologies are created via Mininet, making use of the nodes, edge, and hosts mentioned in Chapter Three. During this process, the Ryu functions as the interface controller, enabling the packet routing to take place within the network design. As for the experiment presented in this work, a fully connected network topology is designed having the following characteristics:

- 16 Hosts
- 20 Switches
- 2 RYU Controllers

Given the fact that network QoS and managing the traffic are among the essential aims of this work; a fully connected topology has been adopted within an SDN OpenFlow protocol, whereby an interconnection is created among all

switches. Figure 4.1 below represents the topology used in this work. The Ryu manager assists in loading the shortest path algorithm and network slicing files into the RYU controllers. Its main role is observing the connections that exist among hosts and switches to determine the shortest paths.

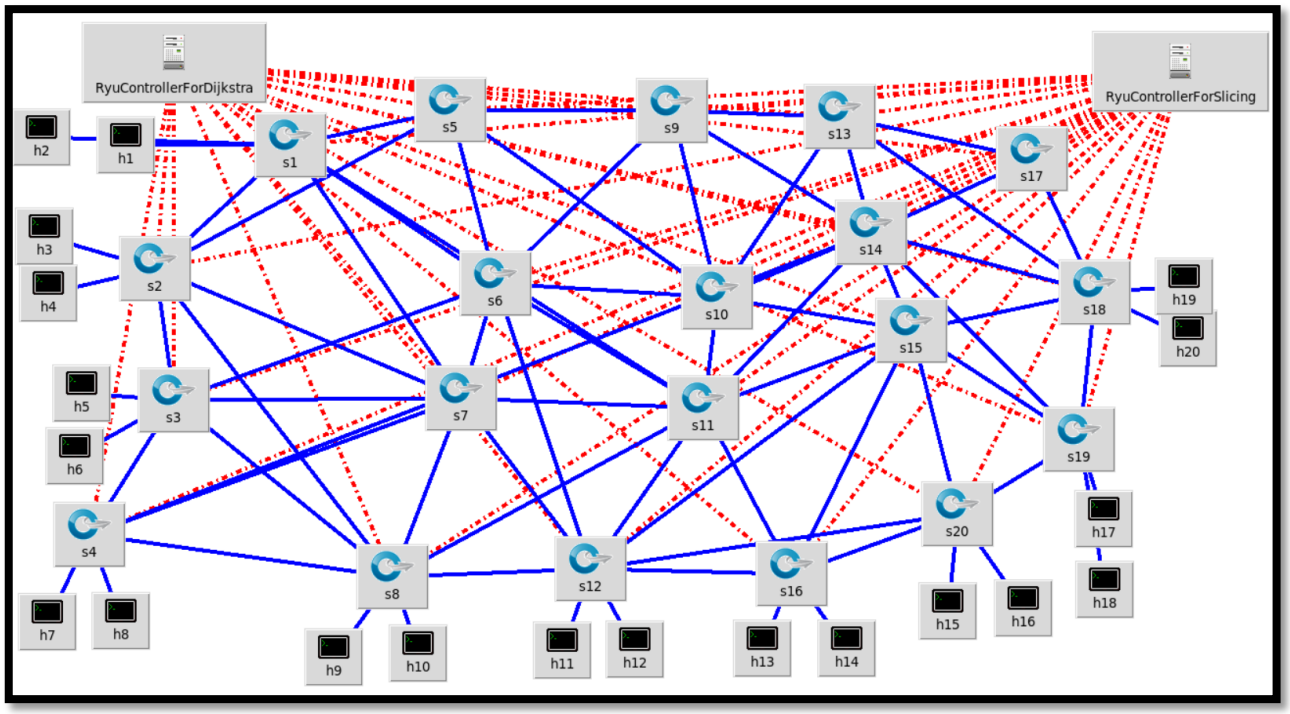


Figure 4.1 Network Topology

4.4 Topology Design

In SDN, the controller maintains all network topology information. In addition, the suggested technique is utilized to determine the shortest path provided within a certain network topology. The network is constructed across hosts and switches, as shown in Figure 4.2. In Mininet, the pingall command shows the connection among all hosts within the network and examines if all hosts are active and can be reached by one another. In case the hosts are active, a (0%) loss is displayed, indicating the receival of all packets. The pingall command takes approximately ten minutes for completing the link between hosts, depending on the size of the topologies.

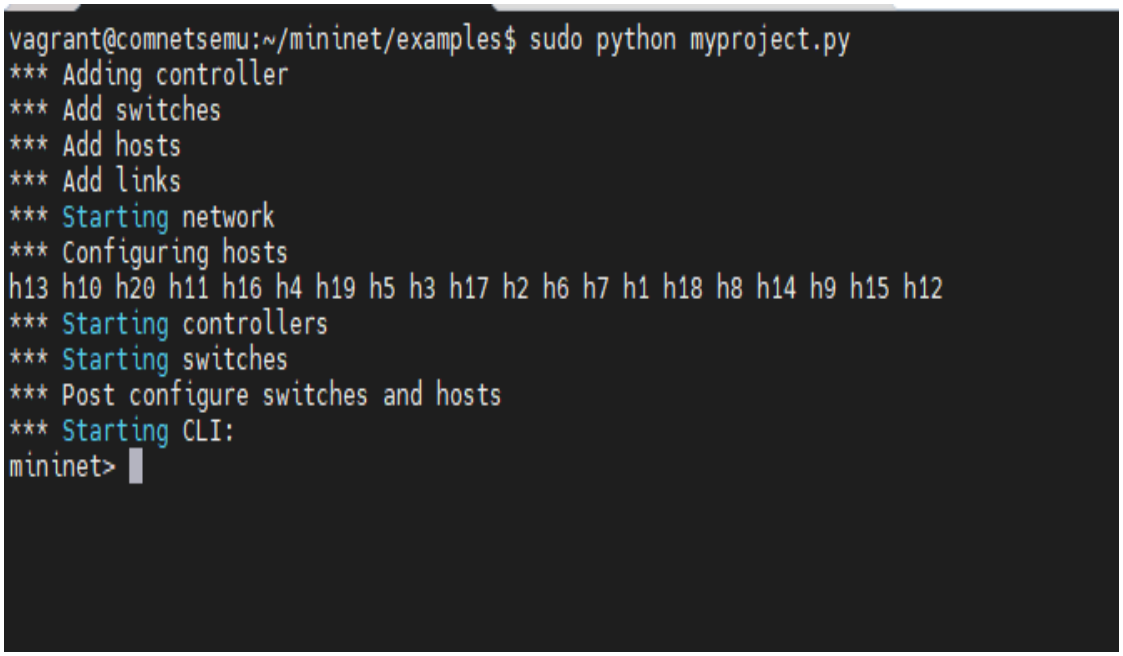


Figure 4.2 generation topology in Mininet

4.5 The connection between Mininet and RYU Controller

Ryu and Mininet have been set up, to establish the topology, Mininet loads a script from a directory. As demonstrated in Figure 4.2 above, MAC addresses are assigned automatically for matching the host names, using the Mininet CLI option mac. This improves the simulation stability and consistency while also simplifying Ryu logic. The Mininet CLI Arp options enable hosts of caching mac addresses to locate other hosts. The CLI option ovsk has been adopted for specifying the switch types. The Open vSwitch, an open-source virtual switch, will be used by Mininet. To ensure compatibility, all network appliances, as well as controllers set to utilize OpenFlow version 1.3. The CLI option remote instructs switches on where to locate the controller. Switches communicate with the controller through the ports 6633 for Dijkstra algorithm and 6634 port for network slicing to establish a link between the network topology and the controller. To keep track of packages transmitted, all switches within the topology are allocated unique ports. The controller script is run to determine

the shortest path or network slicing across this topology. According to Figure 4.3, all topology switches and ofp handler events have been loaded to transmit and receive packets across the switches.

```
vagrant@comnetsemu:~/ryu$ ryu-manager ryu/app/controller.py --observe-links --ofp-tcp-listen-port=6633
loading app ryu/app/controller.py
loading app ryu.controller.ofp_handler
instantiating app ryu/app/controller.py of SimpleSwitch13
instantiating app ryu.controller.ofp_handler of OFPHandler
packet in 13 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 6
packet in 6 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 6
packet in 15 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 5
packet in 12 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 2
packet in 11 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 1
packet in 7 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 4
packet in 7 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 1
packet in 14 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 3
packet in 6 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 3
packet in 4 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 6
packet in 12 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 1
packet in 7 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 3
packet in 11 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 2
packet in 5 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 4
packet in 5 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 9
packet in 15 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 4
packet in 15 7a:da:a6:09:f0:0f 92:73:51:4e:bc:aa 8
packet in 4 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 4
packet in 15 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 3
packet in 5 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 8
packet in 5 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 5
packet in 5 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 6
packet in 14 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 6
packet in 6 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 2
packet in 6 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 4
packet in 6 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 5
packet in 2 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 4
packet in 4 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 5
packet in 4 7a:da:a6:09:f0:0f 92:73:51:4e:bc:aa 6
packet in 4 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 3
packet in 11 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 3
packet in 5 92:73:51:4e:bc:aa ff:ff:ff:ff:ff:ff 7
packet in 5 7a:da:a6:09:f0:0f 92:73:51:4e:bc:aa 9
```

Figure 4.3 Run Ryu controller

4.6 Starting Ryu and Mininet

After loading the Ryu application, the Mininet is started up. It is necessary to enable the Ryu of detecting networks before the interaction with Mininet, as shown in Figure (4.4). After loading, Mininet is used to ping other hosts by running a pingall, enabling the flow table configuration of all switches correctly. This step might need some time to finish, and several attempts might be required before succeeding. Any incoming flows are detected by the Ryu application. The host might run the Iperf in different modes, being either server or client. Port no. 6633 is used by the switches when communicating with the Dijkstra Ryu controller and port number 6634Network slicing Ryu controller.

During this process, unique port numbers will be assigned to each switch to enable the controller of sending replies.

Ryu runs every algorithm individually and finds the short path. Ryu is started after that the following command is typed into the CLI:

$~/ryu/bin/ryu-manager ~/ryu/ryu/app/short-path --observe-links

The Mininet is started up by typing the following command into the terminal:

$sudo python myproject.py --topo mytopo --controller remote --protocols=OpenFlow13 --arp –mac

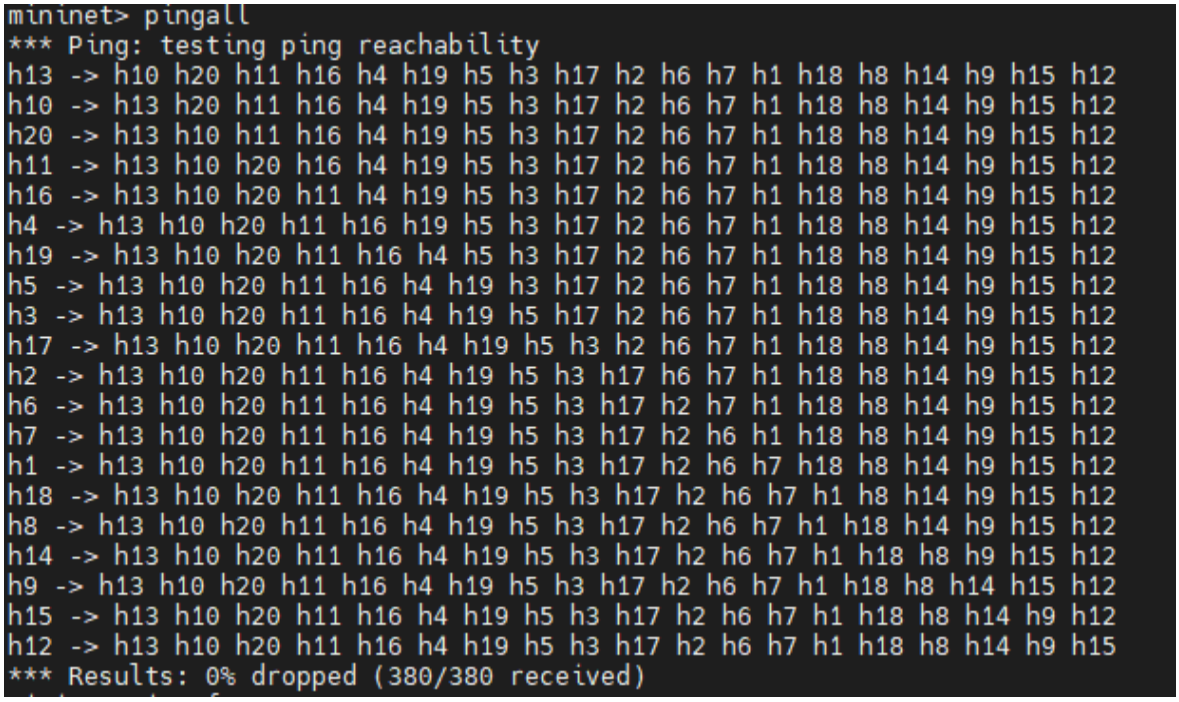


Figure 4.4 pingall

4.7 Validation

The network was evaluated before and after applying the short-path algorithm and Slicing. The testing concentrated on various QoS characteristics like bandwidth, latency, jitter, delay, and packet loss among network hosts. Delay was calculated through the transmission of 20 Internet Control Message Protocol (ICMP) Echo Request packets from the first host to the last host within that

network and, counting the time it took for the ICMP Echo Reply to arrive at source hosts. Throughput, Jitter, and Packet Loss were measured utilizing iPerf, both with TCP and then with UDP, with each test lasting 10 seconds. Examples of the test results are shown in Figures (4.5) to (4.7).

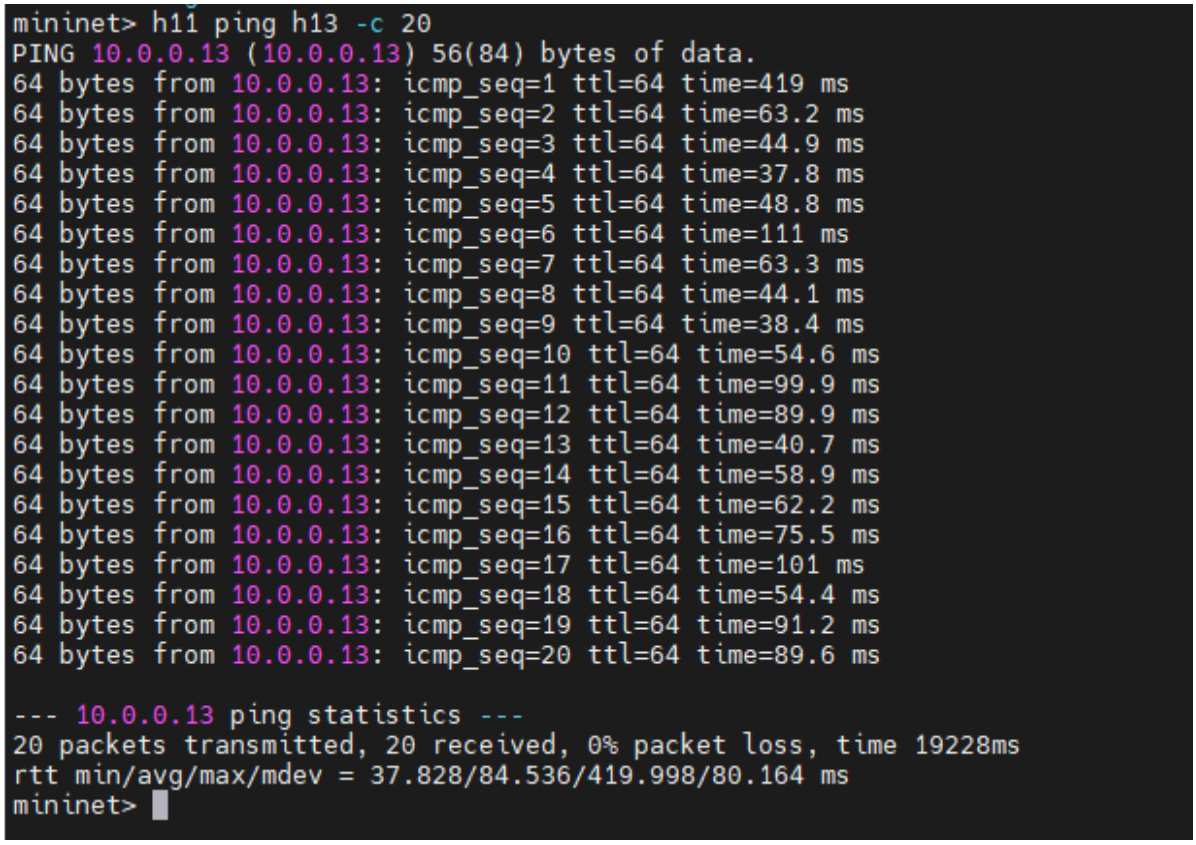


Figure 4.5 ping from h11 to h13 before the algorithm

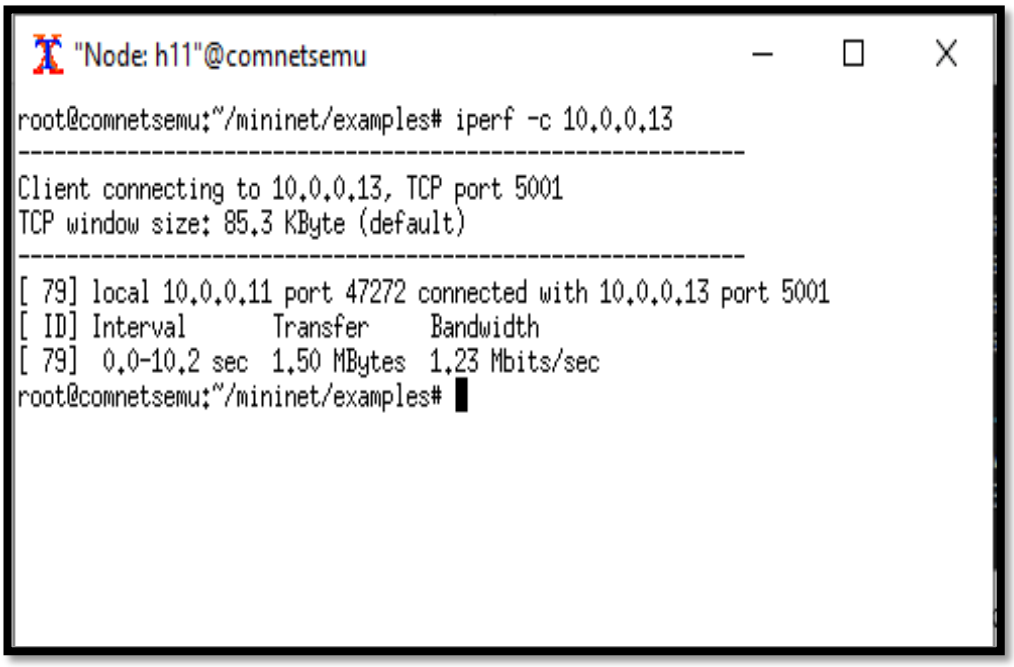


Figure 4.6 iPerf h11 to h13 before loading algorithm – TCP connection.

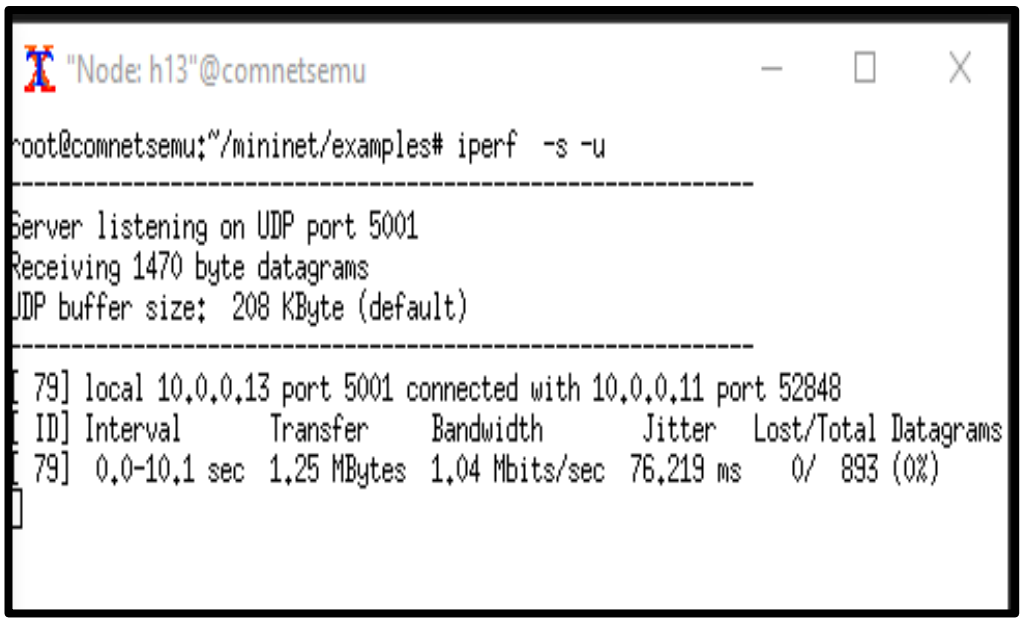


Figure 4.7 iPerf h11 to h13 before algorithm – UDP connection

4.7.1 Test results of the implementation of the Dijkstra algorithm

The network assessment took place in different sizes of hosts both before and after it ran the algorithm; to examine extraordinary behaviour. Table 4.1, notices it decreased the bandwidth with high jitter, delay, latency, and packet loss. When the algorithm was run and the system became stable, we measure the QoS parameter using the IPerf tool; it notices a reduction in jitter, delay, latency, and packet loss. Because packet loss and latency are measured after the system has stabilized, they are nearly zero. Tables 4.1- 4.2 illustrate the results obtained.

Table 4.1 Measure QoS parameters results before implementation algorithm

Numbers of the hosts	TCP		UDP					Delay (ms)
	Bandwidth (Mbits/sec)	Transfer (Mbytes)	Bandwidth (Kbits/sec)	Transfer (Mbytes)	Jitter	Latency(ms)	Packet loss (%)	
15	1.5	1.86	865	1.25	21.867	113.2	0	16.736
20	1.19	1.42	694	1.05	27.38	159.4	0	18.619
24	1.05	1.25	686	1.02	128.242	244.1	18	27.203
28	1.53	1.84	837	1.25	22.716	112.7	0	71.519
34	1.19	1.42	662	1.4	27.86	190.3	0	73.58
38	1.28	1.54	719	1.14	21.619	212.7	8.60	78.964

Table 4.2 Measure QoS parameters after implementation algorithm

Numbers of the hosts	TCP		UDP					Delay(ms)
	Bandwidth (Gbits/sec)	Transfer (GBytes)	Bandwidth (Mbit/sec)	Transfer (Mbyte)	Jitter	Latency(ms)	Packet loss (%)	
15	8.08	9.41	1.5	1.25	0.01	0	0	0.017
20	13.1	15.2	1.5	1.25	0.005	0	0	0.021
24	11	12.8	1.5	1.25	0.009	0	0	0.038
28	18.8	21.9	1.04	1.24	0.014	0	0	0.053
30	15.5	18.1	1.5	1.25	0.021	0	0	0.067
38	12.1	14.1	1.04	1.24	0.041	0	0	0.107

The average performance for the previous tables is calculated as explained in table 4.3.

Table 4.3 Measure average results for QoS parameter

Algorithm	TCP		UDP					Delay(ms)
	Bandwidth	Transfer	Bandwidth	Transfer (Mbytes)	Jitter (ms)	Latency (ms)	Packet Loss %	
Before	1.29(Mbits/sec)	1.55(Mbits/sec)	743.83(Kbits/sec)	1.18	41.61	172.06	4.43	47.77
After	13.09 (Gbits/sec)	12.9(GBytes)	1.34(Mbit/sec)	1.24	0.1	0	0	0.05

A. Performance of the short-path algorithm

A significant improvement has been observed in the network performance after running the algorithm. the network bandwidth was increased from **743.83(Kbits/sec),** to **1.34(Mbit/sec)** after running the algorithm and this result in UDP case, also in TCP just measured. The following Figures (4.8) to (4.11) illustrate both cases (TCP, UDP) before and after implementation.

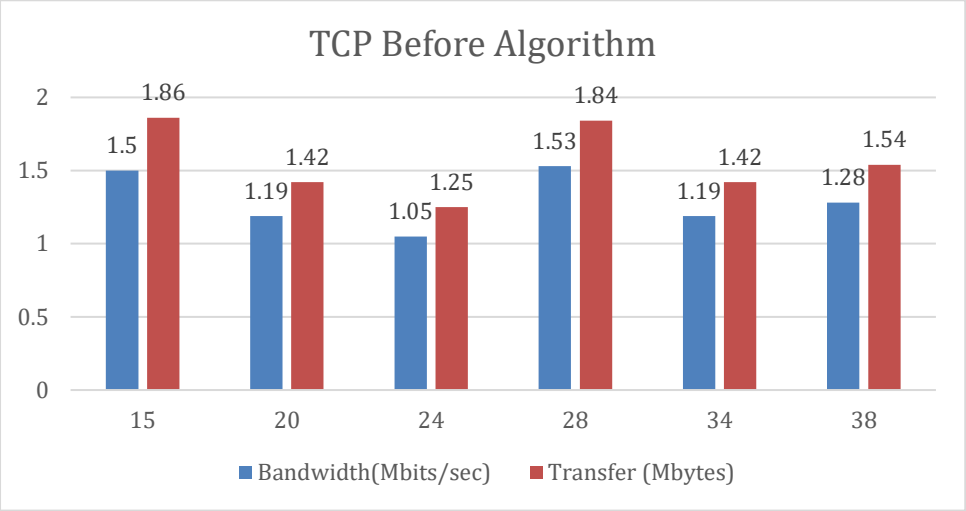


Figure 4.8 Measure TCP Bandwidth before the algorithm

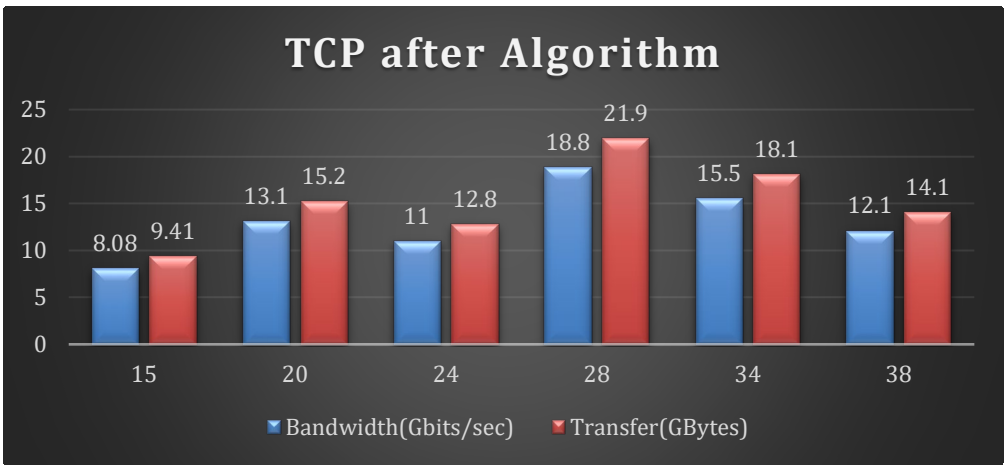


Figure 4.9 Measure TCP Bandwidth after the algorithm

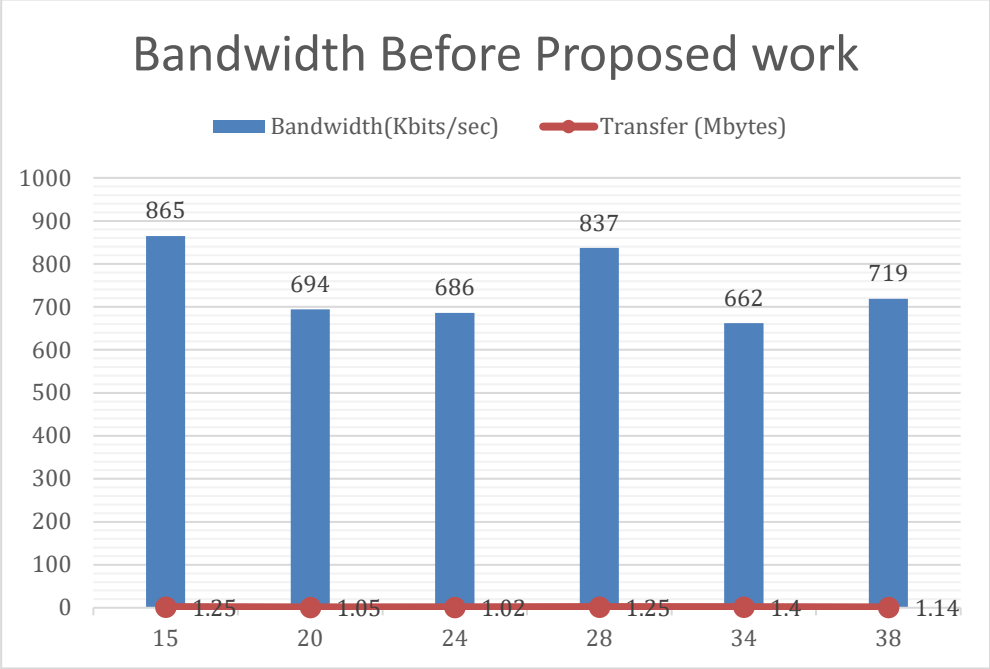


Figure 4.10 QoS parameters Bandwidth before Algorithm

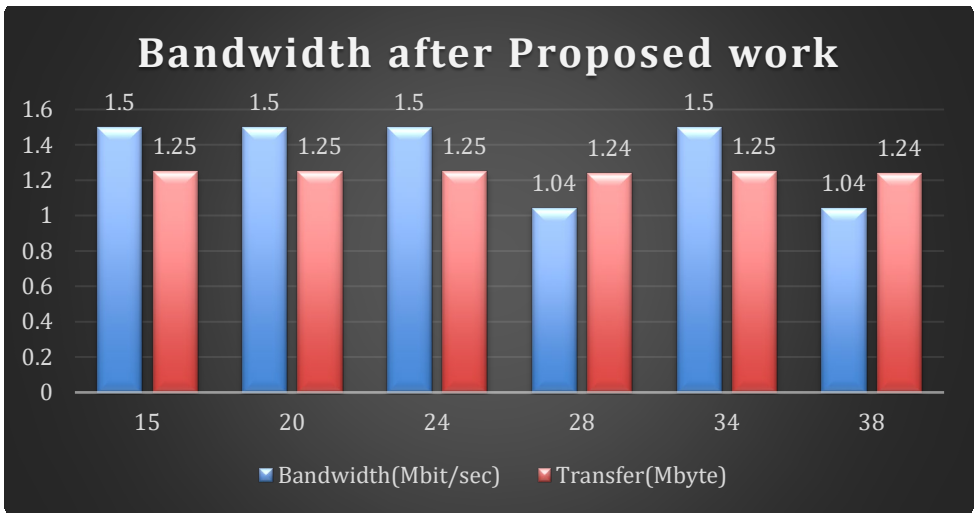


Figure 4.11 QoS parameters Bandwidth after Algorithm

Figures (4.12) to (4.13) show a comparison between jitter before and after the algorithm implementation, The Jitter has decreased from **41.61 to 0.1 (ms).**

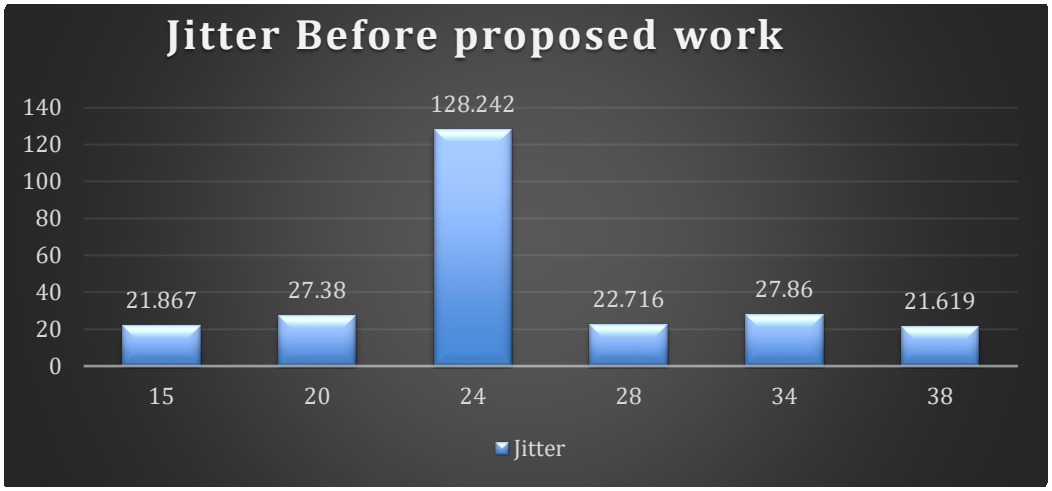


Figure 4.12 QoS parameters Jitter before Algorithm

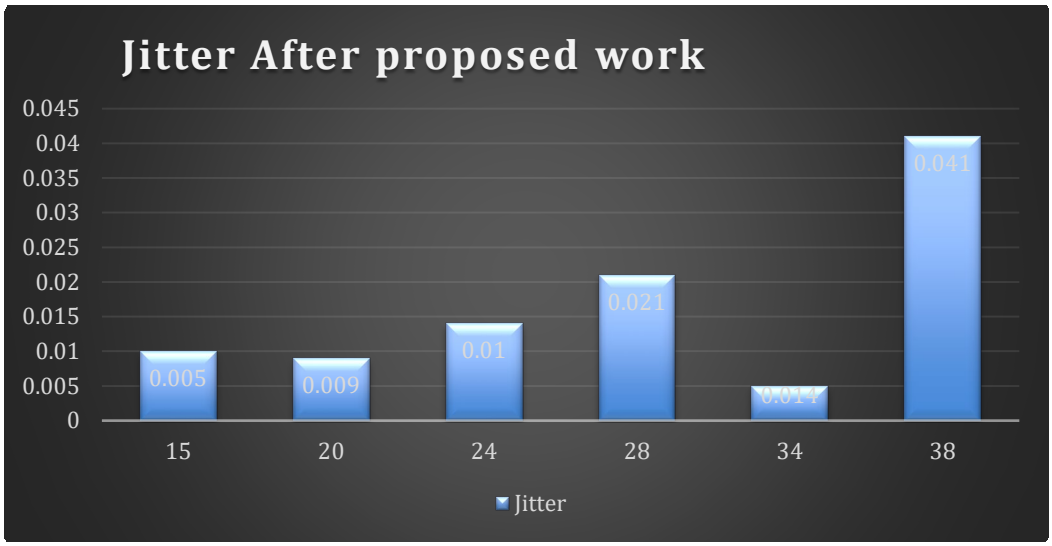


Figure 4.13 QoS parameters Jitter after Algorithm

The delay has decreased from **47.77 (ms)** to **0.05 (ms)**, the results of delay before and after implementation are shown in Figures (4.14) to (4.15) below:

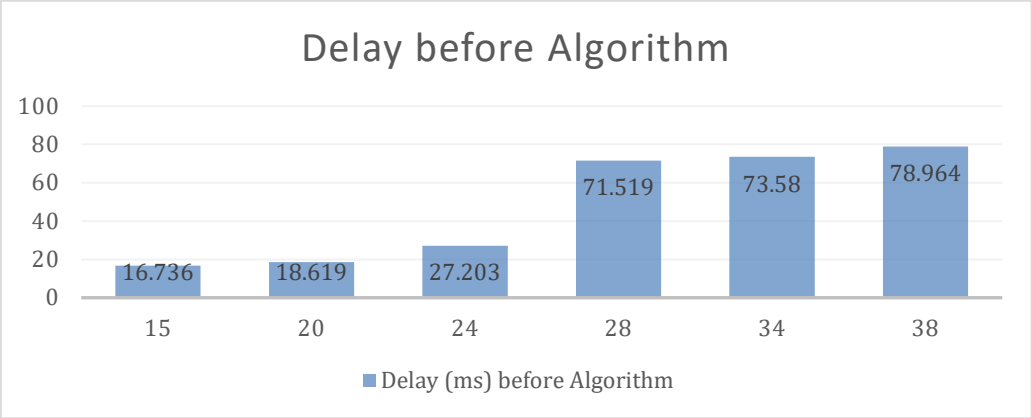


Figure 4.14 QoS parameters Delay before Algorithm

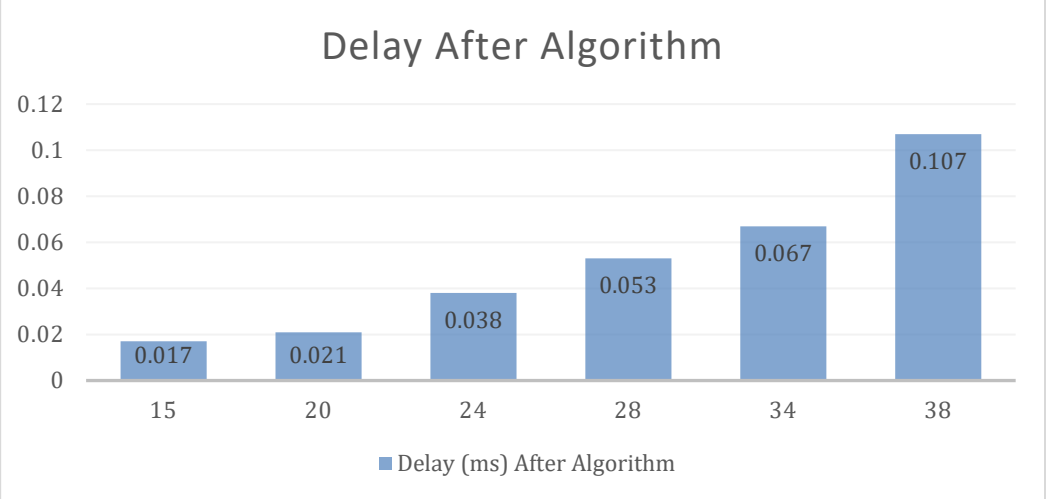


Figure 4.15 QoS parameters Delay after Algorithm

The figures (4.16) to (4.17) show the packet loss and latency were equal to 0 when run the algorithm and measured while the system stable.

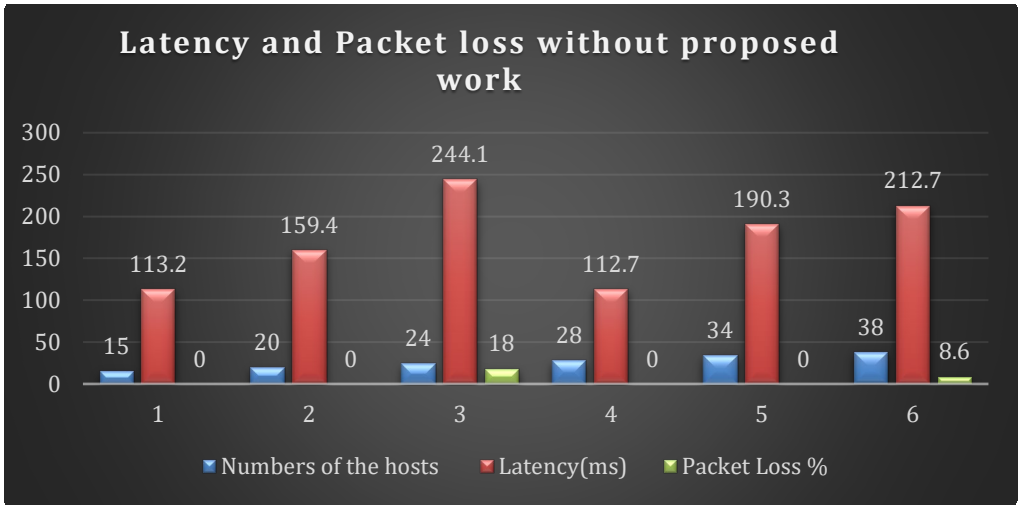


Figure 04.16 QoS parameters Latency and Packet loss before Algorithm

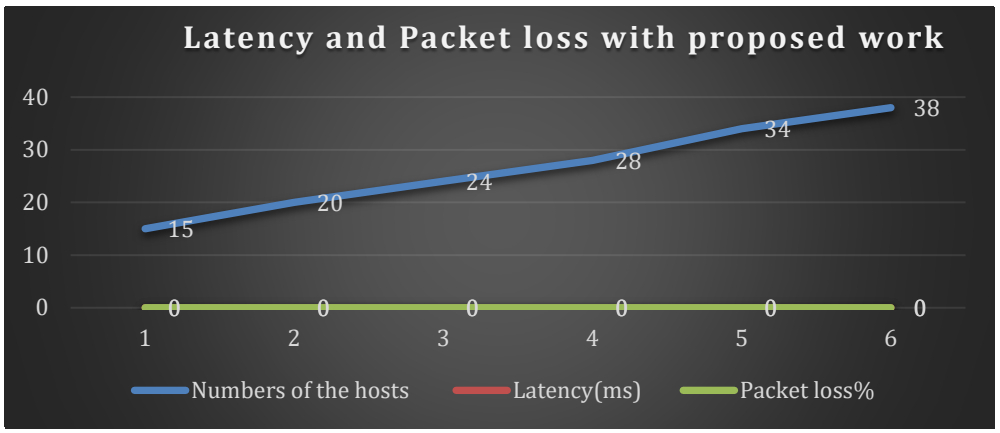


Figure 4.17 QoS parameters Latency and Packet loss after Algorithm

4.7.2 Test results of the Network slicing

The network slicing assessment took place using the following steps:

1. **Connectivity validation via ping.:** given the fact that ICMP packets are created by ping when belonging to the non-video slices, it can therefore be used as a means for determining the reachability of all hosts, as illustrated in Figure (4.18) below.

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8
h2 -> h1 h3 h4 h5 h6 h7 h8
h3 -> h1 h2 h4 h5 h6 h7 h8
h4 -> h1 h2 h3 h5 h6 h7 h8
h5 -> h1 h2 h3 h4 h6 h7 h8
h6 -> h1 h2 h3 h4 h5 h7 h8
h7 -> h1 h2 h3 h4 h5 h6 h8
h8 -> h1 h2 h3 h4 h5 h6 h7
*** Results: 0% dropped (56/56 received)
mininet>
```

Figure 4.18 Test connection

1. **Bandwidth validation using iPerf**: From h2 to h4, a game service is generated involving 11 Mbits/s UDP traffic with the destination port 9998, making use of the video slice. After logging in h1 and h3 within novel terminals, the UDP packets are listened to at port 9998 with h4 functioning as a receiver. Next, these packets are sent to the destination port 99998 via the senders h2 to h4. Figure (4.19) indicate how h2 continues to send the game traffic and h4 receives it a 4.50 Mbits/s bandwidth. This implies that the game traffic makes use of the game slice created.

```
^Croot@comnetsemu:~/mininet/examples# iperf -s -u -p 9998 -b 11M
------------------------------------------------------------
Server listening on UDP port 9998
Receiving 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[ 39] local 10.0.0.2 port 9998 connected with 10.0.0.4 port 57869
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[ 39]  0.0-10.0 sec  5.38 MBytes  4.50 Mbits/sec  22.939 ms    0/ 3835 (0%)
```

```
root@comnetsemu:~/mininet/examples# iperf -c 10.0.0.2 -u -p 9998 -b 11M
------------------------------------------------------------
Client connecting to 10.0.0.2, UDP port 9998
Sending 1470 byte datagrams, IPG target: 1019.56 us (kalman adjust)
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[ 39] local 10.0.0.4 port 57869 connected with 10.0.0.2 port 9998
[ ID] Interval       Transfer     Bandwidth
[ 39]  0.0-10.0 sec  5.38 MBytes  4.50 Mbits/sec
[ 39] Sent 3835 datagrams
[ 39] Server Report:
[ 39]  0.0-10.0 sec  5.38 MBytes  4.50 Mbits/sec   0.000 ms    0/ 3835 (0%)
root@comnetsemu:~/mininet/examples#
```

Figure 4.19 Using iPerf to validate bandwidth

The figure (4.20) below shows another service that is not a gaming slice, by using the TCP protocol.

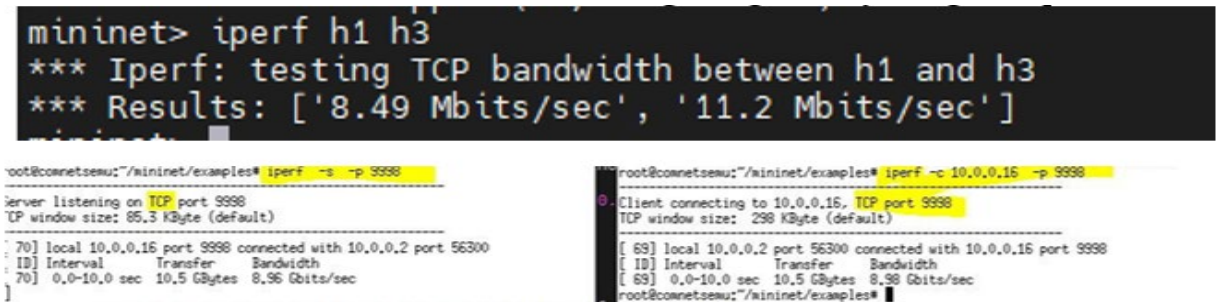


Figure 4.20 Another service not gaming slice by using TCP protocol

2. **Flow entry checking via dump-flows**: Switch s1 receives a UDP packet via port s1_eth3, with a destination address of 00:00:00:00:00:04 through port 9998. Such a case implies that the packet is supposed to be within the game slice. After that s1 checks the flow table and the actions that correspond to it as output s1_eth1, the latter port will map the packets towards the flow using an 11 Mbits/s bandwidth.as shown in figure (4.21).

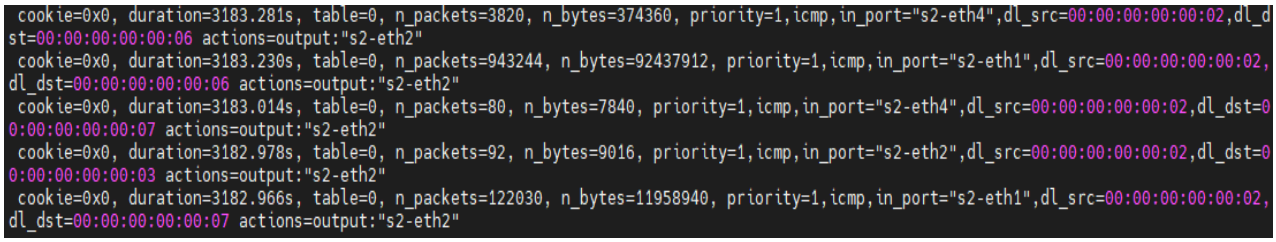


Figure 4.21 dump-flows to check flow entries

Chapter Summary

In this chapter, we have presented an introduction to the topology of the network and Ryu controller and how the runs. Then run the network topology created in Mininet and algorithms that have been written in the Ryu controller for evaluating the QoS and network slicing. Then, the results of performance evaluation before and after the work were analyzed and discussed separately.

Chapter Five

Conclusion and Suggestions

Chapter 5 Conclusion and Suggestions

5.1 conclusion

. Currently, there is a remarkably higher and more complex request for Quality of Service (QoS) The present study works toward increasing the QoS, due to the difficulty in achieving QoS within SDN/NFV enabled 5G networks. To overcome these limitations, this study proposes the implementation of the Dijkstra algorithm is used to find the shortest paths between nodes in the graph. The Network Slicing (NS) technology is adopted for optimizing allocations as well as QoS values. And the following result can be summarized:

1. Significant increase has been observed in regards to the network performance after the algorithm and network slicing were run.
2. The results obtained with the proposed system increase bandwidth and reduce jitter and delay compared to previous works.
3. The network utilization has been improved and the traffic is controlled and managed using Network Slicing because each slice has its own set of requirements to meet the demands of various use cases.
4. The technology efficiently allocates network resources for maximum cost-efficiency.

- There are issues with Ryu when it restarts the following steps to run it again
    1. To exit Ryu, hit CTRL+C twice.
    2. Exit Mininet and wait for it to finish.
    3. Clear cache on the same Ryu terminal by executing #mn -c (root mode)
    4. Restart Ryu and wait till it is fully loaded.
    5. Run Mininet with network topology and ping again

- Ryu controller should use the last version 3.34 and install it on ubuntu 18.04 with python3.8 Otherwise, it does not work with old versions
- Flow Visor controller supports Network Slicing but for now, it is outdated

5.2 Suggestions for Future Work

The following proposals can be worked out in the future:

1. The first idea is that the technique of network slicing will be implemented by using the proposed algorithms that will apply to the network slicing automatically without requiring a user any requirements, such as the port number and the size bandwidth as used in this thesis.
2. The second idea is to examine the performance of the shortest-path algorithm software on several common SDN controllers such as Open daylight, Floodlight, Beacon, and NOX/POX to compare the results.
3. Finally, the suggestion is made to study performing topologies of various sizes other than the fully connected topology that has been used in the project, to see if there are any additional restrictions with the algorithm.

References

[1] Y. Yang *et al.*, *5G System Design Edited by*, vol. 167, no. 2020. Elsevier B.V., 2020. doi: 10.1002/9781119694748.

[2] J. Park, J. Bukhari, and W. Yoon, "An SDN testbed for evaluating wireless heterogeneous networks," *J. Theor. Appl. Inf. Technol.*, vol. 97, no. 13, pp. 3627–3637, 2019.

[3] F. Hu, *Opportunities in 5G Networks: A Research and Development Perspective*. 2016.

[4] M. K. Arjmandi, "5G overview: Key technologies," *Opportunities in 5G Networks: A Research and Development Perspective*. pp. 19–32, 2016. doi: 10.1201/b19698-4.

[5] F. Z. Yousaf *et al.*, "Network slicing with flexible mobility and QoS/QoE support for 5G Networks," *2017 IEEE Int. Conf. Commun. Work. ICC Work. 2017*, no. July, pp. 1195–1201, 2017, doi: 10.1109/ICCW.2017.7962821.

[6] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network Slicing in 5G: Survey and Challenges," *IEEE Commun. Mag.*, vol. 55, no. 5, pp. 94–100, 2017, doi: 10.1109/MCOM.2017.1600951.

[7] S. Costanzo, I. Fajjari, N. Aitsaadi, and R. Langar, "DEMO: SDN-based network slicing in C-RAN," *CCNC 2018 - 2018 15th IEEE Annu. Consum. Commun. Netw. Conf.*, vol. 2018-Janua, no. July, pp. 1–2, 2018, doi: 10.1109/CCNC.2018.8319321.

[8] C. Simon, M. Maliosz, J. Biro, B. Gero, and A. Kern, "5G Exchange for inter-domain resource sharing," *IEEE Work. Local Metrop. Area Networks*, vol. 2016-Augus, no. 671636, 2016, doi: 10.1109/LANMAN.2016.7548842.

[9] K. Ramantas, E. Kartsakli, M. Irazabal, A. Antonopoulos, and C. Verikoukis, "Implementation of an SDN-enabled 5G experimental platform for core and radio access network support," *Adv. Intell. Syst. Comput.*, vol. 725, pp. 791–796, 2018, doi: 10.1007/978-3-319-75175-7_77.

[10] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing and softwarization: A survey on principles, enabling technologies, and solutions," *IEEE Commun. Surv. Tutorials*, vol. 20, no. 3, pp. 2429–2453, 2018, doi: 10.1109/COMST.2018.2815638.

[11] K. S. Ibarra-Lancheros, G. Puerto-Leguizamón, and C. Suárez-Fajardo, "Quality of service evaluation based on network slicing for software-defined 5G systems," *TecnoLógicas*, vol. 21, no. 43, pp. 27–41, 2018, doi: 10.22430/22565337.1066.

[12] L. Ben Azzouz and I. Jamai, "SDN, slicing, and NFV paradigms for a smart home: A comprehensive survey," *Trans. Emerg. Telecommun. Technol.*, vol. 30, no. 10, pp. 1–13, 2019, doi: 10.1002/ett.3744.

[13] W. Shuqi, L. Bei, and F. Yu, "Design of multi-service network slicing scheme based on SDN/NFV," *Proc. - 2018 Int. Conf. Sens. Networks Signal Process. SNSP 2018*, pp. 344–351, 2019, doi: 10.1109/SNSP.2018.00073.

[14] X. Li *et al.*, "Network Slicing for 5G: Challenges and Opportunities," *IEEE Internet*

Comput., no. August, pp. 1–8, 2018, doi: 10.1109/MIC.2018.326150452.

[15] N. Nikaein *et al.*, "Network store: Exploring slicing in future 5G networks," *MobiArch 2015 - Proc. 10th Int. Work. Mobil. Evol. Internet Archit. co-located with MobiCom 2015*, no. July, pp. 8–13, 2015, doi: 10.1145/2795381.2795390.

[16] I. Afolabi, M. Bagaa, T. Taleb, and H. Flinck, "End-To-end network slicing enabled through network function virtualization," *2017 IEEE Conf. Stand. Commun. Networking, CSCN 2017*, pp. 30–35, 2017, doi: 10.1109/CSCN.2017.8088594.

[17] D. Kim and S. Kim, "Network slicing as enablers for 5G services: state of the art and challenges for mobile industry," *Telecommun. Syst.*, vol. 71, no. 3, pp. 517–527, 2019, doi: 10.1007/s11235-018-0525-2.

[18] A. Nayak Manjeshwar, P. Jha, A. Karandikar, and P. Chaporkar, "VirtRAN: An SDN/NFV-Based Framework for 5G RAN Slicing," *J. Indian Inst. Sci.*, vol. 100, no. 2, pp. 409–434, 2020, doi: 10.1007/s41745-020-00160-x.

[19] G. O. Medeiros, J. C. W. A. Costa, D. L. Cardoso, and A. D. F. Santos, "An Intelligent SDN Framework Based on QoE Predictions for Load Balancing in C-RAN," *Wirel. Commun. Mob. Comput.*, vol. 2020, 2020, doi: 10.1155/2020/7065202.

[20] S. Ejaz, Z. Iqbal, P. Azmat Shah, B. H. Bukhari, A. Ali, and F. Aadil, "Traffic Load Balancing Using Software Defined Networking (SDN) Controller as Virtualized Network Function," *IEEE Access*, vol. 7, pp. 46646–46658, 2019, doi: 10.1109/ACCESS.2019.2909356.

[21] O. Akpovi A., E. Seun, A. A. O., and O. F. Y., "Introduction to Software Defined Networks (SDN)," *Int. J. Appl. Inf. Syst.*, vol. 11, no. 7, pp. 10–14, 2016, doi: 10.5120/ijais2016451623.

[22] P. Göransson, C. Black, and T. Culver, *Software Defined Networks: A Comprehensive Approach: Second Edition*. 2016.

[23] J. METZLER and A. Metzler, "Ten Things to Look for in an SDN Controller," no. May, p. 11, 2013.

[24] D. S. Rana, S. A. Dhondiyal, and S. K. Chamoli, "Software Defined Networking (SDN) Challenges, issues and Solution," *Int. J. Comput. Sci. Eng.*, vol. 7, no. 1, pp. 884–889, 2019, doi: 10.26438/ijcse/v7i1.884889.

[25] S. Asadollahi, B. Goswami, and M. Sameer, "Ryu controller's scalability experiment on software defined networks," *2018 IEEE Int. Conf. Curr. Trends Adv. Comput. ICCTAC 2018*, no. February, pp. 1–5, 2018, doi: 10.1109/ICCTAC.2018.8370397.

[26] D. P. Nsrc and A. L. Nsrc, "RYU OpenFlow Controller".

[27] M. Kazuya SUZUKI), Member, Kentaro SONODA, Nobuyuki TOMIZAWA, Yutaka YAKUWA, Terutaka UCHIDA, Yuta HIGUCHI, Nonmembers, Toshio TONOUCHI, and Hideyuki SHIMONISHI, "E97.B_375.pdf."

[28] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "Software-defined networking security: Pros and cons," *IEEE Commun. Mag.*, vol. 53, no. 6, pp. 73–79, 2015, doi: 10.1109/MCOM.2015.7120048.

[29] P. Greendyk, A. Parikh, and S. Tripathi, “Service platforms,” *Build. Netw. Futur. Get. Smarter, Faster, More Flex. with a Softw. Centric Approach*, no. 1, pp. 293–329, 2017, doi: 10.1201/9781315208787.

[30] A. Esmaeily, “Study of RAN slicing using SRS LTE in a real SDN-NFV platform,” *Interciencia*, vol. 489, no. 20, pp. 313–335, 2018.

[31] C. Bouras, A. Kollia, and A. Papazois, “Exploring SDN & NFV in 5G Using ONOS & POX Controllers,” *Int. J. Interdiscip. Telecommun. Netw.*, vol. 10, no. 4, pp. 46–60, 2018, doi: 10.4018/ijitn.2018100103.

[32] F. Z. Yousaf, M. Bredel, S. Schaller, and F. Schneider, “NFV and SDN-Key technology enablers for 5G networks,” *IEEE J. Sel. Areas Commun.*, vol. 35, no. 11, pp. 2468–2478, 2017, doi: 10.1109/JSAC.2017.2760418.

[33] ETSI, “What is NFV?,” 2018, [Online]. Available: www.etsi.org

[34] D. N. K. Jayakody, K. Srinivasan, and V. Sharma, *5G enabled secure wireless networks*. 2019. doi: 10.1007/978-3-030-03508-2.

[35] F. Debbabi, R. Jmal, and L. Chaari Fourati, “5G network slicing: Fundamental concepts, architectures, algorithmics, projects practices, and open issues,” *Concurr. Comput. Pract. Exp.*, vol. 33, no. 20, pp. 1–38, 2021, doi: 10.1002/cpe.6352.

[36] A. G. Kanatas, K. S. Nikita, and P. Mathiopoulos, *New directions in wireless communications systems: From mobile to 5G*. 2017. doi: 10.1201/b21300.

[37] A. Napolitano, A. Giorgetti, K. Kondepu, L. Valcarenghi, and P. Castoldi, “Network Slicing: An Overview,” *IEEE 4th Int. Forum Res. Technol. Soc. Ind. RTSI 2018 - Proc.*, no. September, 2018, doi: 10.1109/RTSI.2018.8548449.

[38] N. Nikaein *et al.*, “Network store: Exploring slicing in future 5G networks,” *MobiArch 2015 - Proc. 10th Int. Work. Mobil. Evol. Internet Archit. co-located with MobiCom 2015*, pp. 8–13, 2015, doi: 10.1145/2795381.2795390.

[39] S. Robert, “Slicing a Network Advanced Computer Networks FlowVisor Software-Defined Network (SDN) SDN Switches and OpenFlow,” *IEEE Commun. Mag.*, pp. 1–5, 2010.

[40] T. Ahmed, A. Alleg, R. Ferrus, and R. Riggio, “On-Demand Network Slicing using SDN/NFV-enabled Satellite Ground Segment Systems,” *2018 4th IEEE Conf. Netw. Softwarization Work. NetSoft 2018*, no. NetSoft, pp. 378–383, 2018, doi: 10.1109/NETSOFT.2018.8460139.

[41] NGMN, “Description of Network Slicing Concept,” *NGMN 5G Proj. Requir. Archit. – Work Stream E2E Archit.*, vol. 1, no. January, pp. 1–7, 2016, [Online]. Available: https://www.ngmn.org/fileadmin/user_upload/161010_NGMN_Network_Slicing_framework_v1.0.8.pdf

[42] X. Li *et al.*, “Network Slicing for 5G: Challenges and Opportunities,” *IEEE Internet Comput.*, vol. 21, no. 5, 2018, doi: 10.1109/MIC.2018.326150452.

[43] S. M. A. S. Liqaa A. Al-Hashime, Ghaida A. Al-Suhail, *New Trends in Information and Communications Technology Applications*. 2018. [Online]. Available: http://link.springer.com/10.1007/978-3-030-01653-1

[44] F. Fossati *et al.*, “Decentralization of 5G slice resource allocation To cite this version : HAL Id : hal-02501918 Decentralization of 5G slice resource allocation,” 2020.

[45] A. Alfoudi, “Slicing-Based Resource Allocation and Mobility Management for Emerging Wireless Networks,” *Thesis*, no. June, 2018.

[46] Q. Wang *et al.*, “Enable Advanced QoS-Aware Network Slicing in 5G Networks for Slice-Based Media Use Cases,” *IEEE Trans. Broadcast.*, vol. 65, no. 2, pp. 444–453, 2019, doi: 10.1109/TBC.2019.2901402.

[47] L. Bonati, M. Polese, S. D’Oro, S. Basagni, and T. Melodia, “Open, Programmable, and Virtualized 5G Networks: State-of-the-Art and the Road Ahead,” *Comput. Networks*, vol. 182, pp. 1–66, 2020, doi: 10.1016/j.comnet.2020.107516.

[48] www.comp.nus.edu.sg, “A short walk-through of Mininet and POX”.

[49] K. Parsons, “iPerf3 User Documentation General Options Command Line Option Description-p,-port n,” pp. 3–4, [Online]. Available: https://d2cpnw0u24fjm4.cloudfront.net/wp-content/uploads/iPerf3-User-Documentation.pdf

[50] C. Computing and S. Id, “Enhancement of Network Throughput in SDN Using Shortest Path Routing Algorithms MSc in Cloud Computing Sayali Kapse National College of Ireland Supervisor : Muhammad Iqbal”.

[51] M. T. Ananta, M. T. Ananta, and M. A. Muslim, “Multicasting with the extended dijkstra’s shortest path algorithm forsoftware defined networking,” *Int. J. Appl. Eng. Res.*, vol. 9, no. 23, pp. 21017–21030, 2014.

[52] F. Granelli, T. G. Nguyen, and H. Wu, *Realizing network slicing*. Elsevier Inc., 2020. doi: 10.1016/B978-0-12-820488-7.00029-3.

Appendix

1. Certification Of Acceptance Paper 1

BICITS'21

1st Babylon International Conference on Information Technology and Science

CERTIFICATE OF ACCEPTANCE

This certificate is granted to

Adian Rasmi Alkhafaji and Dr. Firas Sabah Al-Turaihi

Certifies the acceptance of the research paper entitled:

MULTI-LAYER NETWORK SLICING AND RESOURCE ALLOCATION SCHEME FOR TRAFFIC-AWARE QOS ENSURED SDN/NFV-5G NETWORK

in

BICITS'21

Which will be held on 28-29 April, 2021 in Babylon, IRAQ, by College of Information Technology, University of Babylon and Technically Sponsored by IEEE represented by IEEE Iraq Section.

Prof. Dr. Hussain Attia
Dean of IT College, University of Babylon

Prof. Dr. Sattar B. Sadkhan
IT College, University of Babylon, IEEE Iraq Section

2. Certification Of Acceptance Paper 2

Certificate

of Participation

8th International Conference (ICCITM) 2022

Adian Rasmi Al-Khafaji

Presented a paper entitled

Traffic-Aware QoS Guaranteed SDN/NFV-5G Network with Multi-Layer Network Slicing and Resource Allocation

At the 8th International Conference on Contemporary Information Technology and Mathematics (ICCITM), on 30-31 Augst,2022, that was organized by the College of Computer Science and Mathematics, University of Mosul, Mosul, Iraq, and scientifically sponsored by IEEE.

IEEE

Prof. Dr. Dhuha Basheer Abdullah
Dean of Computer Science and Mathematics College,
Conference Chair

IEEE Iraq Section

Prof. Dr. Sattar B. Sadkhan
IT College, University of Babylon,
IEEE Iraq Section

الخلاصة

في الوقت الحالي ازداد الطلب والحاجة الى Quality of Service (QoS) لتلبي ما لم تستطيع التقنيات المتواجدة في وقتنا خذا تلبيتها. الدراسة الحالية متوجهة بشكل كبير على زيادة جودة الخدمة Quality of Service (QoS) ضمن قيود اتفاقية مستوى الخدمة العادلة (SLA)، نظرًا لصعوبة تحقيق جودة الخدمة داخل شبكات (5G) التي تدعم SDN / NFV.

تم اعتماد تقنية تقطيع الشبكة Network Slicing(NS) وخوارزمية Dijkstra لتحسين استخدامات موارد الشبكة وخدماتها المتعلقة بالجودة وغيرها. حيث تم اختبار الشبكة قبل وبعد تشغيل خوارزمية Dijkstra وتقسيم الشبكة، مع التركيز الرئيسي على معلمات QoS مثل bandwidth, delay, jitter, latency, and packet loss داخل الشبكة.

اما بعد ذلك فتم اختيار Mininet simulation platform بالإضافة الى عناصر اخرى كهيكلة الشبكة والعقد المكونة منها الشبكة بالإضافة الى العناصر الاخرى التي يتم التحكم بها من خلال RYU SDN كما تم استخدام ال iPerf لتقيم أداء الشبكة بعد اختبارها.

تظهر البيانات الناتجة من الاختبار أنه تم ملاحظة زيادة كبيرة فيما يتعلق بأداء الشبكة بعد الخوارزمية وتقطيع الشبكة.، تمت زيادة Bandwidth للشبكة بعد تشغيل الخوارزمية. وانخفض Jitter. ,Latency Packet loss, وقلل delay.

واخيرا، تم تحسين استخدام الشبكة والتحكم في حركة البيانات داخل الشبكة وإدارتها باستخدام Network Slicing لأن كل مجموعه لها متطلباتها الخاصة لتلبية حالات الاستخدام المختلفة. على ضوء ذلك فان التكنولوجيا الحديثة توظف موارد الشبكة بكفاءة لتحقيق أقصى قدر من الكفاءة وبأقل تكلفة ممكنة.

جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعة بابل - كلية تكنولوجيا المعلومات
قسم شبكات المعلومات

مخطط تقطيعي مع جودة الخدمة المضمونة لشبكات SDN/NFV-5G

رسالة مقدمة
الى مجلس كلية تكنولوجيا المعلومات - جامعة بابل وهي جزء من متطلبات نيل
درجة الماجستير في تكنولوجيا المعلومات / شبكات المعلومات

من قبل الطالبة

اديان رسمي حسن بشير

بإشراف
أ.م. د فراس صباح صالح هادي

1444 هـ

2022 م