Service Chaining in Network Function Virtualization

Author

Muhammad Arslan Tariq

Fall-MS2017 (CE) 00000206127

Supervisor

Dr. Muhammad Umar Farooq

DEPARTMENT OF COMPUTER & SOFTWARE ENGINEERING

COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY

ISLAMABAD

NOVEMBER, 2020

Service Chaining in Network Function Virtualization

Author

Muhammad Arslan Tariq

Regn Number

206127

A thesis submitted in partial fulfillment of the requirements for the degree of

MS Computer Engineering

Thesis Supervisor:

Dr. Muhammad Umar Farooq

Thesis Supervisor's Signature: _____

DEPARTMENT OF COMPUTER & SOFTWARE ENGINEERING

COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY,

ISLAMABAD

November, 2020

Declaration

I certify that this research work titled "*Service Chaining in Network Function Virtualization"* is my own work. The work has not been presented elsewhere for assessment. The material that has been used from other sources it has been properly acknowledged / referred.

Signature of Student

MUHAMMAD ARSLAN TARIQ

2017-NUST-MS-CE-00000206127

Language Correctness Certificate

This thesis has been read by an English expert and is free of typing, syntax, semantic, grammatical and spelling mistakes. The work is original contribution of the author and does not contain any plagiarism. Moreover, Thesis is also according to the format given by the university.

Signature of Student

MUHAMMAD ARSLAN TARIQ

2017-NUST-MS-CE-00000206127

Signature of Supervisor

DR. MUHAMMAD UMAR FAROOQ

Copyright Statement

Acknowledgements

I am thankful to my Creator Allah Subhana-Watala to have guided me throughout this work at every step and for every new thought which You setup in my mind to improve it. Indeed I could have done nothing without Your priceless help and guidance. Whosoever helped me throughout the course of my thesis, whether my parents or any other individual was Your will, so indeed none be worthy of praise but You.

I am profusely thankful to my beloved parents who raised me when I was not capable of walking and continued to support me throughout in every department of my life.

I would also like to express special thanks to my supervisor Dr. Muhammad Umar Farooq for his help throughout my thesis and also for Selected Topics in Computer Networks courses which he has taught me. I can safely say that I haven't learned any other engineering subject in such depth than the ones which he has taught.

I would also like to thank Dr. Ali Hassan and Dr. Farhan Hussain for being on my thesis guidance and evaluation committee and express my special Thanks to these faculty members for their help.

Finally, I would like to express my gratitude to all the individuals who have rendered valuable assistance to my study.

Dedicated to my exceptional parents and adored siblings whose tremendous support and cooperation led me to this wonderful accomplishment.

Abstract

Network function virtualization (NFV) is a network built by an architectural concept and technique that involves the different information technologies for the virtualized functioning of different nodes and switches involved in it. Network function virtualization also creates a virtualized environment for commodity hardware that runs the different services by using the Network Function NFs through chaining and placement. Therefore, the involvement of chaining and placement in network function virtualization makes our services and applications easy and fast to accessible. However, it will be more cost-effective and will be a reason for increasing robustness throughout the whole working system.

To optimize the NFV chaining and placement, it is necessary to ensure that resource allocation is carefully carried out and orchestrated, preventing under or over-utilized NFs and service placement. In this order, we have formalized the network function chaining and placement problem and to cope with NFV placement, we propose the lightweight network function placement solution that takes into account dynamic hardware parameters and finding the shortest path to determine the utilization to place the Network function NF. In the process of chaining, we have performed the automation of the system as well. We have also demonstrated that we can run the Network function NFs with minimal 12 MB RAM and one vCPU to acquire our expected outputs.

Key Words: *Network function virtualization, Software defined network, service chaining, floyd warshall*

Table of Contents

List of Figures

List of Tables

CHAPTER 1: INTRODUCTION

For appreciating the need and motivation behind the networking industry's rapid adaption of the network function virtualization, it is important to discuss the history and background of the networking. Moreover, it is important to analyze the challenges that were faced at that time. In past years, technology was not evolved or upgraded that much to contribute and solve various serious issues that were faced by the IT and networking firms and organizations. There was a trend of utilizing the traditional network architecture that includes the use of traditional phone networks and telegrams as well.

1.1 Background, Scope and Motivation

Early over the design criteria and benchmark of the qualities by which the network was judged were the availability, capacity, latency, and throughput for carrying data with minimal loss. However, all the network devices that were traditional and designed in the past years were designed for specific functions. The network data built was designed customized and tailored for meeting the efficiency criteria decided by the developer effectively. The codes or software running to the custom-designed system were tightly coupled to it. They were integrated closely with the silicon programmable field and the customized integrated circuits. However, the traditional network architecture that was utilized before the NFV focused exclusively only on performing the specific functions of the device. In traditional network devices, the service providers were constantly looking for different ways to expand and scale their services without the increase in their cost. Many characteristics of the traditional devices create various constraints that limit the cost of the deployment, operational efficiency of the network, and scalability as well. Some of the limitations are,

- Flexibility limitations
- Issues in manageability of the device
- Time to market challenges
- High costs of operations
- Migration considerations
- Interoperability

It was important to develop or invent some other system that can easily deal with the above-mentioned limitations and the issues that were faced by traditional devices. In the present time, data centers have already proven the technology of the server virtualization approach, in which the stacks of independent serer of hardware are mostly replaced by the virtualized servers. Network function virtualization built over the concept of server virtualization. It means that to have an end to end data steaming it should pass through a series of functions. There are many examples of these functions already been used by different IT and software firms and organizations such as firewalls, software denied, WAN, encryption data etc. This focus of network function virtualization with the help of service training is how it relates to getting on board. Many enterprises are using this virtual network functions and for their correspondence, the NFV plays an important role. Network function virtualization allows many providers who are providing their internet services so that they can implement their key function which can arrange from broadband remote access to internet multimedia systems etc. NFV is a virtual machine that is helping out all the providers in the cloud environment and digital scopes [1].

Network function virtualization carries high future utilization and adaption by the business market and Information Technology firms. There are various pros and cons of utilizing the virtual network connection to provide internet services but one of the key problems is that the virtual machines VM have to be managed for their placement in different cloud services. This management issue can cause an interface and that would be problematic for clients if there is no implementation of the virtual method policies in a broader cloud environment which is not using a single interface. Working on this matter can improve the use of NFV in managing the VM in the future. It is important to consider the cons of NFV to make them more advance and flexible, which can be used in a different perspective in the upcoming time. Moreover, the problem with modular management can be a disadvantage of using this network function virtualization. However, this problem can be solved by using multiple tenants and sharing the sources to have broader control over the multi-cloud services. There are many advantages of using the virtual network such as that it can be set up when there is a heavy traffic flow coming so it can improve the operational efficiency of whatever enterprise it is functioning in. Technology is also broadly beneficial because it has the potential that it can enable the automated provisions of the applications which are loaded upon it. Network function virtualization gives another approach to

make, convey, and operate networking administrations. It is the way toward decoupling the system capacities from restrictive equipment machines so they can keep running over the virtualized environment on standardized hardware. This capacity (firewall, deep packet inspection, deception system and intrusion prevention) becomes a virtual network function (VNF). NFV is intended to convey the systems segments expected to help a foundation absolutely autonomous from equipment. These segments incorporate virtual register, storage, and system capacities. The concept of NFV came from the service providers who wanted to design something which can make the addition of new network or applications over hardware easily and in with a limited time period (faster). The hardware-based appliances allow applying the high standardized IT virtualization technology over there network. This idea was proposed just because of the main reason that was the high investment of money. The individual firewall system or any other system consumes a lot of time, energy and of course a great amount of money to execute the same function or process over the network.

The motivation behind using the service training system is that it allows deploying to deploy and chain together multiple LAN and IP networks. By using this it will provide an opportunity for an open connectivity system among all the software that is launched. This can beneficial even regarding the plugging boxes because it doesn't require any human assistance and can easily run through. There is no need for any networking team member for the surveillance of this service training system.

The NFV is high reliable network appliance. It can deliver

- High performance up to 100Gbps
- High reliability of about 99.999%
- Scalability to millions of the users
- Low-latency delivery of real-time applications
- Ability to integrate with legacy network architectures and link to existing operational and billing systems.

It means that it cuts off all the traditional usage of a network reliability program. For NFV it is compulsory to use high-performance servers especially the Intel-based servers. As it is faster, reduces costs in buying network equipment, flexible and used in broad software horizons, the NFV is the most advanced program. Though it faces challenges such as lacking mature

standards and difficulty using for many operators. It is widely available for the software community around [2].

1.2 Network Function Virtualization

A way of virtualizing the network services is known as the network function virtualization. It includes firewalls, load balancers, and routers, which have traditionally run over the proprietary hardware. However, these services are known or packaged as the "virtual machines" VM over the commodity hardware. A virtual machine allows the services provider to run their network system over the standard servers rather than the proprietary one. With the utilization of network function virtualization, there is no need to deploy dedicated hardware for every individual network function. The network function virtualization improves agility and scalability by allowing the service providers for delivering the new network services and the applications over demand as well. However, NFV didn't require additional hardware resources as well. The architecture of network function virtualization helps in defining and elaborating the standards for the implementation of NFV. ETSI has developed various standards, one of the most significant being the one given below which shows how the NFVI helps us to decouple hardware and software.

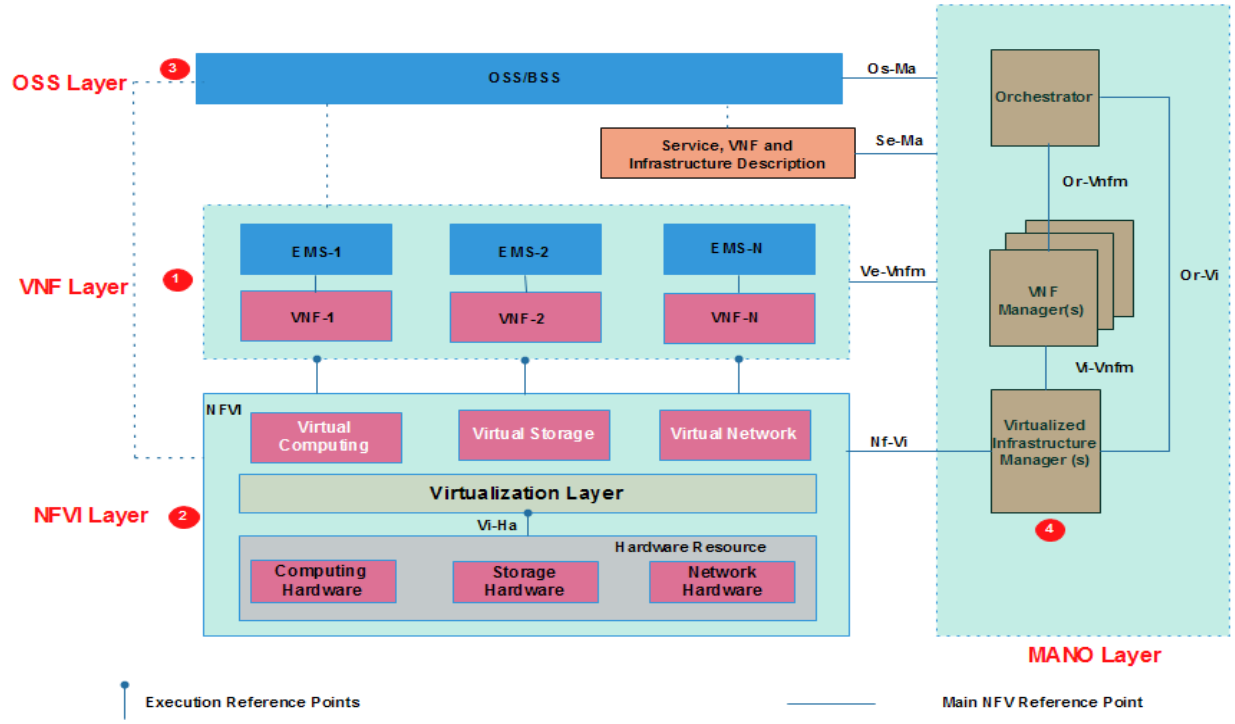

Figure 1.1: Architecture of Network Function Virtualization

1.2.1 Operation Support Subsystem (OSS)

OSS / BSS applies to an Operator's OSS / BSS. OSS performs the maintenance of networks, error control, device management and infrastructure management. BSS performs retail relations, inventory control, order relations and so on.

In the NFV architecture, the operator's decoupled BSS / OSS may be integrated using standard interfaces with the NFV Management and Orchestration. Network function virtualization carries high future utilization and adaption by the business market and Information Technology firms. There are various pros and cons of utilizing the virtual network connection to provide internet services but one of the key problems is that the virtual machines VM have to be managed for their placement in different cloud services. This management issue can cause an interface and that would be problematic for clients if there is no implementation of the virtual method policies in a broader cloud environment which is not using a single interface. Working on this matter can improve the use of NFV in managing the VM in the future. It is important to consider the cons of NFV to make them more advance and flexible, which can be used in a different perspective in the upcoming time. Moreover, the problem with modular management can be a disadvantage of using this network function virtualization. However, this problem can be solved by using multiple tenants and sharing the sources to have broader control over the multi-cloud services. There are many advantages of using the virtual network such as that it can be set up when there is a heavy traffic flow coming so it can improve the operational efficiency of whatever enterprise it is functioning in. Technology is also broadly beneficial because it has the potential that it can enable the automated provisions of the applications which are loaded upon it.

1.2.2 Management and Orchestration (MANO) Layer

Management and Orchestration Layer is also abbreviated as MANO and it includes three components:

- Virtualized Infrastructure Manager(s)
- VNF Manager(s)
- Orchestrator

MANO communicates both with layer NFVI and layer VNF. MANO layer handles all services in the network layer as well as generating and removing resources and controlling their

VNF allocation. The Virtualized Infrastructure Manager (VIM) contains the functionalities used under its jurisdiction to monitor and handle the connection of a VNF with and virtualize, processing, storage and network services. Virtualized infrastructure Manager performs the following:

- Inventory of NFV networking tools, processing , storage and network resources;
- Management and allocation of infrastructure resources e.g. increased VMs, increased energy efficiency etc.
- Hypervisor allocation of VMs, Resource processing, capacity and related network connections
- Root cause review of the output problems from the viewpoint of NFV technology
- Collection of details on network faults
- Collection of capacity planning , monitoring and optimization information

The VNF Manager is responsible for handling the life cycle of the VNF, including launch, updates, demand, scaling up / down, and termination. To assist multiple VNFs for each VNF, a VNF manager may be deployed, or one VNF manager may be deployed. Orchestrator is responsible for organizing and managing NFV infrastructure and software resources and realizing network services.

1.3 Software Defined Network

Software-defined technology to the network management that uses to enable the dynamic, configuration of the programmatically efficient network for improving the performance and monitoring of the network system. It used to make the network more like cloud computing rather than traditional network management. Further, the SDN is also known as the categories of the different technologies, which use to separate the network control plane from forwarding plane for enabling more provisioning automated and the policy-based management of network resources [14]. The driving concepts behind software-defined network creation and implementation are myriad. In reducing the complexity of statistically defined networks, the SDN helps. The implementation of the SDN makes the automation functions simple and intelligent; it also enables the simple provisioning and control of resources from the data center to the large network area.

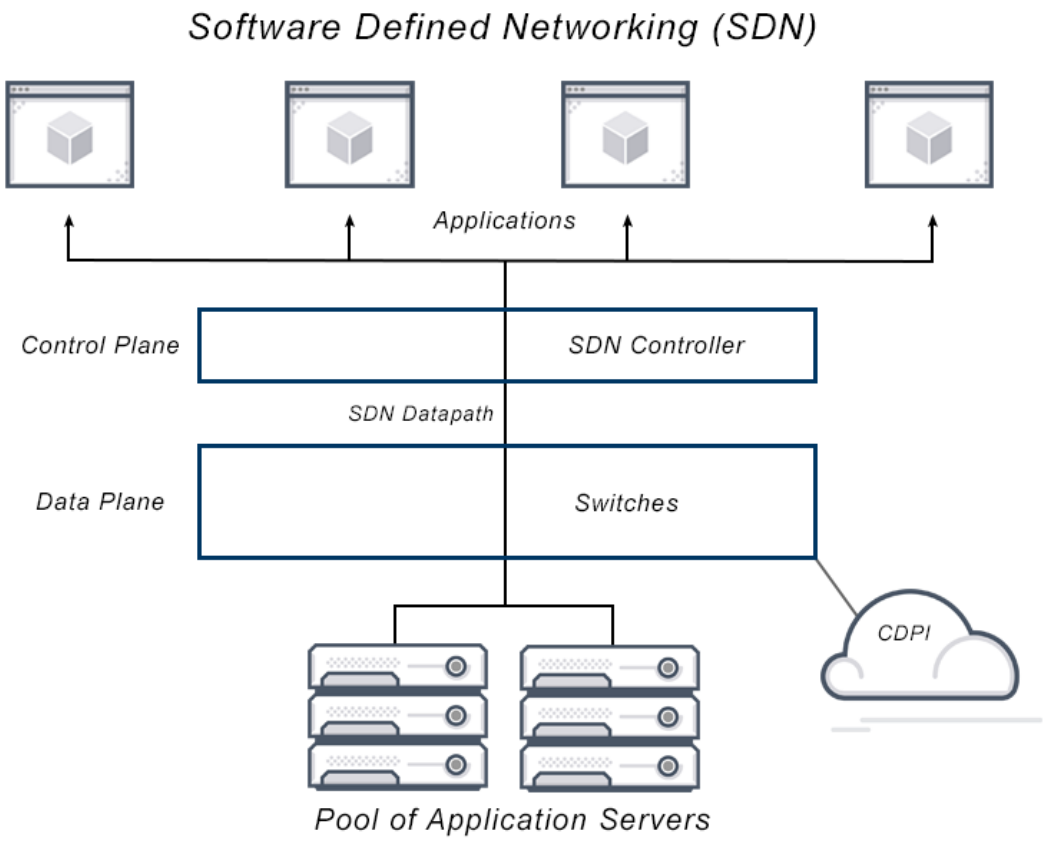

Figure 1.2: Architecture of Software Defined Networking

In SDN the open flow is known as the only one of the canons, however, it is a key component because of its networking software revolution was started. Open flow defines a network of programmable control that is used to manage and direct the traffic among the switches and routers. From past years since the inception of SDN, it has evolved into a reputable technology that is offered by the key vendors. The open network technology foundation develops a myriad of open-source SDN technology as well [4]. SDN broadly consists of three layers:

- Application layer
- Control layer
- Infrastructure layer

1.4 Problem Statement

One of the main challenge in using network devices is that with the passage of time network devices becomes obsolete and demands high cost for replacement and upgradation and complexity to run those devices and later with different vendors it becomes hard to manage all

devices for services running in infrastructure. We have proposed a solution and algorithm to cater these problems using network function virtualization NFV and also designed algorithm using cost of nodes and compute of running node to find the shortest path and automate these whole process.

1.5 Placement, assignment and chaining of NFV

Network function virtualization NFV [5] is an architecture that provides a new way to create, to operate and to distribute the different networking services. It is a simple way to decouple the different NFs (network functions) from miscellaneous hardware appliances. Those decoupled network functions can run over commodity hardware or dedicated cloud infrastructure. We need to implement the different service policies like traffic monitoring, routing and bandwidth utilization for different services running over hardware. With the passing time, it became hard to manage the service policies and security parameters for the services running on the same parameters, so the main agenda of network function virtualization chaining is proposed to replace the ordinary network and other devices with the virtualized network function (vNF). The basic motive to design such a function was to reduce the cost and save time which was spending on other physical hardware devices.

The placement of network function and chaining comprises of interconnecting a lot of system capacities (e.g., firewall, load balancer, and so forth.) through the system to guarantee system streams are given the right treatment. These streams must experience start to finish ways navigating a particular set of capacities. Generally, this issue can be disintegrated into three stages: 1) Placement 2) Assignment 3) Chaining [6]

1.5.1 Placement in NFV

In placement, we decide how many network functions are to be placed for efficient completion of the service requirements. This placement also involves the right places for these functions and it may even involve new NF instantiations. For instance, having a firewall service entertained at multiple points can help minimize latency and add to the network's performance. Usually, network functions are best suited when placed at the network's points of presence (N-PoPs) [7] to easily manage the traffic flows and helps to create the ideal environment for uninterrupted services running on infrastructure

1.5.2 Assignment in NFV

The assignment phase consists of allocating the network functions NFs for specific network tasks i.e. Firewall, load balancer etc. Based on network traffic and service requirement, it modifies the network functions accordingly and assign respective network services required from network function NF. For example, it may be more efficient to deploy more than one network functions NFs of firewall at the network's points of presence (N-PoPs) to effectively distribute the traffic and also monitors the traffic based on defined traffic rules and policies.

1.5.3 Chaining in NFV

Third phase is chaining, in this phase we create the process and service flows through inter connected virtualized network functions NFs throughout the network. Suppose we're running different IoT services simultaneously within same network, the traffic goes through same path and NFs which create latency and congestion issues over network. But with the help of chaining, we can create the different network paths according to their requirements then we can overcome this issue i.e. one service requires user authentication whereas other one is fetching data from server then we can choose different data path and network functions according to their needs.

Effective use of chaining involves flow based traffic engineering. NFV had been a topic to research since 1980s, but they have become more practical after inclusion of software defined network SDN [8], that separates the control and data planes for better data engineering. Hence, this can easily handle the traffic and congestion control over the network and helps us in creating virtualized network functions NFs according to services required.

Network function virtualization gives another approach to make, convey, and operate networking administrations. It is the way toward decoupling the system capacities from restrictive equipment machines so they can keep running over the virtualized environment on standardized hardware. This capacity (firewall, deep packet inspection, deception system and intrusion prevention) becomes a virtual network function (VNF). NFV is intended to convey the systems dministration segments expected to help a foundation absolutely autonomous from equipment. These segments incorporate virtual register, storage, and system capacities. The concept of NFV came from the service providers who wanted to design something which can make the addition of new network or applications over hardware easily and in with a limited time

period (faster). The hardware-based appliances allow applying the high standardized IT virtualization technology over there network. This idea was proposed just because of the main reason that was the high investment of money. The individual firewall system or any other system consumes a lot of time, energy and of course a great amount of money to execute the same function or process over the network. This chaining of NFV (network function virtualization) provides all the terminations (such as firewall, virtualized data inspection, intrusion, and deception) over a single platform, performing the same executing process in a limited or we can say short time and consuming almost no amount of money. So in this paper we'll discuss solution for chaining problem, placement of Network function over commodity hardware and algorithm to automate the assignment the Network function (NFs). Every single node contain different amount of memory and operating systems into it, here we will suppose those systems based on their utilization. The above-mentioned divisions are the inner divisions of every single node placed in our network chaining. In our network, we will place an SDN (software-defined network) and it will act as a virtual programming switch or switch instead of the physical system network. So rather than using ordinary switch for packet forwarding we'll use SDN for traffic controlling and packet forwarding. And determine the node which is enough capacity for the placement of NFs. We need to implement the different service policies like traffic monitoring, routing and bandwidth utilization for different services running over hardware. With the passing time, it became hard to manage the service policies and security parameters for the services running on the same parameters, so the main agenda of network function virtualization chaining is proposed to replace the ordinary network and other devices with the virtualized network function (vNF). The basic motive to design such a function was to reduce the cost and save time which was spending on other physical hardware devices.

1.5.4 NFV Service Chaining Challenges

One of the problems of digital network management is to develop an architectural architecture to enable and drive device and service development by focusing on the concepts of Software Defined Networks (SDN), Network Function Virtualization (NFV), and Cloud. Our performance depends on the notion of abstraction and the availability of stable service manager systems that meet the agility, optimization, and automation needs of next-generation networks.

1.6 IoT in NFV

The advancement and spread of the Internet of Things (IoT) have been massively increased over a decade. Millions of IoT devices and network has already been in production and gathering real-time data per millisecond. However, with the widespread of IoT networks, it's becoming difficult to acquire and execute real time data. Network function virtualization (NFV) enables to provide a flexible and efficient solution for IoT based applications and service management. In NFV based infrastructure, it offers dynamic network functions (NFs) for service chaining and placement (SCP) of different IoT applications. Network function virtualization (NFV) creates a virtualized environment on commodity hardware that runs the different IoT services by using the Network Function NFs through chaining and placement. This paper optimizes the NFV based chaining and placement algorithm to ensure that resource allocation is carefully carried out and orchestrated, preventing under or over-utilized NFs and IoTs service placement. In this order, we have formalized the network function chaining and placement problem, and to cope with NFV placement, we propose the lightweight network function placement solution that takes into account dynamic hardware parameters and finding the shortest path to determine the utilization to place the Network function NF. The Internet of Things (IoT) is becoming a reality with the exponential growth of information and communication technology. Computers, sensors, actuators, and software systems with communication capacity can be the "things" in the IoT. With the number of IoT devices expected to increase to more than 20 billion by 2020, intense network traffic will be on IoT applications. In addition, with more and more IoT technologies emerging, the diversification of IoT-traffic specifications is difficult for conventional network frameworks to match. To avoid these challenges and separate hardware infrastructure for every IoT application to run, network function virtualization NFV provides a new way to create, to operate, and to distribute the different networking services. It is a simple way to decouple the different network functions (NF) from miscellaneous hardware appliances. Those decoupled network functions can run over commodity hardware or dedicated cloud infrastructure. We need to implement the different service policies like traffic monitoring, service authorization, routing, and bandwidth utilization for IoT services running over hardware. With the passing time, it became hard to manage the service policies and security parameters for the IoT services running on the same parameters, so the main agenda of network function virtualization chaining is to replace the ordinary network and other devices with the virtualized

network function (vNF). The primary motive to design such a function was to reduce the cost and save time, spending on other physical hardware devices.

1.7 Thesis Contribution

We have thoroughly studied the behavior of network function virtualization and network function. We have investigated the research challenges and key issues that usually occurs in infrastructure. Finally, we have developed a chaining algorithm and solution, which provides the solution to most of the investigated problems. So, following are the main contributions in thesis:

- A lightweight NF placement solution is provided that take account of the dynamic hardware parameter i.e. compute utilization.
- An algorithm is provided to solve the assignment problem for selecting most suitable NF from multitude of NF instances
- Results from deploying network functions on commodity hardware are provided.

CHAPTER 2: LITERATURE REVIEW

We will highlight some of the most remarkable research work done on the network function virtualization, network function placement and work is done on their chaining process. We will relate the whole section with the efforts recently done to evaluate or to check the technical feasibility over the hardware on which the virtualization to be performed or done.

2.1 Related Work

Yong et al. [9] work over the framework of dynamical service chaining in the software-defined NFV system. It was observed that the cognitive radio and the software-defined radio use to enable the 5G future network. In this framework, the role of SDN and NFV is to enable high efficiency and flexibility in the construction of service chaining. It includes the flows of steering through the required service chain. It is done by the ability of traffic routing of the agile and deployment of the dynamic service. The overall system was carefully designed for the identification of the optimal mechanism, which use to boost up both resource utilization and performance. In this research, an optimization framework and a unified control are elaborated and explained for enabling the SDN-NFV framework, which is utilized for the optimization of the service chaining according to the requirement of the user and the network environment. By the designing of the services, network, and by the development of an optimization technique this system proves to be a beneficial framework for the miscellaneous scenarios of traffic steering. This scenario includes the selection of virtual machines, policy optimization, and function assignment. However, this research work over the chaining of network function virtualization will provide the benefits that can be simulated under a realistic problem or scenario.

Hwang et al [10] also came up with another idea of the utilization of a NetVM “net virtualization machine”, which is known as a very useful source while working over the virtualization or planning anything regarding it. NetVM carries virtualization to the Network by empowering high transfer speed system capacities to work at close line speed while exploiting the adaptability and customization of ease item servers. NetVM permits adjustable information plane handling capacities, for example, firewalls, proxies, and routers to be implanted inside virtual machines. NetVM makes it simple to progressively scale, convey, and reconstruct system capacities. This gives far more authentic reasons for adaptability than an existing reason

mentioned. The function of NetVM is to bring the virtualization to the network by enabling the bandwidth at high power to function near the line speed of network functions. NetVM operates the network function by this technique while gaining the advantage of the flexibility and slow speed of the low-cost commodity servers.

According to Lorenz et al. [11], the number of threat vectors and attacks is increasing with the passing years. Despite the occurrence of the following attacks, the main security system such as the firewall of the network remained unchanged and out of any secondary danger. Besides, various new challenges are raised not only over the level of security provided but also over the manageability and scalability. The manageability includes the deployment of countermeasures, which includes intrusion detections system and firewalls. Further due to the strict integration into the infrastructure of the physical network it is hard to adapt the security measures to the current condition of the network. All over, this research elaborates and demonstrates the various architectural designs for the process of integration of the NFV/SDN based solutions of the security into the enterprise network. The main security system such as the firewall of the network remained unchanged and out of any secondary danger. Besides, various new challenges are raised not only over the level of security provided but also over the manageability and scalability. The manageability includes the deployment of countermeasures, which includes intrusion detections system and firewalls. Further due to the strict integration into the infrastructure of the physical network it is hard to adapt the security measures to the current condition of the network. All over, this research elaborates and demonstrates the various architectural designs for the process of integration of the NFV/SDN based solutions of the security into the enterprise network.

2.2 Network Function Tools

A functional block building into a network infrastructure that contains the well-defined externally managed interfaces and a well-designed or defined functional behavior is known as the network functions. In simple and practical terms the network function is known as a network node or a physical appliance in today's generation. There are various tools of network functions that can be applied for the chaining of network function virtualization. The network function virtualization acts as flexible networking in the matter of the implementation of different tools. However, some important tools of network functions are explained as:

2.2.1 OPNFV

The OPNFV open framework for the virtualization of network functions is an open source platform released by the Linux Foundation in September 2014 [25]. The OPNFV aims to act like a carrier-grade and an integrated platform, which introduces the new and smart products to the company. The OPNFV use to promote the open-source networking that use to bring different organizations together for the acceleration of innovation and new technologies in the market. The main objective of OPNFV is the development of an open-source system to increase and boost up the functionality of the network function virtualization. OPNFV aims to bring the advanced services of NFs in the market. Further, as compared to the traditional network virtualization the OPNFV brings up the components in a simple, upgraded, and smarter way. It creates the simple end to end platform along with the storage and computation of the network virtualization. The main attention of OPNFV is an integration of the various stack testing, components, and automation. However, the architecture of OPNFV sum ups the virtualization process in integration, testing, and some of the new features having complete control of security [28]. As compared to the simple network function virtualization, OPNFV brings up the components in a simple and upgraded way. It creates a simple end-to-end platform along with the computation of storage and network virtualization. This tool pays attention to the integration of different components, stack testing, and the build by automation. However, it sums up the virtualization in testing, integration, and new features having a complete security control into it. Following are main objectives of OPNFV:

- Development of an open-source system to increase the functionality of NFV, bringing-up the advance services into the market
- Working on the standards of OPNFV to meet up-to-the requirements of companies using NFV
- Contribution in to be an active part of an open-source project
- Establishment of an ecosystem.

2.2.2 TripleO

TripleO is an official project of the OpenStack with an aim or goal of allowing the users to manage and deploy a production smart cloud onto the bare hardware metal by using a subset of an existing open stack component. This network function tool aims of the deployment and

utilization of an open stack, it also works for enhancing their documentation more than before. TripleO is a user-friendly tool, which lets users invest or spend their time on the development of script over the respective tool [17]. Moreover, it also helps the system to come up with the beneficial act of strong and smart security and bug fixing elements as well. The architecture of TripleO is designed as a user friendly and smart interface. It allows user to design their script according to the capacity of their RAM, storage, and CPU while working over the virtual machines VM. TripleO accepts and processes all the commands that are generated by the user during the designing and development of virtual machine VM. Moreover, TripleO can handle the two different modes that are PoC "proof of contact" and scale. In tripleo, when the user is working on virtual machines, they can design their script according to the capacity of their RAM, CPU, and disk space. As long as the disk has that much capacity, it accepts all commands created by the user during designing of VM. During the deployment of cloud, components act as a constraint.

Following are main objectives of TripleO:

- Tripleo comes up with the aim of deployment and utilization of an open stack by its self
- They worked to make their documentation more well-enhanced then before
- Users can invest their own time in creating the script over this tool.
- To come up with the beneficial act of strong security and bug fixing elements.

2.2.3 Tacker

Tacker is known as an OpenStack service for the orchestration of NFV that contains a general purpose of managing NFV for the deployment and operational process of virtual network functions and the network services over the platform of network function virtualization. The tacker is used for managing the OpenStack and enabling the remote CPE devices, it also used to do the deployment of the VNFs for providing the local network services. However, the tacker is also making network virtualization for converting its network services into a virtual function.

The architecture of the tacker is divided into the CLI and horizon that is directly connected to the API that is known as a plugin framework. The API is linked to the NFV catalog which works as the NFV descriptors. The NFV is further linked with the two different parts that are NFVO and VNFM. The VNFM works as the basic life cycle of the VNF, which helps in the good monetarization of the deployed VNFs. Whereas NFVO includes the VNF secure placement

policy and end-to-end deployment of network service in the decomposed VNF. Tacker divides its architecture into horizon and CLI which is directly connected by the API (plugin framework), which is linked to the NFV catalog. That catalog works as the NFV descriptors. That NFV is further linked by the two different parts (NFVO and VNFM). VNFM works as the basic life-cycle of VNF, and it helps in good monetarization of deployed VNF. Whereas NFVO includes the VNF secure placement policy and end-to-end deployment of network service in the decomposed VNF.

Following are main objectives of Tacker:

- Tacker is used by an NFV orcharstor and maintains its deployment to VNF in the SP network for providing the advance services to the remote customers' network
- It is also use to manage the open stack and to enable the remote CPE devices. The deployment of VNFs to provide local network services
- Tacker is also making network virtualization for converting its network services into a virtual function

2.2.4 MININET

MININET is known as a software emulator for the prototyping of a large number of networks over a single machine. It allows the users to create quickly, customize, share a software-defined function, and interact for the simulation of the network topology that utilizes the OpenFlow switches. MININET use to facilities over the manipulation of the software-defined network component, moreover MININET aims to access the network of its framework on different virtualization machines. MININET is further used for the support of arbitrary custom topologies and to use a different type of commands to get experience over an open flow system as well. The architecture of MININET is based on different isolated hosts, considered as a group of different nodes working over the network for passing the data packets. The framework of MININET contains different switches and a controller as well. The switch is set with the default Linux bridge system running with the kernel mode of network. However, the system also has applications, SDN application is a part of it as well. A mininet network is based on different isolated hosts, considered as a group of different nodes working over the network for passing the data packets. The architecture/software contains different switches and a controller as well.

Following are main objectives of MININET:

- Mininet facilities over the manipulation of software-defined network component
- To access the network of mininet on different virtualization machines
- To use a different type of commands to get experience over an open flow system
- Mininet also supports arbitrary custom topologies

2.2.5 CLICKOS

By the research and investigation of the researchers, a middlebox was conducted that was a high-performance technological middlebox platform which was termed as ClickOS. ClickOS consist of the Xen-based software-generated through the middle-box, which is the reason for the alterations in the subsystems of I/O, which can travel through the speed of 10 GB/s. The ClickOS is the reason for enabling hundreds of virtual network functions without any kind of delay or interruption in the processing of miscellaneous packets. The ClickOS contains a primary objective of turning the middleboxes into the programmed and virtual entities. This tool of network function also aims to be designed smaller in size (5MB) and less time consuming (30 milliseconds)/network functioning. ClickOS also deploys or implement a different firewall, load balancer, carrier-grade NAT, and others. Moreover, ClickOS is a sufficient way to make functioning smart and successful network function virtualization. Its architecture includes different walls and nodes for the passing of data packets. Initially, a switch is placed for the generation of data. The data is passed through a firewall and then by different nodes and walls as well. These walls are placed to check and clear up all the trash data from data packets. However, this system consumes the minimum time in a possible time limit. These walls are placed to check and clear up all the trash data from data packets.

Following are main objectives of CLICKOS:

- The primary objective of clickOS is to turn the middleboxes into virtual, programmed (software) entities.
- To design clickOS smaller in size (5MB) and less time consuming (30 milliseconds)/network functioning.
- ClicIOS to produce less delay (45 milliseconds).
- To implement different firewall, load balancer, carrier-grade NAT, and others.

In this section, we will discuss the methods or algorithms that we used in the service chaining of network function virtualization. These methods or processes of the shortest path are deployed for conducting the shortest way for the working of nodes throughout the whole chaining process. However, the following methods were deployed over the service chaining process:

- Floyd Warshall algorithm
- Dijkstra algorithm

2.2.6 Floyd Warshall Algorithm

Floyd Warshall algorithm is implemented to conduct the shortest path between all of the pairs of multiple vertices in a defined weighted graph. This algorithm used to work for both directed and undirected graphs but it is not applicable over the negative cycles (graphs having a sum of edges in a negative number). Floyd Warshall can also conduct the transitive closure of the directed graphs, inversion of the real matrices and it also conducts a test that if an undirected graph is a bipartite or not. However, the dynamic programming approach is used in Floyd Warshall for conducting the shortest path. Let's suppose,

v(i,j) is considered as the weight of edges between the i and j vertices, the shortest path can be defined as,

Shortest path (i,j,0)=v(i,j)

Its recursive case is defined as,

Shortest path (i,j,k)= min(shortest path(i,j,k-1), shortest path (i,k,k-1)+shortest path (k,j,k-1))

The above-mentioned formula is known as the main core formula of Floyd Warshall for conducting the shortest path. It works by conducting the shortest path of all pairs one by one till k=n.

2.2.7 Dijkstra Algorithm

Another way or algorithm to conduct the shortest path from the starting node to the ending node in a defined weighted graph is known as the Dijkstra algorithm. This algorithm used to work over the defined tree for conducting the shortest path from their starting vertex, source, and to all the other points that are mentioned in the graph. The graph of Dijkstra can be directed or non-direct, it does not work over the negative weights. Dijkstra algorithm is quite similar to

the Prism's algorithm for minimum spanning tree. The shortest path is generated with sources as the root. In determination through Dijkstra two sets are arranged, one set has the vertices that are included in the shortest path tree, the second set contains the vertices that are not included in the shortest path yet. Allover in every set of Dijkstra algorithm a vertex is conducted that is from the second set and have a minimum distance from source node [29].

The implementation of the modified Floyd Warshall and Dijkstra can be done over the open-flow in network function virtualization. The use of both of the modified algorithms is defined by Furculita et al., in their respective research conference paper. The paper highlights the idea of presenting a gear-box like an algorithm routing system whose results are obtained by the simulation process. However, both of the algorithms can be utilized in their best way for conducting the shortest path through a graphical method in the network function virtualization.

2.3 Network devices

The network chaining in function virtualization comprises the interconnecting of different system capacities.

2.3.1 Firewall

A firewall is designed for preventing unauthorized access or signals from an unknown network. The firewall blocks the malware and dangerous nodes, which are private or unknown in the network function system. Overall, a firewall is a security tool in the virtualization process.

2.3.2 Deep Packet Inspection

An advance and smart method of managing the network traffic in a system is known as the deep packet inspection. It is a kind of packet filtration that uses to identifies, locates, reroutes, block, and classifies the packets contains code payloads or any kind of specific data.

2.3.3 Intrusion Prevention System

The intrusion prevention system is another kind of network security deployed into the virtualization process. It is used for detecting the malware and threats in a system. An intrusion prevention system used to monitor the system in a continuous manner looking for various malicious issues and collecting information regarding them.

2.3.4 Deception System

A deception system is deployed for the post-breach detection of any malware or threats that pop-up or detected in the virtualization system. Moreover, it is also used for minimizing the risks that are faced by the system during the process.

CHAPTER 3: PROPOSED METHODOLOGY

In this chapter, we will explain the methodology of our proposed service chaining algorithm and load ratio mechanism. Our proposed service chaining algorithm is based on Floyd warshall algorithm, whereas load ratio mechanism utilizes compute of commodity hardware, and it is mainly aimed for network function placement. Following subsections explain both the variants of our proposed algorithm in detail.

3.1 Proposed Architecture

In this section, we discussed the network chaining problem and their generated or planned solutions. Secondly we formulize our network as an Integer Linear Programming Model, by using an algorithmic approach.

The designed network architecture system consists of four different layered network functions. A firewall is considered as a security tool which controls network trafficking according to the security rules. A propelled technique for controlling the system traffic is a type of bundle separating finds or recognizes the traffic and to reroute the particular information is known as Virtualized Deep Packet Inspection (vDPI). Intrusion prevention system another layer of the characterized chain is a sort of a system security framework that checks your framework and recognizes the errors, this framework screens your system constantly and watches out for all possible malicious occurrences and accumulate data with respect to it. The last working layer deception is the process which defends the attack by the hackers by transmitting the dummy or any random data to them. It works to identify the hacker's signature to block their access to the system.

In our mentioned chaining system a firewall will be placed at both starting points to check the transmitting of the packets through the network. If the firewall gets any suspicious or unknown packet so the data packet is transferred to the virtualized deep packet inspection for further check-up and inspection. The virtualized data packet inspector checks the packet to decide rather send the packet to the intrusion prevention system or directly to the node through the network. If any suspicious packet is found so the intrusion prevention drops it down automatically. If it found an unverified signature or any zero attacks so the packet is sent to the deception system for the further verification. Having such traffic forwarded through the desired

paths requires to have flow based traffic engineered for each user. This job can be best accomplished by having an SDN controller setting the forwarding rules on all switches.

In chaining, the service policy and chaining problem are tied so, this procedure comprises of making ways that interconnect the system capacities set and allowed in the past stages. This stage considers two significant elements, specifically, start to finish way latencies and unmistakable handling delays included by various virtual system capacities. In chaining all the nodes are chained-up or we can say are connected in a manner able way for the passing of the network traffic from the first router to another.

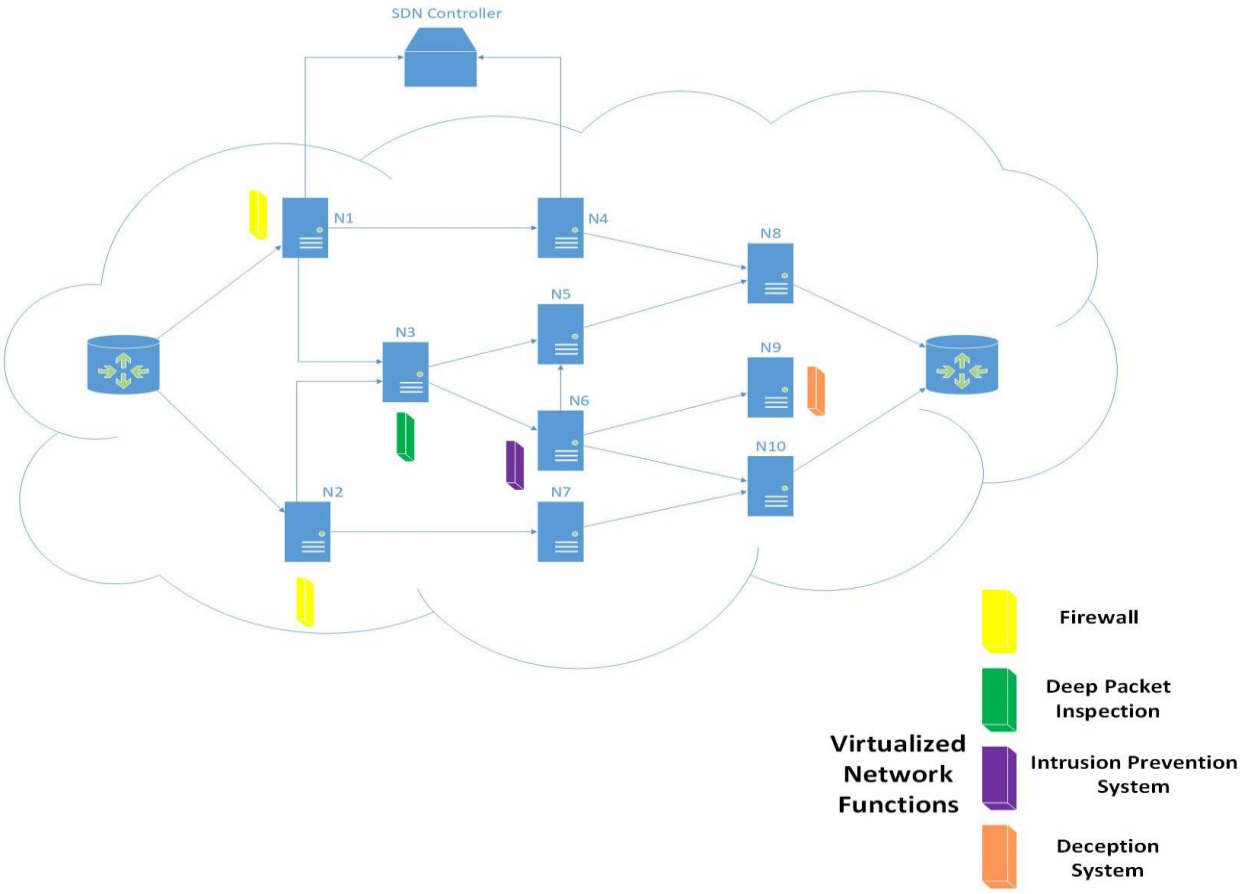

Figure 3.1: Proposed Cloud Infrastructure of multiple Nodes having SDN controller and Network Functions NFs

In this figure 3.1, two switches are set 1 in the beginning and 2 toward the end considered as a closure purpose of the system anchoring. An SDN controller is likewise put outside of the cloud-associated with two nodes (N1, N4). The beginning switch 1 is then additionally associated with two different paths (N1, N2), these paths are associated with the nodes of the firewall and that firewall nodes are associated with one deep packet inspection node (N3) and two simple data passing nodes (N4, N7). The data traffic will be generated by the switch, going through the firewall nodes, the firewall nodes will inspect that data traffic, if they discover any kind of issue so those information packets will be sent through the simply joined nodes for further assessment or inspection process, if there is no malware so data packets will go through the essentially joined nodes. The deep packet inspection is then associated with two different nodes (N5, N6), one of them is the node of the intrusion prevention system (N6) and the other is a simple node. The data packet went through deep packet inspection will go through the simple node if the malware or issue which was discovered will be tackled, yet in the event that there is some error left or the data packet is indicating an issue in the passing traffic so it will be gone through the intrusion prevention system for further checking. In the wake of going through intrusion prevention if the data packets are clear so will be sent through the simple data passing node or if not, as yet having issues so will be sent through the deception system node. This deception node will choose rather move the data traffic to the consummation point (second switch) or to dismiss it. We will utilize Floyd Warshall's algorithm to direct the briefest way in this binding. Floyd Warshall algorithm is implemented to conduct the shortest path between all of the pairs of multiple vertices in a defined weighted graph. This algorithm used to work for both directed and undirected graphs but it is not applicable over the negative cycles (graphs having a sum of edges in a negative number). Floyd Warshall can also conduct the transitive closure of the directed graphs, inversion of the real matrices and it also conducts a test that if an undirected graph is a bipartite or not. However, the dynamic programming approach is used in Floyd Warshall for conducting the shortest path. This algorithm used to work over the defined tree for conducting the shortest path from their starting vertex, source, and to all the other points that are mentioned in the graph. The graph of Dijkstra can be directed or non-direct, it does not work over the negative weights. Dijkstra algorithm is quite similar to the Prism's algorithm for minimum spanning tree. The shortest path is generated with sources as the root. In determination through Dijkstra two sets are arranged, one set has the vertices that are included in

the shortest path tree, the second set contains the vertices that are not included in the shortest path yet. The implementation of the modified Floyd Warshall and Dijkstra can be done over the open-flow in network function virtualization. The use of both of the modified algorithms is defined by Furculita et al., in their respective research conference paper. The paper highlights the idea of presenting a gear-box like an algorithm routing system whose results are obtained by the simulation process. However, both of the algorithms can be utilized in their best way for conducting the shortest path through a graphical method in the network function virtualization.

3.2 Load Ratio Mechanism

We introduced the model having different nodes, starting point and ending for the regulation and process of data packets. Now further, we will discuss the working and its probability in each node placed in this chaining.

Let's suppose our node's mechanism is divided into three different parameters i.e. CPU utilization, RAM and GPU/disk utilization having different probability ratios **(α, β and γ)**. We'll use performance unit PU to evaluate the total capacity of each nodes and after measuring performances of each nodes, we'll again evaluate its utilization ratios w.r.t each parameters to determine the current usage of nodes and to determine the less utilized node and future usability of node as a Network Function NF. Given below is the equation for determining the performance unit:

Performance Unit PU = α x CPU (No. of Cores) + β x RAM (GB) + γ x GPU (No. of Cores)

Whereas α is 0.5, β is 0.3 and γ is 0.2

Let's suppose we want to calculate performance unit of three nodes i.e. Node A, Node B and node C to determine the less utilized node to place network function NF. Node A has 8 GB of RAM, CPU 4 cores and GPU 4 cores and calculating PU is:

PU=0.5x4+ 0.3x8+ 0.2x4

PU=2+2.4+0.8=5.2

Table 3-1: Performance Unit of nodes

Parameters	CPU	RAM	GPU	PU
Node A	4	8	4	5.2
Node B	6	16	4	8.6
Node C	4	4	8	4.8

As we have computed the performance units of 3 different nodes, now we want to find out the utilization ratio UR based on the values of performance unit of all three nodes. So, Utilization ratio is basically utilization of each nodes at which they're currently running and based on this information we can compute the utilization ratio. Suppose Node A has utilization of 60% whereas Node B has 45% and node C 70%. So, we compute utilization ratio as:

UR= 60% of PU of Node A=0.6x5.2=3.1

UR= 45% of PU of Node B=0.45x8.6=3.8

UR= 70% of PU of Node C=0.7x4.8=3.3

Now, we have both performance unit PU and utilization unit UR to calculate the Average performance rate APR to check the remaining unused compute of nodes:

APR=PU-UR

APR of Node A=5.2-3.1=2.1

APR of Node B=8.6-3.8=4.8

APR of Node C=4.8-3.3=1.5

As shown above, and by the rule "The Greater the better" Node B has more unutilized resources than Node A and Node C. So, if required SDN will choose Node B to place new network function NF.

This whole mechanism is tested or performed for the process of chaining, for checking or determining a node which will utilize the minimal resources and provides the least utilized nodes so that we can place our NFs.

Every single node contain different amount of memory and operating systems into it, here we will suppose those systems based on their utilization. The above-mentioned divisions are the inner divisions of every single node placed in our network chaining. In our network, we will

place an SDN (software-defined network) and it will act as a virtual programming switch or switch instead of the physical system network. So rather than using ordinary switch for packet forwarding we'll use SDN for traffic controlling and packet forwarding. And determine the node which is enough capacity for the placement of NFs. At all starting points, a firewall will be placed to verify the propagation of the packets across the network. If any unusual or unexplained packet is received by the firewall, the data packet is passed for further check-up and analysis to a virtualized deep packet inspection. The auditor of the virtualized data packet tests the packet to determine whether to deliver the packet to the intrusion prevention device or directly across the network to the node instead. If any suspicious packet is detected, it is immediately dropped by intrusion protection. If an unverified signature or zero attacks are detected, the packet is submitted for further verification to the deception device. For each user, making those traffic forwarded along the desired paths allows flow-based traffic to be engineered. By having an SDN controller setting the forwarding rules on all switches, this job will better be done. The designed network architecture system consists of four different layered network functions. A firewall is considered as a security tool which controls network trafficking according to the security rules. A propelled technique for controlling the system traffic is a type of bundle separating finds or recognizes the traffic and to reroute the particular information is known as Virtualized Deep Packet Inspection (vDPI). Intrusion prevention system another layer of the characterized chain is a sort of a system security framework that checks your framework and recognizes the errors, this framework screens your system constantly and watches out for all possible malicious occurrences and accumulate data with respect to it. The last working layer deception is the process which defends the attack by the hackers by transmitting the dummy or any random data to them. It works to identify the hacker's signature to block their access to the system.

3.3 Shortest path algorithm

We need to determine the distance of each node to each other and try to find the shortest path. So, we're using Floyd warshall algorithm to determine the shortest path.

"The Floyd Warshall algorithm is an algorithmic method of finding the shortest path in a weighted graph with both positive and negative edge weight, but no negative cycles are allowed in it." Floyd Warshall can also conduct the transitive closure of the directed graphs, inversion of the real matrices and it also conducts a test that if an undirected graph is a bipartite or not.

However, the dynamic programming approach is used in Floyd Warshall for conducting the shortest path.

Floyd warshall compare out all the possible paths with the help of the graph in between the pair of different vertices. The comparisons Θ(|V|^{3}) is use in the graph, maybe there are up to Ω(|V|^{2}) edges in working graph. In this algorithm every combination made over the edges are tested. It does incrementally improving the way to find out the shortest path until the graphical estimation is optimal.

However, after solving the algorithm we conclude a general formula or a form of the matrix that we use to conduct the shortest path in any graph.

General formula:

Shortest path (i,j,k) = min[shortest path(i,j,k-1), shortest path(i,k,k-1) + shortest path(k,j,k-1)]

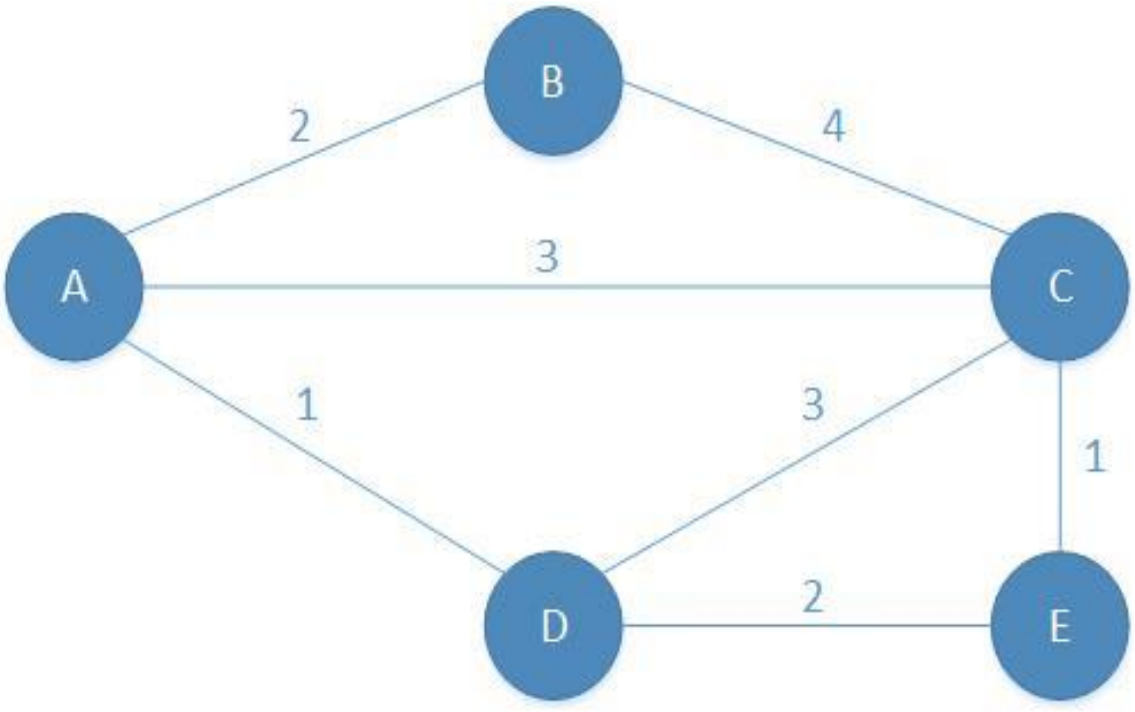

Figure 3.2: Proposed multi-node path

$$\begin{Vmatrix} 0 & 2 & 3 & 1 & 3 \\ 2 & 0 & 4 & 3 & 5 \\ 3 & 4 & 0 & 3 & 1 \\ 1 & 3 & 3 & 0 & 2 \\ 3 & 5 & 1 & 2 & 0 \end{Vmatrix}$$

Calculating the sum of each row gives us total distance cost of all nodes in that respective row. And the summation of these rows is followed as:

$$\begin{Vmatrix} 9 \\ 14 \\ 11 \\ 9 \\ 11 \end{Vmatrix}$$

Now, using load ratio mechanism and Floyd warshall algorithm given the 50-50 split ratio to acquire the results from both methods to determine the next potential network functions NFs on infrastructure as below:

Node A=50% of load ratio mechanism (APR) + 50% of Floyd warshall algorithm (Total node cost)

Node A= 0.5*2.1 + 0.5*9 = 1.0 + 4.5 = 5.5

Node B= 0.5*4.8 + 0.5*14 = 2.75 + 7 = 9.4

Node C= 0.5*1.5 + 0.5 * 11 = 0.75 + 5.5 = 6.25

After getting the mentioned above results, we can do the selection of node by "**The Greater the Better**" method to place the network functions over there.

CHAPTER 4: EXPERIMENTAL SETUP AND RESULTS

In this section, we are discussing the ClickOS tool which we're using for our performance evaluation in real time physical environment. As we discussed earlier that ClickOS contain different nodes joined with each another. ClickOS can easily handle data packets in a million/second. Furthermore, Considering ClickOS have many benefits and technological points discussed in the brief architecture section.

4.1 Network Function Tool

The main objectify points, or we can say that the placement of ClickOS contains the following nodes and switches (i.e., firewall, switches, and nodes). However, ClickOS is not summed up only into these given objects.

The leading architectural designing or the software has a Xen to make the system easy to understand and fast in speed. This system of Xen runs virtual machines of clickos; every described version of clickOS is running over the top of the miniOS. Xen was made a part of this working system because it provides a secure and better performance to the modified operating systems as a guest. However, the guest Operating System (OS) is unmodified. For the running process of ClickOS user do provide a configuration and managed text files of the different interconnected elements. While running the programmable files ClickOS provides the different readable/writable handlers and miscellaneous variables as well. These files and variables can easily change the state of an element at any instant time. ClickOS is depended on the switches or the proc/file system to provide the functioning through these described mechanisms [20].

ClickOS operates the three different parts of the plane of the operating system. First one is the C-based CLI that take care of different tasks, creating and destroying the guest domains coming into ClickOS for the functioning purpose. Whenever a guest domain or the data packet boosts up miniOS, a threat is generated, that is considered as the second part of the controlling the system's plane. This thread creates an entry to the Xen functioning store, a /proc file system is shared between all the guests running domains or data packets and dom0. It is the duty of the control thread system then to watch for the changes if they are made during the entry. However, when the configuration settings are done or written by the user, the written function can take care

of threating issues occurred again and again during the chaining. It merely shows that the different click instances can work at a single time through the single domain of ClickOS.

The third part of the architecture of the plane operating system consists of a new click element named as clickOS control. It controls all the given elements by configuration on one end and the Xen placement store on the second end. The CLI at that point gives users an interface to peruse and keep in touch with component (to read and write to the element) handlers through the Xen store and ClickOS Control. Every one of these activities on the ClickOS side of the things and processed is done or executed in the safe and control thread maintains the environment mentioned above.

4.2 Experimental Setup

In this section, we are discussing the deployment of network functions NFs over commodity hardware using ClickOS to evaluate working of network functions. First, we elaborate the hardware and functions used in this experiment and then the application of this equipment over the virtualization function.

4.2.1 Implementation of ClickOS

In this section, a discussion will be done regarding the setup of a Virtual Machine for running the ClickOS. Use the VM image with the installation of UBUNTU 14.04.4 “64-BIT LTS”, it helps in the downloading of an image and using it as a starting point. However, Virtual Box is needed for downloading the image.

- Configuration of the virtual machine
- Setting the memory to the 1024MB
- Set the total number of the processors to “2”

4.2.2 Installation of XEN

XEN is considered as one of the most known virtualized platforms. In the respective project implementation of XEN is considered for supporting the network function.

- Installation of dependencies for XEN
- Making a directory known tutorial
- Downloading the XEN-4.4.1 in the directory of tutorial
- Compilation and building of XEN
- Configuration of the system for booting from XEN
- Updating some of the required configurations
- Updating the grub and rebooting of the system
- Verification of the installation of XEN

4.2.3 Building the ClickOS

Another mandatory step of the installation of ClickOS includes the following points,

- Getting the source code
- Setting up the environment variables
- Replacement of the mini-OS
- Building the toolchain
- The building of the ClickOS kernel
- Building the COSMOS

4.2.4 Installation of the Open vSwitch

OVS (open virtualization switch) setup should be set according to the MAC's address, memory, and vCPUs (virtual central processing unit) of a virtual domain for the smooth running of a virtual switch. For testing purposes, we make the content depicting system work for simply getting the packet, print OK and afterward send the packet back to the system interface to check the working of system work.

- Installation of the Open vSwitch
- Configuration and running OVS
- Creating a bridge over OVS

4.2.5 Instantly starting a ClickOS

For starting the ClickOS in instance the system have to,

- Create a configuration file for the XEN
- Create a configuration file for the Click
- Start the ClickOS instantly

We compose the content in Click language because the click is a language that can define network function system capacity dependent on click modular. We design the script and execute the click system capacity to check that the clickOS case is running. Network function starts running with 12MB memory and 1 vCPU as shown in figure 4.1 use giving us the yield of the required content.

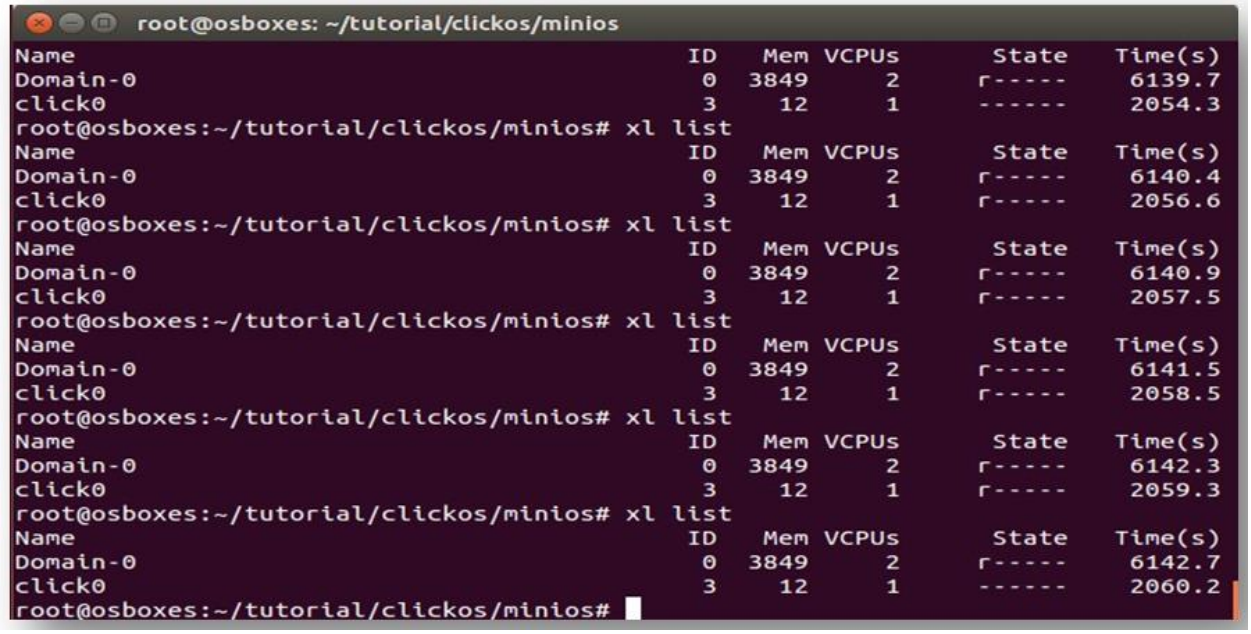

Figure 4.1: Utilization of Network Functions in ClickOS

And in figure 4.2 we can see the output of click0 console network function that receives packets from the network interface, prints the packets, and sends the packets back to the network.

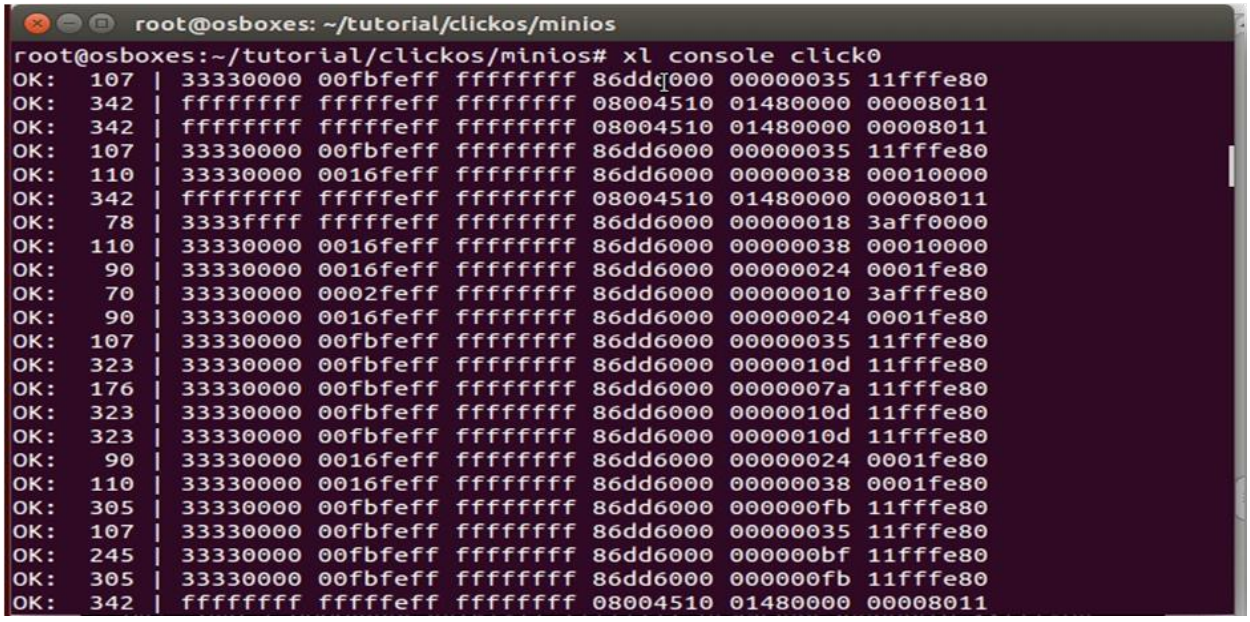

Figure 4.2: Output of Network Function in ClickOS

Also, we have implemented the network address translation NAT over the commodity hardware using network function NF.

```
ubuntu@ubuntu:~/cmpe210_clickos_setup$ sudo vim NAT.click
define($MAC 00:00:00:00:01:00);
source :: FromDevice;
dest :: ToDevice;


AddressInfo(
        IP1 192.168.56.105,//Outer IP1
        IP2 192.168.56.106,//Outer Ip2
        IP3 192.168.56.107,//Outer IP3
        IP4 192.168.56.108,//Inner Ip4 for outer IP1
        IP5 192.168.56.109,//Inner IP5 for outer IP2
        IP6 192.168.56.110);//Inner Ip6 for outer IP3

c :: Classifier(
        12/0806 20/0001 38/c0a83869,//ARP Req for IP1
        12/0806 20/0001 38/c0a8386a,//ARP Req for IP2
        12/0806 20/0001 38/c0a8386b,//ARP REq for IP3
        12/0800 34/08,//ICMP
        -);

f :: IPFilter(
        0 dst host IP1,//ICMP IP1
        1 dst host IP2,//ICMP IP2
        2 dst host IP3,//ICMP IP3
        3 all);//Firewall for all others
```

Figure 4.3: Network Address Translation in ClickOS

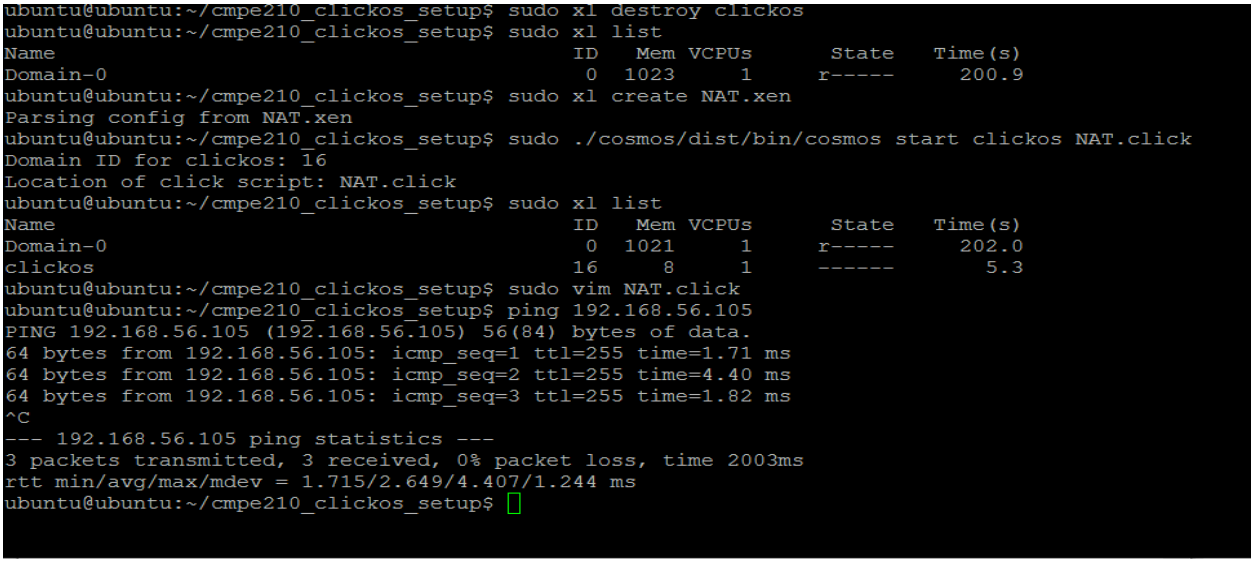

Figure 4.4: Output of Network Address Translation in ClickOS

In Figure 4.3, we have initialized public IPs and private IPs for network address translation, and in figure 4.4, we pinged the public IP through clickOS to check it's NF, and it successfully responded back

4.3 Simulation

We created different NFs in a physical environment for our virtualization process but these NFs are designed for a specific parameter, so we implemented these NFs over a simulator to get our desired result. Furthermore, we implemented a simulator of NS3 with a bandwidth of about 2mbps; recorded time of simulation process is about 3 minutes. We have implemented the cache server placement using the NS3 simulation tool. The proposed approach assumes a scenario of nodes connected and checked the End to End delays from IoT users to all attached nodes for better placement of the cache server with controller help. OpenFlow 1.3 [22] is used for Controller to switch communication, and flow monitor [23] is used for obtaining results. OpenFlow 1.3 module brings the OFSwitch13LearningController class that implements the controller interface to work as a "learning bridge controller." [19]. This learning controller guides the OpenFlow switches to forward incoming frames from one port to the single correct output port whenever possible [24]. Content query routing exceeds the scope of this paper. Therefore, we presume that switches route the requests according to the network routing protocol (e.g., shortest path routing).

Table 4-1: Simulation Results of nodes

Simulator	Bandwidth	Application	Traffic Type	MTU	Data Rate	Simulation Time
NS3	2Mbps	On off Application	Constant Bit Rate	1000 byte	100Kb/s	3 Minutes

	Switch1	Switch2	Switch3	Switch4	Switch5	Switch6	Switch7	Switch8	Switch9	Switch 10	Origin Server
Link Delays (ms)	4	4	7	10	9	8	9	11	11	15	19

Place The cache server in each Open Flow Switch and measure the latency from sender

	Cache server at S1	Cache server at S2	Cache server at S3	Cache server at S4	Cache server at S5	Cache server at S6	Cache server at S7	Cache server at S8	Cache server at S9	Cache server at S10	Origin Server
LATENCY (ms)	12.40	12.40	15.40	18.41	17.41	16.41	17.41	19.41	19.41	23.42	27.42

CHAPTER 5: CONCLUSION AND FUTURE WORK

5.1 Conclusion

Network function virtualization is an architecture that contains the capacity to recreate the different network functions over a virtualized network. After working over the service chaining of virtualization we also concluded that network function virtualization can help us to cater many future problems regarding to the virtualization of NFs and optimization of curricular technical issues.

In this thesis, we formularized the network function chaining and assignment problems in NFs. We broadly discussed that how we can a virtualized environment for commodity hardware that runs the different services by using the NFs. For that purpose we discussed the problem overview and methodology in which we covered the placement, chaining and assignment of our nodes defined in our virtualization process.

Further, we purposed an optimized algorithm *"FLOYD WARSHALL ALGORITHM"* to solve the shortest path for our chaining [13]. Unified control architecture for a dynamic service chaining using SDN controller is also discussed in the related work section. We also discussed our simulation process in which we used a NS3 simulator to test our virtualization process. However, we conducted our simulation result in a physical environment that shows us that the minimum consumption of memory was done (about 12 MB) in a simulation of around 2000 secs. Also, we mean to explore different software to reoptimize the NFs placement, assignments, and chaining. Moreover, we discussed these different softwares in detail as our tools that include OPNFV, TRIPLEO, TACKER, MININET and CLICKOS.

5.2 Future Work

Increased services over the internet has given multiple challenges for network devices to perform different actions for assigned services. In future, the service requirement and chaining of these services will increase exponentially and we have to come up with automated design to cater all these problems. Future research along these lines include the use of mathematical framework optimization to improve the algorithm's scalability and robustness. In order to incorporate a more detailed output model of network devices with respect to the rules they implement, we also aim to enrich our model. Finally, in order to simplify the general issue of

service chaining, where various forms of theoretically highly accessible VNFs have to be put, intertwined and crossed, we plan to expand our model and create an integrated model. A lightweight automated solution should be for chaining of network function NFs for IoT applications. IoT applications have massively increased, and user-based data and applications require different functions to cater to the different service requests. So, network function virtualization provides the solution on commodity hardware to install the required network function instantiations and instantly carry out the results. For automation and placement, we can combine two algorithms, i.e., load ratio mechanism, to check the current commodity hardware utilization. Secondly, finding the shortest path using Floyd warshall algorithm for automation to the nearest available node has enough computed for network function provision.

REFERENCES

[1] M. C. Luizelli, L. R. Bays, L. S. Buriol, M. P. Barcellos, and L. P. Gaspary, “Piecing together the NFV provisioning puzzle: Efficient placement and chaining of virtual network functions,” 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM), 2015.

[2] Y. Li, F. Zheng, M. Chen, and D. Jin, “A unified control and optimization framework for dynamical service chaining in software-defined NFV system,” IEEE Wireless Communications, vol. 22, no. 6, pp. 15–23, 2015.

[3] “IEEE Recommended Practice for Routing Packets in IEEE 802.15.4 Dynamically Changing Wireless Networks.”

[4] A. Hakiri, A. Gokhale, P. Berthou, D. C. Schmidt, and T. Gayraud, “Software-Defined Networking: Challenges and research opportunities for Future Internet,” Computer Networks, vol. 75, pp. 453–471, 2014.

[5] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck and R. Boutaba, "Network Function Virtualization: State-of-the-art and Research Challenges"

[6] O. Soualah, M. Mechtri, C. Ghribi, and D. Zeghlache, “An efficient algorithm for virtual network function placement and chaining,” 2017 14th IEEE Annual Consumer Communications & Networking Conference (CCNC), 2017.

[7] M. C. Luizelli, W. L. D. C. Cordeiro, L. S. Buriol, and L. P. Gaspary, “A fix-and-optimize approach for efficient and large scale virtual network function placement and chaining,” Computer Communications, vol. 102, pp. 67–77, 2017.

[8] M. M. X. N. N. K. O. T. T. Bruno Nunes Astuto, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks," HAL, 2013.

[9] Y. Li, F. Zheng, M. Chen, and D. Jin, “A unified control and optimization framework for dynamical service chaining in software-defined NFV system,” IEEE Wireless Communications, vol. 22, no. 6, pp. 15–23, 2015.

[10] C. Lorenz, D. Hock, J. Scherer, R. Durner, W. Kellerer, S. Gebert, N. Gray, T. Zinner, and P. Tran-Gia, “An SDN/NFV-Enabled Enterprise Network Architecture Offering Fine-Grained Security Policy Enforcement,” IEEE Communications Magazine, vol. 55, no. 3, pp. 217–223, 2017.

[11] G. Carella, J. Yamada, N. Blum, C. Lück, N. Kanamaru, N. Uchida, and T. Magedanz. 2015. “Cross-layer service to network orchestration. In Proceedings of the 2015 IEEE International Conference on Communications” (ICC’15). 6829–6835

[12] Fraunhofer FOKUS. 2016. OpenSDNCore—Reasearch and testbed for the carrier-grade nfv/sdn environment. Retrieved July 25, 2016 from http://www.opensdncore.org/

[13] J. Sherry, S. Hasan, C. Scott, A. Krishnamurthy, S. Ratnasamy, and V. Sekar, “Making middleboxes someone elses problem,” ACM SIGCOMM Computer Communication Review, vol. 42, no. 4, pp. 13–24, 2012.

[14] R. Cziva, S. Jouet, K. J. S. White, and D. P. Pezaros, “Container-based network function virtualization for software-defined networks,” 2015 IEEE Symposium on Computers and Communication (ISCC), 2015.

[15] R. Cziva and D. P. Pezaros, “Container Network Functions: Bringing NFV to the Network Edge,” IEEE Communications Magazine, vol. 55, no. 6, pp. 24–31, 2017.

[16] Docker Inc. 2016. Docker Documentation. Retrieved July 25, 2016 from https://docs.docker.com/

[17] S. V. Rossem, W. Tavernier, B. Sonkoly, D. Colle, J. Czentye, M. Pickavet, and P. Demeester, “Deploying elastic routing capability in an SDN/NFV-enabled environment,” 2015 IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN), 2015.

[18] Rackspace Cloud Computing. 2016. OpenStack Open Source Cloud Computing Software. Retrieved July 25, 2016 from https://www.openstack.org/

[19] Linux Foundation. 2016. The OpenDaylight Plataform. Retrieved July 25, 2016 from http://www.opendaylight.org

[20] J. Deng, H. Hu, H. Li, Z. Pan, K.-C. Wang, G.-J. Ahn, J. Bi, and Y. Park, “VNGuard: An NFV/SDN combination framework for provisioning and managing virtual firewalls,” 2015 IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN), 2015.

[21] A. Gupta and R. K. Jha, “A Survey of 5G Network: Architecture and Emerging Technologies,” IEEE Access, vol. 3, pp. 1206–1232, 2015.

[22] Pedro Fortuna and Manuel Ricardo, "FlowMonitor - a network monitoring framework for the Network Simulator 3 (NS-3)" 2009.

[23] Luciano Jerez Chaves, Islene Calciolari Garcia, and Edmundo Roberto Mauro Madeira, "OFSWITCH13: Enhancing ns-3 with Openflow 1.3 Support," 2016

[24] Yong Cui , Jian Song, Minming Li , Qingmei Ren, Yangjun Zhang, and Xuejun Cai, “SDN-Based Big Data Caching in ISP Networks”,pp. 1-4, 2018.

[25] AT&T, Telecom Italia, Netronome, Intel, ServiceMesh, PLUMgrid, and Cisco Systems. 2015. PoC#16—NFVIaaS with Secure, SDN-controlled WAN Gateway. Technical Report. The European Telecommunications Standards Institute.

[26] S. E. C. a. L. S. Guerassimov, "NFV and OPNFV," Network Architectures and Services, 2016

[27] C.-H. Lin, J.-C. Liu, M.-H. Liou, and W.-C. Wu, “Shortest Driving Time Computation Based on Cloud Technologies and Genetic Algorithm,” 2014 5th International Conference on Intelligent Systems, Modelling and Simulation, 2014.