

**Virtualized Network Function (VNF) based placement and chaining
solution for enterprise network security**



By

Muhammad Safwan Qureshi
(Registration No.: 00000320984)

Supervisor

Dr. Muhammad Umar Farooq

DEPARTMENT OF COMPUTER & SOFTWARE ENGINEERING
COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING
NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY
ISLAMABAD

June, 2023


THESIS ACCEPTANCE CERTIFICATE


Certified that final copy of MS/MPhil thesis written by NS **Muhammad Safwan Qureshi** Registration No. 00000320984, of College of E&ME has been vetted by undersigned, found complete in all respects as per NUST Statutes/Regulations, is free of plagiarism, errors and mistakes and is accepted as partial fulfillment for award of MS/MPhil degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have also been incorporated in the thesis.

Signature : _____ 

Name of Supervisor: **Dr Muhammad Umar Farooq**

Date: 14-06-2023

Signature of HOD: _____ 
(Dr Usman Qamar)
Date: 14-06-2023

Signature of Dean: _____ 
(Brig Dr Nasir Rashid)
Date: 14 JUN 2023

Dedicated to my outstanding teachers and family, whose unwavering help and collaboration helped me achieve this fantastic goal.

Acknowledgements

I am appreciative to my Creator Allah Subhana-Watala for providing me with His guidance at every turn of this work and for each fresh idea You implanted in my thoughts to make it better. True enough, without Your invaluable support and direction, I would have been powerless. Nobody else should receive honor besides You because everyone who assisted me with my thesis, including my parents and anyone else, did so at Your command.

I owe my devoted parents a huge debt of gratitude for raising me when I was unable to walk and for supporting me through every phase of my life.

Additionally, I want to extend a special thank you to my thesis advisor, Dr. Muhammad Umar Farooq, for his guidance throughout the writing of my thesis and for teaching me Selected Topics in Computer Networks courses. I can confidently claim that the technical topics he has taught are the only ones I have studied in such depth.

Furthermore, I would like to express my gratitude to Drs. Ali Hassan and Farhan Hussain for serving as members of my thesis counselling and evaluation committee.

Finally, I would want to offer my gratitude family and to everyone who has helped my research in any way

Abstract

The Internet of Things (IOT) is the new standard; everything is connected to the internet and is managed by technology. All spheres of life are now interconnected under one umbrella, thanks to information technology. However, such a system has also presented other networking concerns, including issues with over-resource use and security. Major issues in an enterprise network include network security, processing power, latency, bandwidth, and energy use, which have emerged as top difficulties. All of the aforementioned problems are experienced with traditional network deployment approaches. Although some of these issues have diminished since the introduction of Software Defined Network (SDN), there is still room for improvement. It is essential to plan and direct network traffic in such a way that the goal can be achieved in less time with greater precision and accuracy in order to meet today's problems in the network area. Without embracing cutting-edge technologies like Software Defined Network and Network Virtual Functions, it would not be able to deliver such an effective solution in the quickly expanding information technology industry. In our work, SDN-NFV based network topology has been demonstrated which offers all most all the required functionality of a small level to a medium level enterprise network. It demonstrates how to develop a security solution based on SDN-NFVs utilizing the ONOS SDN Controller, Zodiac FX OpenFlow switches, and virtual network functions (VNF). VNF consists of Virtual Security Functions (VSF), which include Pfsense, an IPS, and an IDS for detecting intrusions. The deployment of an enterprise security solution using an SDN-NFV platform and commodity hardware is one of the major achievements of this work. Moreover, in this work simulation based SDN-NFV based service function placement and chaining has been tested to present that how Virtualized Network Function (VNF) based placement and chaining solution for enterprise network security can be overall effective.

Key Words: *Network function virtualization, Software defined network, NFV Placement service function chaining, Network security Function (VSF), IDS(Intrusion Detection System), Intrusion Prevention System (IPS).*

Table of Contents

Dedication	ii
Acknowledgement	ii
Abstract	iii
Table of Contents	iv-v
List of Figures	vi
List of Tables	vii
CHAPTER 1: INTRODUCTION	1
1.1 Network Function Virtualization	5
1.1.1 Architectural framework of NFV:	6
1.1.2 Virtualization Network Function (VNF) Layer	9
1.1.3 NFV Infrastructure (NFVI) Layer	10
1.1.4 Management and Orchestration (MANO) Layer	10
1.2 Software Defined Network	11
1.2.1 CONTROL PLANE.....	13
1.2.2 DATA PLANE	14
1.2.3 SDN CONTROLLER.....	14
1.3 Scalability.....	19
1.4 Telemetry	19
1.5 Resilience and Fault Tolerance	19
1.6 How ONOS works?.....	19
1.7 OpenFlow Protocol	21
1.8 Security	24
1.9 Problem Statement	24
1.10 Placement, assignment and chaining of NFV.....	24
1.10.1 Placement in NFV	25
1.10.2 Assignment in NFV	25
1.10.3 Chaining in NFV	25
1.10.4 NFV Placement and Service Chaining Challenges for an enterprise security	26
1.11 Thesis Contribution.....	27
CHAPTER 2: LITERATURE REVIEW and SFC Algorithms and Network Devices	28
2.1 Related Work	28
2.2 Service Chaining Algorithm.....	29
2.2.1 Floyd Warshall Algorithm	30
2.2.2 Dijkstra Algorithm.....	30
2.2.3 Shortest path algorithm	31

2.3	Network Security devices and Tools.....	33
2.3.1	Firewall.....	33
2.3.2	Deep Packet Inspection.....	33
2.3.3	Intrusion Detection System.....	33
2.3.4	Intrusion Prevention System.....	34
2.3.5	Snort as IDS and IPS.....	34
2.3.6	Pfsense.....	34
CHAPTER 3: PROPOSED METHODOLOGY.....		36
3.1	Proposed Architecture.....	39
3.1.1	Server Farm.....	39
3.1.2	DMZ (Demilitarized Zone).....	39
CHAPTER 4: VNF Placement and Chaining, Experimental Evaluation.....		44
4.1 VNF Placement:.....		44
4.2	Environment Setup VNF Placement.....	54
4.3 Service Function Chaining.....		57
.....		61
CHAPTER 5: VNF Placement and Deployment: Results and Evaluation.....		62
5.1	SYSTEM OVERVIEW.....	64
5.2	PROPOSED IMPLEMENTATION.....	65
5.3	EXPERIMENTAL EVALUATION.....	67
CHAPTER 6: Conclusion and Future Work.....		70
6.1 Conclusion.....		70
6.2 Future Work.....		71
REFERENCES.....		73

List of Figures

Figure 1: Architectural framework of NFV	9
Figure 2: MANO Architecture	10
Figure 5: Network Architecture for an Enterprise Security	41
Figure 6: Architecture Nodes, Layers, Zones.....	42
Figure 7: Service Function Chains - Zones Nodes	43
Figure 8: SDN-NFV based deployment of firewall using commodity Hardware	62
Figure 9: Zodiac OpenFlow Switch.....	63
Figure 10: Zodiac Configuration.....	64
Figure 12: Pfsense Firewall Deployed on VM	68
Figure 11: SDN Controller ONOS.....	68
Figure 13: SDN-NFV based Firewall LAN Policy	69
Figure 14: SDN-NFV Based Firewall WAN Policy.....	69

List of Tables

Table 1: SDN - NFV based Architecture Core Component Resources Details.....	52
Table 2: NFV Nodes Cost Matrix	52

CHAPTER 1: INTRODUCTION

It is now more difficult than ever to manage IT infrastructures because of the proliferation of devices, data, and users. Businesses need a solution to handle this complexity because most IT expenditures are stagnant or declining, and many are now turning to artificial intelligence for assistance. [Juniper]. In the fast-growing field of Information technology (IT), networking plays important role to deliver data between two points. Networking, a process that involves connecting computers, media, and devices for networking, can be used to achieve data communications between remote parties. Local area networks (LANs) and wide area networks (WANs) are the two basic categories into which networks fall. Different traits and functionalities distinguish these two different kinds of networks. The LANs and WANs that make up the internet are connected via internetworking devices [1]. But, when both types come together to fulfill the requirements of an enterprise the following challenges occurs: 1- computational capacity, 2-delays, 3- bandwidth and 4- security. With the rapid evolution of IT, conventional methodologies for IT devices configuration and deployment has also evolved, but such techniques and methodologies along with providing required benefits introduces some challenges like: Cost, Security, Bandwidth, Computational Capacity Delay and Energy. To overcome such problems, we proposed a network model along Software Defined Network (SDN) and Network Function virtualization (NFV). Which assures that by adopting this advance network model the problems like Operational Cost and over utilization of resources (computational capacity, bandwidth delay and energy) can be solved. In our proposed work, first, we adopted SDN-NFV based platform, NFV is used to place light weight Network Security Functions (NSF), as from medium level enterprise to a large level enterprise, the advanced network security design almost remains same, taking that into consideration we designed a network which offers almost all the scenarios which an enterprise requires to fulfill its connectivity with DMZ, Server Farm and Internet. Starting from our network model which fulfills the basic to advance level functionality regarding routing switching and security. In our work, we opted solution based on SDN and NFV. Software-based controllers or application programming interfaces (APIs) are used in the SDN networking paradigm to control traffic on a network and communicate with the underlying hardware infrastructure. When compared to traditional networks, which control network traffic with specialized hardware (such switches and routers), this architecture is different. In addition to using conventional hardware,

SDN also makes use of software to create and manage virtual networks. While software-defined networking offers a new method of managing the routing of data packets through a centralized server, NFV enables organizations to segment different virtual networks within a single physical network or to connect devices on different physical networks to create a single virtual network. [1]. Because SDN has so many benefits over traditional networks, we chose to incorporate SDN into our design to address the location and resource utilization issues. SDN provides a significant advancement over conventional networking since it makes it possible for: increased flexibility and quickness with improved control: Developers can simply program an open standard software-based controller to govern the flow of traffic over a network instead of manually programming numerous vendor-specific hardware devices. Since they can select a single protocol to communicate with any number of hardware devices through a central controller, networking managers also have more freedom when selecting networking equipment. Robust security: A software-defined network provides visibility across the whole network, giving security risks a more comprehensive perspective. With the increase of internet-connected smart devices, SDN offers a number of advantages over conventional networking. To prevent hacked devices from infecting the rest of the network, enterprise network administrators can immediately quarantine infected devices [4] or create separate zones for devices that need varying levels of security. The next step is Network functions virtualization (NFV), which involves replacing the hardware of network appliances with virtual machines. A hypervisor is used by the virtual machines to operate networking applications and procedures like load balancing and routing. Load balancing, intrusion detection, and firewalls are just a few examples of the hardware-dependent functions that NFV transforms into VNFs that can be quickly deployed and scaled. As a result, specialized hardware is no longer required for network tasks. NFV allows for flexible resource sharing, rapid service creation, and deployment. With the use of NFV, communication services may be separated from specialized hardware like routers and firewalls. Because of this division, network operations can offer new services on demand and without buying new hardware. With network functions virtualization, network components can be deployed in a matter of hours as opposed to months as with conventional networking. The virtualized services can also run on less expensive generic servers as opposed to pricey proprietary hardware. Other justifications for utilizing network function virtualization include: Pay as you go: Pay-as-you-go NFV models can lower costs as they only pay for what they require; 2- Fewer appliances are required, which lowers operating expenses

because NFV works on virtual computers rather than actual machines; 3. Scalability: Using virtual machines to scale the network architecture is quicker, simpler, and doesn't involve the purchase of extra hardware. The functionality offered by discrete hardware networking components is essentially replaced by network functions virtualization. This means that software that performs the same networking tasks as the conventional hardware is executed on virtual machines. Instead of using hardware components, software handles tasks including load balancing, routing, and firewall security. Network engineers can program each of the various components of the virtual network, and even automate the provisioning of the network, using a hypervisor or software-defined networking controller. Through a single pane of glass, IT managers may quickly customize a variety of network functioning features. Many service providers believe that the advantages of virtualizing network functions outweigh the hazards. Traditional hardware-based networks require network administrators to buy specialized hardware units, manually configure them, then join them to form a network. This takes a lot of time and requires advanced networking knowledge. NFV makes it possible for virtual network functions to be operated on a generic standard server under the management of a hypervisor, which is significantly less expensive than buying specialized hardware. A virtualized network makes it much easier to configure and administer the network. The best part is that because the network is operated on virtual machines, which are simple to provide and administer, network capabilities may be updated or added instantly. These VNFs are merged into Service Function Chains (SFC) in the actual network to fulfil certain requirements. SFC provides on-demand traffic forwarding and filtering and defines these VNFs in a certain order. In order to meet users' service expectations in a changing network when user demands and traffic change dynamically, VNF scaling can dynamically allocate or reclaim related resources for SFC, increasing network flexibility. Scaling-out can install new VNFs to expand the resources allotted to the network when its capacity is exceeded. Scaling-in lowers the resources allotted to the network when resource utilization is low by eliminating extra VNFs. The majority of the related research in this area concentrates on the efficient deployment of VNFs when building SFC and primarily takes user requests and physical machine capacity into account. The issue with SDN-NFV based NSF deployment, available resources, and transmission delay brought on by the link between physical machines (PMs) where VNFs are deployed, however, is disregarded. As a result, when deploying VNF, NSF deployment, compute network resource utilization, and transmission delay must all be taken into account. Consequently, this essay makes two contributions: 1) Using

commodity hardware, we offer SDN/NFV Enabled Security for an Enterprise Network, which is essentially NFV placement. 2) We suggest an effective scaling-out VNF placement and chaining algorithm based on the suggested scaling management mechanism, which takes into account the resource utilization of PMs and the transmission delay of SFC, and then optimizes the physical machine resource utilization and the transmission delay. Following is the rest of this thesis. The second section reviews the literature and the current state of research on the placement of VNFs and service function chaining for scaling. A few service function chaining algorithms, network security tools, and deployment strategies are also covered in this section. The third portion then introduces the suggested methodology, in which the chosen methodology's architecture and brief are provided. The fourth describes the hardware, controllers, configurations, and deployment of VSF utilizing SDN-VNF platform and discusses placement of VSF. The effectiveness of the proposed service function chaining algorithms is presented in the fifth section. The sixth section presents this work conclusion and its future directions. The cost of deployment, network operating effectiveness, and scalability are all constrained by numerous characteristics of older devices. Some of the restrictions include: High costs of operations, Complex Configurations, Time Taking Deployments, Flexibility limitations, Issues in manageability of the device, Time to market challenges, Migration considerations and 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 server 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. 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. 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 be 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: Performance up to 100Gbps, Reliability of about 99.999%, Scalability to millions of the users, Low-latency delivery of real-time applications. 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].

1.1 Network Function Virtualization

The implementation of next-generation communication networks is anticipated to take place on virtualized infrastructures, in which network services are deployed on virtual machines rather than on the currently used proprietary hardware. The dominant option and direction in current and future communication and computing infrastructures appear to be moving away from an

architecture based on a multitude of black boxes that are outfitted with specialized network hardware and pre-loaded with specialized software to a new architecture consisting of a "white box" running a multitude of specialized network software. This is also necessary to support the 5G vision, which calls for a new network design that instructs adaptable, dynamically adjustable network components to deliver on-demand customized services to traffic demands that may be fluctuating in time and place, all while enabling heterogeneity and diversity. Software defined networking (SDN) and network function virtualization (NFV) have been shown to be two promising technologies for controlling future networks in order to realize such a technological paradigm change. A new paradigm called Network Function Virtualization (NFV) uses cloud infrastructures to enable telecommunication services. The service functions, which are also known as Virtual Network Functions (VNF), are carried out by Virtual Machines that are housed in data centers that are spread out geographically. Without having to increase their capital investments for specialized hardware device acquisition and middleboxes installation, service providers can now offer highly specialized network services customized to the needs of end users. The advantages of NFV are numerous and include decreasing capital and operating costs as well as the time it takes for new services to reach the market. As numerous market and technological reasons combine, SDN has become a significant force for innovation and change in networking. The development of cloud applications and services across business and cloud providers, the emphasis on converged infrastructures (compute/storage/network), and software-defined datacenters are a few examples of such reasons.

1.1.1 Architectural framework of NFV:

In Figure 1, the European Telecommunication Standardization Institute (ETSI)'s (the organization that developed the NFV architecture) architectural framework is shown Network Function Virtualization (NFV) is a network architecture approach that aims to transform traditional network infrastructures by decoupling network functions from dedicated hardware appliances and implementing them in software. NFV provides a flexible and scalable framework that enables network services to be virtualized, instantiated, and deployed on standard servers, storage, and switches.

Key Concepts of NFV:

Virtualized Network Functions (VNFs): VNFs are the building blocks of NFV. They represent network functions such as firewalls, routers, load balancers, and intrusion detection systems, which are traditionally implemented as dedicated hardware appliances. In NFV, these network functions are virtualized and implemented as software instances that can run on general-purpose servers or cloud platforms.

NFV Infrastructure (NFVI): NFVI consists of the underlying hardware and software resources that support the execution of VNFs. It includes servers, storage devices, switches, and hypervisors or container platforms that provide the necessary virtualization capabilities. NFVI provides the infrastructure for hosting and managing VNF instances.

NFV Orchestrator: The NFV Orchestrator is responsible for managing the lifecycle of VNFs. It handles tasks such as VNF instantiation, scaling, chaining, and termination. The NFV Orchestrator interacts with the NFVI to provision resources and ensure optimal utilization and performance of VNFs.

Benefits of NFV:

Agility and Flexibility: NFV enables network services to be dynamically provisioned, scaled, and orchestrated, allowing for greater agility and flexibility in deploying and managing network functions. It enables service providers to quickly introduce new services, adapt to changing demands, and scale resources on demand.

Cost Efficiency: By virtualizing network functions and leveraging standard hardware infrastructure, NFV can significantly reduce capital and operational expenses. It eliminates the need for expensive dedicated hardware appliances, reduces power consumption, and enables efficient resource utilization through dynamic allocation and scaling.

Scalability and Elasticity: NFV enables the rapid scaling of network functions to accommodate changing traffic patterns and user demands. It allows service providers to scale resources up or down based on real-time requirements, improving service quality and optimizing resource utilization.

Service Innovation and Time-to-Market: NFV facilitates service innovation by providing a flexible and programmable platform for developing and deploying new network services. It reduces the time-to-market for introducing new services, enabling service providers to respond quickly to customer needs and market trends.

Network Resilience and High Availability: NFV offers enhanced network resilience and high availability through features like redundancy, automated failover, and rapid service restoration. It enables the migration of VNFs to different physical resources in case of failures, ensuring uninterrupted service delivery.

Challenges and Considerations:

Performance: Virtualizing network functions introduces performance considerations, as software-based implementations may not achieve the same level of performance as dedicated hardware appliances. Performance optimization techniques, such as hardware acceleration, need to be considered to address these challenges.

Security: NFV introduces new security concerns, as the virtualized network functions are vulnerable to attacks. Proper security measures, including isolation techniques, secure management interfaces, and traffic encryption, must be implemented to ensure the integrity and confidentiality of the virtualized infrastructure. Interoperability and Standards: Ensuring interoperability between different VNFs, NFV infrastructures, and management systems is crucial for a successful NFV deployment. Standardization efforts by organizations like ETSI (European Telecommunications Standards Institute) and open-source projects like OPNFV (Open Platform for NFV) aim to address these challenges. Management and Orchestration: Effective management

and orchestration of VNFs and NFVI resources are essential for achieving the full benefits of NFV.

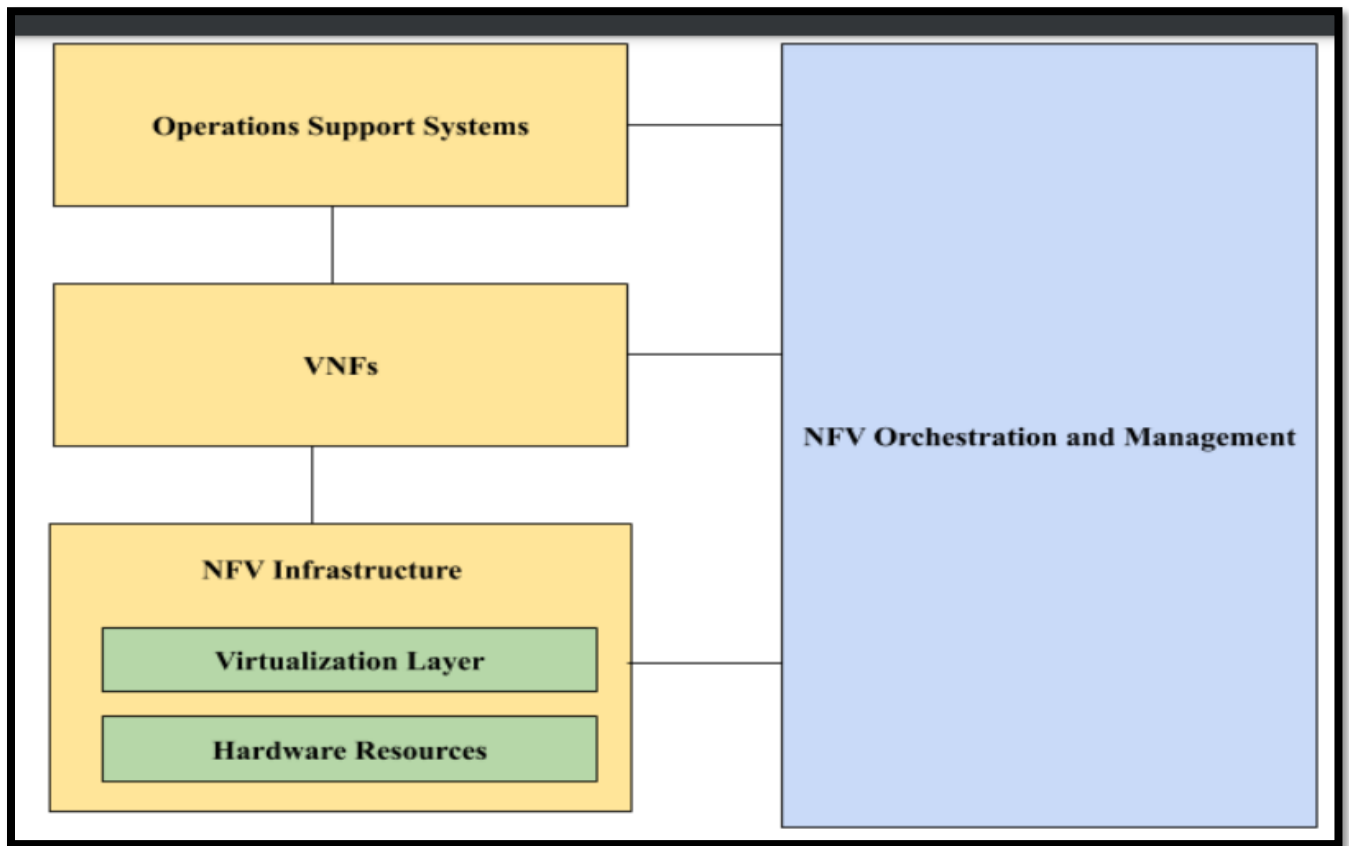


Figure 1: Architectural framework of NFV

1.1.2 Virtualization Network Function (VNF) Layer

It has two subsections: Element Management System (EMS) and VNF Operation Support Subsystem (OSS). The fundamental building component of an NFV architecture is a VNF. Network operations were virtualized. For instance, when a router is virtualized, we refer to it as a router VNF. Similarly, when a base station is virtualized, we refer to it as a base station VNF. VNF stands for Virtual Network Function, and it refers to any virtualized network function. VNF is functionally managed through the Element Management System (EMS). The management tasks include management of faults, configurations, accounting, performance, and security. Through exclusive interfaces, an EMS may control the VNFs. One EMS may be able to manage several VNFs or one EMS may exist for each VNF. EMS can be implemented as a VNF.

1.1.3 NFV Infrastructure (NFVI) Layer

The totality of the hardware and software elements that go into creating the environment in which VNFs are set up, maintained, and used is known as the NFV Infrastructure. The network connects these locations so they can be a part of the NFV infrastructure, even when it physically spans over multiple places. NFV Infrastructure includes following

- Hardware Resources
- Virtualization Layer
- Virtual Resources

1.1.4 Management and Orchestration (MANO) Layer

NFV MANO architecture

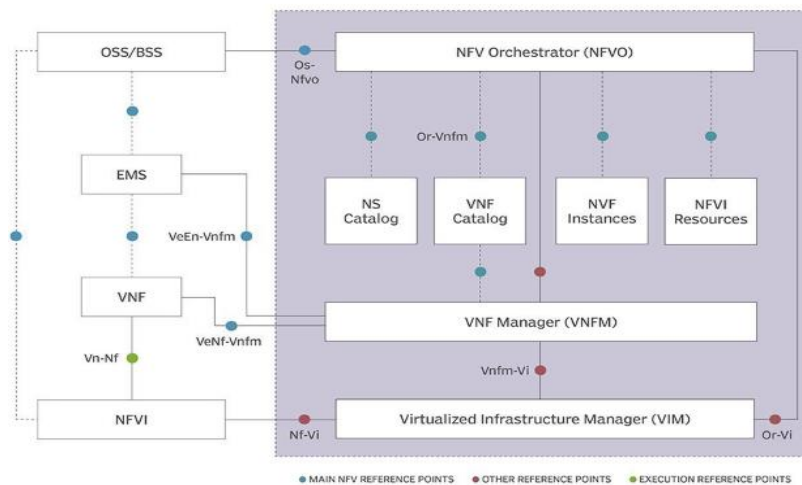


Figure 2: MANO Architecture

MANO, which stands for Management and Orchestration, is a critical component of network function virtualization (NFV) architectures. It provides a framework for managing and orchestrating the lifecycle of virtualized network services, including the deployment, scaling, and monitoring of virtual network functions (VNFs). The primary goal of MANO is to enable the efficient operation and management of virtualized networks, ensuring the delivery of network services with high reliability, scalability, and performance. MANO consists of three key functional blocks:

NFV Orchestrator (NFVO): The NFVO is responsible for the overall orchestration of the virtualized infrastructure. It handles the coordination and management of resources, VNF lifecycle management, and service provisioning. The NFVO interacts with the Virtualized Infrastructure Manager (VIM) to allocate and manage the underlying physical and virtual resources.

Virtualized Infrastructure Manager (VIM): The VIM is responsible for managing the virtualized infrastructure, including the compute, storage, and networking resources. It provides APIs that allow the NFVO to interact with the infrastructure, enabling resource allocation, monitoring, and control.

Virtual Network Function Manager (VNFM): The VNFM is responsible for managing the lifecycle of VNFs. It handles tasks such as VNF instantiation, scaling, termination, and monitoring. The VNFM interacts with the NFVO to ensure proper orchestration of VNFs within the virtualized infrastructure. MANO provides a standardized and interoperable approach to managing and orchestrating NFV environments. It allows service providers to automate and streamline the deployment and management of network services, leading to improved agility, cost efficiency, and scalability. By leveraging MANO, operators can effectively deliver a wide range of network services, including virtual firewalls, load balancers, routers, and more. Overall, MANO plays a vital role in the successful implementation and operation of NFV architectures. It empowers service providers to harness the benefits of virtualization and software-defined networking, enabling them to deliver innovative and scalable network services in a dynamic and rapidly evolving landscape.

1.2 Software Defined Network

Service providers are always changing in order to suit the varying and erratic service needs of the end users. This has an impact on both the services being offered and the network that facilitates the delivery of those services. Transport infrastructure is a significant component of it. The task of forwarding aggregated traffic requests from many end users consuming diverse services across several cities, regions, or continents falls on transport networks. The features and traffic demand of traditional services (such Internet access or VPNs) were created and designed with traditional transport network designs, which historically were assumed to be predictable in terms of origin and destination as well as their pattern. The networks of traditional carriers are regarded as being highly complex and unable to adapt to shifting traffic demands. In the past, network operation has

been done either directly via manual procedures with direct access to the nodes or by using Network Management Systems (NMSs), where available, as a part of Operation Support Systems (OSSs). Multiple manual setup operations are needed in different network nodes for a straightforward capacity boost on a path, which prevents network operators from optimizing the available resources and usually pushes them to over dimension the network. Every year, hundreds of thousands of unique node configurations are created because a mid-sized network may need a few thousand nodes for each of its numerous tiers (such as interconnection, backbone, distribution, aggregation, etc.). The absence of automation in these processes not only necessitates lengthy periods of planning and execution (e.g., careful planning of configuration templates, scheduling of maintenance windows, etc.), but also leaves room for error, slowing the rollout of new services and capacity expansions. In a network operator, the choices have typically been repeated in each technology silo. These business practices are no longer viable or sustainable, especially in light of the widespread adoption of cloud and virtualization technology. When the 5G generation of mobile communications, which features new concepts like network slicing for adapting to certain service characteristics, begins to be deployed, this will become even more clear. The available control and management capabilities have a significant impact on network operations. Then changes must be made to both the OSS and the network's physical structure. By lowering the number of layers and consolidating network operations into fewer nodes, the network transformation should result in a simpler network architecture, cheaper network upgrade costs, and fewer nodes that need to be operated in the network. The OSS transformation should provide a better level of automation with a uniform set of components. To fully realize the potential of such advances, the idea of Software Defined Networking (SDN) becomes the main point of network evolution, addressing the bulk of current gaps. By using SDN principles, network administration should be simplified while providing quick responses and adaptability to network changes caused by shifting traffic patterns or just by service configuration. SDN is being examined as a technique to support management duties, such as the gathering of real-time data that would enable the automatic building and activation of configurations in network elements, as prompted by the OSSs, in addition to network element control capabilities. A novel approach to network programmability, or the capacity to dynamically initialize, control, alter, and regulate network behavior via open interfaces, is known as "software-defined networking" (SDN). By separating the data forwarding plane from the control plane and introducing an abstraction for it, SDN highlights the importance of software in managing

networks. The division between the system (control plane) and the sending capacity (data plane) is referred to as SDN. Before transmitting data to a network device, the task of the rules is formed in the controller while the controllers direct the logic controlling network behavior. SDN is a more contemporary network that may be able to overcome the drawbacks of older conventional networks that have been in use for many years.

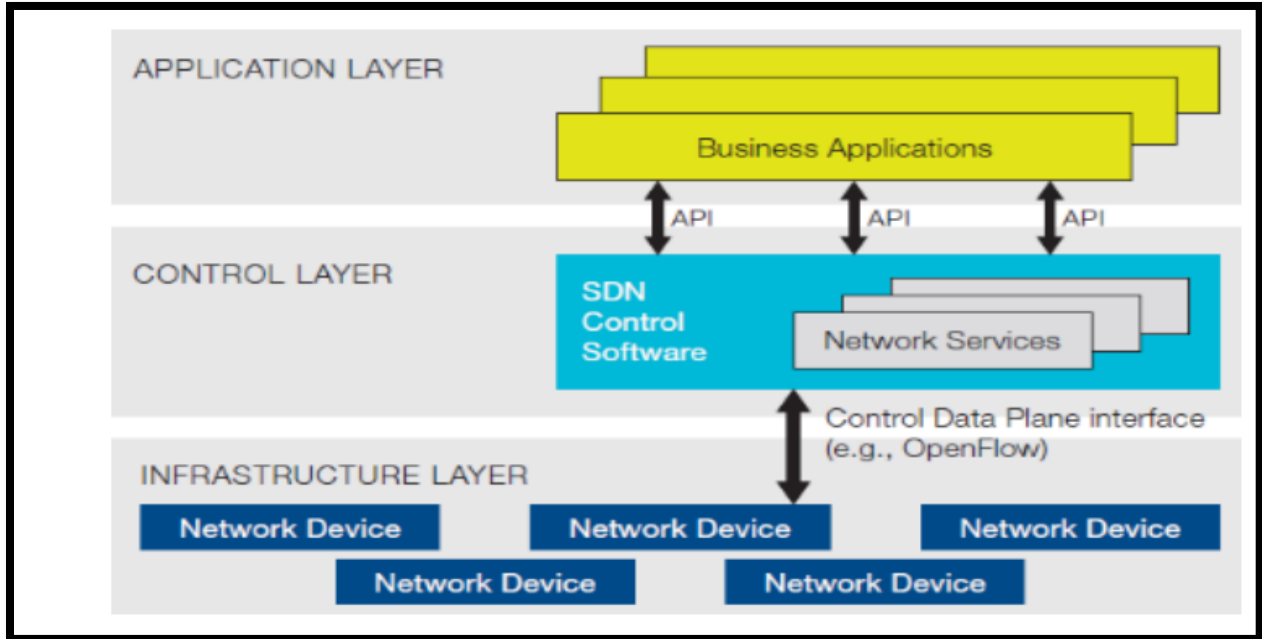


Figure 3: Basic SDN architecture

The logical representation of SDN is shown in Fig. 2. By transferring the control plane from switches to a logically centralized controller, Software Defined Networks (SDNs) make network management and monitoring easier. The controller(s) and switches, which stand in for the control and data planes, respectively, are the two key elements of an SDN architecture.

1.2.1 CONTROL PLANE

The control plane, a component of the network architecture, decides on traffic routing and network topology. Data packet forwarding, or how data is delivered from one location to another, is controlled by the control plane, a component of a network. The creation of a routing table, for instance, is regarded as occurring within the control plane. Network paths are identified by routers using a variety of protocols, and these paths are then stored in routing tables.

1.2.2 DATA PLANE

SDN relies on the idea of centralization to ensure that all network control choices are made in a centralized location and in a centralized manner. SDN decouples the control plane and data from the conventional routing devices to move it into a centralized place and give centralized control. The routing devices are still a part of the data plane, but they are now only used as forwarding devices, with the controller acting as the key decision-making entity. The controller essentially consists of an x86 server that is connected to the forwarding devices and interacts with them to make all control decisions, including routing. The foundation of NFV is the virtualization of network functions into software that runs on a powerful server. This type of virtualization makes network setup and migration simple. By following the above definition, the following salient features of an SDN network can be deduced:

1. Programmable Network
2. Centralized Management
3. Dynamically Configurable Networks
4. Open Protocols and Interfaces
5. Easily Monitored and Fault Tolerant Network

1.2.3 SDN CONTROLLER

The core component of the SDN control plane is controller. In SDN networks, the task of creating rules and uploading them to routing devices is loaded on the controller. Based on existing rules in an OpenFlow-enabled switch, hijacked packets are buffered or dropped by the controller until the appropriate rules are loaded for them. One of the significant roles of a controller is to control traffic on core network elements by means of a set of directions. SDN has a wide diversity of controllers. The controller communicates with switches through northbound interfaces and network applications through southbound interfaces, for example, OpenDayLight (ODL), Ryu. Southbound protocols are used in the SDN sector in a variety of ways, from relatively simple ones like Simple Network Management Protocol (SNMP) to more complex ones like OpenFlow, Open v Switch Database Management Protocol (OVSDB), and NETCONF. Five SDN controllers are categorized based on a few categories that are shown in Table 1.

a:POX/NOX: POX one of the open source controllers that has facilitated the development of SDN applications. The OF protocol, which is used as the de facto communication protocol between

switches and controllers, is where this controller provides a practical way to put it to use. A greater degree of flexibility is attained by using the POX controller to operate various applications, including load balancers, switches, firewalls, and hubs. This controller, which evolved from the NOX controller, is a Python-based SDN controller. The centralized NOX controller, on the other hand, was created in C++. It should be noted that the NOX, which was unveiled in 2008, is the first controller to have an open-source functionality. This controller, however, is added on top of the server. Meanwhile, the selective server is crucial in preserving the most crucial information for the network's overall view. A few switches and one or more network-attached servers make up a NOX-based network's core elements. These particular servers host the NOX controller software, which controls the management programmers that use NOX. Typically, a single NOX software is installed on each server in the network, and the NOX controller software can be thought of as including multiple processes that are carried out by various controllers as well as a single network view that is stored in any server in the network's database. The whole set of network observation findings from NOX are available in the network view, and all apps use this state to make management decisions. To get around this restriction, a suitable switch offering the OF abstraction must be used. The degradation of network switch behavior is ascribed to using the NOX to regulate network traffic.

b: TREMA It is a framework for C and Ruby that creates a software environment for OF protocol developers. The simplest way to use open-source software for nothing.

c: RYU It is an SDN controller built using components. Ryu consists of a variety of distinct built-in parts. Where these elements can be added to, combined with, and replaced to create new, tailored applications for contemporary controller. Thus, any programming language can be used to design a new developed component.

1.2.3.1 ONOS SDN Controller

ONOS (Open Network Operating System) is an open-source SDN controller designed to provide a scalable and high-performance platform for managing and controlling software-defined networks. Developed by the ONF (Open Networking Foundation), ONOS is built on a modular architecture and offers a range of features and capabilities for network management and orchestration.

Key Features of ONOS SDN Controller:

Centralized Network Control: ONOS acts as a centralized controller that manages and controls network devices in an SDN environment. It provides a unified view of the network and enables administrators to define and enforce network policies.

Scalability and Performance: ONOS is designed to scale horizontally, allowing for the addition of new controller instances to handle growing network sizes and traffic loads. It employs distributed and parallel processing techniques to achieve high-performance operation even in large-scale deployments.

Network Abstraction: ONOS provides a network abstraction layer that allows developers and applications to interact with the network using high-level abstractions rather than dealing with low-level network details. This abstraction simplifies network programming and enables the development of network applications and services.

Northbound and Southbound Interfaces: ONOS supports northbound and southbound interfaces for communication with external applications and network devices, respectively. The northbound interface exposes APIs for network applications to interact with the controller and access network information, while the southbound interface communicates with network devices through protocols like OpenFlow.

Modular Architecture: ONOS follows a modular architecture, where functionality is divided into different modules or components. This modular design allows for easy extensibility, customization, and integration of additional features and services.

Fault Tolerance and High Availability: ONOS provides mechanisms for fault tolerance and high availability to ensure continuous network operation. It incorporates redundancy and failover mechanisms, allowing multiple controller instances to handle control plane tasks and seamlessly recover from failures.

Application Ecosystem: ONOS has a vibrant and growing ecosystem of network applications and services. It supports the development and deployment of custom applications to meet specific network requirements, enabling innovation and customization.

Community Support: ONOS benefits from an active open-source community that contributes to its development, enhancement, and maintenance. The community provides support, documentation, and resources to help users and developers leverage ONOS effectively.

Use Cases of ONOS:

Network Monitoring and Analytics: ONOS can collect real-time network data, perform analytics, and provide insights into network performance, traffic patterns, and resource utilization. It enables the development of network monitoring and analytics applications for troubleshooting, capacity planning, and optimizing network operations.

Traffic Engineering and QoS: ONOS can dynamically manage network paths, optimize traffic flows, and enforce Quality of Service (QoS) policies. It enables the development of traffic engineering applications that improve network efficiency, reduce congestion, and prioritize traffic based on service-level agreements.

Virtualized Networks: ONOS integrates with virtualization platforms like OpenStack and supports the orchestration of virtualized network functions. It enables the creation, management, and chaining of virtual network functions (VNFs) to deliver network services in a virtualized infrastructure.

Campus and Data Centre Networks: ONOS can be used to manage and control networks in campus environments and data centres. It provides centralized control, automation, and policy-based management, facilitating efficient network operations and enabling network programmability.

Multi-Domain and Carrier Networks: ONOS supports multi-domain and carrier networks, where multiple network operators or domains need to interconnect and coordinate. It enables the creation of multi-domain network orchestrators and facilitates dynamic resource allocation, service provisioning, and inter-domain communication.

ONOS has gained adoption and deployment in various industry segments, including telecommunications, cloud service providers, academic research, and large-scale network deployments.

Service Provider networks are the primary emphasis of ONOS's distributed, scalable, and stable design. The transition "brown field" networks to SDN "green field" networks is only supported by the Open Network Operating System SDN controller platform.

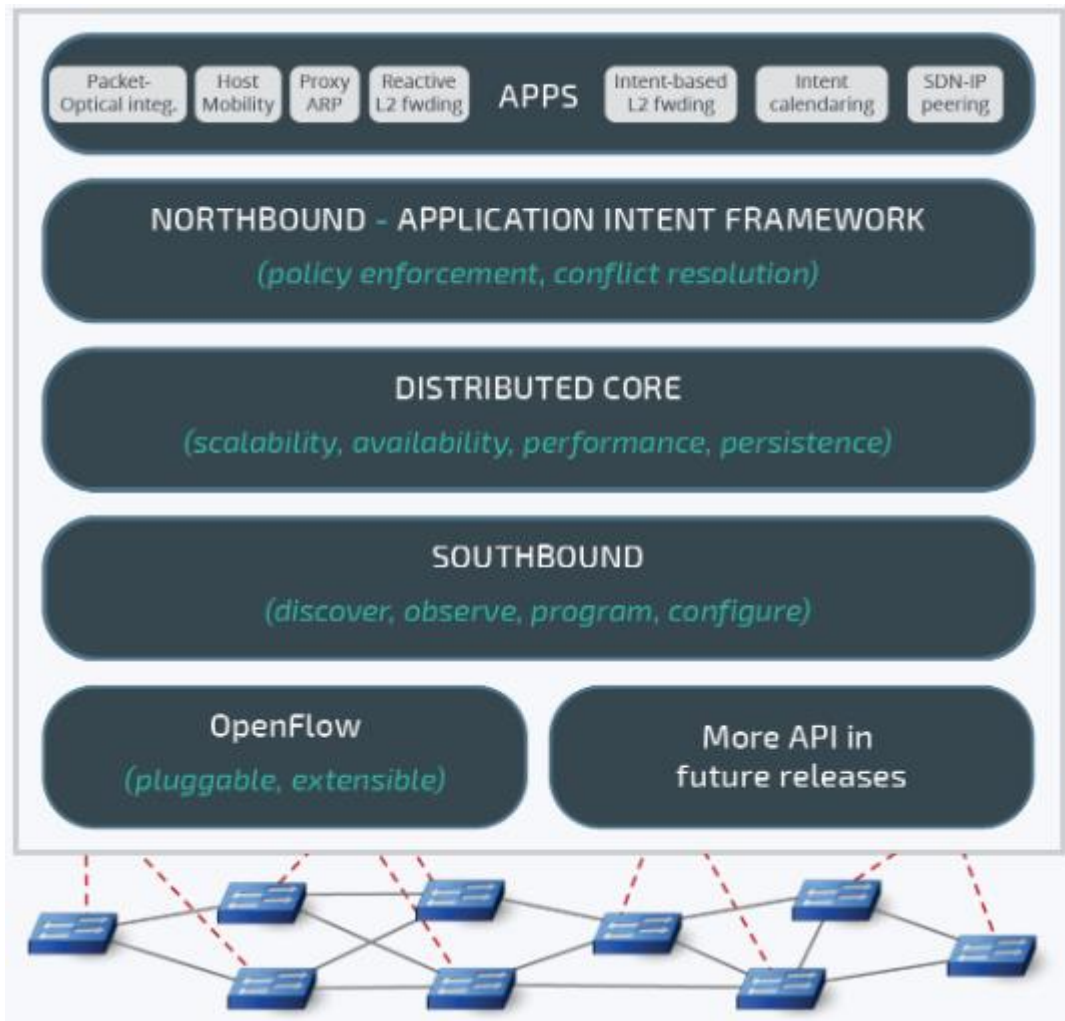


Figure 4: ONOS Architecture

This opens up innovative new capabilities for network operators as well as disruptive deployment and operational cost points.

Tier 1 is made up of protocol-related modules that interact with network devices (shown as Southbound in the figure); Tier 2 is the ONOS core that offers network status independent of any particular protocol; and Tier 3 is made up of ONOS apps that make use of Tier 2's network state data.

1.3 Scalability

In order to achieve performance and geo-redundancy across small regions, ONOS is particularly made to scale horizontally.

Scalability of clusters:

Additional controllers can dynamically join and leave the cluster with ease, providing flexibility over time. The cluster configuration is simple.

The Atomic distributed datastore should reduce cluster partitioning-related disruptions since all servers are guaranteed to have the right data. However, communication and coordination activities quickly increase as a cluster grows, limiting the performance benefits per new cluster member.

Scalability of architecture

- ONOS has built-in BGP routing features that help manage traffic between SDN islands.
- There are multiple instances of ONOS being effectively used in a geo-redundant design to handle huge SD-WANs, including ICONA and SDN-IP.

1.4 Telemetry

Telemetry streams are accessible using pluggable software modules; the most recent release includes plug-ins for Influx DB and Grafana.

1.5 Resilience and Fault Tolerance

Cluster administration in ONOS is fairly straightforward. The remarkably simple cluster administration system of ONOS includes native commands for adding and removing members. Fault tolerance is offered by the Open Network Operating System in systems with an odd number of SDN controllers. In the event of a failing master node, a substitute leader would be selected to take over the network.

1.6 How ONOS works?

ONOS (Open Network Operating System) works as a centralized controller for managing and controlling software-defined networks (SDNs). It follows a distributed architecture and employs various components and protocols to handle network management tasks. Here's an overview of how ONOS works:

Controller Cluster: ONOS can be deployed in a clustered environment, where multiple controller instances collaborate to manage the network. The controller instances form a cluster and communicate with each other to share network information and synchronize their states. This distributed architecture enables scalability, fault tolerance, and high availability.

Southbound Interface: ONOS communicates with network devices (switches, routers, etc.) using southbound protocols such as OpenFlow, NETCONF, and P4Runtime. These protocols establish a communication channel between ONOS and the network devices, allowing the controller to gather network information, configure devices, and control their behaviour.

Network Abstraction: ONOS provides a network abstraction layer that abstracts the underlying network infrastructure, making it easier for applications and services to interact with the network. The abstraction layer presents a simplified view of the network, hiding low-level details and providing high-level abstractions and APIs for network management and control.

Northbound Interface: ONOS exposes a northbound interface that allows applications and network services to interact with the controller. The northbound interface provides APIs and protocols that enable applications to request network information, configure network behaviour, and receive notifications about network events.

Application Framework: ONOS offers an application framework that enables the development and deployment of custom network applications and services. Developers can build applications using the provided APIs, allowing them to control and manage the network based on specific requirements and use cases.

Network State and Flow Management: ONOS maintains a global view of the network state, including information about devices, links, and network flows. It collects data from network devices, processes it, and stores it in a distributed database. This global network view enables network-wide control and optimization.

Event-Driven Architecture: ONOS follows an event-driven architecture, where events such as network topology changes, device failures, or traffic events trigger actions and notifications within the controller. Applications can subscribe to these events and take appropriate actions based on the received notifications.

Distributed Core Services: ONOS provides core services that are distributed across the controller cluster. These services include topology management, device discovery, flow management, traffic

engineering, and fault detection. The distribution of these services across the cluster ensures scalability, fault tolerance, and load balancing.

Lifecycle Management: ONOS manages the lifecycle of network functions and services. It handles VNF (Virtual Network Function) instantiation, scaling, chaining, and termination in virtualized network environments. It also provides mechanisms for updating and upgrading network functions while maintaining uninterrupted service.

Community Contributions: ONOS benefits from an active open-source community that contributes to its development, enhancement, and maintenance. The community provides bug fixes, new features, and performance improvements, ensuring the continuous evolution and stability of the platform.

By leveraging its distributed architecture, network abstraction, and application framework, ONOS enables centralized control and management of software-defined networks, providing a scalable, flexible, and programmable network operating system.

1.7 OpenFlow Protocol

Software used in SDN architecture is called the OpenFlow (OF) Protocol. It describes the communication between a network device or agent and an SDN controller. The Open Networking Foundation (ONF) presently oversees OpenFlow, which was created by Stanford University in 2008. An SDN controller and a switch communicate with each other via the southbound protocol known as OpenFlow.

The application sends information to the SDN controller, which transforms it into flow entries. That are subsequently sent via OF to the switch. Between the controller and switch, there must be IP connectivity in order to establish an OF connectivity.

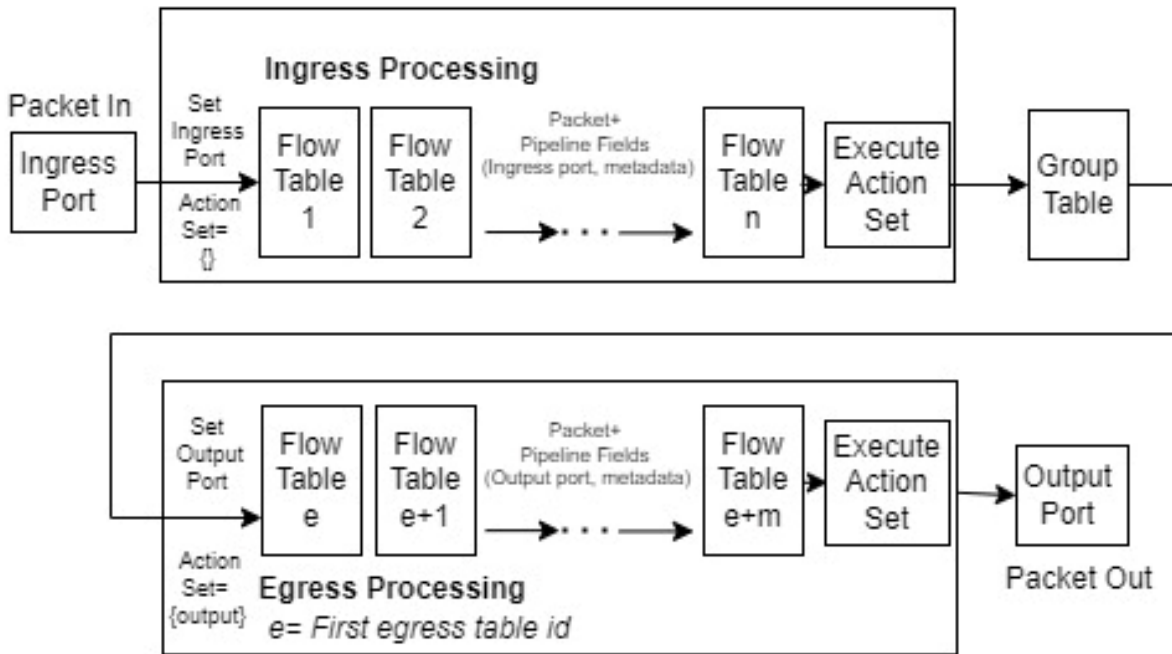


Figure 5: OpenFlow Protocol General Working

Following a successful TCP 3-Way Handshake, OF creates a channel. OpenFlow tables and flow entries are maintained by this protocol. As in figure 5, Through flow tables, packets are handled after entering through the ingress port. As soon as the operation is finished by the stated execution action, the group table acquires data and starts the process. Before the data is sent to the egress output port, the egress tables perform processing on it. Similar to the MAC/CAM table on a traditional switch, flow tables are used to maintain the hardware addresses of the hosts. The SDN switch has to have a lot of knowledge about a packet when it enters an incoming port. Before choosing the best-matching flow entry from the database and carrying out the related action, the switch will examine a variety of information, including IP address, port number, MAC address, and VLAN ID. The packet may be flooded, dropped, forwarded through a different port, or routed to the controller for additional analysis. If a switch does not have an entry for a specific packet, it may have a default entry or "TABLE MISS" entry. The packet could be dropped or sent to the controller depending on which entry has the lowest priority. When a controller receives a packet

of this kind from a switch, it will transmit it to an application that is executing at the application layer. The programme analyses the packet and notifies the controller if a new flow entry has to be added to the switch's flow table. If so, the controller will modify the switch to include a flow entry. The switch will handle it appropriately and execute the proper action the following time because the second packet of the same kind already has an entry. The network is managed significantly more successfully as a result. OpenFlow and SDN are superior than conventional networks in a variety of ways.

- OpenFlow provides a straightforward mechanism of communication between the controller and switch in an existing network.
- Although the bulk of current devices do not have OpenFlow enabled by default, we can quickly do so and leverage it to make the transition to SDN.
- To prevent DoS attacks and spying on the controller and/or network, it provides security via a TLS connection.
- OpenFlow only modifies the flow tables, which specify a packet's path, switch's configuration is not altered by it.

OpenFlow-based networks have specific capabilities. For instance, a single controller can operate several switches. Software can also be used to analyse traffic statistics. Furthermore, different traffic types can be divided into flows that can be managed, and forwarding information can be dynamically updated. Using these abilities, the research community has experimented with novel concepts and put forth fresh applications. The areas where OpenFlow has been studied the most include network management, security, availability, network and data centre virtualization, and wireless applications. They have been used in a variety of settings, including networks made up of real or virtual hardware and simulations. Research has also concentrated on assessing the functionality of OpenFlow networks and formulating strategies for enhancing their functionality [17].

The OpenFlow protocol specification is available in many iterations. The initial version was OpenFlow 0.2.0, which was made available in March 2008. Next, in May 2008, came versions 0.8.0 and 0.8.1. October 2008 saw the release of version 0.8.2. The most extensively used version of OpenFlow, version 1.0, was released in December 2009.

1.8 Security

The groundbreaking idea of software defined networking (SDN), which is used in the next-generation trend, is growing in step with other network technologies. As a result of the separation of the control and data planes in SDN, unique network characteristics like centralized flow management and network programmability have been made possible. In order to improve crucial network deployment qualities including flexibility, scalability, network-wide visibility, and cost-effectiveness, these aspects encourage the introduction of new and better network functions. Although SDN is showing signs of rapid evolution and is shaping this technology as a key enabler for future implementations in heterogeneous network scenarios, including datacenters, ISPs, corporate, academic, and home settings, it is still far from being regarded as reliable and secure, which prevents its agile adoption. The scientific community has recently been drawn to look into SDN security in an effort to close the adoption gap for SDN.

1.9 Problem Statement

The biggest difficulties in employing SDN/NFV- Based security for an enterprise is placement and chaining of NFV are: well architected network design, NFV deployment, and resource management. To overcome the said challenges, this work presents well architected and optimized NFV placement and Chaining solution of security function.

1.10 Placement, assignment and chaining of NFV

Network function virtualization [5], is an architecture that offers a new approach to develop, manage, and distribute the various networking services. Decoupling various NFs (network functions) from various hardware appliances is a straightforward process. These separated network services can operate on either general-purpose hardware or specialized cloud infrastructure. For various services running on hardware, we must implement various service regulations, such as traffic monitoring, routing, and bandwidth utilization. The major goal of network function virtualization chaining is to replace the conventional network and other devices with virtualized network functions as managing service policies and security settings for services operating on the same parameters became more difficult with time. The main reason for creating such a feature was to cut back on the money and effort spent on other physical hardware devices.

In order to ensure that system streams are handled properly, network function and chaining placement involves linking numerous system components (such as firewalls, load balancers, and other components) throughout the system. These streams must navigate a specific set of capacities from beginning to end. This problem can generally be broken down into three stages: 1) Placement 2) Assignment 3) Chaining

1.10.1 Placement in NFV

In order to efficiently fulfil the service needs, we select how many network functions should be put. Additionally, this placement calls for the appropriate locations for these functions, and new NF instantiations may even be necessary. An example of how to reduce latency and improve network performance is to have different locations host a firewall service. Typically, network functions are best placed at the network's points of presence (N-PoPs) [7] to control traffic flows and aid in creating the ideal setting for continuously operating services on infrastructure.

1.10.2 Assignment in NFV

During the assignment phase, network functions (NFs) are assigned to particular network activities, such as load balancing and firewalls. It adjusts the network functions in accordance with the network traffic and service requirements and assigns the necessary network services from the NF. Installing several network functions (NFs) of firewall at the network's points of presence (N-PoPs) may be more efficient for correctly allocating traffic and monitoring it in accordance with established traffic rules and regulations.

1.10.3 Chaining in NFV

Chaining is the third phase, during which we build process and service flows using interconnected virtualized network functions (NFs) over the network. If we run multiple IoT services concurrently on the same network, the traffic would likely use the same path and NFs, which may cause latency and congestion problems. However, with the aid of chaining, we are able to create various network paths in accordance with their needs, which allows us to solve the problem where one service needs user authentication while another is retrieving data from a server. In this case, we can select various data paths and network functions in accordance with their requirements. Chaining is most effectively used when traffic engineering is flow-based. Since the 1980s, NFV has been a research topic. However, NFV has been more practical since the adoption of software defined networks

(SDN) [8], which separate the control and data planes for better data engineering. Thus, this facilitates the establishment of virtualized network functions NFs in accordance with the services required and makes it simple to manage network traffic and congestion. Network function virtualization offers a different method for creating, delivering, and using networking administrations. It is a method for separating the system capacities from constrained hardware units so they can continue to operate in a virtualized environment on standardized hardware. A virtual network function is created for this capability (firewall, deep packet inspection, deception system, and intrusion prevention). NFV is designed to deliver the system administration components necessary to support a foundation that is completely independent from equipment. These sections include system, storage, and virtual register capabilities. The idea for NFV originated with service providers who sought to create something that would allow the deployment of new networks or apps over hardware simple and quick (faster). The use of highly standardized IT virtualization technology over a network is made possible by hardware-based appliances. The primary justification for this suggestion was the significant financial outlay. To carry out the same function or process over the network, a single firewall system or any other system needs a lot of time, effort, and of course- money. This NFV (network function virtualization) chaining offers all terminations (such as firewall, virtualized data inspection, intrusion, and deception) over a single platform, performing the same executing process in a constrained or, should we say, condensed amount of time and costing almost no money. We will therefore explore a chaining problem solution, the placement of network functions on commodity hardware, and an algorithm to automate the assignment of the network functions in this study (NFs).

1.10.4 NFV Placement and Service Chaining Challenges for an enterprise security

Employing SDN/NFV-based security solutions for enterprises presents several challenges, with the primary ones being network design, NFV deployment, and resource management. These difficulties can hinder the effective placement and chaining of NFVs, limiting the overall security capabilities of the network. This work presents a well-architected and optimized solution for NFV placement and chaining of security services to address these issues and improve the security infrastructure. By carefully positioning NFVs inside the network architecture, the suggested solution attempts to increase the effectiveness and efficiency of security services. By considering

factors such as traffic patterns, performance requirements, and resource availability, the solution optimizes the placement of NFVs to ensure optimal security coverage and minimal latency.

Moreover, the proposed solution emphasizes the importance of NFV chaining, which involves sequencing multiple security functions to create comprehensive security service chains. By intelligently chaining NFVs, the solution enables the creation of customized security workflows tailored to the enterprise's specific security requirements. This approach enhances the network's ability to detect and mitigate threats, improving overall security posture.

Through careful design and optimization, the proposed NFV placement and chaining solution provide enterprises with a robust and scalable security infrastructure. By addressing the challenges associated with network design, NFV deployment, and resource management, the solution offers an effective means to leverage SDN/NFV-based security capabilities and strengthen the overall security posture of the enterprise.

1.11 Thesis Contribution

The behavior of network function virtualization and network function has been carefully examined. We have looked into the major problems and research challenges that typically arise in an architecture, NFV deployment and resource management. Finally, we have created a chaining technique and solution that offers a fix for the majority of the difficulties under investigation.

Therefore, the following are the thesis' significant contributions:

- ✓ A well architected SDN-NFV based design for enterprise security function placement and chaining.
- ✓ SDN-NFV based Network security function deployment using commodity hardware has presented in this work
- ✓ An algorithm is formulated overcome the constraints of resource utilization and to make it more optimized.
- ✓ Python based simulation to present the working of proposed algorithm

CHAPTER 2: LITERATURE REVIEW and SFC Algorithms and Network Devices

In this section, we'll showcase some of the most outstanding studies on network function virtualization, network function placement, and study on the 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 on the software-defined NFV system's dynamic service chaining framework. It was noted that the software-defined radio and cognitive radio are used to enable the future 5G network. SDN and NFV serve the purpose of enabling great efficiency and flexibility in the service chaining construction in this framework. It also comprises the guiding flows through the necessary service chain. This is accomplished through the use of dynamic service deployment and flexible traffic routing. The overall system was thoughtfully created to identify the best strategy for increasing performance and resource utilization. This study develops and explains an optimization framework and a unified control for the SDN-NFV framework, which is used to optimize service chaining in accordance with user needs and network environment. This system proves to be an advantageous framework for the many situations of traffic steering thanks to the designing of the services, network, and the development of an optimization technique. In this case, function assignment, policy optimization, and virtual machine selection are all included. This study on the chaining of network function virtualization will offer advantages that can be emulated under a real-world issue or scenario.

Hwang et al [10] also developed the idea of using a NetVM "net virtualization machine," which is well regarded as a very helpful resource when working on virtualization or making any plans related to it. By enabling high transfer speed system capacities to operate at close line speed and utilizing the adaptability and customization of ease-of-use item servers, NetVM introduces virtualization to the network. NetVM enables the implantation of configurable information plane handling components such as routers, proxies, and firewalls inside virtual machines. Progressively scaling, conveying, and reconstructing system capacities are made simple by NetVM. This

provides much more compelling justifications for adaptation than the one already presented. By allowing the bandwidth at high power, NetVM serves to bring virtualization to the network.

According to Lorenz et al. [11], With each passing year, there are more and more attack types and threat vectors. Despite the occurrence of the subsequent attacks, the network's firewall and other primary security measures remained unaltered and safe from any additional threats. In addition, a number of fresh difficulties are emerging with regard to manageability and scalability as well as the level of protection offered. The implementation of countermeasures, like as firewalls and intrusion detection systems, is part of the manageability. Furthermore, it is challenging to adjust the security measures to the current state of the network due to the close integration into the physical network's infrastructure. This study explores and illustrates several architectural approaches for the integration of NFV/SDN-based security solutions into the enterprise network.

2.2 Service Chaining Algorithm

Optimal solution algorithms are algorithms used to find the best solution to a problem within a given set of constraints. These algorithms are commonly used in optimization problems where the goal is to maximize or minimize an objective function subject to certain constraints.

There are several algorithms that can be used to find optimal solutions, including:

Linear programming (LP): A method for optimizing a linear objective function under linear constraints is called linear programming (LP). In business and economics, it is frequently used to find solutions to issues like resource allocation, production planning, and transportation.

Dynamic programming (DP): DP is a method for solving problems where the solution can be broken down into smaller subproblems. It is commonly used in computer science to solve problems such as shortest path algorithms, sequence alignment, and knapsack problems.

Branch and bound: Branch and bound is a method for solving combinatorial optimization problems by systematically exploring the search space. It is commonly used in scheduling problems, traveling salesman problems, and job sequencing problems.

Simulated annealing: A stochastic optimization approach called simulated annealing is motivated by the metallurgical annealing process. Finding the global minimum or maximum of a challenging, nonlinear objective function is a typical challenge that it is frequently employed to resolve.

Genetic algorithms: Natural selection serves as the inspiration for the class of optimization algorithms known as genetic algorithms. They are frequently employed to find solutions to issues in engineering, robotics, and artificial intelligence, such as control system optimization, parameter tweaking, and feature selection. These are just a few examples of the many algorithms that can be used to find optimal solutions to problems. The choice of algorithm depends on the specific problem being solved and the constraints involved.

2.2.1 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 $(i,j,k)= \min(\text{shortest path}(i,j,k-1), \text{shortest path}(i,k,k-1)+\text{shortest path}(k,j,k-1))$

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 $k=n$.

2.2.2 Dijkstra Algorithm

The Dijkstra algorithm is a different approach or method for determining the shortest path in a specified weighted graph between the starting and finishing nodes. The shortest path from the initial vertex, the source, and all other locations listed in the graph were determined using this approach as it was applied to the defined tree. Although the Dijkstra graph can be either directed or non-directed, it cannot handle negative weights. The Dijkstra algorithm and the Prim's minimal spanning tree algorithm are very similar with sources as the root, the shortest path is produced. In

Dijkstra determination, two sets are arranged, one containing the vertices that are part of the shortest path tree and the other containing the vertices that have not yet been part of the shortest path. Everywhere in the Dijkstra algorithm set, a vertex from the second set that is conducted has a minimum distance from the source node [29].

2.2.3 Shortest path algorithm

In order to discover the shortest path, we must first calculate the distance between each node. So, in order to find the shortest path, we're utilizing the Floyd-Warshall method.

"Floyd Marshall's algorithm, also known as the Roy-Warshall algorithm, is an algorithmic method of finding the shortest path in a considered weighted graph with both positive and negative edge weight but no negative cycles are allowed in it". Floyd warshall compare out all the possible paths with the help of the graph in between the pair of different vertices. The comparisons $\Theta(|V|^3)$ is use in the graph, maybe there are up to $\Omega(|V|^2)$ edges in working graph. In this algorithm every combination made over the edges are tested. It does incrementally improve 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)]

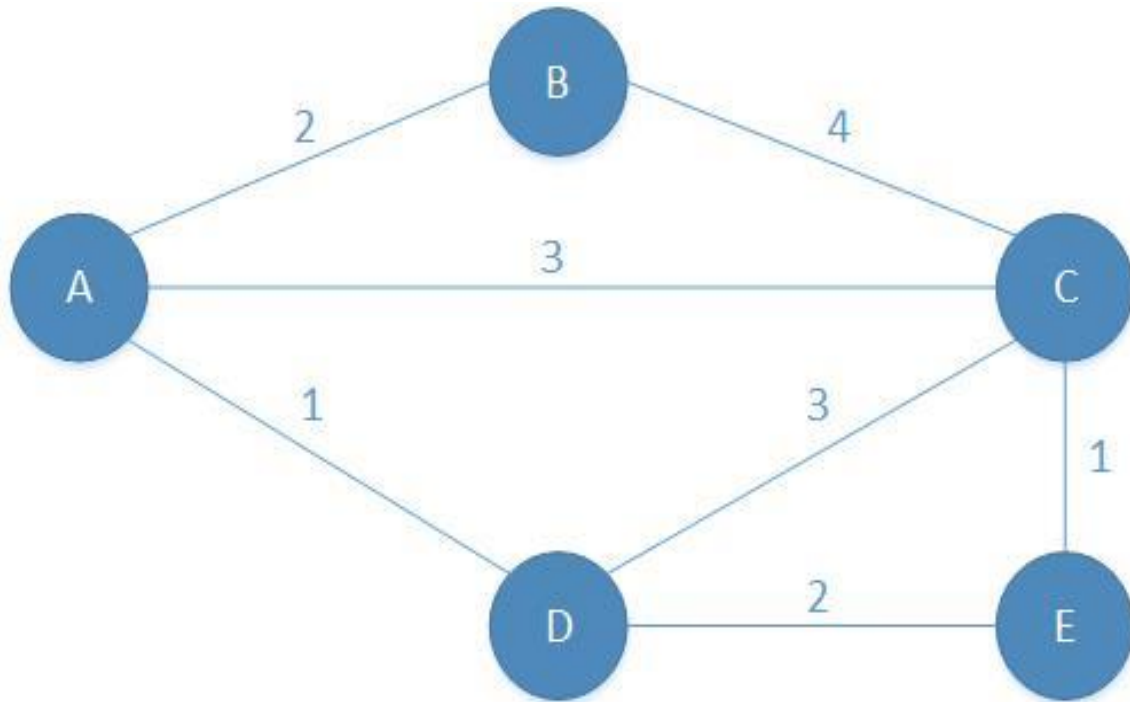


Figure 3.2: Proposed multi-node path

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

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)

$$\text{Node A} = 0.5 * 2.1 + 0.5 * 9 = 1.0 + 4.5 = 5.5$$

$$\text{Node B} = 0.5 * 4.8 + 0.5 * 14 = 2.75 + 7 = 9.4$$

$$\text{Node C} = 0.5 * 1.5 + 0.5 * 11 = 0.75 + 5.5 = 6.25$$

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.

2.3 Network Security devices and Tools

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

2.3.1 Firewall

A firewall is meant to block illegal access or signals coming from unidentified networks. The network function system's private or unidentified harmful nodes are blocked by the firewall together with malware. An overall security tool in the virtualization process is a firewall.

2.3.2 Deep Packet Inspection

Deep packet inspection is an advanced and clever approach of controlling the network traffic in a system. It is a form of packet filtration that uses location, rerouting, blocking, and classification to identify, find, and classify packets that include code payloads or any other kind of particular data.

2.3.3 Intrusion Detection System

A tool for identifying unusual behavior in network traffic and generating alerts when it is discovered is an intrusion detection system (IDS). It is software that checks networks and computer systems for harmful activity and rules that have been broken. A security information and event management (SIEM) system is frequently used to collect centralized data or to receive reports of any illegal activity or violation. A SIEM system integrates outputs from several sources and uses alarm filtering techniques to distinguish between real and fake alarms.

2.3.4 Intrusion Prevention System

An additional form of network security used throughout the virtualization process is the intrusion prevention system. It is employed for the purpose of finding risks and malware on a system. An intrusion prevention system was utilized to continuously monitor the system, searching for various malicious issues and gathering information about them.

2.3.5 Snort as IDS and IPS

The most popular Open-Source Intrusion Prevention System (IPS) in the world is Snort. With the use of a set of rules, Snort IPS can detect malicious network activities. After that, it looks for packets that satisfy those requirements and alerts users when it does. Installing Snort inline can also stop these packets from passing through. Snort can be applied in one of three ways: for network traffic debugging, packet recording, or as a complete network intrusion prevention system. Similar to tcpdump, it can be used as a packet sniffer. Snort can be downloaded and set up for both personal and professional use. In order to combine anomaly, protocol, and signature inspection techniques to discover potentially malicious behaviors, SNORT uses a rule-based language. SNORT is a tool that network administrators can use to identify buffer overflows, stealth port scans, Common Gateway Interface (CGI) attacks, distributed denial-of-service (DDoS), and DoS attacks. SNORT establishes a set of standards for identifying harmful network activity, identifying malicious packets, and alerting users.

Open-source software called SNORT is accessible for both private and professional use. The SNORT rule language specifies which network traffic should be captured and what should happen when malicious packets are found. Similar to how sniffers and network intrusion detection systems work, this snorting feature can be used to detect malicious packets. It can also be utilized as a full network IPS solution that monitors network traffic and identifies and blocks potential attack vectors.

2.3.6 Pfsense

Pfsense software provides edge firewall, router, and VPN functionality to homes, businesses, educational institutions, and government organizations - literally throughout every continent - strengthening networks. It is an exceptionally powerful, dependable, and easy-to-use solution. The pfSense project is a free network firewall distribution that also contains third-party free software

packages for additional functionality. It is based on the FreeBSD operating system. The package concept enables the pfSense software to provide more functionality than traditional commercial firewalls without the use of any artificial limitations. In several deployments around the world, it has effectively replaced nearly every well-known commercial firewall you can think of, including Check Point, Cisco PIX, Cisco ASA, Juniper, Sonicwall, Netgear, Watchguard, Astaro, and more. All of the included components can be configured using the web interface provided by the pfSense software. You never have to use the command line, you never have to be knowledgeable with UNIX, and you never have to manually edit any rule sets. There may be a learning curve for those who are used to using commercial-grade firewalls, but they rapidly become used to the online interface.

CHAPTER 3: PROPOSED METHODOLOGY

Software-Defined Networking (SDN) and Network Function Virtualization (NFV) technologies have emerged in response to the growing need for effective resource utilization in enterprise networks. While NFV enables network services to be virtualized and deployed as software on low-cost hardware, SDN allows for centralized network control and programmability. This combination presents a powerful opportunity to optimize resource utilization in enterprise networks by intelligently placing and chaining NFV instances. The methodology presented here outlines a systematic approach to develop an SDN-based NFV placement and chaining solution aimed at optimizing resource utilization, including CPU, storage, and network link capacity. By carefully analyzing the requirements of the enterprise's network infrastructure, including servers, switches, and links, the methodology defines the desired optimization objectives and creates a network model that represents the infrastructure accurately. With a focus on resource monitoring, the methodology emphasizes the collection of real-time data on CPU utilization, storage usage, and link bandwidth utilization. Leveraging this data, algorithms are developed to determine the optimal placement of NFV instances on servers, aiming to distribute the instances effectively while balancing resource utilization and avoiding server overloading. Furthermore, the methodology addresses the crucial aspect of chaining NFV instances, considering their communication requirements and traffic flows. By optimizing the chaining of NFV instances, latency can be minimized, and the utilization of network links can be maximized.

Integration with an SDN controller allows for the seamless implementation of the placement and chaining algorithms, enabling real-time control and decision-making based on dynamic resource utilization data. The solution is designed to be adaptive, with mechanisms in place to adjust the placement and chaining dynamically as resource utilization and traffic patterns change. The methodology encompasses evaluation, optimization, and ongoing maintenance phases to ensure the effectiveness and efficiency of the solution. Performance evaluations are conducted to assess the solution's capabilities, and optimization is achieved through iterative improvements based on defined objectives. By following this methodology, enterprises can harness the power of SDN and NFV technologies to achieve optimized resource utilization in their networks. The result is improved performance, reduced resource wastage, and a flexible and scalable network

infrastructure that can adapt to changing demands and workloads. Methodology for SDN-based NFV Placement and Chaining Solution to Optimize Resource Utilization

Requirement Analysis:

Understand the enterprise's network infrastructure, including servers, switches, and links. Identify the NFV applications to be deployed and their resource requirements (CPU, Storage, and bandwidth). Define the desired optimization objectives (e.g., minimizing resource wastage, maximizing performance).

Network Modeling:

Create a network model representing the enterprise's infrastructure, including servers, switches, and links. Define the available resources (CPU, Storage, link capacity) for each server and network link. Incorporate the NFV applications into the model, considering their resource requirements.

Resource Monitoring:

Implement monitoring mechanisms to continuously collect resource utilization data from servers and network devices. Collect CPU utilization, storage usage, and link bandwidth utilization information. Store and update the collected data in a centralized database or monitoring system.

Placement Algorithm:

Develop an algorithm to determine the optimal placement of NFV instances on servers based on resource utilization. Consider the resource requirements of each NFV application and the available resources on the servers. Aim to distribute the NFV instances across servers to balance resource utilization and avoid overloading any particular server.

Chaining Algorithm:

- Design an algorithm to determine the optimal chaining of NFV instances based on their communication requirements.
- Consider the traffic flows between NFV instances and the available network links.
- Strive for the highest possible network link utilization and the lowest possible communication latency.

SDN Controller Integration:

- Integrate the placement and chaining algorithms with the SDN controller.
- Enable the controller to receive real-time resource utilization data and control the placement and chaining decisions.
- Implement APIs or protocols to communicate between the controller and the NFV infrastructure.

Dynamic Adjustment:

- Implement mechanisms to dynamically adjust the NFV placement and chaining based on changing resource utilization and traffic patterns.
- Continuously monitor the resource utilization and adapt the placement and chaining decisions accordingly.
- Consider traffic demand changes, server failures, or addition/removal of NFV instances.

Evaluation and Optimization:

- Conduct performance evaluations to assess the effectiveness of the solution.
- Use simulation or emulation environments to test various scenarios and workloads.
- Analyze the results and optimize the algorithms based on the defined objectives.

Implementation and Deployment:

- Implement the SDN-based NFV placement and chaining solution in the enterprise's network infrastructure.
- Test the solution in a controlled environment before deploying it in production.
- Monitor and fine-tune the system during the initial deployment phase.

Ongoing Maintenance:

- Continuously monitor and analyze the system's performance and resource utilization.
- Incorporate feedback from users and operators to improve the solution over time.
- Stay updated with new technologies and research advancements in SDN and NFV to enhance the solution.

3.1 Proposed Architecture

In this section, we presented and discussed the network architecture, Figure 5 demonstrates the proposed architecture to implement the Virtualized network-based placement and chaining solution for an enterprise. This architecture is comprised of following zones: 1-Server Farm, 2-DMZ, 3-VLANs.

3.1.1 Server Farm

A server farm is a collection of numerous linked servers that are kept in the same physical location. By running one or more applications or services concurrently, a server farm offers the combined computational capability of numerous servers. Typically, a server farm is a component of a supercomputer or a section of an enterprise datacenter. A server cluster or computer ranch are other names for a server farm. A server farm is made to offer an enormous and redundant source of computing power for applications that require a lot of processing power. Although server farms often have thousands of servers, depending on the company and the underlying needs, their size might vary. The server farm's servers are connected through a network to one another and to a central management server. These servers' general operations are managed by the central server, including task assignment, resource balancing, scheduling, security, upgrades, and more. Although server farms are typically used for business and scientific applications, they can also be utilized for other services like load balancing, data and application backup, and core computing services for the primary application (ERP/CRM).

3.1.2 DMZ (Demilitarized Zone)

A DMZ network, often known as a "demilitarized zone," serves as a subnetwork that houses an organization's exposed, externally facing services in terms of computer security. It serves as the open entry point to the internet, an unreliable network. A DMZ's function is to give a company's local area network an additional layer of security. The content that is available in the DMZ can be accessed by a guarded and monitored network node that faces away from the company's internal network, but the remainder of the network is kept secure behind a firewall. A DMZ network, when configured properly, provides businesses with additional security by spotting and managing security breaches before they impact the internal network, which contains important assets. To safeguard the hosts that are most susceptible to attacks, there is a DMZ network. Email, web, and

DNS servers are the most typical instances of these hosts. These hosts frequently provide services to users outside of the LAN. They are positioned in the monitored subnetwork because to the elevated risk of attack and to aid in securing the rest of the network in the event that they are hacked. Since the data carried across the DMZ is less secure, the hosts in the DMZ have very restricted access rights to other services within the internal network. To further fortify the controlled border zone, communications between hosts in the DMZ and the outside network are also limited. Because the firewall isolates and controls all traffic shared between the DMZ and the internal network, hosts in the protected network can communicate with both the internal and external networks. Often, a second firewall will keep anything on the external network from accessing the DMZ. If a DMZ is employed, any services that users can use to communicate from an outside network can—and should—be put there. Some of the most popular services include: online servers. A DMZ may require the presence of web servers in order to maintain connection with an internal database server. This increases the security of the internal database, which typically houses critical data. The web servers can then directly access the internal database server or use an application firewall to do so, all while staying inside the DMZ's security perimeter.

Mail servers: The user database used to store login information and private communications is frequently stored on servers without direct internet access. To communicate with and access the email database without directly exposing it to potentially hazardous traffic, an email server will be developed or set up inside the DMZ.

Direct file interaction is made possible via **FTP servers**, which can also hold crucial website material for an organization. Therefore, key internal systems should never have full connectivity to an FTP server.

DMZ Designs

DMZs can be utilized in many network configurations. A single firewall (also known as a three-legged model) or twin firewalls are the two main strategies. It is possible to develop each of these systems to produce intricate architectures made to satiate network requirements. Use of a single firewall with at least three network interfaces would constitute a minimal strategy for network design. We'll put the DMZ inside of this firewall. The first network device establishes a connection with the ISP, the second establishes a connection with the internal network, and the third manages connections inside the DMZ. Dual firewall: Combining two firewalls into one firewall is the more

secure method. Frequently referred to as the "backend" firewall, the second firewall simply manages traffic going from the DMZ to the internal network. Using firewalls made by different

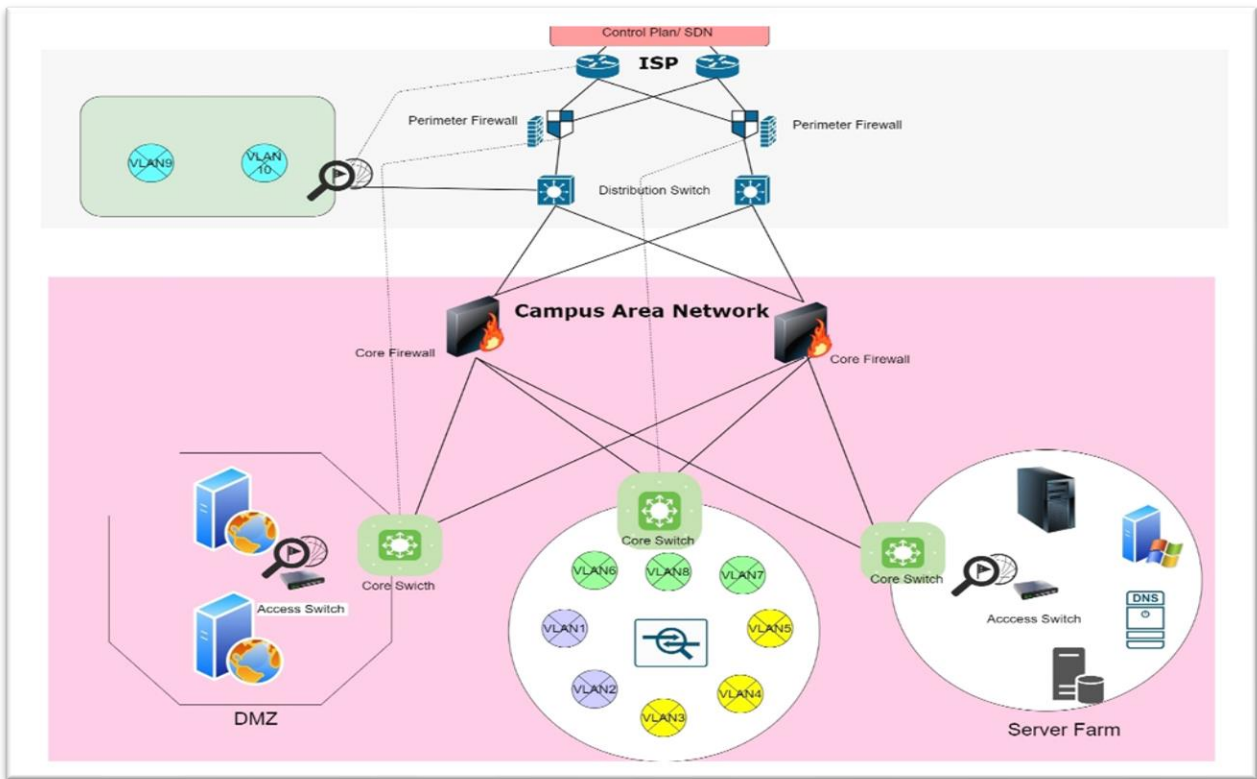


Figure 3: Network Architecture for an Enterprise Security

providers is a great method to further increase protection because they are less likely to share security concerns. The cost of applying this method across a wide network could be higher even though it is more cost-effective. In Proposed given below architecture, we opted dual firewall

DMZ design to strengthen the security of DMZ and Server Farm Zone. The abovementioned design is a very basic demonstration of any enterprise network with limited number of devices and resources but when becomes large, it starts becoming expensive, complex and difficult to manage for the executive management and network administrator. Therefore, to address this challenge we opted for Virtualized network-based placement and chaining solution. To address the second part of our proposed solution we had to architecture such network which offers service function chains, thus on the basis of Figure:6 we created number of service function chains as mentioned in Figure:7. It contains five service function chains. All SFC are comprised of network functions as mentioned in Figure6. In traditional network all functions were hardware based but these functions

are to be SDN-NFV based virtualized functions. So that, they can be controlled and managed through software-based platform rather hardware as in conventional networks.

SFC is crucial in an NFV network for traffic forwarding in order to implement network services in a predetermined order. When a service needs to be changed, all that needs to be done is alter the SFC sequence—the network configuration stays the same. Network services can be supplied quickly in this fashion. In Below mentioned figure, Service Function chains are presented.

Depending on their functions, network devices play a variety of roles in an SFC architecture. The

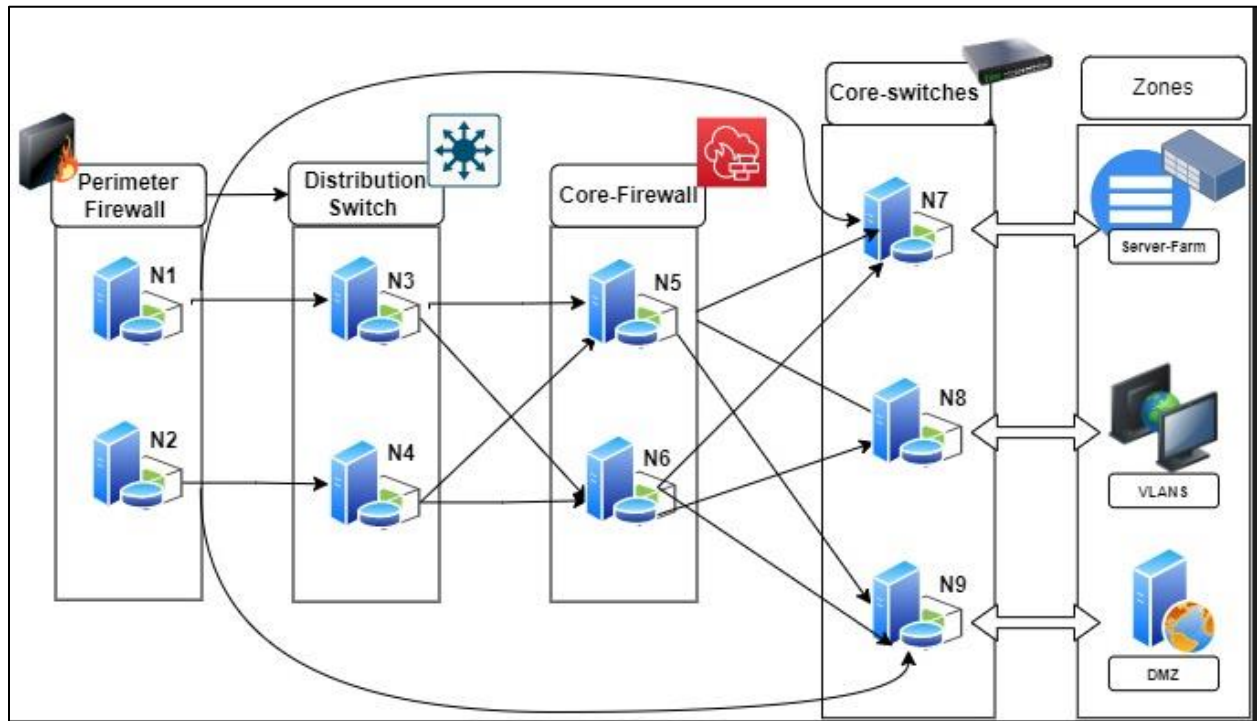


Figure 4: Architecture Nodes, Layers, Zones

following are SFC roles:

Service classifier (SC): At the entrance of an SFC domain is where a SC is situated. The SC classifies the packets, assigns service identities, and encapsulates the packets with service packet headers once they have entered the SFC domain.

Service function (SF): Devices known as SFs offer value-added services including load balancers and firewalls. SFs are divided into NSH-aware SFs and NSH-unaware SFs based on their awareness of the network service header (NSH) encapsulation. NSH-aware SFs can recognize and forward NSH packets, while NSH-unaware SFs are unable to recognize and discard NSH packets.

Service function forwarder (SFF): An SFF connects to an SF, recognizes service flow data, and then transmits packets in accordance with the SF service flow data.

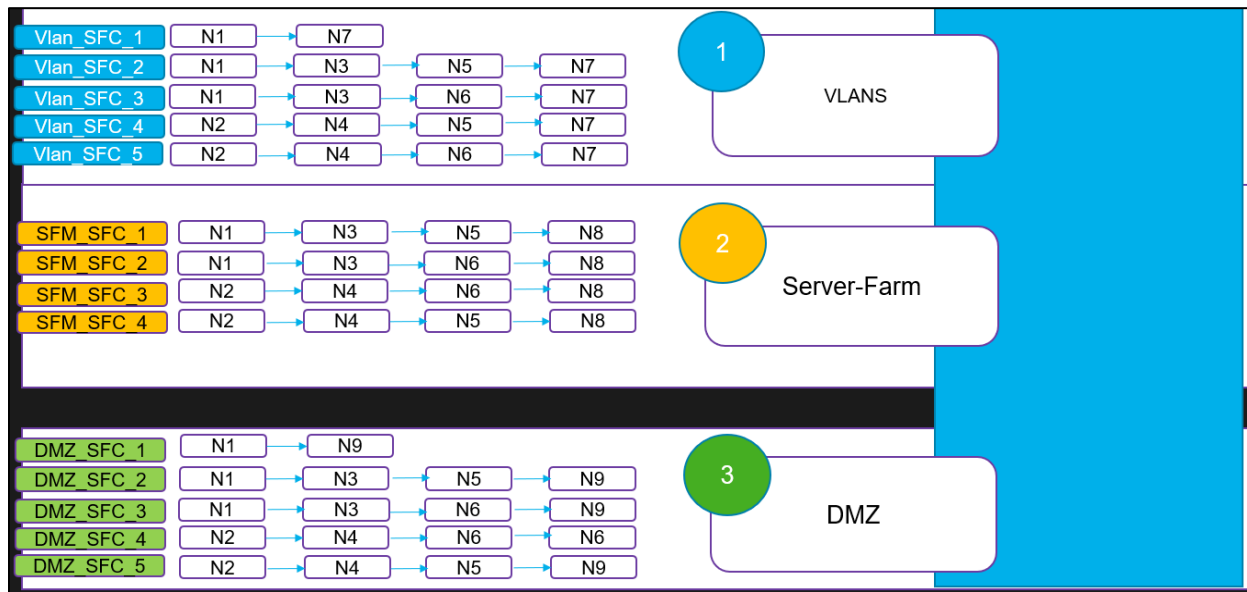


Figure 5: Service Function Chains - Zones Nodes

Figure 7, demonstrates the Number of critical and non-critical Service functions chains for the targeted zones (VLAN, Server Farm and DMZ). Total number of SFC are 14 and the distribution is as follows: 12 SFC are for critical services which includes the perimeter firewall and core firewall nodes. Whereas, number of non-critical services are 2. Which includes only perimeter firewall and core switch to reach the desired zone in the proposed architecture.

CHAPTER 4: VNF Placement and Chaining, Experimental Evaluation

Virtual Network Function (VNF) Placement and Chaining is a crucial aspect of NFV, enabling efficient deployment and orchestration of network services in software-defined networking (SDN) environments. This solution aims to optimize the allocation of VNFs onto suitable physical resources while considering network connectivity requirements and performance objectives. Additionally, VNF chaining allows the sequential connection of multiple VNFs to form complex network service chains, catering to specific service requirements. In this work, we provide a thorough experimental evaluation of the performance of a VNF Placement and Chaining solution. Our solution makes use of sophisticated algorithms and heuristics to decide where VNFs should be placed inside the network architecture in order to maximize resource utilization and decrease latency. We also solve the difficulties associated with VNF chaining, enabling the development of service chains utilizing several VNFs while ensuring end-to-end connection and Quality of Service (QoS). We ran a number of tests in a realistic network testbed to determine how well our approach worked. Different traffic loads, VNF setups, and service requirements were taken into account when simulating various network scenarios. We evaluated the solution's scalability, resource usage, latency, and overall network efficiency using rigorous performance measures. The experimental results provide valuable insights into the strengths and limitations of our VNF Placement and Chaining solution, highlighting its potential for real-world deployment and its impact on network service delivery. The remainder of this work is organized as follows: First, it provides an overview of related work in VNF Placement and Chaining. Secondly, it presents our proposed solution, detailing the algorithms and methodologies used. Thirdly, it describes the experimental setup and methodology. Fourthly, we discuss the obtained results and provides an analysis of the solution's performance. Finally, Section 6 summarizes the findings, highlights the contributions of this work, and outlines future research directions in the field of VNF Placement and Chaining.

4.1 VNF Placement:

The location of virtual network functions (VNFs) is essential for maximizing network infrastructure performance, resource utilization, and overall efficiency. The provided pseudo code

outlines a decision-making process for VNF placement based on various factors such as service request criticality, available server resources, distance to VMs, and resource conditions.

Certainly! Let's elaborate further on the decision-making process for VNF placement based on the given pseudo code.

The pseudo code presented employs a multi-step approach to determine the optimal placement of VNFs in the network infrastructure. It takes into account the criticality of the service request, available server resources, distance to VMs, and resource conditions to make informed decisions.

Here's a more detailed explanation:

Service Request Criticality:

The code initially checks whether the service request is critical or non-critical. This differentiation is essential because critical services often require prioritized resource allocation and higher performance guarantees compared to non-critical services.

Available Server Resources and Percentage Calculation:

The code counts the available server resources, which typically include CPU, RAM, storage, and link capacity. It then calculates the percentage of resources that are currently available. This information provides a baseline understanding of the available capacity in the network.

For Critical Service Requests:

ARMean Calculation: The code proceeds to calculate the ARMean (Average Response Mean) for each server. This metric helps assess the average response time and performance characteristics of each server. Servers with lower ARMean values are generally preferred due to their faster response times.

DVM-Z Calculation: The code calculates the DVM-Z (Distance to VM in the same zone) for each VM and sorts them in ascending order. This calculation quantifies the distance between the VNF and the VMs in the same zone, typically based on network topology or latency measurements.

DVM-Z Mean Calculation: The code computes the mean value of the DVM-Z distances. This mean serves as a reference point to evaluate the distances of individual VMs from the VNF.

Selection of Servers: Servers with DVM-Z values greater than the DVM-Z mean are selected. These servers are closer to the VMs and can potentially offer lower latency and better performance.

Resource Availability Check: The code verifies if the selected servers have enough available resources to accommodate the placement of the VNF. It checks whether the remaining resources,

after placing the VNF, are above a threshold (30% of total capacity). This condition ensures that sufficient resources are available for both the VNF and other applications running on the server.

Placement Decision: If the selected servers meet the resource availability condition, the code places the VNF on the server with the least distance to the VMs, prioritizing lower latency. If the remaining resources exceed the threshold, this server is preferred. Otherwise, the VNF is placed on the second-greatest server based on distance.

For Non-Critical Service Requests:

Similar to the process for critical service requests, the code calculates the ARMean for each VM and the DVM-Z for each VM, sorting them in ascending order.

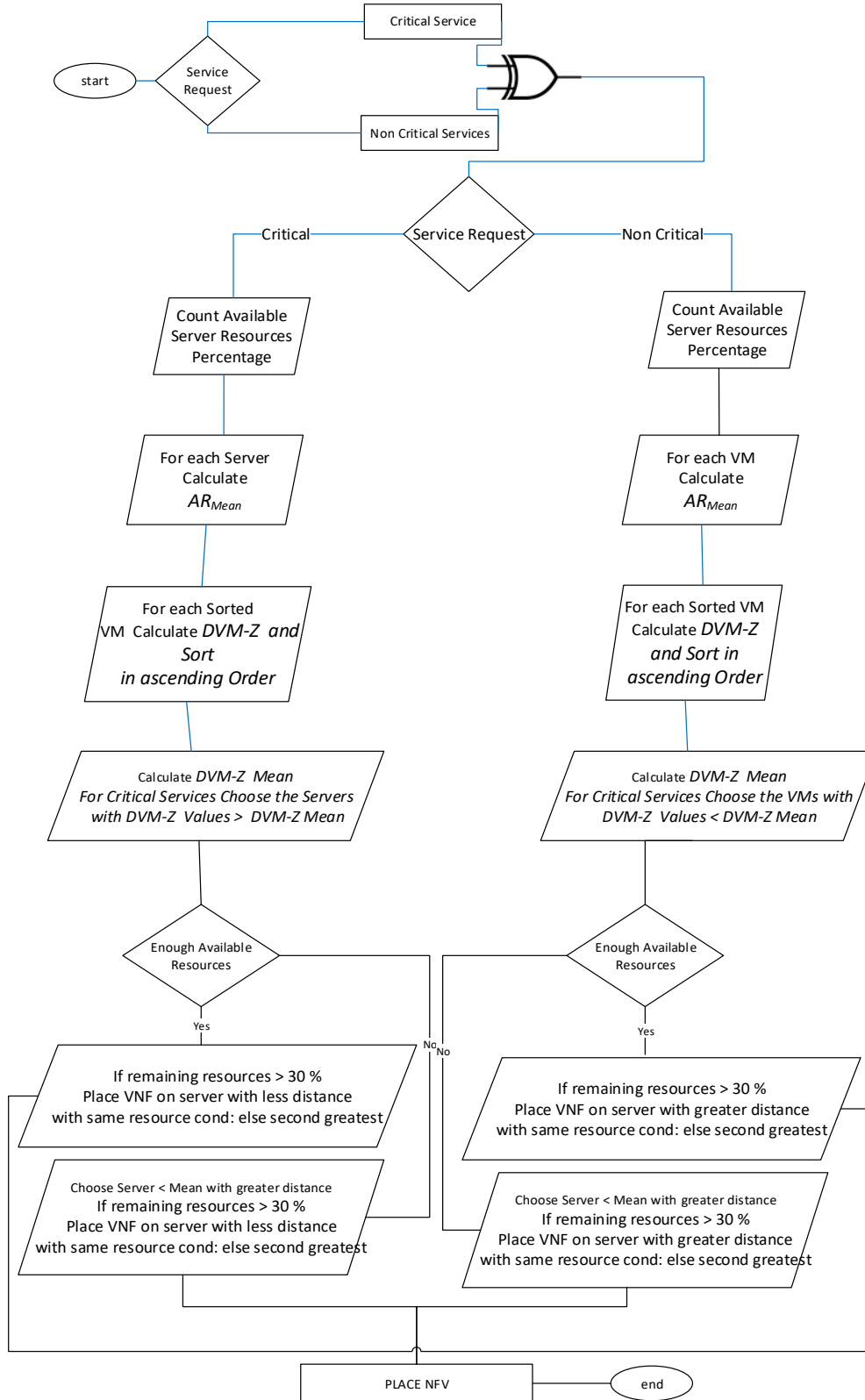
DVM-Z Mean Calculation: The code computes the DVM-Z mean, representing the average distance between the VNF and the VMs in the same zone.

Selection of VMs: In this case, VMs with DVM-Z values less than the DVM-Z mean are chosen. These VMs have shorter distances to the VNF, potentially reducing latency.

Resource Availability Check: The code verifies if the selected VMs have enough available resources to accommodate the placement of the VNF, following the same threshold condition as in the critical service request scenario.

Placement Decision: If the selected VMs meet the resource availability condition, the code places the VNF on the server with greater distance to the VMs, prioritizing resource optimization. If the remaining resources exceed the threshold, this server is preferred. Otherwise, the VNF is placed on the second-greatest server based on distance.

Flowchart 1 : NFV Placement



Start

Requested Zone (VLAN, Server FARM, DMZ)

Target Zone

Service Request (Critical or Non-Critical)

if Service Request is Critical:

 Count Available Server Resources

 Calculate Percentage

For each Server:

Calculate ARMean

 For each Sorted VM:

Calculate DVM-Z and Sort in ascending Order

 Calculate DVM-Z Mean

 Choose Servers with DVM-Z Values > DVM-Z Mean

 if Enough Available Resources:

if Remaining Resources > 30%:

 Place VNF on server with less distance with same resource condition, else second
greatest

 else:

 Choose Server < Mean with greater distance

if Remaining Resources > 30%:

 Place VNF on server with less distance with same resource condition, else second
greatest

 Place NFV_VM

End

Pseudo Code 1 : NFV Placement for Critical Service

The provided pseudo code outlines a decision-making process for placing Virtual Network Functions (VNFs) in a network infrastructure based on certain criteria. Here's a brief explanation of the code:

1. The code starts by gathering the information about the requested zone, target zone, and service request (whether it is critical or non-critical).
2. If the service request is critical, the code proceeds to calculate the available server resources and the percentage of resources available.
3. For each server, the code calculates the ARMean (Average Response Mean) and performs the following steps:
 - a. It calculates the DVM-Z (Distance to VM in the same zone) for each sorted Virtual Machine (VM) in ascending order.
 - b. It calculates the mean of the DVM-Z values.
 - c. It chooses servers with DVM-Z values greater than the DVM-Z mean.
 - d. If there are enough available resources on the chosen servers, it checks if the remaining resources after placing the VNF (Virtual Network Function) are more than 30%. If true, it places the VNF on the server with the least distance that meets the same resource condition. Otherwise, it places it on the second greatest server based on distance.
 - e. If there are not enough available resources on the chosen servers, it chooses a server with resources less than the mean and greater distance. It then follows the same condition as described in step d to place the VNF.
- 4- Finally, the code places the NFV_VM (Network Function Virtualization Virtual Machine) based on the determined placement strategy.

In summary, this pseudo code represents a decision-making algorithm for placing VNFs in a network infrastructure based on the availability of server resources, distance to VMs, and the criticality of the service request.

```

else if Service Request is Non-Critical:
    Count Available Server Resources
    Calculate Percentage
    For each VM:
        Calculate ARMean
        For each Sorted VM:
            Calculate DVM-Z and Sort in ascending Order
            Calculate DVM-Z Mean
            Choose VMs with DVM-Z Values < DVM-Z Mean
        if Enough Available Resources:
            if Remaining Resources > 30%:
                Place VNF on server with greater distance with same resource condition, else second
greatest
            else:
                Choose Server < Mean with greater distance
            if Remaining Resources > 30%:
                Place VNF on server with greater distance with same resource condition, else second
greatest
            Place NFV_VM
        End
    end
end

```

Pseudo Code 2: NFV Placement for Non Critical Service

The pseudo code 2 (NFV Placement for Non-Critical Service) is a continuation of the previous code and represents the decision-making process for placing Virtual Network Functions (VNFs) when the service request is non-critical. Here's a brief explanation of the code:

1. If the service request is non-critical, the code proceeds to calculate the available server resources and the percentage of resources available.

2. For each Virtual Machine (VM), the code calculates the ARMean (Average Response Mean) and performs the following steps:
 - a. It calculates the DVM-Z (Distance to VM in the same zone) for each sorted VM in ascending order.
 - b. It calculates the mean of the DVM-Z values.
 - c. It chooses VMs with DVM-Z values less than the DVM-Z mean.
 - d. If there are enough available resources on the chosen VMs, it checks if the remaining resources after placing the VNF are more than 30%. If true, it places the VNF on the server with greater distance that meets the same resource condition. Otherwise, it places it on the second greatest server based on distance.
 - e. If there are not enough available resources on the chosen VMs, it chooses a server with resources less than the mean and greater distance. It then follows the same condition as described in step d to place the VNF.

3. Finally, the code places the NFV_VM (Network Function Virtualization Virtual Machine) based on the determined placement strategy.

In summary, this pseudo code represents a decision-making algorithm for placing VNFs in a network infrastructure when the service request is non-critical. It considers the available server resources, distance to VMs, and aims to utilize servers with greater distance while meeting the resource condition.

Server	Virtual Machines	CPU (Dual Core)			RAM (GB)			Storage (GB)			Link Capacity (Gbps)		
		Total	Utilized	Available	Total	Utilized	Available	Total	Utilized	Available	Total	Utilized	Available
VM1	Perimeter Firewall	12	4	8	256	16	240	500	35	465	10	2	8
VM2	Distribution Switch	8	4	4	512	16	496	1000	25	975	10	2	8
VM3	Core Firewall	16	8	8	256	16	240	500	15	485	10	4	6
VM4	Core Switches	32	12	20	512	24	488	1000	45	955	10	6	4

Table 1: SDN - NFV based Architecture Core Component Resources Details

Performance Areas of Compute Nodes					
Nodes	CPU (Core)	RAM (GB)	BW (Gbps)	Latency (ms)	Energy (Watt)
N1 (PF)	2	8	1	0.03	20
N2 (PF)	2	8	1	0.04	15
N3 (DS)	2	8	1	0.09	12
N4 (DS)	2	8	1	0.09	13
N5 (CF)	4	8	2	0.07	15
N6 (CF)	4	8	2	0.09	16
N7 (CS)	4	8	2	0.1	12
N8 (CS)	4	8	2	0.09	17
N9 (CS)	4	8	2	0.07	18

Table 2: NFV Nodes Cost Matrix

The table generated against VMs (Perimeter firewall, distribution switch, core firewall, and core switch) provides a comprehensive overview of the available resources, including CPU, RAM, storage, and link capacity, along with their respective status: total, utilized, and available. This table serves as a valuable reference for understanding the resource allocation and capacity management of these network components. By presenting the total, utilized, and available resources for each VM type, the table allows network administrators and operators to easily monitor and assess the current resource usage and capacity utilization. This information is crucial for ensuring optimal performance, planning for future growth, and making informed decisions regarding the deployment of virtual network functions.

The table provides insights into the following aspects:

CPU, RAM, and Storage: These columns display the total capacity, the amount currently being used (utilized), and the remaining capacity (available) for each VM type. Monitoring these resources helps ensure that the VMs have sufficient computing power, memory, and storage to handle their respective tasks effectively.

Link capacity: This column indicates the total available bandwidth or link capacity for each VM type. It helps administrators gauge the capacity of the network connections associated with these VMs, ensuring that they have adequate bandwidth to handle the expected traffic load.

By regularly updating and referring to this table, our solution can easily identify any resource bottlenecks or areas where capacity may be reaching its limits. This information enables them to proactively address any potential issues, such as resource contention or performance degradation, by appropriately allocating resources, scaling up or out as needed, or planning for network upgrades to accommodate increasing demands.

In summary, the above tables provide a clear overview of the available resources for different VM types, allowing solution to effectively manage and optimize the network infrastructure, ensuring efficient resource utilization and maintaining a high level of performance and reliability.

4.2 Environment Setup VNF Placement

Download the appropriate installer for your operating system (Windows, macOS, or Linux).

Run the installer and follow the instructions to install Python 3.10.3.

- Install Visual Studio Code:
- Install the Python extension for Visual Studio Code:
- Create a new Python project:
- pip install Libraries...
- Coding: Run the Python code:

4.3 Simulation -Code- Python

SDN_NFV Simulation

Libraries Used:

```
from typing import List
import prettytable
import matplotlib.pyplot as plt
from tabulate import tabulate
import numpy as np
```

Main Blocks:

- Server Resources
- Number of Servers
- Node Resources
- Service Request
- Zones
- Distance to Zones
- Servers Matrix [Resources, Distances to Zone]
- Apply Algorithm

4.4 NFV Placement Simulation Results:

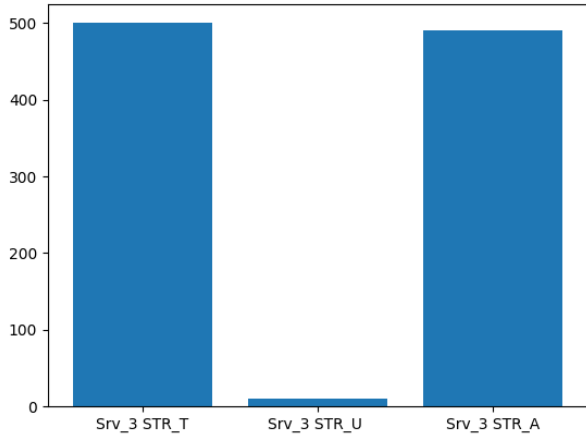
```

Server1      T_CPU      T_RAM      T_STORAGE   T_LINK
Server2      24         256        500         20
Server3      24         256        500         20
Server4      24         256        500         20
Avg node cost value: 15.2
Maximum server nodes: 53
Total Nodes for servers: 26
Average nodes per server: 7
Get Value from API: critical service
server 1 resource average: 478
server 2 resource average: 478
server 3 resource average: 748
server 4 resource average: 478
[478, 478, 748, 478]
['Server1', 478, 1]
['Server2', 478, 2]
['Server3', 748, 3]
['Server4', 478, 4]
Not Suitable for NFV placement
Not Suitable for NFV placement
[[748]]
3
Please assign required cpu10
20
Please assign required ram10
Please assign required storage10
Please assign required link10
-----
Total Compute  Utilized Compute  Available Compute
24             10                14
-----
-----
Total RAM      Utilized RAM      Available RAM
256           10                246
-----
-----
Total Storage  Utilized Storage  Available Storage
500           10                490
-----
-----
Total Link     Utilized Link     Available Link
20            10                10
-----

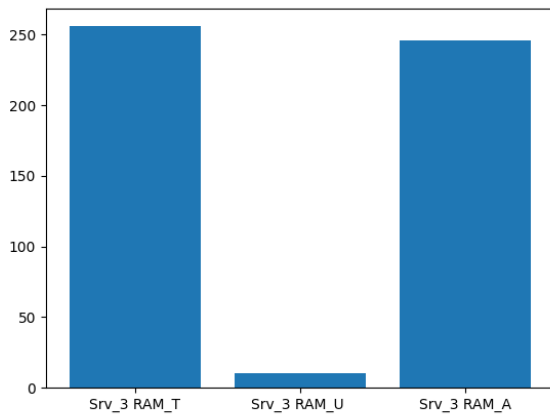
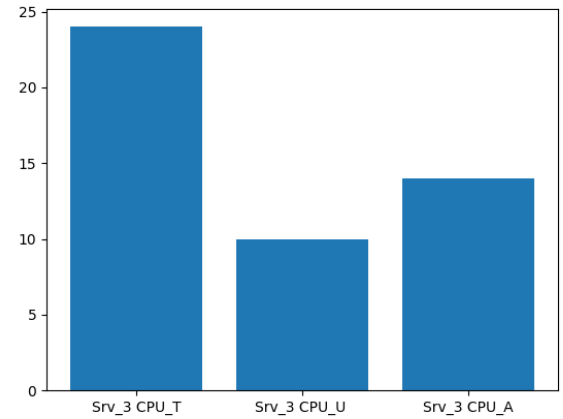
```


4.4 NFV Placement PLOTS:

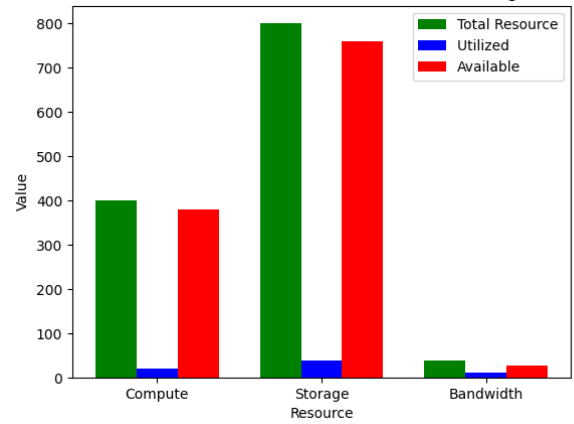
Plot 1 : NFV Placement Server (Storage)



Plot 2 : NFV Placement Server (CPU)



Plot 3: NFV Placement Server (RAM)



Server Selection for VNF using the Proposed Algorithm

Selected Server: No. 3

Graph Presenting - Resources:

1-Storage 2-RAM 3-CPU

4.3 Service Function Chaining

- Service function chaining (SFC) is a concept in computer networking that involves the sequential traversal of multiple network services to fulfill a specific task or request. It enables the creation of customized network service paths by chaining together various functions or applications. Each service function performs a specific operation, such as traffic filtering, deep packet inspection, encryption, or application-specific processing.
- The key idea behind service function chaining is to direct network traffic through a predefined sequence of service functions, allowing for the creation of flexible and dynamic service chains based on specific requirements. SFC can be implemented in various networking environments, including traditional networks, software-defined networks (SDNs), and network function virtualization (NFV) infrastructures.

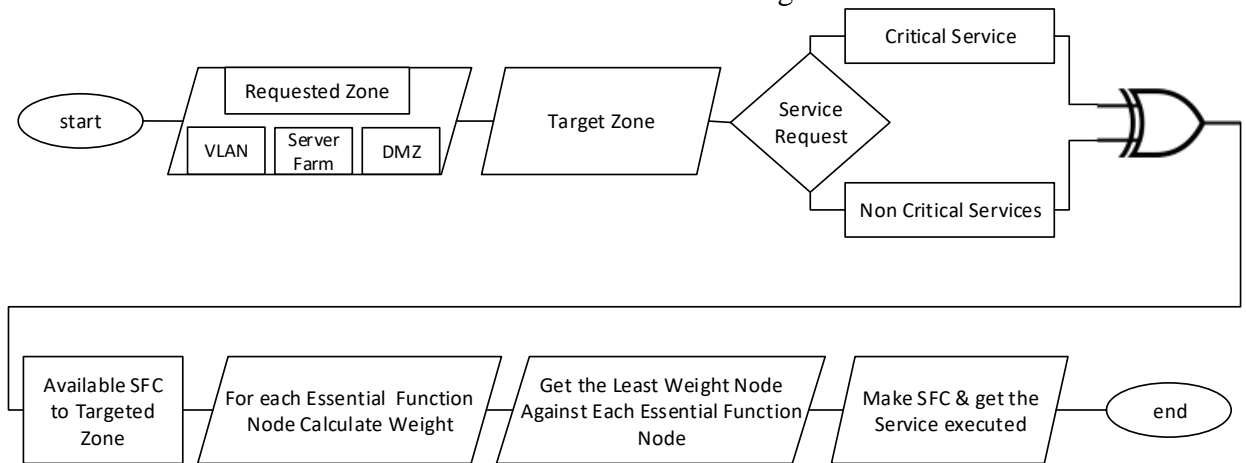
Here's how service function chaining typically works:

- **Classification:** Incoming network traffic is classified based on specific criteria, such as the source or destination address, protocol, application type, or quality of service (QoS) requirements.
- **Service Path Determination:** The network controller or orchestrator determines the appropriate sequence of service functions based on the classification result. It selects the service functions that need to be traversed and defines the order in which they should be applied.
- **Traffic Steering:** The network controller directs the traffic to the appropriate service functions by encapsulating packets with service headers or using other methods, ensuring that they follow the defined service path.
- **Service Function Processing:** Each service function in the chain processes the traffic according to its specific purpose. For example, a firewall might inspect and filter packets, while an encryption service encrypts the data.
- **Service Chain Termination:** After all the necessary service functions have been traversed, the traffic is delivered to its final destination, either within the network or to the external network.

- Service function chaining offers several benefits:
- Flexibility: SFC allows for the dynamic creation and modification of service chains, enabling network operators to adapt to changing service requirements or network conditions.
- Service Composition: Multiple service functions can be combined to create complex service chains tailored to specific needs, providing customized network services.
- Efficient Resource Utilization: By chaining service functions, network resources can be efficiently utilized, as traffic only passes through the necessary functions.
- Scalability: SFC can scale to support large-scale networks and accommodate diverse service requirements.

Overall, service function chaining enhances network flexibility, agility, and customization by enabling the orchestrated traversal of multiple service functions. It plays a crucial role in modern networking architectures, helping optimize traffic flow, improve security, and deliver specialized services.

Flowchart 2 : Service Function Chaining



start

Requested Zone (VLAN, Server FARM, DMZ)

Target Zone

Service Request (Critical or Non-Critical)

if Service Request is Critical:

SFC = FindCriticalSFCChains(Target Zone)

else:

SFC = FindNonCriticalSFCChains(Target Zone)

EssentialFunctions = GetEssentialFunctions(SFC)

ResourcesWeight = GetResourcesWeight(SFC, EssentialFunctions)

OptimalSFC = SelectOptimalSFC(SFC, EssentialFunctions, ResourcesWeight)

end

Pseudo Code 3 : Service Function Chaining

The pseudo code assumes the existence of the following functions:

FindCriticalSFCChains(TargetZone): Finds the Service Function Chains (SFC) to the Target Zone that are suitable for critical service requests.

FindNonSFCChains(TargetZone): Finds the Service Function Chains (SFC) to the Target Zone that are suitable for Critical/non-critical service requests.

GetEssentialFunctions(SFC): Retrieves the essential functions required for the selected SFC.

GetResourcesWeight(SFC, EssentialFunctions): Calculates the resources weight for each essential function within the available SFCs.

SelectOptimalSFC(SFC, EssentialFunctions, ResourcesWeight): Selects the SFC with the greatest total weight for all essential functions.

4.5 NFV Placement Results:

Zones
Total SFC = 14
Nodes = 9
Critical SFC = 12
Non-Critical Services 2

Request: Critical service
Essential Functions:

- Perimeter Firewall
- Distribution Switch
- Core Firewall
- Core switch
- No of PF Node = 2
- Number of DS Node = 2
- Number of CF Node = 2
- Number of CS Node = 3

```
Service Request: critical service
service: 0
name: pf
Largest value of pf: 11.841666666666667
Node 1
service: 1
name: ds
Largest value of ds : 13.508333333333333
Node 1
service: 2
name: cf
Largest value of cf: 10.674999999999999
Node 2
service: 3
name: cs
Largest value of cs: 12.674999999999999
Node 3
['n1', 'n1', 'n2', 'n3']
```

CHAPTER 5: VNF Placement and Deployment: Results and Evaluation

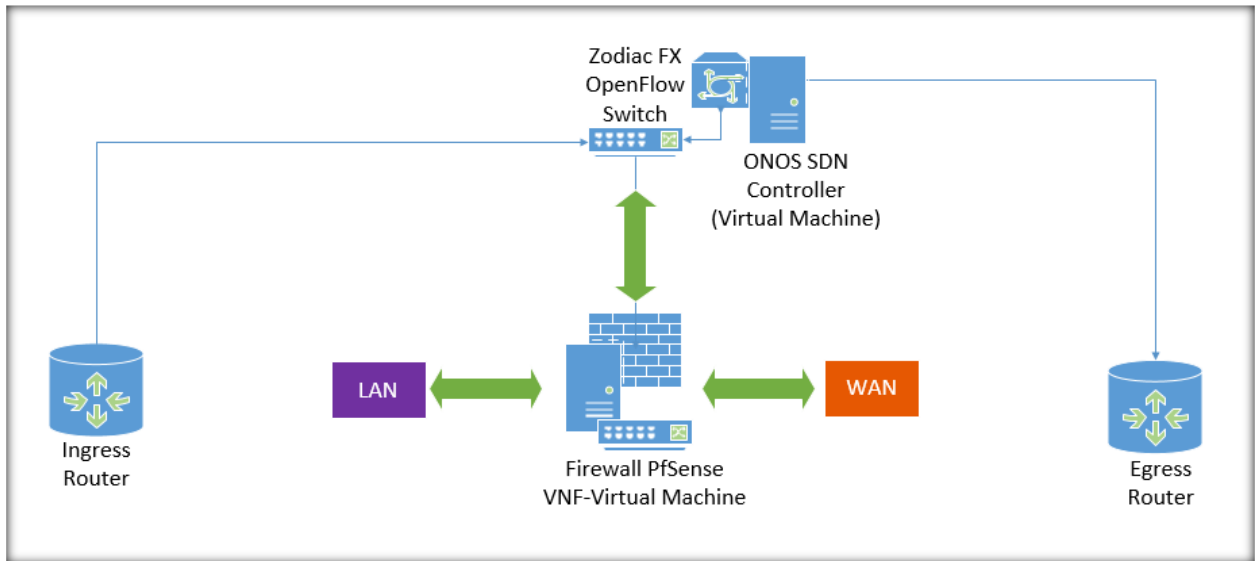


Figure 6: SDN-NFV based deployment of firewall using commodity Hardware

With this quickly developing technology and the of physical devices integration in established networks, management of resource and adaptive scalability are tough jobs, especially when it comes to network security measures. This research focuses on security solutions based on software defined networks (SDN) and network function virtualization (NFV) to address challenges with network and security management. The deployment, configuration, and use of SDN/NFV-based security solutions continue to have substantial issues. In order to solve this research question, we demonstrate the implementation of an SDN-NFVs-based network security solution in this section. The suggested approach is based on the use of Zodiac FX Openflow switches along with virtual network functions (VNF), open network operating system (ONOS), and SDN Controller. A firewall, an IPS, and an IDS are examples of virtual security functions (VSF), which make up a virtual network function (VNF). This work's design and implementation of a security solution for an enterprise using an SDN-NFV platform and affordable hardware is one of its primary accomplishments. The suggested NFV-based network security solution for a company is successfully deployed, configured, and implemented in this work.

Zodiac switch has been used [30], Opensource switch.

As shown in Figure 9, It features three ports for network device connections that support OpenFlow. To communicate with the ONOS controller, it features a Zodiac controller communication port. Additionally, it contains a JTAG connection, erase jumper, serial interface protocol (SIP), and USB connector. As shown in Figure 9, It has ports for network device connections that support OpenFlow, for

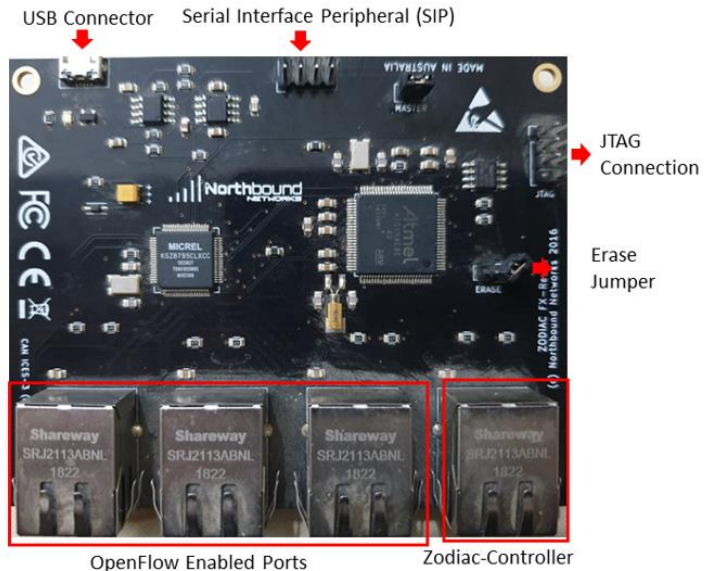
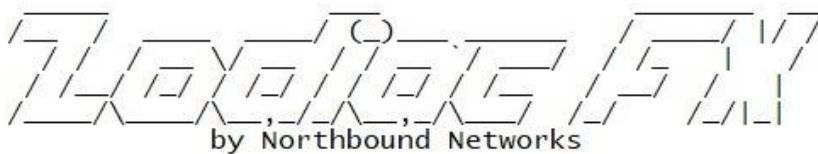


Figure 7: Zodiac OpenFlow Switch

the communication with controller, it features a Zodiac controller communication port. Additionally, it contains a JTAG connection, erase jumper, serial interface protocol, USB connector.

We set up Pfsense firewall as network security functions (NSF) for the placement of network virtualization services. Which is an excellent method for safeguarding networks from hackers. They can provide a system that has a wall isolating it from the outer world that is reasonably secure. There are various types of firewalls, among them filtering network packets play vital role. [31].



Type 'help' for a list of available commands

```
Zodiac_FX_01# config
Zodiac_FX_01(config)# show config
```

```
-----
Configuration
Name: Zodiac_FX_01
MAC Address: 70:B3:D5:6C:D3:A8
IP Address: 192.168.0.151
Netmask: 255.255.255.0
Gateway: 192.168.0.1
OpenFlow Controller: 192.168.0.188
OpenFlow Port: 6633
Openflow Status: Enabled
Failstate: Secure
Force OpenFlow version: Disabled
EtherType Filtering: Disabled
Port Stats Interval: 1
-----
```

```
Zodiac_FX_01(config)# exit
Zodiac_FX_01# openflow
Zodiac_FX_01(openflow)# show status
```

```
-----
Status: Connected
Version: 1.3 (0x04)
No tables: 1
No flows: 5
Total Lookups: 0
Total Matches: 0
-----
```

Figure 8: Zodiac Configuration

5.1 SYSTEM OVERVIEW

Every company's network contains a number of networking devices, including firewalls, IPS, IDS, DNS, DHCP, and others; nevertheless, this list is not all-inclusive. Traffic will be sent to the firewall, IDS, or IPS in accordance with the rules. Figure 1 displays an enterprise network that we considered for our work. When traffic enters the OpenFlow switches, a variety of traffic faults are permitted, and the ONOS controller will choose where to deliver the traffic data. Ingress routers are positioned at the start of the network for traffic inflow, whereas egress routers are positioned

at the end of the topology. The first SDN-based regulating mechanism is the ONOS controller [14]. The best open source controller for developing SDN/NFV systems in the future is this one. The reasons for selecting the ONOS platform are as follows. It is, first and foremost, a distributed, scalable, and modular SDN controller. Second, it's simple and straightforward to configure, install, and manage new software, hardware, and services. Not to mention its scalability, high performance, reliability, and support for next-generation devices, ONOS controller offers affordable solutions. In our topology, we've used the Zodiac switch [30], an SDN-based switch. It features three ports for network device connections that are OpenFlow enabled, as seen in Figure 9. To communicate with the ONOS controller, it features a Zodiac controller communication port. Additionally, it incorporates a Zodiac OpenFlow switch USB connector. JTAG connection, the SIP interface protocol, and the erase jumper. For the placement of network virtualized functions, we installed IDS, IPS, and firewall as network security functions (NSF). An IDS is a device or piece of software that keeps an eye out for malicious activities or policy violations on a network or internal system. A hardware or software component known as an IDS examines a network or system for malicious activity or policy violations. To centrally gather and report on any intrusion activity or violation, a security information and event management (SIEM) system is typically utilized. In order to discriminate between actual and false alarms, a SIEM system aggregates the outputs from many sources. The erase jumper, the SIP interface protocol, and the JTAG connection. We set up Pfsense firewall as network security functions (NSF) for the placement of network virtualization services. Packet filtering firewalls, firewalls that serve as application or circuit gates, hybrid firewalls, and firewalls that safeguard virtual private networks (VPNs) are just a few of the several types of firewalls available. The fact that next-generation firewalls (NGFW) offer all the capabilities in a single device and serve to strengthen and improve security management for a business network further contributes to their popularity today. [31]

5.2 PROPOSED IMPLEMENTATION

Here, we offer a recommendation for the NFV-based security solution's suggested use of commodity hardware. A virtual computer running Ubuntu Linux has been set up using the free open-source virtualization application virtual box. Two methods of installation and usage are available for ONOS SDN Controller. The first is intended for system administrators who want to set up standalone or clustered ONOS instances, while the second is intended for programmers who

want to develop and use their own special network applications. Neither of the modifications needs Zodiac switching after installation or use in our testing environment. The integrated development environment we utilised to create the development version of ONOS is IntelliJ IDEA. For usage by administrators, we have downloaded and installed the most latest version of ONOS. On Ubuntu Linux, we also configured an SDN controller. After installation, ONOS's command line interface (CLI) and graphical user interface (GUI) were both initialised. As was already indicated, we used Zodiac FX switch in our test setup. The switch's single port is used for communication between the SDN controller and the switch. The switch's other three ports, which are OpenFlow enabled and may be managed by the SDN controller, can be used for standard device communication. The Zodiac's configuration is seen in Figure 10. The latest recent firmware (0.86), was installed on the Zodiac FXAs. Through APIs, SDN Controller connects with apps and network hardware. There are two different kinds of APIs: northbound APIs and southbound APIs. Northbound APIs are used for communication between the SDN controller and the network applications. In this case, the most popular northbound API, REST, was used. The firewall for this project was Pfsense. A firewall was implemented on the VM.. The pfsense software has an integrated web interface for setting all integrated components. Thanks to the package approach, it may provide capacity that is equivalent to or more than that of traditional commercial firewalls without any artificial limitations. The pfsense program includes the online configuration interface for all of the included components. In order to integrate IDS and IPS, SNORT has been installed on two virtual machines. It is a strong open-source IDS and IPS that provides network traffic analysis and in-flight data packet monitoring. SNORT uses a rule-based language that incorporates anomaly, protocol, and signature inspection approaches to identify possibly malicious actions. SNORT is a tool that network administrators can use to identify denial-of-service (DoS), distributed denial-of-service (DDoS), CGI attacks, buffer overflows, and stealth port scans. SNORT created a set of rules that describe malicious network activity, identify malicious packets, and alert users. In this case, the general issues with the gadget are fi. We disregarded the FLWS presented against IN Ports 1 and 2 for our VMs. The ONOS controller will divert the incoming traffic to the proper devices connected to the Zodiac switch. The following information and features are also displayed on the controller interface.

5.3 EXPERIMENTAL EVALUATION

We built a scenario in which the firewall operated properly, as the VSF has guaranteed, in order to assess our setup. Figure 6 displays the system requirements, graphical user interface, and CLI interface for the Pfsense firewall running on a virtual machine. The experimentation procedure is set up in the manner below. ONOS controller monitors every bit of data that passes through and enters the Zodiac switches, and it can determine which users should utilise the firewall and which users should make use of IPS/IDS, based on the switch. Traffic is sent in the direction of the firewall when it reaches the zodiac switch and ONOS determines that it needs to go there, as seen in Figure 11. Pushing rules to the switch does this. The firewall's LAN/WAN rules will decide whether to approve or reject the user when it receives the traffic, as shown in Figures 12. We set up two interfaces, one for LAN and the other for WAN, as shown in Figure 12.

In this configuration, all incoming traffic is routed through the rules set on the ONOS controller to the Zodiac switch port, as illustrated in Figure 10. The configured rules are forwarding all WAN

STATE	PACKETS	DURATION	FLOW PRIORITY	TABLE NAME	SELECTOR	TREATMENT	APP NAME
Added	0	226	40000	0	ETH_TYPE:bddp	imm[OUTPUT:CONTROLLER], cleared:true	*core
Criteria: ETH_TYPE:bddp Treatment Instructions: imm[OUTPUT:CONTROLLER], cleared:true							
Added	0	226	40000	0	ETH_TYPE:ldp	imm[OUTPUT:CONTROLLER], cleared:true	*core
Criteria: ETH_TYPE:ldp Treatment Instructions: imm[OUTPUT:CONTROLLER], cleared:true							
Added	102	226	40000	0	ETH_TYPE:arp	imm[OUTPUT:CONTROLLER], cleared:true	*core
Criteria: ETH_TYPE:arp Treatment Instructions: imm[OUTPUT:CONTROLLER], cleared:true							
Added	21,876	57	60000	0	IN_PORT:2	imm[OUTPUT:1], cleared:true	*core
Criteria: IN_PORT:2 Treatment Instructions: imm[OUTPUT:1], cleared:true							
Added	24,943	77	60000	0	IN_PORT:1	imm[OUTPUT:2], cleared:true	*core
Criteria: IN_PORT:1 Treatment Instructions: imm[OUTPUT:2], cleared:true							

Figure 10: SDN Controller ONOS

Figure 9: Pfsense Firewall Deployed on VM

traffic to firewall. As a very basic rule to just allow incoming traffic to a corporate network, we set a rule with the parameters protocol IP4 and source with an asterisk (*) indicating that any IP4 from any source can access the network in order to test the functionality of the firewall. Three rules are displayed in Figure 14; the first two are the defaults, and the third rule is our successfully configured rule with a green checkmark. The same outcomes were obtained for LAN. Figure 13

illustrates the LAN regulations, emphasizing that any host on the LAN can access the internet and that outgoing traffic can reach the LAN.

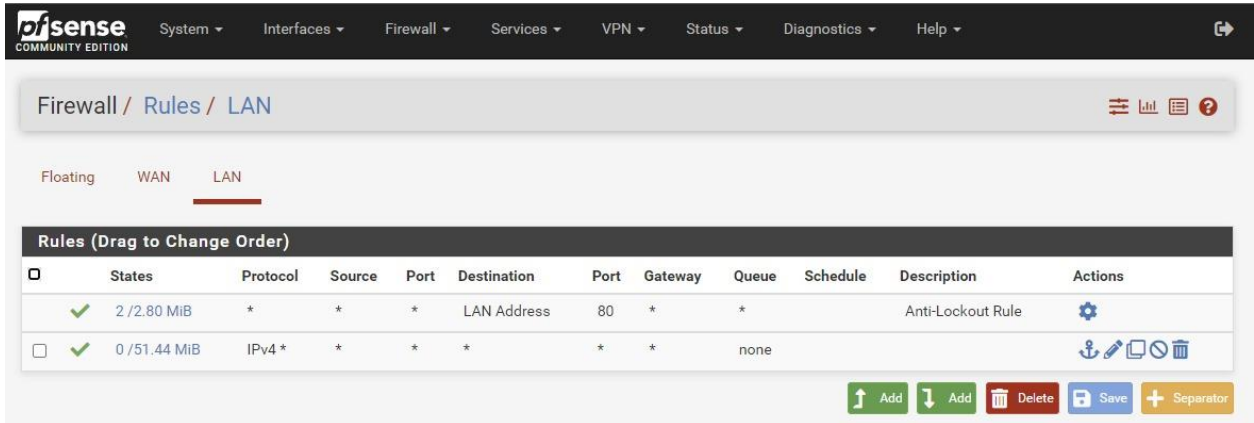


Figure 11: SDN-NFV based Firewall LAN Policy

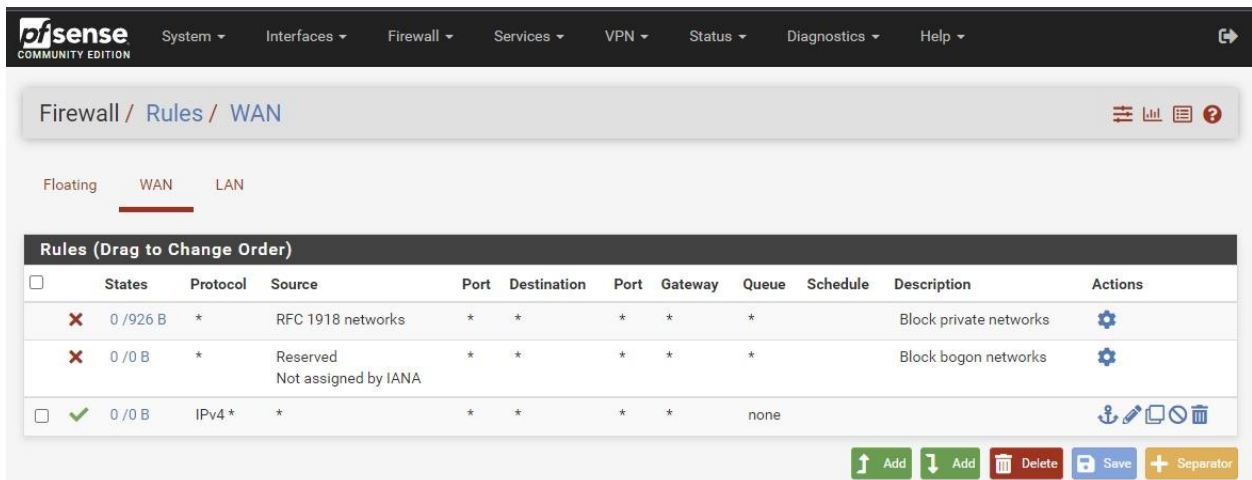


Figure 12: SDN-NFV Based Firewall WAN Policy

CHAPTER 6: Conclusion and Future Work

6.1 Conclusion

In conclusion, the methodology presented for SDN-based NFV placement and chaining solution offers a systematic and comprehensive approach to optimize resource utilization in enterprise networks. By leveraging the capabilities of SDN and NFV technologies, enterprises can achieve efficient allocation of CPU, storage, and network link resources, resulting in enhanced performance and reduced resource wastage. The methodology emphasizes the importance of understanding the enterprise's network infrastructure and requirements, which serves as the foundation for creating an accurate network model. By incorporating real-time resource monitoring and data collection, the placement and chaining algorithms can make informed decisions based on up-to-date information. The integration of the solution with an SDN controller enables dynamic control and adaptability, allowing for adjustments in NFV placement and chaining based on changing resource utilization and traffic patterns. This flexibility ensures that the solution remains effective and efficient as the network evolves over time. Through evaluation, optimization, and ongoing maintenance, the methodology ensures the continuous improvement of the solution's performance and resource utilization. By staying updated with advancements in SDN and NFV technologies and incorporating feedback from users and operators, enterprises can drive further enhancements and adapt the solution to their specific needs. By implementing the SDN-based NFV placement and chaining solution based on this methodology, enterprises can unlock the full potential of their network infrastructure. The optimized resource utilization leads to improved performance, scalability, and cost-efficiency. Ultimately, this methodology empowers enterprises to meet the demands of modern network environments while maximizing their return on investment and delivering superior network services to their users.

6.2 Future Work

As a future work, there are several avenues for extending and improving the SDN-based NFV placement and chaining solution methodology:

1. **Multi-objective Optimization:** The current methodology focuses on optimizing resource utilization. Future work could explore multi-objective optimization techniques to consider additional objectives, such as minimizing latency, maximizing energy efficiency, or improving fault tolerance. This would enable enterprises to achieve a broader range of optimization goals based on their specific requirements.
2. **Machine Learning and AI Techniques:** Incorporating machine learning and artificial intelligence techniques can enhance the intelligence and adaptability of the solution. By leveraging historical data and predictive analytics, the system can learn from past resource utilization patterns and make proactive decisions regarding NFV placement and chaining. This can lead to improved resource allocation and better adaptability to dynamic network conditions.
3. **Dynamic Workload Balancing:** While the current methodology aims to balance resource utilization during the initial placement of NFV instances, future work could focus on dynamic workload balancing. This involves continuously monitoring resource utilization and traffic patterns, and dynamically redistributing workloads across servers to optimize resource utilization and maintain performance under changing conditions.
4. **Network-aware Placement:** The methodology can be extended to consider the network topology and traffic patterns during the placement of NFV instances. By taking into account factors such as network latency, bandwidth constraints, and communication requirements, the solution can optimize both resource utilization and network performance, leading to improved overall system efficiency.

5. **Hybrid Placement Strategies:** Instead of solely focusing on placement within a single data center, future work could explore hybrid placement strategies that incorporate both local and cloud-based resources. This would enable enterprises to leverage the benefits of both on-premises and cloud infrastructure, optimizing resource utilization while ensuring scalability and cost-effectiveness.

6. **Integration with Intent-based Networking (IBN):** Intent-based Networking is an emerging paradigm that focuses on expressing high-level business objectives, allowing the network to autonomously translate and enforce those objectives. Future work could explore the integration of SDN-based NFV placement and chaining solutions with IBN, enabling the network to dynamically adapt its configurations and resource allocations based on high-level business intent.

7. **Security and Resilience:** Enhancing the solution's capabilities in terms of security and resilience is crucial. Future work can focus on incorporating mechanisms for ensuring secure NFV placement, secure communication between NFV instances, and resilience against network failures or attacks. This would provide enterprises with a more robust and secure network infrastructure.

By exploring these avenues for future work, the SDN-based NFV placement and chaining solution can be extended and improved to address evolving network challenges and meet the changing needs of enterprises. These enhancements would further optimize resource utilization, enhance network performance, and enable enterprises to leverage the full potential of SDN and NFV technologies.

REFERENCES

- [1] M. C. Luizelli, L. R. Bays, L. S. Buriol, M. P. Barcellos, and L. P. Gaspar, "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. Zeglache, "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. Gaspar, "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] J. Hwang, K. K. Ramakrishnan, and T. Wood, "NetVM: High Performance and Flexible Networking Using Virtualization on Commodity Platforms," IEEE Transactions on Network and Service Management, vol. 12, no. 1, pp. 34–47, 2015.

- [11] 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.
- [12] 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
- [13] Fraunhofer FOKUS. 2016. OpenSDNCore—Research and testbed for the carrier-grade nfv/sdn environment. Retrieved July 25, 2016 from <http://www.opensdncore.org/>
- [14] J. Sherry, S. Hasan, C. Scott, A. Krishnamurthy, S. Ratnasamy, and V. Sekar, “Making middleboxes someone else’s problem,” *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 4, pp. 13–24, 2012.
- [15] 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.
- [16] 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.
- [17] Mohammed Alsaeedi; Mohd Murtadha Mohamad”*Toward Adaptive and Scalable OpenFlow-SDN Flow Control: A Survey*”
- [18] 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.
- [19] Rackspace Cloud Computing. 2016. OpenStack Open Source Cloud Computing Software. Retrieved July 25, 2016 from <https://www.openstack.org/>
- [20] Linux Foundation. 2016. The OpenDaylight Platform. Retrieved July 25, 2016 from <http://www.opendaylight.org>
- [21] 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.

- [22] A. Gupta and R. K. Jha, "A Survey of 5G Network: Architecture and Emerging Technologies," *IEEE Access*, vol. 3, pp. 1206–1232, 2015.
- [23] ITU towards "IMT for 2020 and beyond." Retrieved September 28, 2017 from <http://www.itu.int/en/ITU-R/study-groups/rsg5/rwp5d/imt-2020/Pages/default.aspx>.
- [24] Y. Kyung, T. M. Nguyen, K. Hong, J. Park, and J. Park, "Software defined service migration through legacy service integration into 4G networks and future evolutions," *IEEE Communications Magazine*, vol. 53, no. 9, pp. 108–114, 2015.
- [25] J. Ordonez-Lucena, P. Ameigeiras, D. Lopez, J. J. Ramos-Munoz, J. Lorca, and J. Folgueira, "Network Slicing for 5G with SDN/NFV: Concepts, Architectures, and Challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 80–87, 2017.
- [26] A. Lombardo, A. Manzalini, G. Schembra, G. Faraci, C. Rametta, and V. Riccobene, "An open framework to enable NetFATE (Network Functions at the edge)," *Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft)*, 2015.
- [27] 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.
- [28] S. E. C. a. L. S. Guerassimov, "NFV and OPNFV," *Network Architectures and Services*, 2016
- [29] 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.
- [30] D. Gonzalez, C. Mellado, K. Waltam, and A. Lara, "Low-cost SDN ´ switch comparison: Zodiac FX and raspberry PI," in *2019 IV Jornadas Costarricenses de Investigacion en Computaci ´ on e Inform ´ atica ´ (JoCICI)*, 2019, pp. 1–5.
- [31] M. Arefin, M. Uddin, N. A. Evan, M. R. Alam et al., "Enterprise network: Security enhancement and policy management using nextgeneration firewall (NGFW)," in *Computer Networks, Big Data and IoT*, 2021, pp. 753–769