Performance and Cost Analysis in Building Network Address Translators (NATs) as Virtual Network Functions (VNFs) in NFV-based Systems

You-Sheng Liu and Quincy Wu
National Chi Nan University
Nantou, Taiwan
{s107321024, solomon}@ ncnu.edu.tw

ABSTRACT

As the fifth-generation (5G) of cellular networks is thriving, it provides higher network speed and lower latency to mobile network users. However, the service providers need to face several challenges like installing new network devices with considerable financial investments or quickly deploying new network services. Network Function Virtualization (NFV) is proposed to solve these problems in 5G core networks. In this paper, we will introduce the advantages and challenges of NFV, briefly walk through its framework. Then, we will discuss how to develop Virtualized Network Functions (VNF), by taking network address translation (NAT) as the sample network function. Finally, we will conduct an experiment to demonstrate that deploying the virtualized NAT on the generic server is not only with lower cost but also with expected performance.

Keywords: NFV, NAT

I. INTRODUCTION

Network Function Virtualization (NFV) is a network architecture that leverages the virtualization technology to decouple the network functions from the dedicated hardware appliances onto the standard commercial off-the-shelf while the network functions are realized as virtualized entities commonly referred to Virtualized Network Functions (VNFs). In another word, VNF is a network function running on the virtual machine (VM) instead of running on the dedicated hardware appliances. NFV enables flexible network function deployment, reducing any extra purchases on dedicated hardware appliances for special network functions; therefore, NFV reduces the overall capital expenses (CAPEX). Compared to legacy networks which we must configure each network device manually, NFV deployment can be fully automated. NFV also reduces operational expenses (OPEX) by making use of software tools automatically without on-site installation or configuration.

There are more logical benefits that can be provided by NFV. First, the life cycle of VNF can be shorter and dynamic compared to physical devices because these functions can be added when they are required, provisioned easily through automated software tools that do not require any on-site installation or configuration, and torn down to free up the resources when there is no need for these functions. Second, NFV allows VNFs to expand or shrink their resource distribution through various methods. Vertically, as long as the server has enough resources, VNF is able to adjust its resources on-demand. From a horizon perspective, it can also create a new instance that implements the same network function to split the load with the existing VNF.

Challenges do exist while realizing NFV [1]. Performance is always an important issue about VNF. Some factors may lead to degradation of performance. For example, using the general-purpose server without the hardware acceleration, single VNF or multiple VNF configuration, the choice of hypervisor. However, it still leverages virtualization technology, such as single-root I/O virtualization (SR-IOV) or Peripheral Component Interconnect (PCI) passthrough [2], to improve the performance.

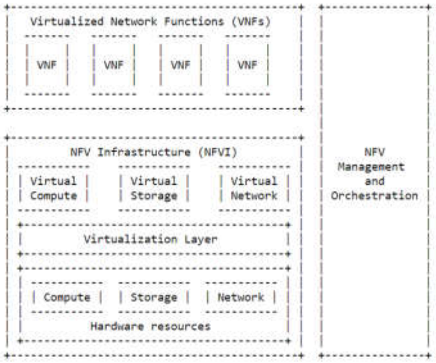

Figure 1: ETSI NFV Framework [**Error! Reference source not found.**]

European Telecommunications Standards Institute (ETSI) NFV Framework (Fig. 1) is composed of three domains.

1) Virtualized Network Functions (VNFs) that virtualized entity implements network functions such as firewall, router, and so on. Each network service (NS) can be composed of one or more VNFs.

2) NFV Infrastructure (NFVI) is in charge of virtualizing physical resources like CPUs, memories, disks, and network adaptors to virtual resources and distributing them to the VNFs running over the NFVI.

3) NFV Management and Orchestration (NFV MANO) manages all the activities about virtualization in the NFV architecture such as the lifecycle management of VNFs, the resource management of NFVI, and the orchestration of NS.

In this paper, we will use Openstack [5] as NFVI and Open Source MANO (OSM) [6] [7] as NFV MANO.

Openstack is an open project hosted by Rackspace Technology and NASA and it aims to provide a common service for cloud infrastructure. In order to complete the goal, openstack is divided into many software sub-projects. For example, Nova is for computing service, Neutron is for networking service, Keystone is for identity service, etc; therefore, we can install each sub-project individually to build our own infrastructure based on requirements.

OSM is an open project hosted by ETSI and it aims to develop an open source NFV MANO that is aligned with the ETSI NFV standard information model. OSM covers the three sub-domains in the NFV MANO, there are NFV Orchestrator (NFVO), VNF Manager (VNFM), and Virtualized Infrastructure Manager (VIM) respectively. NFVO is responsible for managing the lifecycle of NS, VNFM is responsible for managing the lifecycle of VNF and VIM is responsible for managing the resources in NFVI. The operations mentioned above are wrapped in the osm tool.

II. How to develop VNF

Service function chain (SFC) [4] is usually used to illustrate the network service and the order of VNFs that are going to be applied to. As shown in Fig. 2, packets pass through a VNF firewall, then they are distributed by a VNF load balancer and finally they reach services like web service or something else. (PNF stands for Physical Network Function; SaaS means Software as a Service)

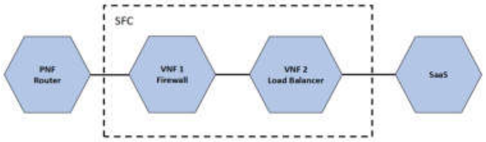

Figure 2. Example of SFC

We usually create more instances for a VNF to keep the availability and efficiency with the expansion of the service scale. These VNFs can be used to distribute the network traffic. Some VNFs may be used in different SFCs. As shown in Fig. 3, there are two services, service A and service B. Service A is composed of VNF1, VNF2, and VNF3. Service B is composed of VNF4, VNF2, and VNF5. If the network traffic is low, then we can launch a single VNF2 instance to satisfy the requirement as well as minimize the cost. When the network traffic is growing up, we can dynamically launch more VNF2 instances to distribute the traffic. However, this may introduces some problems, such as how to determine the path. Logically, VNF2A, VNF2B and VNF2C are all VNF2 but physically they could be deployed on different servers. Therefore, the placement of VNF is also an issue. [8] explored how to arrange the VNFs to the appropriate VM so that the VNFs are satisfied with the resources and the delay is also in the tolerated range? However, that situation is more complicated and beyond the scope of this paper. We will focus on the comparison of performance and cost between the dedicated server and the generic server.

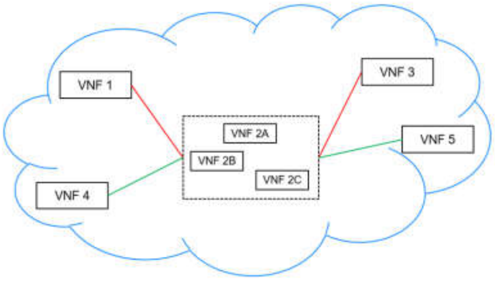

Figure 3. Multiple SFC in cloud

We will briefly introduce the steps of developing VNFs. Assume that Openstack and OSM are already installed and configured. First, we register the Openstack as NFVI to OSM. You should provide the URL of API, project name, account information. Second, using the osm tool to create a VNF package which we focus on the VNF Description (VNFD), cloud-init, and charm. In VNFD, we can specify the virtual resource to the VNF and which software image, cloud-init file, and charm to be used. In cloud-init, we use this file to initialize our virtual machine. Normally, we use it to configure the network setting and account setting of the virtual machine. Charm is created by juju, a free and open-source application modeling tool developed by Canonical Ltd., which includes a series of commands for initializing the VNF. Charm is also responsible for the runtime operation of VNF. Third, using the osm tool to create an NS package which we only focus on the NS Description (NSD). In NSD, we can define that which VNFs we will use, the connectivity among them, and how to deploy the NS on NFVI. Finally, we can launch the NS for providing the service.

III. NAT as VNF

Network address translation (NAT) [9] is an essential network function for an organization of a certain size, such as small and medium-sized enterprises or campuses. In the original design, every device must possess a public IP address to communicate with other devices on the Internet However, the amount of public IP addresses is limited and it is impossible to assign a public IP address to each user. What to do if the users of an organization want to access the Internet but there is not enough public address? NAT is proposed to solve this problem.

NAT, which is the address translation between private and public network addresses and provides transparent routing to end hosts, is proposed to solve the exhaustion of IPv4 addresses. It also provides a certain level of security that avoids the end hosts being directly accessed through the Internet. There are three types of NAT about address binding: static address translation, dynamic address translation, and network address port translation (NAPT).

1) Static address translation is one-to-one address mapping for end hosts between the private network and public network during the lifetime of NAT operation.

2) Dynamic address translation, opposite to static address translation, the mapping relationship between private and public network is dynamically based on usage requirements and

NAT server will free up the address when the binding is terminated so the address can be reused.

3) Network address port translation is the most common NAT type. It can transform transport identifiers, like TCP and UDP port numbers, ICMP query identifiers, in addition to transforming IP addresses. As a result, the public addresses depletion problem can be extremely mitigated by mapping multiple private addresses to a single public address.

NAT is an essential and necessary network function in enterprise and campus networks, so we want to design a virtualized NAT as a VNF, and test its performance to verify whether it can load the traffic of the usual campus network, and hope to use a cheaper cost to get better feedback.

IV. Cost/Performance under campus network

In this section, we will compare the cost and performance between the dedicated network device and the general-purpose server. Table I illustrates the basic information about the two devices. Cisco Firewall 2120 has the network processor units (NPU) that specifically for processing the network packets and it also supports more network functions. These features make it more powerful than a generic server, so its price is consequently much more expensive. Before determining this dedicated server as your best choice, we should consider if we will use all of the network functions that it supports, if we can maximize the resource utilization, and how familiar our users are with the operating system.

TABLE I. Comparison with Cisco Firewall 2120 and x86 Generic Server

Server	Cisco Firewall 2120	x86 Generic Server
Resource	CPU: 6 NPU: 8 Memory: 16 GB NPU Memory: 8 GB Disk: 2 SSD Slot	CPU: 16 Memory: 32 GB Disk: 1 TB, SSD
System	Internetwork Operating System (IOS)	Any Systems
Network Functions	Firewall, IPS, And more	Customized Network Functions
Price	About 560,000 NTD	About 20,000 NTD

References

[1] M. Chiosi et al., “ Network Functions Virtualisation: An Introduction, Benefits, Enablers, Challenges & Call for Action.,” ETSI White Paper, Oct. 2012.

[2] A.L. M. Falkner, I. Lambadaris and G. Kesidis, “Performance analysis of virtualized network functions on virtualized systems architectures,” Computer Aided Modelling and Design of Communication Links and Networks (CAMAD) 2016 IEEE 21st International Workshop on, 2016.

[3] A.R. C. J. Bernardos, J. C. Zuniga, L. M. Contreras, P. Aranda and P. Lynch, “Network Virtualization Research Challenges,” IETF RFC 8568, April 2019.

[4] E. J. Halpern, Ericsson, C. Pignataro, Ed., “ Service Function Chaining (SFC) Architecture,” IETF RFC 7665, October 2015.

[5] Daniel Grzonka, “The analysis of openstack cloud computing platform: Features and performance,” Journal of telecommunications and Information Technology, 2015.

[6] L. Mamushiane, A. A. Lysko, T. Mukute, J. Mwangama and Z. D. Toit, "Overview of 9 Open-Source Resource Orchestrating ETSI MANO Compliant Implementations: A Brief Survey," 2019 IEEE 2nd Wireless Africa Conference (WAC), 2019, pp. 1-7

[7] YILMA, Girma M., et al. “On the challenges and KPIs for benchmarking open-source NFV MANO systems: OSM vs ONAP,” arXiv preprint arXiv:1904.10697, 2019.

[8] H. Hawilo, M. Jammal and A. Shami, "Network Function Virtualization-Aware Orchestrator for Service Function Chaining Placement in the Cloud," in IEEE Journal on Selected Areas in Communications, vol. 37, no. 3, pp. 643-655, March 2019

[9] K.E. P. Srisuresh, “Traditional IP Network Address Translator (Traditional NAT),” IETF RFC 3022, January 2001.