NI: A Benchmark Suite for NfvInsight

ABSTRACT

Keywords

Network Function Virtualization;

1. INTRODUCTION

Network function virtualization (NFV) has become a hot topic, both in industry and academia. Since the publication of NFV Introductory White Paper [] of ETSI in 2012, a lot of works have been emerged in this field. Modification works of NFV were done on the whole software stack (or maybe both SW and HW stack?). There are de facto industrial NFV platform OPNFV [] as well as advanced NF allocation frameworks like OpenNF [], CoMb [] and E2 []. Also, there are works like OpenBox [] and Netbricks [] to rewrite or modify the NFs.

However, a suite of easy-to-use NFV benchmark is not yet existed. It is unavoidable to experience a time consuming process finding both open source software and proper chaining policies. According to our observation, most NFs used in papers are different open source implementations linked in different kinds of NF chain policies. So that, there is not yet a general baseline for measurement and comparison. Furthermore, the workload generator and network traffic trace used are also different, and real-world traces need to be provided to test different NF chains.

We also did a survey among top conference researchers who have experiences setting up NFV environment. In our survey, we found that the deploying time varies much due to the scale. In average, it takes around 1 month to build up a NF cluster having less than 10 VM instances. But when the scale of instances increase to over 50, the build up process can take 3-4 months or more. One of our respondent said that they were still keeping on iterating and improving their testbed constantly.

There are more complains pointing out their pain points: 1) Automate the setting up and testing process. 2) Configure and stabilize NFs (?). 3) Write rules to set up topology and enforce flow control.

In this paper, we develop a suite of NF benchmarks, which is supposed to have the following characteristics: 1) Representing typical NFs 2) Easy to use 3) Plenty metrics for measurement.

According to our research and observation, the representative of an NFV benchmark should be satisfied in three aspects:

1) Representative NFs. For the demand of representative NFs, we referred to the NFV Introductory White Paper [], which defined ten scenario of NFV use cases. In the first version of our benchmark, the open source implementation of NFs we collected covers half of the ten use cases. Table **??** lists the basic information of NFs used in our benchmark. **2) Representative NF chains.** However, not only single NFs should be typical, but also the NF chains. We referred to ETSI standard documents of SFC (Service Function Chaining) [**?**] for the typical use case of service chains in the scenarios of both enterprise user and datacenter. We also consulted our industrial partners for real world chaining policies. The typical NF chains our benchmark provides are listed in Table **??**. **3) Proper workload generator.** Since each NF chain serves at different network level, only one workload generator is not enough to test all the scenarios. So we select different clients for each chain.

The goal of the 'easy to use' design is to achieve one-week setting up as well as one-click test running, no matter the scale of the testing environment. To finish a test, users only need to touch one configuration file and execute one single command. In the end, the measurement report will be output to a file. To implement our design, we leverage Docker and Kubernets to pack NFs in docker, manage the images, and do allocation automatically. We use OVS to do switching and packet force forwarding. Pre-written scripts and Openflow rules are written to implement chaining and flow control.

In our benchmark, we provide the most concerned metrics for measurement, that is latency and throughput. We measure latency in the granularity of per-packet and per-NF, and output cumulative distribution of latency in the measurement report.

The main contribution of our work is

2. BACKGROUND AND MOTIVATION

The idea of our work is based on an observation that

2.1 Network Function Virtualization

2.2 A Survey of NFV Testing Environment Deploying

To further prove our observation, we did a survey among top conference authors dedication in the field of NFV. We delivered the survey to sixteen people who are the first authors of papers published on top conferences and we finally collected eight responses.

The question we concerned most is the time they spent on deploying a NFV testing environment. To precisely measure the labor time, we use the metric of man-month which indicates the number of months used if the work is done by one person. We also asked for the scale of physical servers and VM instances, as well as the virtualization technology they used. The result is shown in Table 1.

	Man-Month Used	# of Servers	# of VMs	Virtualization Technology
1	<0.5	4	11-20	KVM
2	0.5-1	2	1-5	Hyper-V
3	0.5-1	4	1-7	Container
4	0.5-1	4	6-10	KVM
5	1-2	4	6-10	KVM, Container and other
6	4	10	100	KVM
7	6	24	72	KVM
8	6	4	1-5	Xen

Table 1: The result of our survey reflects the relationship between labor consuming and the scale of the testing environment.

From the responses, we can see the average time to deploy a testing environment for NFV is around 1-2 months. It can be quite fast for a experienced person, as the responser No.1, who used less than half a month setting up a cluster including 4 physical servers and 11-20 VMs. However, for the others, the deploying process can be time taking and painful.

> We wrote a bunch of scripts that simplified most of the process. The primary trouble was in figuring out how to determine that NFs were done starting up and ready to forward packets.
>
> Automating the process of setting up the testbed end to end took us a lot of time and many iterations.

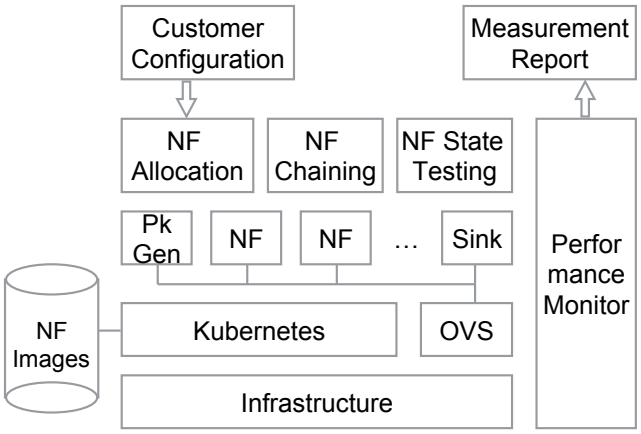

Figure 1: An overview of our benchmark design.

> Two parts for me: 1. Install, configure and stabilize the open-source software, especially under heavy workload. 2. Figure out appropriate workload to test different types of VNFs. It would be great if there is some real-world traffic/workload traces.

Traffic generation and topology configuration

Setting up the datapath (interfaces, DPDK, routing tables, etc)

If there is a system that will take as input key parameters (e.g., number of nodes, topology, VM images, choice of hypervisor) and then automatically generates a ready to go set up, that will be of a huge value!

Installing proper rules in OpenFlow-enabled switches to enforce chaining

The setup significantly evolved overtime and became much easier to manage when switching from a home-brewed Xen setup to OpenStack

3. BENCHMARK DESCRIPTION

3.1 Overview

Representative NF K8s+ovs measurement Since NFV involves multi-layers in the whole software stack, it is insufficient only providing several software implementations.

3.2 Design

Connection test

3.3 NF

We select NFs which needs no modification of the kernel. Most promising NFV use cases.

[**?**, **?**]

4. IMPLEMENTATION

5. EVALUATION