논문 19-44-12-18 The Journal of Korean Institute of Communications and Information Sciences '19-12 Vol.44 No.12
https://doi.org/10.7840/kics.2019.44.12.2303

SFC 테스트 설계를 위한 YAML 기반 프로파일러

응웬쭝당찐[◆], 유 명 식[°]

A YAML-Based Profiler for Designing SFC Tests

Trinh Nguyen[◆], Myungsik Yoo[°]

요 약

네트워크 기능은 네트워크 기능 가상화 (NFV)를 통해 새로운 차원으로 발전하고 있다. 가상화 된 인프라를 활용함으로써 NFV는 보다 유연하고 확장 가능한 방식으로 서비스를 제공할 수 있다. 많은 장점에도 불구하고, NFV는 서비스 지연, 테스트, 특히 SFC 테스트 등과 같은 많은 문제들이 남아있다. 또한 현재의 연구는 SFC 테스트 또는 테스트 케이스 설계를 위한 맞춤형 방법에 대한 부분이 많이 부족한 실정이다. 따라서 본 논문에서는 SFC 테스트를 설계를 위한 YAML 기반 프로파일러를 제안한다.

Key Words : NFV, SFC, Testing, YAML, Test Profiler

ABSTRACT

Network operations have been evolving to a next level with Network Function Virutualisation (NFV). By leveraging virtualized infrastructure, NFV provides more flexible and extensible ways to deploy services. Despites the great benefits, NFV also brings many challenges such as service latency, testing, especially SFC testing, etc. Moreover, current research does not focus on testing SFC and lacks of a customizable method to design the test cases. Hence, this article proposes a YAML-based profiler for designing the SFC tests.

Ⅰ. Introduction

Virtualisation technologies have been matured to a level that they are adopted in most of today infrastructure. Hence, by building the network services based on those virtual resources, NFV created a new paradigm of orchestration and life-cycle management of the value-added features. It also reduces capital expenditures (CAPEX) and operating expenses (OPEX) by replacing vendor-specific middle-boxes with Virtual Network Functions (VNFs) which are software. In traditional networks, SFCs are chains of network functions that are directly linked together to provide a service. In case of NFV, SFCs are logically linked the VNFs so that it can create more sophisticated service to the end users. Figure 1 describes the core elements of NFV that consists of a Network Function Virtualisation Orchestrator (NFVO), a VNF Manager (VNFM) to manage the life-cycle of the VNFs, and

※ This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2018-2017-0-01633) supervised by the IITP(Institute for Information & communications Technology Promotion)

◆ First Author : Trinh Nguyen, Department of ICMC Convergence Technology, Soongsil University, Seoul, Republic of Korea, dangtrinhnt@soongsil.ac.kr, 학생회원
° Corresponding Author : Myungsik Yoo, School of Electronic Engineering, Soongsil University, Seoul, Republic of Korea, myoo@ssu.ac.kr, 종신회원

논문번호 : 201910-223-D-RN, Received October 4, 2019; Revised October 11, 2019; Accepted October 11, 2019

a Virtual Infrastructure Manager (VIM) that manages the virtual infrastructure. Besides all of those benefits, NFV also brings us challenges because of the complexity of its virtualisation layers as well as the softwarization of the network functions. Therefore, NFV has to evolve with a new way of testing its components (e.g., SFCs, VNFs, etc.) in order to create a dependable system.

Unlike traditional network, SFCs testing in NFV require not only testing the functionalities but also verifying the softwarized components. That increases the complexity and effort for SFC testing in NFV especially in the designing test case phrase of the test procedure. In addition, state-of-the-art testing methods do not focus on how to design the test cases. Hence, there is a need for a more sophisticated tool to allow us to build the test case. In this article, we leverage the power of the YAML language to design such tool and evaluate it.

The rest of this paper is organized as follows. Section II describes the background and discusses the disadvantages of existing work. Section III illustrates the overall architecture of the proposed test case design tool. Finally, section IV concludes the paper and discusses future works.

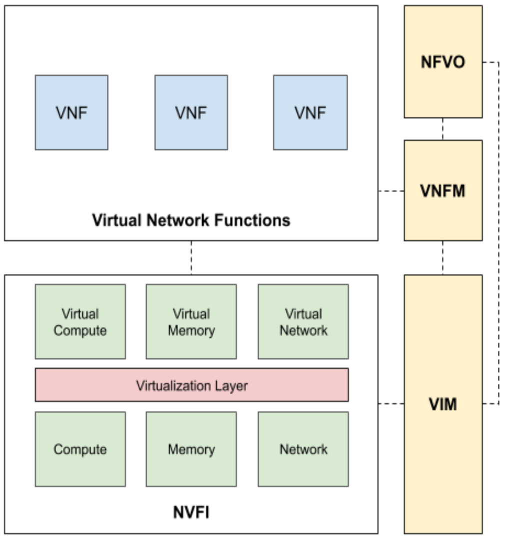


Fig. 1. Core components of the NFV architectural framework

Ⅱ. Background

With the new network paradigm of NFV, the approach of traditional network functions of single vendor evolves to a series of virtualised functions offered by many vendors running on a virtualised infrastructure. Therefore, we need to design new tools to support testing of the SFCs in NFV which requires the understanding of existing testing methods in NFV. Currently, there are two major testing methodologies that are conformance and interoperability testing.

2.1 Conformance testing

Conformance testing is the process to test a system and is not usually exposed to the users. The main targets of conformance testing is to test whether a function is working correctly. In addition, all the input and output information can be collected during this testing process. Only one System Under Test (SUT) can be processed using this testing method. Figure 2 illustrates the main concept of conformance testing.

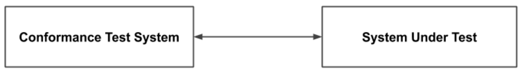


Fig. 2. Conformance testing

2.2 Interoperability testing

Different from conformance testing which targets the functionality, interoperability testing focuses on testing the service that the SUT offers users. The SUT may consist of many Functions Under Test (FUT) built by different manufacturers. Interoperability testing validates that a service with at least two features is properly functioning. The testing process of interoperability testing procedure is executed through the Application Programming Interfaces (APIs). Since this testing process requires developers to look for many elements when implementing it, there are not much effort putting into it. Therefore, there is a need to design an interoperability testing system for the NFV system to prevent failures, and the first step is finding out a way to better design

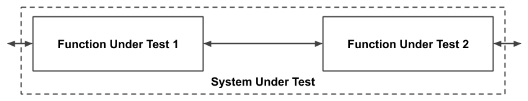


Fig. 3. Interoperability testing

sophisticated test cases. Figure 3 illustrates the main concept of interoperability testing.

2.3 YAML

Testing SFCs involves many steps such as validating the target NFV platform access privileges, deploying the test scenarios and acquiring the management address of the children VNFs, and checking if the package forwarding flows of the SFCs are working correctly, etc. Those steps require a sophisticated way to describe the test scenarios so that the testing procedure can produce meaningful results for network operators. Hence, we chose YAML[15] as the foundation for the testing profiler to design the test cases. Figure 4 shows a sample of YAML structure.

YAML which stands for YAML Ain't Markup Language is a human-readable data-oriented language structure used as the input format for different kind of software. YAML is not a markup language used for document markup such as XML, HTML, etc. The users (e.g., network operators, system administrators, etc.) define the data in a YAML file which then read by the applications. For instance, YAML is used to specify authentication credentials or deploy software packages on the systems. There are some well-known systems that are using YAML such as Kubernetes for container orchestration, Ansible or SaltStack for configuration management, and OpenStack Tacker, an NFV

```
item1:
  key1: value1
  key2: value2
item2:
  - sub_item1
  - sub_item2
...
```

Fig. 4. Sample YAML structure

orchestration and VNF manager engine to launch VNFs on OpenStack infrastructure.

There are several works have been proposed for NFV testing but mostly focuses on conformance testing and do not leverage the power of YAML language to design the test cases. The GYM framework in [5] that provides an end-to-end system for testing the VNFs can only test the VNF functionalities due to lack of a flexible test profiler. The 5TANGO project[6] offers a way to test the NFV system but does not support testing the orchestration and SFCs and test cases are not easy to built. SFCPerf[7] creates repeatability for examining many virtual network functions using both active and passive data collection for analysis. Despite of having such advantages, SFCPerf has some disadvantages such as the management module features overlap with the NFV MANO, active data collection service is only deployed at the end points, and the test cases cannot be designed flexibly. Moreover, the work in [8]-[12] are open source projects that aim mainly on performance and functional testing of the NFV components and do not mention or focus on how to construct the test cases. Therefore, a sophisticated test case profiler is necessary to enhance existing solution or should be supported in new testing frameworks. This article propose a YAML-based profiler to support testing of complicated SFCs. Detail architecture of the proposed system will be discussed in the next section.

Ⅲ. The Yaml-Based Profiler for Designing Sfc Tests

3.1 Scope of the proposed mechanism

Solving the disadvantages of the existing work, the proposed system is trying to offers NFV developers the ability to design the test cases with different complexity levels of data structure in NFV. The scope of proposed test profiler are aiming at providing a tool to design SFC test cases which are to give sophisticated and executable instructions as output to check SFCs.

3.2 Test profiles

The Test Profiler provides the ability to design the test cases in form of profiles. The profiles are written using a descriptive language called YAML. The test profile can be defined as illustrated in Figure 5. A test profile should have these fields:

test_profile_version: this field indicates the version of the test profile definitions.

description: this field describes the test profile to give network operator what this test is about.

```
test_profile_version: version_of_the_test_profile_definition

description: Description of this test profile

system_under_test:
  name: name of the target system under test
  auth_url: authentication url
  username: admin username
  password: admin password

services_under_test:
  - name_of_network_service_under_test_1
  - name_of_network_service_under_test_2

test_criteria:
  - test_rule_1: test_rule_1_value
  - test_rule_2: test_rule_2_value
```

Fig. 5. A sample YAML test profile

system_under_test: this section tells us what the target system Tesuto will test. It includes the authentication information of the target system such as: name, authentication URL, username, and password.

services_under_test: this field shows the list of network services we need to test.

test_criteria: a list of test criteria and expected values.

3.3 Architecture

As shown in Figure 6, the proposed Profiler includes three components which is described below.

YAML Parser: the parser parses the input YAML file which describes the test profile.

Profile Translator: the translator translates and validates the input data from the YAML file and transform them into test profile objects or test instructions that can be used by the Test Controller to execute the actual tests. The Test Controller can be implemented with an interface that can take the output of the proposed YAML-based Profiler as input.

The Test Controller is out of this article's scope.

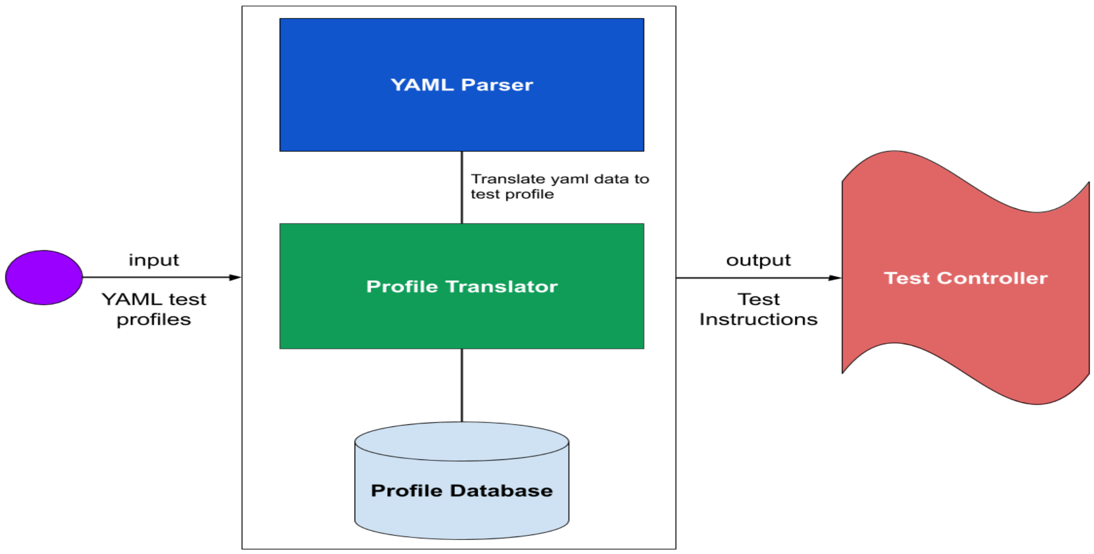


Fig. 6. Architecture of the proposed YAML-based profiler for SFC tests

Ⅳ. Experiment

An implementation is made to evaluate the proposed YAML-based test profiler in term of feasibility. OpenStack Tacker[13] was used as the NFV MANO under test and other services are deployed using Devstack[14]. A regular computer was used to instantiate the test scenario as illustrated in Figure 7. The test scenario is a single SFC that consists of two VNFs, a firewall and a Content Filtering VNF. The target point of the service under test is a HTTP Server. The SFC under test does not allow traffic going through port 8080 (VNF1) and blocks access to the "/restricted" URI (VNF2) of the HTTP Server. This experiment proves that the proposed test

profiler can offer appropriate instructions for the test controller to execute the test cases.

Table 1 is the configuration used for the experiment.

We designed the test case simply by creating a YAML file with the information such as authentication information (i.e., system_under_test), the criteria to test (i.e., test_criteria) as shown in Figure 8.

A simple test controller was implemented to show

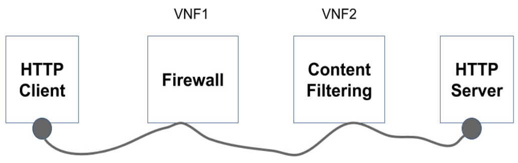
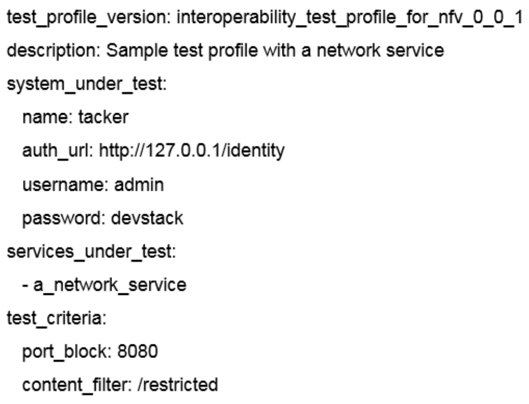


Fig. 7. The test scenario

Table 1. Specifications of the Experiment

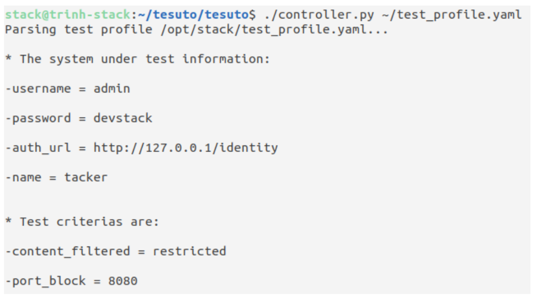
Item	Configuration
CPU	Intel (R) Xeon (R) CPU E3-1240 V3 @ 3.40GHz, 8 cores
Memory	16GB DDR3, 1333 MT/s
Operating System	Ubuntu 18.04 LTS
Software	OpenStack Tacker, Devstack, Ubuntu, Python 2.7



```
test_profile_version: interoperability_test_profile_for_nfv_0_0_1
description: Sample test profile with a network service
system_under_test:
  name: tacker
  auth_url: http://127.0.0.1/identity
  username: admin
  password: devstack
services_under_test:
  - a_network_service
test_criteria:
  port_block: 8080
  content_filter: /restricted
```

Fig. 8. The test profile used in the experiment.

that with the proposed YAML-based profiler, the information can be fetch in to the test process. Figure 9 displays the results when the controller reads the input from the proposed profiler.



```
stack@trinh-stack:~/tesuto/tesuto$ ./controller.py ~/test_profile.yaml
Parsing test profile /opt/stack/test_profile.yaml...
* The system under test information:
-username = admin
-password = devstack
-auth_url = http://127.0.0.1/identity
-name = tacker

* Test criterias are:
-content_filtered = restricted
-port_block = 8080
```

Fig. 9. Results of the experiment

Ⅴ. Conclusion and Future Work

Even though testing in NFV is a mature topic, state of the art works only focus on conformance testing which is only cover performance and functionality of the VNFs, and lack of a sophisticated way for network developers to design the test cases efficiently. This article by reviewing some of the related work in NFV testing methodologies proposes an YAML-based Test Profiler for SFCs. Future work will focus on enhancing the Profiler's features so that it can support more complicated test cases for SFC testing.

References

[1] ETSI, Network Function Virtualisation (NFV); *Architectural Framework.* [Online], Available: https://www.etsi.org/deliver/etsi_eg/202100_202199/202107/01.01.01_60/eg_202107v010101p.pdf

[2] International Telecommunication Union, *X.290 : OSI conformance testing methodology and framework for protocol Recommendations for ITU-T applications.* [Online], Available: https://www.itu.int/rec/T-REC-X.290/en

[3] ETSI, Methods for Testing and Specification (MTS); *Planning for validation and testing in the standards-making process.* [Online], Available: https://www.etsi.org/deliver/etsi_eg/202100_202199/202107/01.01.01_60/eg_202107v010101p.pdf

[4] ETSI, Network Functions Virtualisation (NFV); Testing Methodology; *Report on NFV Interoperability Testing Methodology*. [Online], Available: https://www.etsi.org/deliver/etsi_gs/NFV-TST/001_099/002/01.01.01_60/gs_nfv-tst002v010101p.pdf

[5] R. V. Rosa, C. Bertoldo, and C. E. Rothenberg, "Take your VNF to the Gym: A testing framework for automated NFV performance benchmarking," *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 110-117, 2017.

[6] P. Twamley, et al., "5GTANGO: An approach for testing NFV deployments," *IEEE 2018 EuCNC*, Ljubljana, Slovenia, Jun. 2018.

[7] I. J. Sanz, Diogo M. F. Mattos, and Otto Carlos M. B. Duarte, "SFCPerf: An automatic performance evaluation framework for service function chaining," *NOMS 2018-2018 IEEE/IFIP Netw. Operations and Management Symp.*, Taipei, Taiwan, Apr. 2018.

[8] OPNFV Yardstick, [ONLINE], Available: https://wiki.opnfv.org/display/yardstick/Yardstick

[9] OPNFV QTIP, [ONLINE], Available: https://wiki.opnfv.org/display/qtip/Platform+Performance+Benchmarking

[10] OPNFV Bottlenecks, [ONLINE], Available: https://wiki.opnfv.org/display/bottlenecks/Bottlenecks

[11] OpenStack RefStack, [ONLINE], Available: https://refstack.openstack.org/

[12] OpenStack Tempest, [ONLINE], Available: https://docs.openstack.org/tempest/latest/

[13] OpenStack Tacker, [ONLINE], Available: https://docs.openstack.org/tacker/latest/

[14] OpenStack Devstack, [ONLINE], Available: https://docs.openstack.org/devstack/latest/

[15] YAML, [ONLINE], Available: https://yaml.org/

응웬쫑당찐 (Trinh Nguyen)

Trinh Nguyen received his B.Eng. degree in Computer Networking from University of Information Technology, VNU-HCM, Ho Chi Minh City, Vietnam, in 2012. He has been pursuing the Master's degree in ICT at Soongsil University since Fall 2017. His research interests include Software-Defined Networking, Network Function Virtualization and Cloud Computing.

유 명 식 (Myungsik Yoo)

Myungsik Yoo received his B.S. and M.S. degrees in electrical engineering from Korea University, Seoul, Republic of Korea, in 1989 and 1991, and his Ph.D. in electrical engineering from State University of New York at Buffalo, New York, USA in 2000. He was a senior research engineer at Nokia Research Center, Burlington, Massachusetts. He is currently a professor in the school of electronic engineering, Soongsil University, Seoul, Republic of Korea. His research interests include visible light communications, sensor networks, Internet protocols, control, and management issues.