This is a preprint. The final article is published at: M. Peuster et al., "Introducing Automated Verification and Validation for Virtualized Network Functions and Services," in IEEE Communications Magazine, vol. 57, no. 5, pp. 96-102, May 2019. doi: 10.1109/MCOM.2019.1800873

Introducing Automated Verification and Validation for Virtualized Network Functions and Services

Manuel Peuster*, Stefan Schneider*, Mengxuan Zhao†, George Xilouris‡, Panagiotis Trakadas§, Felipe Vicens¶, Wouter Tavernier‖, Thomas Soenen‖, Ricard Vilalta**, George Andreou‡‡, Dimosthenis Kyriazis††, and Holger Karl* *Paderborn University (manuel.peuster@upb.de), †Easy Global Market, ‡NCSRD Demokritos, §Synelixis Solutions, ¶ATOS, ‖University of Ghent (imec), **CTTC, ††University of Piraeus, ‡‡Huawei

***Abstract*—Network function virtualization (NFV) and software defined networks (SDN) will transform network management and operation tasks into agile development tasks and software artefacts which are managed and deployed as composite services using DevOps principles. Those softwarised networks rely on complex technology stacks, starting with low-level virtualization technologies and ranging up to machine learning-based orchestration solutions. One of the main challenges in those environments is to verify that the deployed functions and services operate correctly and meet the quality goals, set by the stakeholders, *before* they are put to production.**

We tackle this challenge by introducing the novel concept of a verification and validation (V&V) platform for NFV, which enables automatic testing and qualification of single network functions and complex services. By adding such a platform to the NFV ecosystem, new business models emerge as we discuss in this article. We evaluate our proposed concepts by presenting a case study that uses our open-source V&V platform to test a real-world network service.

I. Introduction

The upcoming 5th generation of networks (5G) is expected to be the backbone and enabler of many innovative services, ranging from those that require ultra-low latency to those with ultra-high bandwidth demands. Examples are the emerging *vertical use cases for 5G*, such as smart manufacturing (industry 4.0), immersive media, connected vehicles, or public protection and disaster relief, which cannot be efficiently implemented in legacy, general-purpose networks [1]. To tackle this, technologies like software-defined networks (SDN) and network function virtualization (NFV) are emerging and will allow to apply agile methods and DevOps concepts to the networking domain [2].

The latter introduces even more complexity into future softwarised network scenarios, raising a series of questions about quality control and availability assurance. First, how to verify that all involved components of the technology stack, and especially the virtualized network functions (VNF), work correctly? Second, how to validate that complex service function chains (SFC), consisting of multiple, chained VNFs, correctly implement the intended service? Third, how to dimension virtualized resources to meet quality of service (QoS) goals? And finally, how to know about the aforementioned characteristics before a single VNF or service is deployed to production?

In this article, we present solutions for those questions by introducing and applying verification and validation (V&V) methods, in form of our novel *V&V platform*, to the networking and especially the NFV domain.

The contributions of this article are as follows: First, we analyse an NFV scenario and identify how and where V&V concepts can be applied (Sec. II), before giving an overview about state-of-the-art solutions in Sec. III. After that, we introduce the novel concept of a *V&V platform for NFV ecosystems* in Sec. IV. Finally, we present a case study, using our open-source prototype platform, called *5GTANGO* [3], highlighting the usefulness of our approach by providing test results of a real-world NFV service in Sec. V.

II. Verification & Validation for NFV

In the software engineering community, *verification & validation (V&V)* is a way to determine whether a software product operates correctly and meets all predefined requirements [4]. V&V concepts were introduced in the early 80s and have evolved to modern software development practices, including automated testing and continuous integration principles. Practically, V&V processes are an inherent part of all modern, agile development processes and usually appear as part of a fully automated continuous integration (CI) pipeline, for example, as code style checks, unit tests, smoke tests, regression tests, integration tests, test coverage analysis and many more.

The application of those V&V concepts to the NFV domain promises to reduce the time-to-market for NFV services even further and improve the reliability, interoperability, and quality of softwarised network solutions. V&V mechanisms are even more important for the emerging vertical use cases that have their very own, strict, perhaps contradicting requirements. Approaches that allow to verify and validate wether a certain VNF or service (strictly) meets its requirements, if deployed in a given environment, will be a key enabler for wide adoption of softwarised 5G technologies and agile service deployments [1].

This leads to the question how automated V&V concepts can be applied to the NFV domain and how our networks can benefit from them? To answer this question, we take a closer look at a typical NFV scenario shown in Fig. 1. It shows a network service, consisting of four VNFs that are chained together to form an SFC. Each VNF is a virtualized entity (e.g., virtual machine or container) that is executed on top of a given (maybe distributed) NFV infrastructure (NFVI). The

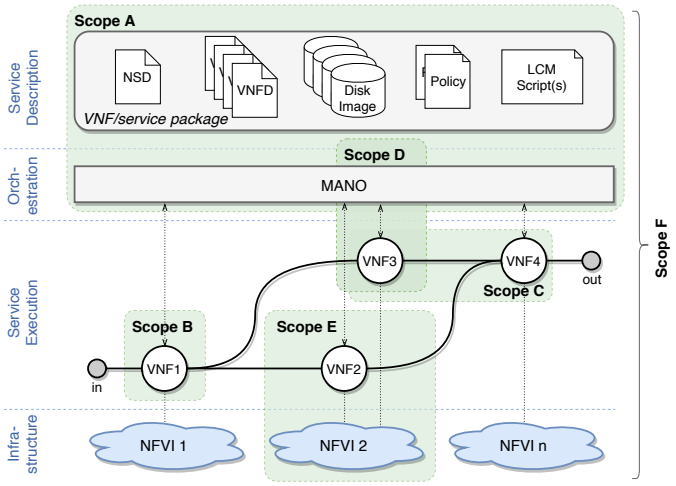

Fig. 1: Typical NFV scenario. The dashed boxes highlight the different scopes in which V&V concepts and methods should be applied, as further detailed in Table I.

complete deployment is under control of the management and orchestration (MANO) system that requests the instantiation of the virtualized entities, controls the lifecycle of the VNFs, and manages their respective SFC. On top of the MANO layer, the VNF, service, and lifecycle management (LCM) descriptions are shown, which specify how a MANO system should deploy and manage a certain service and its VNFs.

Fig. 1 shows that NFV scenarios are different from typical software projects because they do not only contain the actual application code that implements the VNF's packet processing capabilities, but also many additional artefacts, like descriptors for VNFs and services, LCM scripts, or disk images for different NFVI technologies. All of them are key artefacts for the proper operation of a service and thus need to be tested. We highlight this by defining different *scopes* in the shown NFV scenario and map each scope to testing approaches in Table I to show where the presented V&V concepts are applied. Each of these scopes deals with components within the scope itself; external reference points, e.g., lines that leave a scope, are not considered. A V&V solution for NFV should support all of these scopes and needs to fit into existing NFV ecosystems, as we discuss in Sec. IV.

III. State of the Art

Recently, verification and validation of VNFs and SFCs has started to attract the research community's attention. In [5], authors define a domain-specific language, named NetKAT, for specifying and verifying network packet-processing functions based on packet header modifications and network topology encoding. NetKAT is based on the formalism of Kleene algebra with tests defining a generic algebraic system for reasoning about partial correctness. A similar approach is followed in [6], providing formal verification processes of forwarding graphs within networks. Those approaches can be considered as specific test definition approaches and could be added to our proposed V&V platform.

With respect to standardization efforts, ETSI specifies first guidelines for pre-deployment testing [7], mainly targeting performance assessment of the NFVI and its ability to fulfill performance and reliability requirements of certain VNFs and services as well as the testing of the integration between VNF, NFVI, and MANO. Additionally, work in IETF [8] defines a list of service verifications but focusses mainly on the verification of SFC forwarding graphs, e.g., loop detection. Our work takes the standardized testing methodologies into consideration and focuses more on their practical and architectural implementation in the NFV ecosystem.

Further, there are a few NFV testing solutions. In [9], the authors argue the necessity of developing a framework that will allow VNF performance validation and benchmarking and demonstrate the benefits of the developed NFV-VITAL framework by analysing three open-source VNF implementations. A more recent work [10] checks the consistency of the VNF description on real deployments, introducing the concept of augmented network topology, in conformance with the principles of the NetKAT formalism. Another recent work [11] introduces Gym, a framework that allows automated performance benchmarks of NFV artefacts. Gym defines a minimum set of standardized interfaces while allowing user-defined tests along a catalogue of reusable VNF testing procedures and reports multiple system configuration descriptors and workload parameters. Finally, the work in [12] focuses on end-to-end performance tests for complete services, arguing that testing the performance of single VNFs in isolation does not yield representative results. Even though those solutions already offer a couple of usable implementations, most of them focus on performance validation only but do not cover every scope to be tested in NFV scenarios as we show in this article. They can be considered as one part of our V&V concepts and thus can be integrated with our work.

IV. A V&V-enabled NFV ecosystem

Bringing V&V concepts to the NFV domain requires some extensions of the NFV ecosystem, e.g., to add testing facilities or to make existing components aware of test results. To do so, we introduce the novel concept of adding a *V&V platform* to the NFV ecosystem that offers the service of verifying and validating VNFs or services against a pre-defined or custom set of tests. To do so, a V&V platform uses a test infrastructure which is similar to the production environment and may be based on different NFVIs controlled by different virtualized infrastructure managers (VIM) and MANO solutions. Having this, a customer of the V&V platform can submit VNFs and services to the platform, the platform verifies and validates them against different test cases in a variety of environments, and finally returns the test results to the customer. An example for this is an intrusion detection system VNF that is submitted in one or multiple compliant VNF packages and then tested against different environments, e.g., using OpenStack, Kubernets, or AWS as infrastructure controlled by different MANO systems, e.g., 5GTANGO, OSM, and ONAP.

This novel V&V platform is operated by a *V&V provider*, tightly integrates into the NFV ecosystem, and adds new

TABLE I: Mapping of V&V concepts to different scopes as shown in Fig. 1.

Scopes in Fig. 1	static analysis	model checking	unit tests	integration tests	smoke tests	system tests	performance tests	security tests	compliancy tests	stability tests
Scope A: Validating VNF, service, and management descriptions and their semantics against given schemas and data models.	•	•							•	
Scope B: Testing the packet processing software within a single VNF, e.g., IDS rules.			•		•		•	•		
Scope C: Verifying the interoperability of multiple, chained VNFs, e.g., if the output traffic of the first VNF can be processed by the second.				•			•	•		
Scope D: Ensuring compatibility between MANO systems and managed VNFs, e.g., configuration, management, and monitoring interfaces.				•	•			•	•	
Scope E: Testing compatibility between VNF and the NFVI that executes it. Also, examining the performance of a VNF or service on a given NFVI.							•	•	•	
Scope F: Checking the complete end-to-end deployment of a service with a given NFVI and MANO system.					•	•	•	•		•

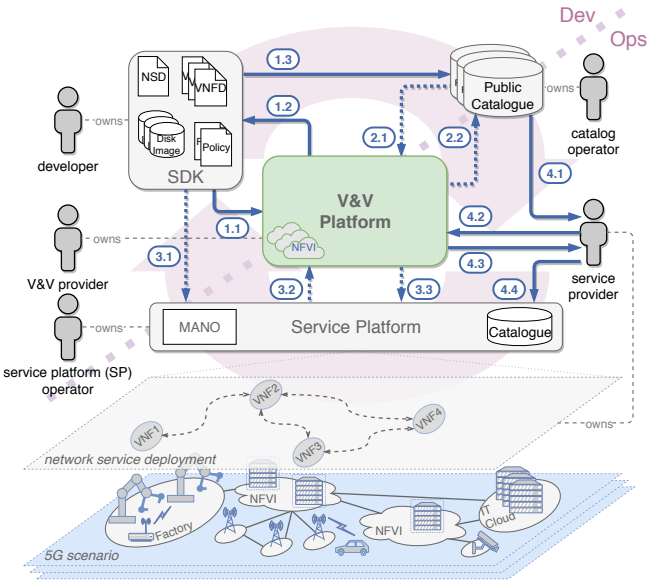

Fig. 2: End-to-end NFV scenario with our proposed V&V platform showing the involved roles and workflows

roles and business models to it, as we show in Fig. 2. This starts with the *VNF and service developer* who uses NFV-enabled service development kits (SDK) [13] to create new VNFs and services. These new VNFs and services might then be uploaded to public catalogues owned by *catalogue operators* to share them with their potential customers, the *service providers*. The *service providers* pick up existing VNFs and services and deploy them into production using service platforms (SP) operated by *SP operators*.

In the *developer-centered business model*, we consider the case where the *developer* submits the developed artefacts to the V&V platform (1.1) to have them tested before they are shared (1.3). This also gives early feedback about the compatibility of the developed artefacts to a variety of environments, which the *developer* might not be able to test on her own, e.g., in her lab (1.2). In the *catalogue-centered business model*, the *catalogue operator* is, in turn, interested in verifying and validating all artefacts uploaded to the corresponding catalogues. This can be done by first sending the uploaded artefacts to the V&V platform (2.1) and only storing them in the catalogues if all tests have passed (2.2), e.g., to ensure that no malicious VNFs or services enter the catalogues. The *service platform-centered business model* is similar to the second one, but here the *SP operator* has an interest in getting artefacts verified and validated before they are on-boarded to the service platform (3.1). In this model, the pre-testing of artefacts (3.2 and 3.3) mitigates the risk of on-boarding incompatible or broken VNFs and services to a production service platform. Finally, the *service provider* has an interest in using the V&V platform to test third-party VNFs and services, being the fourth model, called *service provider-centered business model*. The *service provider* browses the available catalogues and selects the building blocks for his services (4.1). Even though the catalogues might already offer pre-tested VNFs and services, the *service provider* might still be interested in running those third-party artefacts against his own set of tests. He can do this by uploading those artefacts and his custom tests to the V&V platform (4.2), which verifies and validates them and sends them back (4.3). Finally, the *service provider* can decide if those artefacts fulfill his requirements and put them to production (4.4).

The presented V&V platform solution has the benefit that not every party needs to setup own testing infrastructure, which is costly and often not feasible. For example, most VNF and service developers do not have different NFVIs and MANO solutions available. *SP operators* and *catalogue operators*, in contrast, do not want to test new artefacts in their existing production infrastructure. A V&V platform allows them to outsource these tasks and save resources by using the V&V platform's test resources on-demand.

Besides the potential resource savings, the time required to put new VNFs and services into production can be reduced as well. One reason for this is that *service providers* will know about the compatibility of the deployed artefacts beforehand and time-consuming bug-fixing tasks on freshly deployed

services will be reduced. But more importantly, less re-testing and test repetitions are required if we assume that all roles in the described scenario trust the *V&V provider*. This is because verified and validated artefacts are annotated with the verification and validation results, all signed by the *V&V provider*. Then, every other role in the system can check the integrity of the the existing results and reuse them without requiring new test runs. For example, a *developer* gets a VNF back from the V&V platform which already attest that the VNF runs smoothly on-top of OpenStack and Amazon AWS. This VNF and its test results can then be uploaded to a catalogue and the *catalogue operator* can trust the signed test results and skip those tests in his own verification and validation phase.

The presented V&V platform concept can be seamlessly integrated into today's NFV ecosystems in which SDKs, catalogues, service platforms, and NFV infrastructure are already present. Generated test results are added as additional metadata to the VNFs and services exchanged between those entities.

V. Case Study: The 5GTANGO approach

We designed and implemented an open-source V&V platform prototype as part of the 5GTANGO NFV framework [3] to evaluate the feasibility of the presented concepts. We use it to perform a case study in which we test a virtualized content delivery network service (vCDN) to give the reader detailed insights in how a V&V platform works.

A. Building a V&V platform

Fig. 3 shows the internal architecture of the 5GTANGO V&V platform and its surrounding building blocks. It consists of the following main components that enable a fully automated V&V workflow: (i) The *V&V gateway*, exposing APIs towards the V&V platform users, allowing them to submit packages for verification and validation; (ii) the *test invoker*, responsible for the test case configuration, scheduling, and maintenance of the test state; (iii) the *V&V catalogues* holding the artefacts to be tested, e.g., VNFs and network services; multiple repositories, i.e. (iv) the *test repository*, the (v) *test result repository*, are used to store tests, test results, as well as raw monitoring metrics collected during the tests; (vi) the *test engine* responsible to control the execution of tests in the test queue using an extensible set of test plugins. The V&V platform uses the concept of plug-able *test platform drivers* to abstract and unify the interface towards the *test execution platforms* (vii) on which the VNFs or services under test (SUT) are deployed and the tests are actually executed. Finally, there is a set of *test analysis tools* (viii) to process the resulting test data.

B. V&V platform workflow

We now describe the V&V platform's workflow, starting with test definition, followed by automated test management and execution, and finally discussing result collection and management.

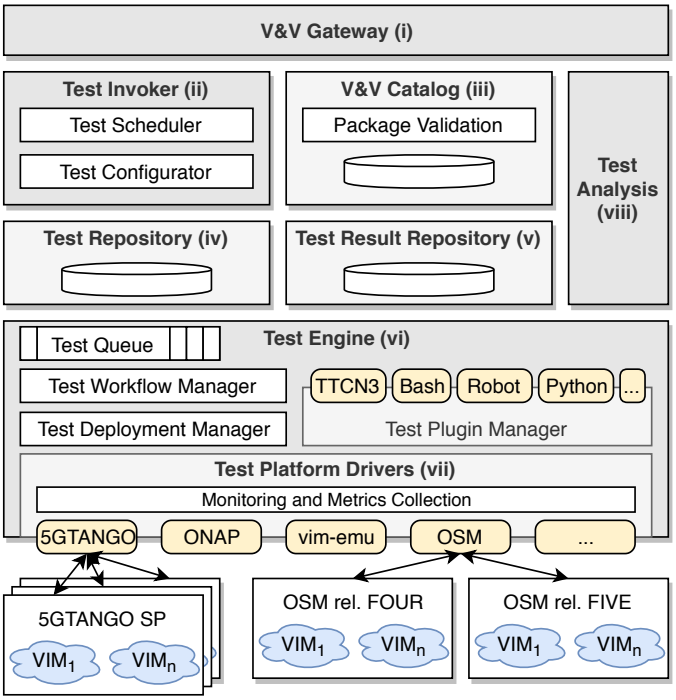

Fig. 3: 5GTANGO V&V platform architecture with several connected test execution platforms

1) Test definition and implementation: Tests may be single test cases or a more complex battery of tests, i.e., a test suite. They can either be pre-uploaded to the V&V platform as standalone tests or uploaded side-by-side with a VNF or a network service. The latter enables tests that are custom-tailored to a specific VNF or network service supporting the business models defined in Sec. IV.

A special challenge is the definition and implementation of tests to be executed on a single or multiple V&V platforms. Not to tie our implementation to any specific test definition approach, the 5GTANGO V&V platform offers a *test plugin* system as part of its *test engine*. The plugin mechanism utilizes container technology, i.e., Docker, to allow packaging and integration of new test plugins, ranging from simple, script-based tests (e.g., Bash, Python) to more advanced testing technologies, like TTCN-3 or the Robot test automation framework. The latter is used by ETSI to build tests for interface specifications and can be re-used to test compliance to ETSI standards. This extensible design allows to always pick a suitable technology to build tests for all scopes described in Table I.

Each test implementation is accompanied by a *test descriptor* defining against which types of VNFs and services a test can be executed and which environments are needed. Such a *test descriptor* can be compared to a *Jenkinsfile*, known from general purpose CI systems. We aligned the test descriptors with the ETSI data models for VNF and service specification and published them along with our open-source prototype [3].

2) Test management and execution: When a VNF or network service is uploaded, the V&V platform needs to decide which of the tests, either from the set of already available tests or from the tests uploaded with the VNF or service, should

be executed. To automate this decision, we added a tagging system to our test descriptors as well as to the descriptors of VNFs and network services, which allows to flexibly categorize tests. We start with high-level test categories, like functional and performance tests; more detailed categories based on the scopes defined in Table I down to detailed test categories, like latency tests, TCP throughput tests, and so on. Using the tagging approach, developers can also specify on which target environments a test should be executed, e.g., a network service should be tested on 5GTANGO 4.1, OSM rel. FOUR and OSM rel. FIVE.

VNFs or services uploaded to the V&V platform are automatically matched against those tags, e.g., a firewall VNF could indicate that it can be tested using end-to-end throughput tests using arbitrary layer 2 traffic. Alternatively, customers of the V&V platform may manually select the set of tests to be executed. All test execution requests are then queued in the test engine and executed once the required testing resources become available. While first in, first out (FIFO) queuing may often be sufficient, more sophisticated queuing mechanisms are easy to realise (e.g., earliest deadline first or prioritizing tests for premium users). This creates new research opportunities for the NFV community, since it is desirable that a V&V platform optimally utilizes the connected test execution infrastructure while ensuring that deadlines are met and test executions are properly isolated.

To finally execute the tests, the test engine forwards the VNF or network service to be tested (the SUT) to the target test platforms. This is done through an intermediate *testing platform driver* layer which abstracts and unifies the access to different kinds of test platforms, e.g., a 5GTANGO service platform, OSM, ONAP, or emulated test environments, like vim-emu. Each of these test platforms offers a particular configuration, e.g., different connected NFVIs, such as OpenStack, Kubernets, Amazon AWS, or even specific hardware accelerators, all known by the test engine. The test engine then instructs a selected test platform to deploy the SUT and may add additional test probes to the deployed service, e.g., traffic generators, to stimulate the SUT. Once the SUT is up and configured, the tests are executed. When all tests are done and the results are stored in the test result repository, the SUT is terminated to free resources.

3) Test result collection and management: During test execution, the test engine collects monitoring data from the test execution platforms and stores it in the test result repositories. This is done through the test platform drivers, which not only abstract the control interfaces of those platforms, but also connect to and translate from platform-specific monitoring solutions. Besides the raw monitoring data recorded during test executions, the test results produced by the tests themselves are stored in the result repositories. Those results can either be simple binary *pass*- or *fail*-like results or more complex results, like raw performance metrics. To simplify the use of those results they can also be represented by statistical information or be stored as trained machine learning models, again opening an interesting opportunity for further research: How to best represent and share NFV test results?

Furthermore, the authenticity and integrity of all test results have to be ensured when they are shared with other entities or components (Fig. 2). 5GTANGO uses *packages* as first-class artefacts to exchange test results. To support this, we introduced an advanced packaging format that goes beyond existing cloud and NFV packaging concepts, like ETSI's SOL004 [14]. 5GTANGO packages can either contain VNFs, entire network services, test definitions, test results, or a combination of these. It is also possible to reference a package from another package. All packages are immutable and always signed by the entity which created them. Using this, a V&V platform produces a test result package, referencing the VNF or service package that was tested, and sings it with the private key of the V&V provider. Any other party can then decide which packages to accept (e.g., those which are verified by a trusted V&V and/or created by a trusted developer) by checking their signatures.

C. Verifying and validating a network service

To evaluate the proposed concepts, we use the 5GTANGO V&V platform to verify and validate an example network service, a vCDN implementation, following the developer-centered business model described in Sec. IV. The used vCDN service is a multi-VNF network service with a load balancer VNF (HAproxy) and one or multiple caching proxies (Squid) interconnected to a single SFC. Both VNFs are implemented as VMs and are compatible with OpenStack-based NFVIs. We used the ETSI-compatible 5GTANGO service description format to compose this service, which can then be deployed using a 5GTANGO service platform registered as a test execution platform to the V&V.

We used three Dell RX730 servers, each with dual socket Intel(R) Xeon(R) CPU E5-2640 v4 @ 2.40GHz CPU and 128 GB memory, to install our platform and execute the presented experiments on top of an OpenStack Pike (based on OPNFV 5.0) installation. The three servers were interconnected by 10G Ethernet links connected to a Pica8 P-3297 SDN switch. On top of this infrastructure, the V&V automatically deploys the example service to be tested and terminates it once all tests have been performed. In addition to the VNFs of the service, deployed with 1 vCPU and 4 GB memory each, the V&V instructs the test execution platform to deploy two additional VMs acting as traffic source (4 vCPU, 8 GB memory) and traffic sink (16 vCPU, 8 GB memory) for the tests.

We performed three types of tests. First, a series of functional tests was performed ensuring the correct instantiation and configuration of the service. More specifically, the VNF on-boarding, the VNF instantiation and configuration, the SFC and forwarding graph setup, horizontal scaling, as well as an end-to-end traffic forwarding is tested as shown in the *V&V test report* in Fig. 4. The report shows that all tests have passed except for the scaling test, which was expected as we have intentionally used an example service that does not support horizontal scaling. These tests verify that our example service is compatible and works well on a 5GTANGO service platform.

Second, a series of performance tests using *throughput* as main metric was performed. Third, performance tests with *end-to-end service latency* are performed and the results of

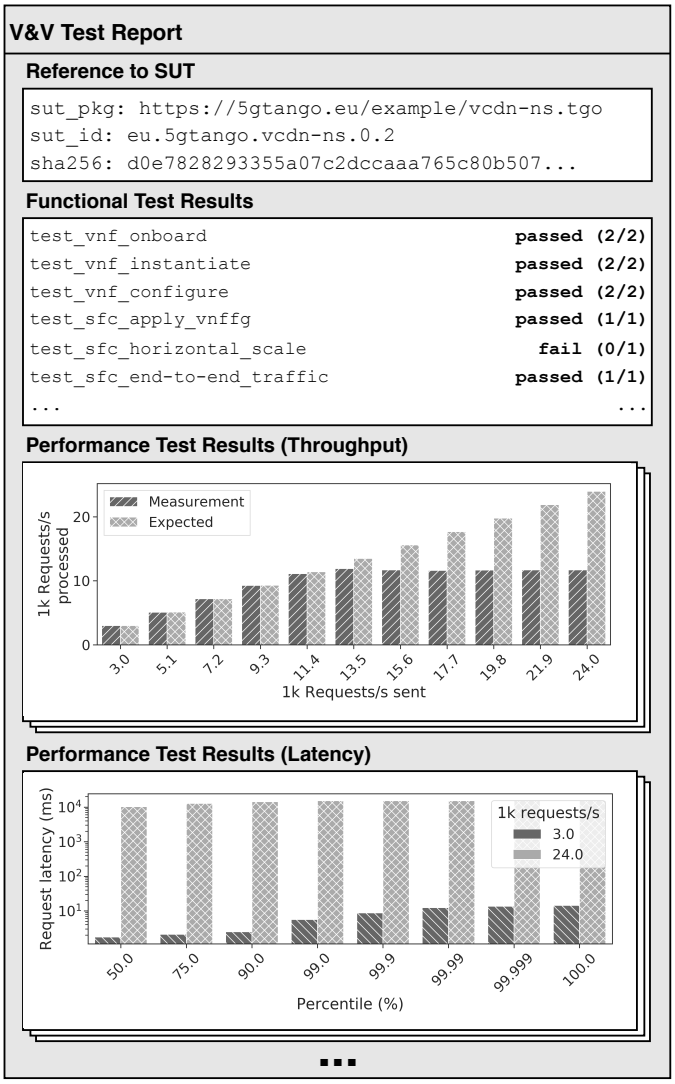

Fig. 4: V&V test report referencing the service under test and showing test results of different test types

both test types are also shown in Fig. 4. They have been done by using the tool *Wrk* as traffic source and an *Nginx* instance as traffic sink. Each test was configured to do 100 parallel HTTP requests, over 30 seconds using request rates between 3,000 and 24,000 requests/s. The results show that the throughput stagnates at about 13,000 requests/s and thus identifies a performance limit if the vCDN service runs with 1 vCPU and 4 GB memory per VNF. This test can be repeated with other resource configurations to learn more about the service's behavior under different resource assignments. The results show how the latency of the service increases under high load and can help developers to optimize it.

Besides the test results, the report in Fig. 4 also shows how the package of the tested service is referenced and its integrity is ensured by using a checksum. It is worth noting that all these tests have been performed in a completely automated manner, without human interaction, after the example service was uploaded to our V&V platform by the service developer. After the test process has finished, the test results are signed by the V&V platform, to verify which platform actually performed the tests, and are presented to the service developer for analysis, debugging, or to further present them to third parties, e.g., potential customers of the VNFs or service.

VI. Conclusions and Future Work

As in traditional software projects, verification and validation plays an important role in future, softwarised networks. It is still a novel discipline and existing solutions mostly focus on small parts of the overall technology stack, which is not enough as the presented analysis of test scopes shows. Using the concept of a trusted V&V platform as part of the NFV ecosystem, we presented the first end-to-end approach for automated verification and validation in the NFV domain, opening the door for new business models and opportunities. Our approach is complementary to most existing testing solutions and allows to integrate them by using flexible, plugin-based designs.

Following the presented concepts, new research questions about flexible, platform-independent test definition approaches, optimized test scheduling, automated test selection and execution algorithms, as well as generic test result representation formats emerge. The presented V&V platform, which is available online [3], is a step towards the realisation of the proposed concepts in real-world scenarios.

Acknowledgments

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. H2020-ICT-2016-2 761493 (5GTANGO), and the German Research Foundation (DFG) within the Collaborative Research Centre "On-The-Fly Computing" (SFB 901).

References

[1] IEEE 5G Initiative, "5G and Beyond Technology Roadmap," https://5g.ieee.org/images/files/pdf/ieee-5g-roadmap-white-paper.pdf, Accessed on 11-13-2018.

[2] H. Karl, S. Dräxler, M. Peuster, A. Galis, M. Bredel, A. Ramos, J. Martrat, M. S. Siddiqui, S. van Rossem, W. Tavernier *et al.*, "DevOps for network function virtualisation: an architectural approach," *Transactions on Emerging Telecommunications Technologies*, vol. 27, no. 9, pp. 1206–1215, 2016.

[3] 5GTANGO project consortium, "5GTANGO Development and Validation Platform for Global Industry-specific Network Services and Apps," https://5gtango.eu, Accessed on 11-13-2018.

[4] B. W. Boehm *et al.*, *Software engineering economics*. Prentice-hall Englewood Cliffs (NJ), 1981, vol. 197.

[5] D. Kozen, "NetKAT: A formal system for the verification of networks," in *Proc. 12th Asian Symposium on Programming Languages and Systems (APLAS 2014)*, vol. 8858. Springer, 2014.

[6] S. Spinoso, M. Virgilio, W. John, A. Manzalini, G. Marchetto, and R. Sisto, "Formal Verification of Virtual Network Function Graphs in an SP-DevOps Context," in *Service Oriented and Cloud Computing - 4th European Conference*, 2015, pp. 253–262.

[7] ETSI GS NFV-TST 001, "Network Functions Virtualization (NFV); Pre-deployment Testing; Report on Validation of NFV Environments and Services," Accessed on 11-13-2018.

[8] M.-K. Shin, K.-H. Nam, S. Pack, S. Lee, and R. Krishnan, "Verification of NFV Services : Problem Statement and Challenges," Working Draft, Accessed on 11-13-2018. [Online]. Available: http://www.ietf.org/internet-drafts/draft-irtf-nfvrg-service-verification-05.txt

[9] L. Cao, P. Sharma, S. Fahmy, and V. Saxena, "NFV-VITAL: A framework for characterizing the performance of virtual network functions," in *NFV-SDN*. IEEE, 2015, pp. 93–99.

[10] J. Pelay, F. Guillemin, and O. Barais, "Verifying the configuration of virtualized network functions in software defined networks," in *NFV-SDN*. IEEE, 2017, pp. 223–228.

[11] R. V. Rosa, C. Bertoldo, and C. E. Rothenberg, "Take Your VNF to the Gym: A Testing Framework for Automated NFV Performance Benchmarking," *IEEE Communications Magazine*, vol. 55, no. 9, pp. 110–117, 2017.

[12] M. Peuster and H. Karl, "Profile Your Chains, Not Functions: Automated Network Service Profiling in DevOps Environments," in *2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, 2017.

[13] S. V. Rossem, W. Tavernier, D. Colle, M. Pickavet, and P. Demeester, "Introducing Development Features for Virtualized Network Services," *IEEE Communications Magazine*, vol. PP, no. 99, pp. 2–10, 2018.

[14] P. Twamley, M. Müller, P. Bök, G. K. Xilouris, C. Sakkas, M. A. Kourtis, M. Peuster, S. Schneider, P. Stavrianos, and D. Kyriazis, "5GTANGO: An Approach for Testing NFV Deployments," in *2018 European Conference on Networks and Communications (EuCNC)*, June 2018, pp. 1–218.

AUTHOR BIOGRAPHIES

Manuel Peuster received his MSc degree in computer science from Paderborn University in 2014, where he is currently doctoral researcher in the Computer Networks group. His research interests are NFV, SDN, and performance benchmarking.

Stefan Schneider received his MSc degree in computer science from Paderborn University in 2017, where he is currently doctoral researcher in the Computer Networks group. His research interests are NFV, SDN, and machine learning.

Mengxuan Zhao joint Easy Global Market as a research engineer in 2015 after her PhD in computer science from University of Grenoble. Her research interests are data management and standardization in IoT, as well as NFV related to 5G.

George Xilouris is holder of MSc graduate in Automation Systems since 2000. He is fellow researcher at Media Networks Lab, at the Institute of Informatics and Telecommunications at NCSR Demokritos. His current interests are next generation and software networks.

Panagiotis Trakadas is collaborating with Synelixis Solutions as a Project Manager in EU-funded projects. He is also an Associate Professor at TEI of Sterea Ellada. His research interests include routing and virtualization technologies in next generation networks.

Felipe Vicens is a member of ATOS Research & Innovation department Telecom team. He has a long experience and in-depth knowledge in networking, virtualisation and cloud environments. His current interests are in 5G, SDN networks and cloud-native.

Wouter Tavernier received his MSc degree in Computer Science in 2002 from Ghent University. He obtained a PhD in 2012 and is currently professor at the same university. His interests focus on performance aspects of SDN, NFV and large-scale routing.

Thomas Soenen obtained his MSc degree in Physics and Astronomy in 2012 from Ghent University. Currently, he is a researcher at IDLab at Ghent University - imec. His interests focus on new network paradigms such as SDN and NFV.

Ricard Vilalta (MSc in telecommunications engineering 2007, PhD in 2013) at the Universitat Politcnica de Catalunya (UPC). He is a senior researcher at CTTC and he is an active contributor in several standardization bodies such as ONF, ETSI and IETF.

George Andreou is senior software engineer at Huawei Technologies Ireland. His interest are big data, machine learning, and SDN/NFV.

Dimosthenis Kyriazis Dimosthenis Kyriazis is an Assistant Professor in University of Piraeus. His research focuses on virtualization technologies in distributed infrastructures, ranging from 5G environments to cloud and edge computing, while also analysing topics related to data management and analytics.

Holger Karl received his PhD in 1999 from Humboldt University Berlin; afterwards, he joined Technical University Berlin. Since 2004, he is Professor for Computer Networks at Paderborn University. His main research interests are wireless communication and architectures for the Future Internet.