UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

LUCAS BONDAN

NFV Environments Security through Anomaly Detection

Thesis presented in partial fulfillment
of the requirements for the degree of
Doctor of Computer Science

Advisor: Prof. Dr. Lisandro Zambenedetti Granville

Porto Alegre
July 2019

CIP — CATALOGING-IN-PUBLICATION

Bondan, Lucas

NFV Environments Security through Anomaly Detection / Lucas Bondan. – Porto Alegre: PPGC da UFRGS, 2019.

142 f.: il.

Thesis (Ph.D.) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2019. Advisor: Lisandro Zambenedetti Granville.

1. Network Functions Virtualization. 2. Security. 3. Anomaly Detection. I. Granville, Lisandro Zambenedetti. II. Título.

“Success is not final, failure is not fatal:
it is the courage to continue that counts.”
— Winston Churchill

ACKNOWLEDGMENTS

First of all, I would like to thank my family. Thanks to Marli my mom, and Gentil, my dad, for being my example, for understanding those times I wasn't able to be around, for all the support over the years, the patience, the love. Thanks to my brothers Ulysses (a.k.a Taia) and Moisés (a.k.a. Bob), real friends that life gave to me. I don't think it is usual for a middle brother to say this, but you guys are the best! To my love, Camila, thank you "principessa mia"! Eight years side-by-side, always together even in the face of physical distance.

Thanks to Lisandro, my advisor, for choosing me as one of his "disciples" during the long path from my master's degree to the Ph.D. pursuit. I would also like to thank my colleagues at Gent University, especially Filip, Bruno, and Tim, who received me with arms wide open and for all the orientation during my stay in Belgium. I am grateful to all my friends and partners of the Computer Networks Group for all the support and discussions, especially those not related to work but life. The long path of this Ph.D. would have been much harder without you guys. I want to thank also professors and staff of the Informatics Institute from UFRGS for all the assistance over my academic years.

Last but not least, I would like to express my gratitude to those people that influenced directly or indirectly the person I am now. Character and moral are made with both good and bad examples. It's up to you to use them wisely.

ABSTRACT

The employment of Network Functions Virtualization (NFV) solutions has increased in both academic and commercial environments in the last years, given the clear advantages of NFV for flexible, scalable, and cost-effective service provisioning. Such advantages are achieved by chaining together Virtualized Network Functions (VNFs) in Service Function Chains (SFCs), adaptable to customers' needs. As such, SFCs and VNFs became central elements of NFV environments, but despite their unquestionable importance, there is still a lack of proposals for ensuring the availability, confidentiality, and integrity of these elements. As network elements, NFV elements are susceptible to many different threats, such as DoS attacks, information leakage, and unauthorized access. Thus, efforts to overcome such security threats have emerged recently. However, NFV-specific threats still lack a classification to help network operators in designing and employing the most suitable countermeasures. First, this thesis investigates and classifies the main security threats that may affect NFV environments. Then, with this investigation, an NFV Security Module (NSM) to the standard NFV architecture is proposed, providing anomaly detection mechanisms to NFV Orchestrators (NFVO), and analyzing the operation of NFV elements executing under NFVOs' control. The proposed NSM architectural framework enables the design and deployment of different anomaly detection mechanisms in NFV environments. NSM is validated through the implementation and evaluation of different anomaly detection mechanisms, designed to deal with heterogeneous information. The obtained results obtained show the effectiveness of the designed mechanisms in the face of realistic SFC and VNF datasets, achieving accuracies over 95% and proving the feasibility of using NSM as a framework for anomaly detection in NFV environments.

Keywords: Network Functions Virtualization. Security. Anomaly Detection.

Segurança em Ambientes NFV Através de Detecção de Anomalias

RESUMO

O emprego de soluções baseadas em Virtualização de Funções de Rede (*Network Functions Virtualization* ou NFV) tem aumentado tanto em ambientes acadêmicos quanto em comerciais nos últimos anos, dadas as evidentes vantagens de NFV para provisionamento de serviços flexível, escalável e econômico. Tais vantagens são alcançadas através do encadeamento de Funções de Rede Virtualizadas (*Virtualized Network Functions* ou VNFs) em Serviços de Funções em Cadeia (*Service Function Chains* ou SFCs), adaptáveis às necessidades dos clientes. Como tal, SFCs e VNFs tornaram-se elementos centrais de ambientes NFV, mas apesar de sua importância inquestionável, propostas para garantir disponibilidade, confidencialidade e integridade desses elementos ainda são escassas. Como elementos de rede, os elementos NFV são suscetíveis a muitas ameaças diferentes, como ataques DoS, vazamento de informações e acesso não autorizado. Desta forma, esforços para superar tais ameaças de segurança tem emergido recentemente. No entanto, ameaças específicas de NFV ainda não possuem uma classificação para ajudar operadores de rede a projetar e empregar as contramedidas mais adequadas. Esta tese investiga e classifica as principais ameaças de segurança que podem afetar ambientes NFV. Com esta investigação, propõe-se um Módulo de Segurança NFV (*NFV Security Module* ou NSM) para ser adicionado à arquitetura NFV padrão, fornecendo mecanismos de detecção de anomalias para orquestradores NFV através da análize da operação de elementos NFV operando sob seu controle. A arquitetura proposta para o NSM permite o design e a implementação de diferentes mecanismos de detecção de anomalias em ambientes NFV. O NSM é validado através da implementação e avaliação de diferentes mecanismos de detecção de anomalias, projetados para lidar com informações heterogêneas. Os resultados obtidos mostram a eficácia dos mecanismos projetados em face dos conjuntos de dados realisticos de SFCs e VNFs, alcançando precisões acima de 95% e comprovando a viabilidade de usar o NSM como *framework* para detecção de anomalias em ambientes NFV.

Palavras-chave: Virtualização de Funções de Rede, Segurança, Detecção de Anomalias.

LIST OF ABBREVIATIONS AND ACRONYMS

4G	Fourth generation wireless network
5G	Fifth generation wireless network
AAA	Authentication, Authorization, and Accounting
BSS	Business Support System
CAPEX	Capital Expenditures
CDN	Content Delivery Network
COTS	Commercial-off-the-Shelf
CPE	Customer-Premise Equipment
DC	Data Center
DDoS	Distributed Denial of Service
DoS	Denial of Service
DHCP	Dynamic Host Configuration Protocol
DPI	Deep Packet Inspection (DPI)
DPF	Deep Packet Filtering
EMS	Element Management System
EPC	Evolved Packet Core
ETSI	European Telecommunications Standards Institute
ID	Identifier
IDS	Intrusion Detection System
IETF	Internet Engineering Task Force
IMS	IP Multimedia Subsystem
IRTF	Internet Research Task Force
ISG	Industry Specification Group
LTE	Long-Term Evolution

MANO	MANagement and Orchestration
MED	Merged Entropy-based Detector
MitM	Man-in-the-Middle
NAT	Network Address Translation
NED	Numerical Entropy-based Detector
NFV	Network Functions Virtualization
NFV-SEC	NFV ISG Security Working Group
NFVI	NFV Infrastructure
NFVO	NFV Orchestrator
NFVRG	NFV Research Group
NS	Network Services
NSH	Network Service Header
NSM	NFV Security Module
OAD	Orchestrator Abstraction Driver
OPEX	Operational Expenditures
OPNFV	Open Platform for NFV
OS	Operating System
OSS	Operations Support System
PNF	Physical Network Function
QoS	Quality of Service
RG	Residential Gateway
RQ	Research Question
ROC	Receiver Operating Characteristic
SED	Single Entropy-based Detector
SDN	Software-Defined Networking
SDHN	Software-Defined Home Networks

SFC	Service Function Chaining
SFCWG	IETF SFC Working Group
SLA	Service-Level Agreements
VIM	Virtual Infrastructure Manager
VM	Virtual Machine
VNF	Virtualized Network Function
VNFFG	VNF Forwarding Graphs
VNFM	VNF Manager
VoLTE	Voice Over LTE

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

Introduced by the European Telecommunications Standards Institute (ETSI), the concept of Network Functions Virtualization (NFV) is currently a reality in both production and experimental networks (CHIOSI et al., 2012). Industry and academia are exploring the concepts introduced by ETSI's NFV Industry Specification Group (ISG) to boost innovation in network service provisioning and management, as well as to reduce Capital Expenditures (CAPEX) and Operational Expenditures (OPEX) (CHIOSI et al., 2012). Born in Data Centers (DCs) and maturing in the campus, NFV comprises the virtualization of functions usually performed by dedicated devices in different network environments (MIJUMBI et al., 2016b). Such "softwarized" functions are called Virtualized Network Functions (VNFs) and are considered the core of the NFV architecture.

Home environments with Residential Gateways (RGs) and Customer-Premise Equipments (CPEs), network services like Dynamic Host Configuration Protocol (DHCP) and Network Address Translation (NAT), network security solutions such as Deep Packet Inspection (DPI) and Intrusion Detection System (IDS), and fourth and fifth generation (4G and 5G) wireless networks systems with the virtualization of Evolved Packet Core (EPC) are just a few examples of the broad applicability of NFV to create virtualized versions of traditional network functions.

As Software-Defined Networking (SDN) provides flexibility and innovation capabilities to network control plane, the virtualization of network functions provided by NFV brings flexibility to network operators regarding the service delivery process, since customers' specific demands can be individually supplied and dynamically adjusted by chaining VNFs together, composing Service Function Chains (SFC) (or VNF Forwarding Graphs – VNFFG – according to ETSI) (QUINN; ELZUR, 2016; QUITTEK et al., 2014).

As in any innovative networking paradigm, NFV brings many particular challenges to be overcome as its adoption increases over time, along with well-known challenges of virtualized network environments (MIJUMBI et al., 2016a). For example, since software generally performs slower than dedicated hardware, performance optimizations have been explored to improve the VNFs' processing time (KOURTIS et al., 2015; MCGRATH et al., 2015), as well as efficient resource allocation for VNFs (RANKOTHGE et al., 2017b; HERRERA; BOTERO, 2016) and optimal SFC placement in the NFV Infrastructure (NFVI) (LI; QIAN, 2016; DALLA-COSTA et al., 2017). Some challenges faced to employ NFV solutions are addressed through the NFV MANagement and Orchestration (MANO) layer (QUITTEK et al., 2014). NFV MANO is designed to the management and orchestration of all elements and resources (physical and virtual) in the NFV environment, including computing, networking, storage, and virtualization hosts (*i.e.,* Virtual Machines – VMs – or containers). While the management side of MANO regards to the active participation of network operators into the NFV environment operation, either by defining network policies or directly adjusting NFV elements operation, orchestration solutions aim to provide a certain level of autonomy to operations related to VNFs and services life-cycle (BARI et al., 2016).

With the increasing deployment of NFV-based solutions, as well as advancements to overcome NFV challenges becoming widespread and leading to the consolidation of whole NFV ecosystems, security-related issues started to gain attention (BRISCOE et al., 2014; YANG; FUNG, 2016). As a network virtualization concept, NFV environments may present both virtualization and networking vulnerabilities, which can lead to different types of threats to NFV elements operation. Container engines (COMBE; MARTIN; PIETRO, 2016), hypervisors (THONGTHUA; NGAMSURIYAROJ, 2016), and virtual machines (WANG et al., 2016b) are examples of exploitable NFV elements that can have their operation compromised. Furthermore, undisclosed vulnerabilities (so-called zero-day threats) are under constant research investigations and security firms (FIREEYE, 2015). Once a new vulnerability is discovered, it can be explored in two different ways: *(i)* shared with the research community, aiming to find the best way to close the vulnerability; or *(ii)* sold in public or black marketplaces for the best price, opening the possibility to explore such vulnerability for malicious purposes. By becoming either public or sold for the best price, the consequences of exploring NFV vulnerabilities can be devastating to the operator's ecosystem.

Considering the importance of protecting NFV environments, ETSI created the NFV ISG Security working group (NFV-SEC) focused on discussing security-related issues of NFV, evidencing the importance of protecting such environments (BRISCOE et al., 2014). In the same way, solutions from both industry and academia have been proposed to turn NFV environments reliable, resilient, and safe, especially approaches based on anomaly detection are receiving attention, given their effectiveness into detecting undesirable or malicious behaviors of NFV elements. As shown by Chandola *et al.* (CHANDOLA; BANERJEE; KUMAR, 2009), many anomaly detection techniques have been proposed to identify abnormal behaviors in network environments, playing an important role to identify threats in different network security contexts. In summary, anomaly detection refers to the process of recognizing patterns in observable data that do not match with expected behaviors, which indicates a so-called anomaly.

In the NFV context, anomalies in the network and virtualized elements can indicate a series of different threats for the overall NFV environment operation. For example, missing elements, misconfiguration, and traffic redirection can lead to the interruption of service delivery and, in some extreme cases, can indicate attacks with enough power to compromise the entire network operation. Therefore, anomalies in the network should be detected as soon as possible, enabling network operators and mitigation mechanisms to quickly apply countermeasures, avoiding major injuries to service delivery for customers. Anomaly detection in VNFs operation (GIOTIS; ANDROULIDAKIS; MAGLARIS, 2015), NFV services (KOURTIS et al., 2016), and Service-Level Agreements (SLA) violations (SAUVANAUD et al., 2016) are some examples of the applicability of anomaly detection in NFV environments.

Unfortunately, available solutions do not take full advantage of the standardization efforts proposed by ETSI for the integration of their security mechanisms with NFV environments. Most proposals provide solutions to specific problems in specific network contexts. Since one of the main objectives of NFV is the flexibility to operate in different networking environments, the demand for an adaptable security solution for NFV environments able to cope with specific operators' needs becomes imperative. Taking into account both the wide variety of NFV environments, as well as the proven effectiveness of anomaly detection mechanisms in identifying possible threats in different network contexts, network operators can take advantage of an integrated platform capable to provide customizable anomaly detection mechanisms for their NFV environments, using ETSI NFV architecture to provide flexibility to operate in different network scenarios.

In this thesis, an architectural framework for NFV elements security is presented, considering the integration between anomaly detection and NFV standardization efforts. By adding anomaly detection capabilities to the NFV architecture, it is possible to provide a flexible architectural framework to network administrators implement security solutions adaptable to their networking environments and specific needs. The proposed architectural framework allows the implementation of different anomaly detection mechanisms to analyze information acquired by NFV Orchestrators (NFVO)s from the NFV environment based on ETSI information model. Thus, any NFVO following ETSI NFV architecture is capable of performing anomaly detection through the proposed architectural framework, providing interoperability and full integration with ETSI standardization efforts.

1.1 Hypothesis & Research Questions

To overcome the limitations exposed in the context of security NFV environments, particularly in terms of services and virtualization elements availability, confidentiality, and integrity, this thesis presents the following hypothesis.

***Hypothesis*: the employment of anomaly detection mechanisms in conjunction with network orchestrators can properly identify anomalous behaviors related to security threats to NFV virtualization elements in different networking environments.**

In order to guide the investigations conducted in this thesis, the following research questions (RQ) associated with the hypothesis are defined and presented.

RQ I. *What are the threats that may affect NFV environments?*

RQ II. *What information should be analyzed to keep the NFV environment safe and how such information should be acquired?*

RQ III. *How to provide a flexible way to analyze different threats in NFV environments?*

The methodology employed to show the feasibility of the proposed solution relies on the development of a prototype following the specifications of the architectural framework. Different entropy-based anomaly detection mechanisms were implemented to prove the implementation flexibility of the architectural framework proposed, analyzing the detection of different types of anomalies (SHANNON, 1948; BEREZINSKI; JASIUL; SZPYRKA, 2015). Such mechanisms are designed based on two types of information available: (*i*) *qualitative* information, which is interpreted as characteristics and descriptors (*i.e.,* textual information), such as identifiers, IP addresses, member VNFs; and (*ii*) *quantitative*, analyzed and processed as numerical values, such as the bandwidth limit for customers or services. When performing anomaly detection, false-positives may occur (*i.e.,* declare the existence of an anomaly when it does not exist), potentially compromising the operation of NFV elements. For this reason, the false-positive rate should be reduced as much as possible during the detection process.

The prototype is evaluated in case-studies based on two operator network scenarios as presented by the ETSI NFV-SEC (BRISCOE et al., 2014). The first network scenario evaluated is the *monolithic operator* scenario, in which the same organization that operates VNFs deploys and controls their resource consumption, *i.e.,* a private NFV deployment scenario. The second one is the *network operator hosting virtual network* *operators* scenario, similar to the monolithic scenario, except that the network operator hosts other virtual network service providers along with its own VNFs. In both monolithic and hosted operators scenarios, customers can subscribe to services with different requirements, with dedicated SFCs deployed to deliver services fulfilling their demands.

The prototype implemented also allows network operators to configure the anomaly detection analysis into two different operational modes: oriented by *polling*, where NFV elements are analyzed based on a predefined time interval; and oriented by *events*, where NFV elements are analyzed only when NFVO signals a new NFV event. The operational modes are analyzed and compared, discussing the pros and cons of each approach.

1.2 Main Contributions

Many contributions are expected during the development of this thesis, advancing the state-of-the-art of NFV security area and also providing new solutions for overcoming technological challenges in such subject. In the following, the main contributions of this thesis are highlighted.

1. Classifying NFV security threats in different domains according to the nature of NFV elements and monitored information.
2. Adding support to anomaly detection in NFV environments through an integrated solution with the standard ETSI NFV architecture.
3. Revisiting anomaly detection principles for network environments to find the most suitable mechanisms for NFV environments.
4. Identifying anomalies in NFV environments by analyzing NFV elements operation without the direct intervention of network operators.
5. Mapping characteristics of anomalies into possible threats based on the information acquired from NFVOs.
6. Creating programmable and customizable anomaly detection solutions, allowing network operators to adapt their detection mechanisms to the threats their environments are more susceptible.

1.3 Thesis Roadmap

The remainder of this thesis is organized as follows.

In Chapter 2, the background concepts and the most important studies related to this thesis are presented. First, the evolution of NFV is presented, starting with the first virtualization efforts until the emergence of SDN and networking planes separation and network programmability. Then, the standardization efforts led by ETSI take place, creating the concept of NFV which started to be explored by both industry and academia. Finally, academic efforts dealing with more particular research challenges in this context are presented and discussed in details.

In Chapter 3, the classification of NFV security threats is presented, based on different domains according to the nature of NFV elements and monitored information. Then, a deep discussion regarding the main research investigations and solutions for security threats in each NFV security domain is provided. Next, the motivation of this thesis is presented in details.

In Chapter 4, the architectural framework proposed in this thesis is presented, providing details about all functional blocks and components composing the architectural framework, their functionality and their role in the whole anomaly detection process. The architectural framework is based on the addition of a new module to the NFV MANO architecture, communicating directly with NFVOs to obtain information regarding NFV elements operation and forwarding the results of the anomaly detection analysis back to NFVOs. Then, two different operational modes of the proposed solution are introduced, highlighting their advantages and disadvantages.

In Chapter 5, the validation of architectural framework is presented, based on the prototype developed as a proof-of-concept, called NFV Security Module (NSM)[1]. The mechanisms considered to perform anomaly detection using NSM are introduced, detailing the scenarios considered for NSM evaluation and the anomaly detection mechanisms designed to validate NSM operation.

In Chapter 6, the details of the NSM evaluation are presented, detailing the data set used as well as the parameters involved in the experiments. Moreover, the results obtained through the experimental evaluation are discussed in terms of accuracy. The accuracies of anomaly detection algorithms are compared considering different types of anomalies, as well as the impact of the sample sizes and the detection times of the detection process

[1] NSM is available at https://github.com/ComputerNetworks-UFRGS/nsm/

and the operational modes.

In Chapter 7, some final remarks and conclusions are presented. In addition, answers to the fundamental questions proposed along this thesis are discussed and justified. Moreover, opportunities for future work are presented.

2 BACKGROUND

In this chapter, the origin of NFV concepts and some of the most important events that lead to NFV conception are presented. Section 2.1 details the origin of NFV, from the adoption of virtualization solutions to the definition of a common NFV architecture by ETSI. Then, the evolution of NFV since its birth is presented in Section 2.2, highlighting important events that lead NFV to become a networking trend.

2.1 From Network Virtualization to NFV

The virtualization of network elements has been adopted by both industry and academia over the last years. Network virtualization brings many advantages, such as easier deployment and management of network services and underlying network resources, as well as the potential for operational cost reduction and innovation boosting (ESTEVES; GRANVILLE; BOUTABA, 2013). Network virtualization extracts connectivity and services logic from dedicated hardware to be performed as software on top of the physical network, bringing flexibility for network operators to scale up/down their services according to customers' needs in a centralized way.

As network virtualization became widely adopted, the Software-Defined Networking (SDN) paradigm of separating control, forwarding, and management planes started to receive attention. The ability to provide virtualized networks without physical changes to the underlying infrastructure, and especially the possibility to centrally program the network control plane turned SDN into a major networking solution for flexible service provisioning (WICKBOLDT et al., 2015). Thus, given the advantages of both network virtualization and SDN, as well as their inherent similarities, joint solutions have emerged for many different network environments, such as data center networks, wireless access networks, and home networks. Likewise, security-related aspects have also been investigated by both industry and academia in such environments (NUNES et al., 2014).

However, SDN was designed to separate control and forwarding planes, bringing intelligence to the control plane without dealing with problems related to the forwarding plane, such as the high costs involved to deploy new network functions, their complex management, and the integrity of deployed functions and services. Therefore, network companies and academia have started to design their dedicated VNFs to improve the forwarding flexibility, programmability, and security, as SDN did to the control plane, with no concern about interoperability nor integration of solutions in a standardized way.

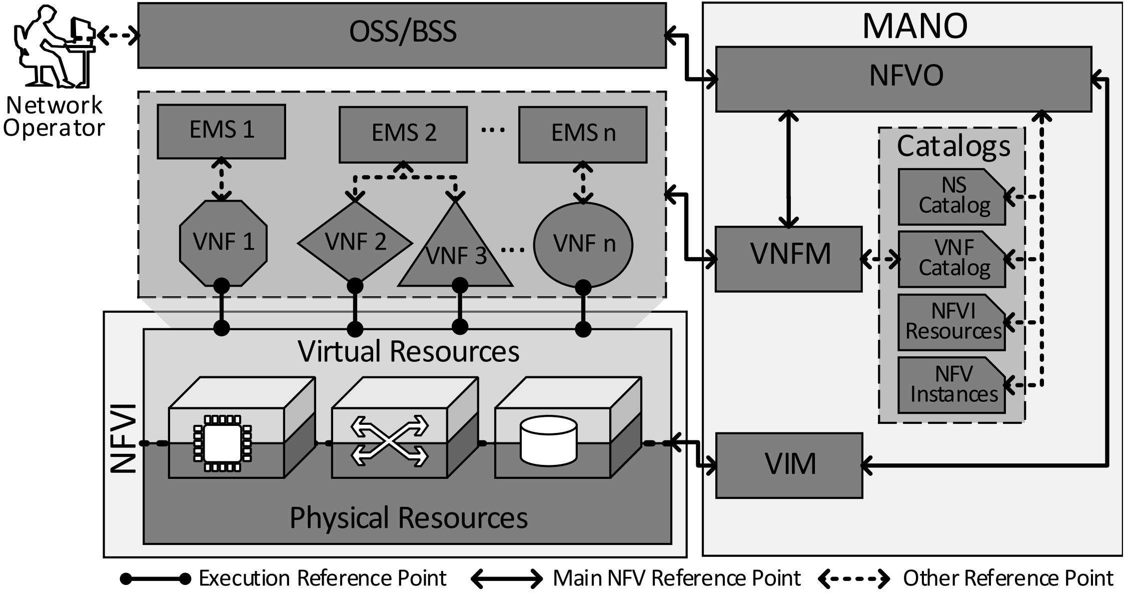
Concerned with the interoperability and management of VNFs, European Telecommunications Standards Institute (ETSI) established an Industry Standardization Group (ISG) forming a consensus with over 150 companies of the networking market to present the concept of NFV (CHIOSI et al., 2012). Originally designed to reduce both CAPEX and OPEX through flexible service deployment, delivery, management, and interoperability, NFV has become an enabler for boosting innovation (MIJUMBI et al., 2016b) in many different aspects of service provisioning. Examples of NFV applicability are resource optimization and self-adaptable services, as well as network and service management, integrity, and security.

NFV is highly complementary to SDN, taking care of forwarding plane functions and fulfilling the gap let by SDN in providing programmability to the forwarding plane. In turn, SDN concentrates on the control plane, but SDN does not depend on NFV (or vice-versa). For example, from the network security point-of-view, security-related VNFs can cooperate with SDN controllers to identify malicious behaviors in the network, using SDN network programmability capabilities to mitigate potential threats automatically. In turn, the centralized network control provided by SDN can be used to collect traffic information to be analyzed by specific VNFs, which can be migrated to different points of the network to analyze suspicious traffic.

The first step of ETSI NFV ISG was the publication of the *Network Functions Virtualization – Introductory White Paper* (CHIOSI et al., 2012). In this non-proprietary white paper, authored by network operators, ETSI outlines the benefits, enablers, and challenges for NFV adoption, encouraging both industry and academia to collaborate in deploying interoperable solutions based on high volume industry standard servers. To boost NFV adoption, ETSI NFV ISG presented a high-level architectural framework and design principles of VNFs, as depicted in Figure 2.1.

The central elements of the NFV architecture are the VNFs, *i.e.,* software implementations of physical network functions deployed on NFV Infrastructures (NFVI). By moving from dedicated hardware to software, network functions become cheaper, more flexible, and easier to deploy, manage, and scale up/down (MIJUMBI et al., 2016b). Moreover, NFV has the power to boost innovation, easing the time-to-market of new network functions. Such advantages are the key to provide customizable service delivery to customers with different requirements. To do so, VNFs are connected composing the chain of functions needed to deliver specific services, called SFCs. Taking into account

Figure 2.1: NFV architectural framework proposed by ETSI (CHIOSI et al., 2013a)



the importance of SFC for service delivery, the Internet Engineering Task Force (IETF) created a working group focused on defining an architecture to handle the deployment of such chains, the IETF SFC Working Group (SFCWG) (QUINN; ELZUR, 2016).

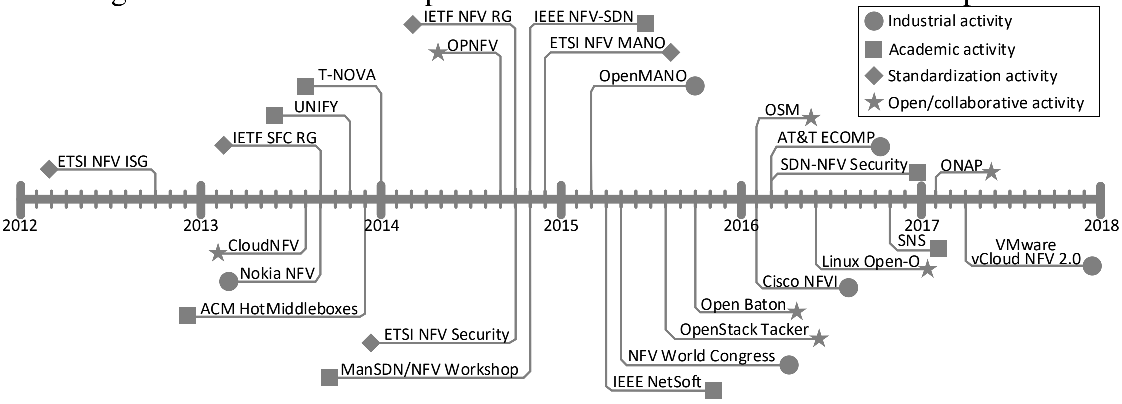
In the NFV architecture, Operations and Business Support Systems (OSS/BSS) are responsible for enforcing access control rules in data centers shared with different network operators. Despite not being mandatory, when present OSS/BSS elements are responsible for granting network operators access to NFVO. An NFV Management and Orchestration (MANO) plane is designed to handle operations related to services and function life-cycle management, as well as resource sharing among virtual elements (QUITTEK et al., 2014). NFVO is the NFV MANO element responsible for bringing intelligence to service provisioning and composition processes, directly interacting with VNF Managers (VNFM) for managing VNF operation life-cycle. Likewise, NFVI virtual and physical resource sharing orchestration among different virtualized elements is performed by NFVOs through Virtual Infrastructure Managers (VIM). In the NFV MANO plane, NFVO has access to the Network Services (NS) and VNF Catalogs, which maintain information regarding available services and functions, as well as NFV instances operating in the NFVI and NFVI physical and virtual resources. Every service and VNF should be registered in such catalogs before being deployed in the NFVI, and their operation is continuously monitored and updated by the NFVOs. Every service and VNF should be registered in the catalogs present in NFV MANO before being deployed in the NFVI. Once registered, their

their operation is continuously monitored and the catalogs are updated by the NFVOs.

2.2 NFV Adoption and Evolution

Figure 2.2 depicts a timeline with some of the most important events related to NFV in the last years, taking as the starting point the publication of the first white paper proposed by ETSI NFV ISG (CHIOSI et al., 2012). This document has caught the attention of the research community and, one year later, ETSI released the first proposal of the NFV architectural framework, aiming not to standardize NFV, but to present a consensus among computer networking companies through a common architecture (CHIOSI et al., 2013b). ETSI's NFV architectural framework defines functional blocks and communication interfaces to decouple network functions from dedicated hardware to run as software in commercial-off-the-shelf (COTS) servers as VNFs.

Figure 2.2: Timeline of important NFV-related events since its conception.



From the publication of ETSI's first NFV white paper, different NFV initiatives from both industry and academia emerged. Nokia announced its NFV initiative for Voice Over LTE (VoLTE) in September 2013, proposing the first commercial NFV-based IP Multimedia Subsystem (IMS) solution compliant with the ETSI NFV architecture. The first edition of the ACM Workshop on Hot Topics in Middleboxes and Network Function Virtualization (HotMiddleboxes) took place in November 2013, together with ACM CoNEXT conference, the first event with a clear focus on NFV. Also in 2013, the first open NFV project appeared: CloudNFV, aiming to be an open, flexible, and cloud-driven collaborative implementation of the ETSI NFV specifications. Moreover, IETF created in the same year the Service Function Chaining Working Group (SFCWG), focused on providing a new approach to service delivery and operation considering the introduction of virtualized network functions in operator networks.

At the end of 2013 and beginning of 2014, two important FP7 projects started: UNIFY[1], formed by a consortium of service providers, vendors, universities, and research institutes to provide features such as service orchestration capabilities, automated service chaining deployment, and dynamic VNF placement; and T-NOVA[2], with the objective of providing a MANO platform for the automated provisioning, configuration, monitoring, and optimization of VNFs in NFV environments. Supported by these projects, works in many different NFV-related areas have emerged, covering challenges such as VNF and SFC placement (SAHHAF et al., 2015b; MCGRATH et al., 2015), performance optimization (CERRATO; ANNARUMMA; RISSO, 2014; PAGLIERANI, 2015), and NFV MANO (GIANNOULAKIS et al., 2014; SAHHAF et al., 2015a).

In September 2014, the Open Platform for NFV (OPNFV) project was created, a collaborative project (now supported by the Linux Foundation) to support the development of different open source NFV solutions. In the same year, important networking conferences, workshops, and meetings took place. Among them, the first edition of the IEEE NFV-SDN and the ManSDN/NFV workshop, maturing concepts and stimulating discussions through research projects and dedicated solutions for software-defined and virtualized environments. Late in 2014, IETF NFVRG was created to deal with many different NFV-related challenges, such as services verification and orchestration (SHIN et al., 2017; BERNINI et al., 2017).

Also in 2014, ETSI announced two major NFV initiatives: NFV Security, identifying potential security vulnerabilities that may affect NFV elements, as well as exploring realistic NFV deployment scenarios; and NFV MANO, providing details about the management and orchestration framework for VNFs and services on NFV environments. With the proposal of a MANO plane for NFV and the investigation of security-related issues, ETSI made clear the importance of guaranteeing security of NFV environments, monitoring the operation of NFV elements and acting to enforce resilient service provisioning.

Telefónica announced the OpenMANO initiative in March 2015 as an open source project to provide practical realization of NFV MANO under ETSI's NFV ISG standardization, later becoming the main component of Telefónica's NFV Reference Lab. Other open projects also were born late in 2015, dealing with the orchestration process of NFV. Among these projects, OpenStack Tacker, an orchestration extension of the OpenStack cloud operating system; and Open Baton, a framework for the orchestration of VNFs in

[1] http://www.fp7-unify.eu/

[2] http://www.t-nova.eu/

heterogeneous infrastructures, are noteworthy. The importance of open projects to mature NFV is indisputable since the NFV community has been designing solutions for many NFV challenges using tools provided by these projects (MEDHAT et al., 2016; CHEN et al., 2017; BELLAVISTA et al., 2017; CARELLA et al., 2017) Later in 2015, both industry and academia kept reinforcing the development of NFV solutions through the organization of the first NFV World Congress, as well as the first IEEE NetSoft conference.

In 2016, Cisco and AT&T announced their leading NFV solutions, definitely entering the NFV market with Cisco NFVI and AT&T ECOMP. The Linux Foundation also announced the Open-O project, an open-source framework to integrate SDN and NFV for agile service provisioning. Still in 2016, the research community hosted the first edition of the ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization (SDN-NFV Security) and the IEEE International Workshop on Security in NFV-SDN (SNS), both aimed to boost NFV security-related research. In 2017, both AT&T's ECOMP and Linux's Open-O are merged together to create ONAP, putting together the benefits of ECOMP and Open-O to build a platform for real-time VNF orchestration with support to policy-based automation, with a dedicated security coordination committee responsible for managing identified vulnerabilities and coordinating the necessary security-related activities in the platform. In April, VMWare announced its vCloud NFV 2.0 solution, an NFV platform in accordance with ETSI's specifications designed for software-defined data centers focused on service provisioning flexibility.

Through the analysis of the NFV timeline, it is clear the attention NFV is receiving over the recent years from both academia and industry, pushing service provisioning to a new level. The organization of networking conferences, workshops, and meetings focused on NFV is increasing. With the growth of NFV-based solutions adoption, the security of such environments is a vital matter, as addressed by the ETSI NFV Security Group. According to (AVIZIENIS et al., 2004), when addressing security, three key attributes must be properly addressed: (*i*) availability of correct services for authorized actions only, (*ii*) confidentiality for the absence of unauthorized disclosure of information, and (*iii*) integrity to guarantee the absence of unauthorized system alterations.

In the next chapter, the first step towards addressing NFV security is taken through the study and classification of important threats of NFV environments, classifying them in different security domains.

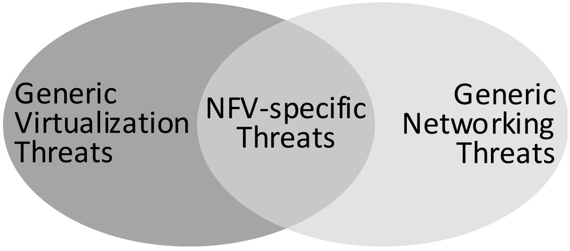
3 NFV THREAT CLASSIFICATION

Security is an important aspect of any networking-related area, especially in emerging networking paradigms and technologies. As innovative solutions mature, security-related aspects start to be investigated together with the consolidation of NFV-based solutions, aiming to provide trustworthy environments for both network operators and customers. As NFV concepts crystallized and became widely explored in different scenarios, security-related aspects of NFV started to be investigated. To support and answer the RQ I, in the next sections, the separation of NFV in security domains is presented, as well as research efforts and solutions within the context of each security domain.

3.1 Security Domains

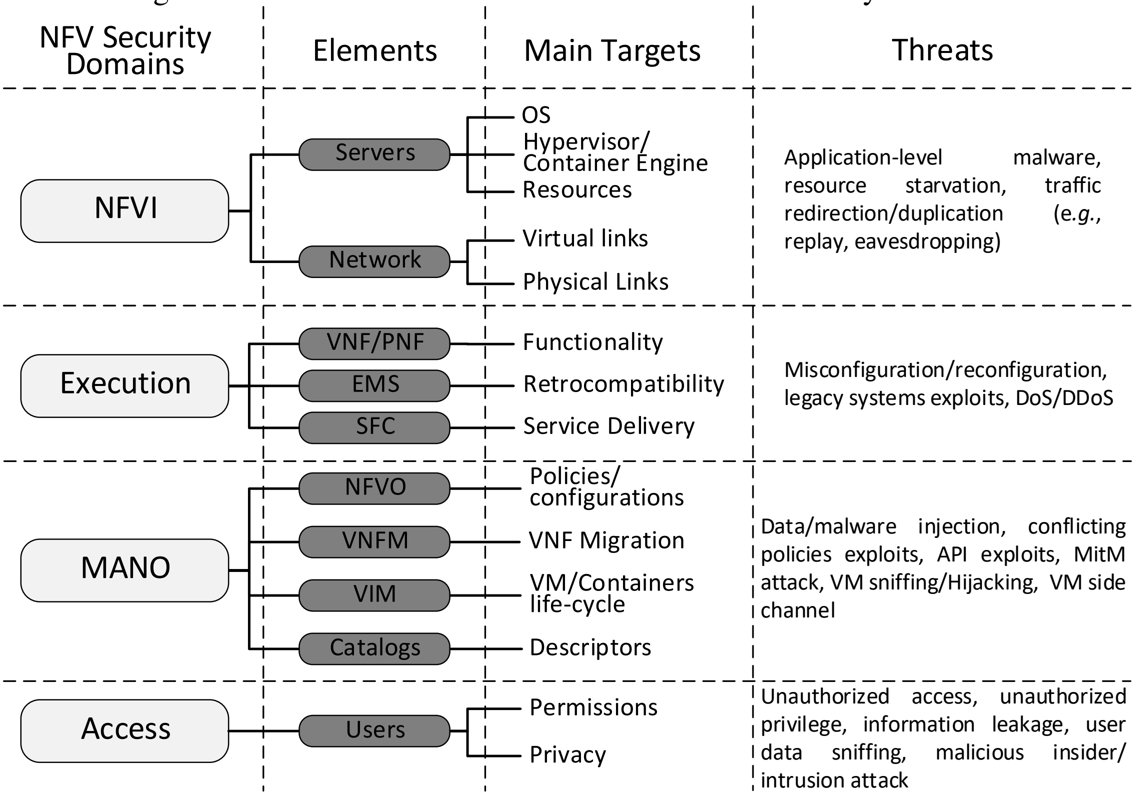
The NFV threats classification presented in this work aims to provide an easy way to analyze the vulnerabilities and attacks that may affect NFV environments. As stated by Hawilo *et al.* (HAWILO et al., 2014), security in computer networks can be organized in different security domains. However, implementing network functions in a virtualized environment renders NFV susceptible to threats related to the combination of virtualization vulnerabilities and network-based attacks (BRISCOE et al., 2014), as illustrated in Figure 3.1.

Figure 3.1: NFV-specific threats belong to the intersection of virtualization and networking threats (BRISCOE et al., 2014).



Considering their combined nature, NFV-specific threats deserve updated and refined classification, enabling network operators to adopt the most suitable security solutions for their environments. The NFV Security working group published a series of specifications related to security aspects of the ETSI NFV architectural framework. In such specifications, they point out important aspects that must be provided by technical solutions employed in NFV environments (*e.g.,* hypervisors, VMs, containers engines) to guarantee aspects such as software integrity protection, verification, secure logging, cryptography, hardware-based root of trust, among others. However, these specifications do not cover the protection of the NFV elements introduced by the ETSI architectural framework but specify requirements to network operators select technologies to employ their NFV environment. For this reason, generic networking and virtualization threats are analyzed to define an NFV-specific threat classification, depicted in Figure 3.2.

Figure 3.2: Classification of NFV threats based on security domains.


NFV Security Domains
Elements
Main Targets
Threats
NFVI
Servers
Network
OS
Hypervisor/ Container Engine
Resources
Virtual links
Physical Links
Application-level malware, resource starvation, traffic redirection/duplication (e.g., replay, eavesdropping)
Execution
VNF/PNF
EMS
SFC
Functionality
Retrocompatibility
Service Delivery
Misconfiguration/reconfiguration, legacy systems exploits, DoS/DDoS
MANO
NFVO
VNFM
VIM
Catalogs
Policies/ configurations
VNF Migration
VM/Containers life-cycle
Descriptors
Data/malware injection, conflicting policies exploits, API exploits, MitM attack, VM sniffing/Hijacking, VM side channel
Access
Users
Permissions
Privacy
Unauthorized access, unauthorized privilege, information leakage, user data sniffing, malicious insider/ intrusion attack

Threats, in general, have specific main targets, which is a feature or functionality provided by an NFV element. The NFV architecture presents elements related to execution, user access, resources, management, and orchestration, according to their roles in the NFV architecture (CHIOSI et al., 2013a). Therefore, the NFV threat classification presented is based on different security domains, defined according to roles of the NFV elements, their main targets, and threats that may affect each domain, as explained in detail in the following.

NFVI domain

In an NFV environment, all hardware and software resources required to deploy VNF instances are provided by the NFVI (CHIOSI et al., 2013a). Processing, storage, and connectivity are provided through a virtualization layer, abstracting physical resources to

VNFs. As such, the main elements in this security domain are (*i*) the servers responsible for providing processing and storage resources to hosted VNFs and (*ii*) the network itself (composed of routers, switches, and links), which provides connectivity needed for service delivery. One of the main characteristics of NFV is the ability to employ standard COTS servers to build the NFVI, bringing flexibility to infrastructure providers (WRIGHT; HU; REID, 2015). Servers have both software and hardware capabilities, with the Operating System (OS) and the hypervisor or container engine responsible for handling virtualized elements as the supporting software running on these servers, and the resources available for the virtualized elements execution figuring as the hardware substrate provided by NFVI servers.

As in any computer system, software running on servers is a potential target for malware such as viruses, worms, botnets, and Trojan horses (SCHAFFER, 2006). Despite being generic software-related threats, their impact in virtualized environments such as NFV deployments can be devastating (VAUGHAN-NICHOLS, 2008). Additionally, the installed software may vary from one server to another, rendering the creation of security patterns more complex. For example, hypervisors and container engines may present different vulnerabilities, leading to different strategies to protect these elements (THONGTHUA; NGAMSURIYAROJ, 2016; COMBE; MARTIN; PIETRO, 2016). Virtual and physical links are the network resources available to provide connectivity for VNFs and services. In NFV, traditional outside traffic-based threats, such as Distributed Denial of Service (DDoS) attacks, can be easily overcome through VNF migration or VNF scaling up (RASHIDI; FUNG, 2016; RASHIDI; FUNG; BERTINO, 2017; AL-HARBI; ALJUHANI; LIU, 2017). However, network operators should consider such attacks coming from the inside network (YUSOP; ABAWAJY, 2014). Additionally, flow duplication and redirection solutions to mitigate DDoS attacks can be exploited in the network for malicious purposes, such as acquiring sensitive data or unauthorized access to certain services through replay attacks and eavesdropping techniques (THING; LEE; SLOMAN, 2005; SALVADOR; NOGUEIRA, 2014).

Execution domain

The execution domain embraces the elements responsible for executing network functions and for providing service delivery in NFV, *i.e.,* VNFs and Physical Network Functions (PNFs), Element Management Systems (EMSs), and SFCs. As such, the main security vulnerabilities targeted in the execution domain are related to functionalities, especially those provided by VNFs. Since VNFs are pure software implementations of network functions, they are more susceptible to misconfiguration exploits, such as changing the functionality provided by a VNF or taking advantage of an exploitable code snippet to perform an attack (SULATYCKI; FERNANDEZ, 2015; MANSFIELD-DEVINE, 2017).

In the NFV architecture, EMSs are responsible for providing the management interfaces to both VNFs and PNFs. In the case of PNFs, additional drivers may have to be designed, enabling NFVOs to properly orchestrate such functions together with VNFs (MECHTRI et al., 2017). Different from VNFs, PNF functionalities are usually hard-coded, rendering misconfiguration exploits harder to realize in such devices. However, legacy PNFs may present outdated system versions, becoming vulnerable to more recent threats, which can exploit vulnerabilities in the additional software drivers and abstraction layers implemented to communicate with EMSs.

Finally, any problem in VNF and PNF operation may stop the whole SFC operation, *i.e.,* causing a DoS, which may not be necessarily related to traditional traffic-based attacks (*e.g.,* Xmas-tree and TCP SYN flood) (ASHKTORAB; TAGHIZADEH, 2012). For example, an attacked VNF may change received packets before forwarding them to the next element in the SFC, adding malicious code to compromise service delivery. In addition, a compromised VNF can be changed to spoof traffic passing through to obtain sensitive information regarding the SFCs that the VNF is part. Furthermore, in NFV environments where services are also virtualized, they may suffer from the same threats of VNFs, also leading to service delivery problems.

MANO domain

Complementary to the execution domain, the MANO domain encompasses the elements related to the management and orchestration of execution elements. MANO provides to network operators access to all functions and services through four main elements: NFVO, VNFM, VIM, and Repositories (QUITTEK et al., 2014). The NFVO is the MANO element responsible for bringing intelligence to service provisioning and composition processes, directly interacting with VNFMs for managing VNF operation life-cycle. Likewise, NFVI virtual and physical resource sharing orchestration among different virtualized elements is performed by NFVOs through VIMs. NFVO has access to network services and VNF catalogs, that maintain information regarding available services and functions, as well as NFV instances operating in the NFVI and its physical and virtual resources. Every service and VNFs should be registered in such catalogs before deployment in the NFVI, being continuously monitored and adjusted by NFVOs.

Given the importance of MANO elements for the NFV environment operation, a

MANO security domain is designed, concentrating most of the potential threats to the NFV environment operation. Among them, policies and configurations defined by network operators in the NFVO to provide automated orchestration are the main targets in this element, taking advantage of conflicting policies to explore possible vulnerabilities (XIONG; ZOU; CAI, 2015). When indicated by the NFVO, VNFMs can, for example, migrate VMs, to support the increasing demand for specific services or to improve resource utilization. Thus, live migration solutions aim to move VMs along the infrastructure to minimize service delivery interruption, but this approach can turn the NFV environment open to threats such as Man-in-the-Middle (MitM) attacks (VACCA, 2016). VM/container life-cycle management provided by VIM may, in turn, suffer from threats such as VM sniffing to acquire sensitive information or even take control over specific VMs or containers through VM hijacking or side channel attacks (RAZAVI et al., 2016; ZHANG et al., 2011). In addition, VMs may suffer from over/under allocation, in which vulnerable VMs/containers may have their resource requirements scaled up or down, directly affecting resource allocation strategies defined by network operators and potentially causing an interruption in the service delivery (DUNCAN et al., 2013).

Many different solutions for MANO elements have been explored over the last years, especially for VNF management and orchestration, such as Tacker[1], Open Baton (BELLAVISTA et al., 2017), OSM[2], and ONAP[3]; and infrastructure resource management through VIMs like OpenStack[4], CloudStack[5], and Aurora Cloud Manager (WICKBOLDT et al., 2014). As such, each solution has its communication API to allow interaction with the remaining NFV elements, which may also present vulnerabilities that can be exploited for malicious purposes (CHEN et al., 2016)[6]. Additionally, unprotected catalogs may have VNF and service descriptors changed, for example, by removing/inserting data in the catalogs to obtain VNF access or injecting malware in the VNF code.

Access domain

The last NFV security domain deals with the access of different users to NFV environments. NFV has been explored in many use cases, such as mobile networks

[1] https://wiki.openstack.org/wiki/Tacker

[2] https://osm.etsi.org/

[3] https://www.onap.org/

[4] https://www.openstack.org/

[5] https://cloudstack.apache.org/

[6] Two paper were publish related to the MANO domain. The first (ManSDN/NFV 2014) was related to management requirement of a specific solution. The second (ISCC 2014) was a comparison of different virtualization solutions from the MANO perspective. More details are provided in Annex G and Annex E, respectively.

(LIU et al., 2016a), Content Delivery Networks (CDN) (HERBAUT et al., 2017), and VNF marketplaces (XILOURIS et al., 2014). In all scenarios, users play specific roles in the whole network operation. For example, in a VNF marketplace, the infrastructure providers (or marketplace maintainers) are responsible for providing the resources needed to host VNFs, as well as for managing the marketplace. As such, they have the rights to perform almost every operation in the marketplace environment. In turn, developers (or vendors) can publish their VNF solutions in the marketplace to be acquired by customers, which need the approval of the marketplace maintainers (or reviewers).

Taking advantage of the existence of users with different permission levels in the network environment, attackers can target network access control mechanisms to have unauthorized access or privileges to certain elements in the network infrastructure. Moreover, user data is under constant monitoring, even for Quality of Service (QoS) or to enforce that the users' behavior is not violating network policies (RISTENPART et al., 2009). During the acquisition of such information, user data can be sniffed or sensitive information leaked by attackers (PENG; CHOO; ASHMAN, 2016). Furthermore, despite the availability of many different mechanisms to counter outsider attacks, NFV environments are still vulnerable to malicious insiders, which can assume a high permission-level user profile to obtain specific access rights to the infrastructure (DUNCAN et al., 2013).

3.2 Research Efforts and Solutions

Now that the threats are classified, the specialized literature is investigated, gathering solutions designed to protect different NFV deployment scenarios. A systematic literature review was performed to confirm that security is a relevant topic in the NFV field and, once confirmed the relevance of the topic, summarize the most relevant works in the area. To do so, we chose to use Scopus[7] research tool and database of peer-reviewed literature, which indexes, among others, ACM and IEEE digital libraries, two of the most important research databases available. Scopus is considered one of the largest available database for academic research, allowing the definition of curatorship for indexed documents, a key feature to filter scientifically irrelevant results. The systematic research was performed using the following search query:

[7] <https://www.scopus.com>

TITLE-ABS-KEY (((NFV OR VNF OR SFC OR "function virtualization" OR "virtualized function" OR "function chaining") AND ((security OR integrity OR confidentiality OR availability) W/3 (threats OR anomaly OR detection OR orchestration)) ANDNOT (medicine OR disease OR patient OR food OR biology))) AND (LIMIT-TO (PUBYEAR , 2018) OR LIMIT-TO (PUBYEAR , 2017) OR LIMIT-TO (PUBYEAR , 2016) OR LIMIT-TO (PUBYEAR , 2015) OR LIMIT-TO (PUBYEAR , 2014) OR LIMIT-TO (PUBYEAR , 2013) OR LIMIT-TO (PUBYEAR , 2012))

The key aspect of the query is to filter recent documents that explicitly cites costs associated with the orchestration of virtual functions. A time window of five years was used, since the technologies we consider in this work, such as container virtualization, were not prominent prior to the considered years. On January 2019, the query defined resulted in a total of 93 documents, which were systematically categorized following the approach defined by Kitchenham (KITCHENHAM, 2004).

As a result of this investigation, it is possible to organize security-related proposals for NFV into two approaches. First, solutions started to be designed using NFV concepts to overcome traditional network security flaws, vulnerabilities, and threats. Then, considering the emergence of NFV-based scenarios, solutions have been designed to improve the security of such emerging environments. In this section, both NFV-based solutions for traditional security issues and security approaches to emerging NFV environments are presented, discussing in detail their coverage based on the security threat classification presented in this work.

3.2.1 Security Solutions based on NFV Concepts

One of the biggest advantages of NFV is its ability for dynamic service provisioning, adjusting SFC composition and auto-scaling VNFs according to customers' demand. Given the flexibility to quickly instantiate new VNFs, adjust their resource allocation, and migrate them across the network to supply customers' needs, security-based solutions have emerged based on the creation of security-related VNFs. Among them, DDoS detection and mitigation strategies for different scenarios are often found in the literature.

Given the importance of good firewalling strategies to overcome DDoS attacks, Mauricio, Rubinstein, and Duarte proposed and evaluated a firewall implementation in the OPNVF platform (MAURICIO; RUBINSTEIN; DUARTE, 2016), showing that with more instances, firewall VNFs may achieve good performance when compared to traditional physical firewall appliances. Additionally, taking into account the existence of multiple NFV firewalls, Gray *et al.* proposed and evaluated different synchronization schemes for clustered firewalls state synchronization (GRAY et al., 2017). Jakaria *et al.* considered resource allocation optimization to design a dynamic DDoS defense architecture using NFV, controlling the number of instances needed to overcome SYN flood attacks (JAKARIA et al., 2017).

Taking into account cloud environments, Guenane, Nogueira, and Serhrouchni propose an NFV-based firewalling service for cloud environments, presenting an architecture with security elements focused on DDoS mitigation (GUENANE; NOGUEIRA; SERHROUCHNI, 2015). In the same vein, Fung and McCormick propose a DDoS mitigation strategy focused on mitigating traditional IP-based DDoS attacks based on the implementation of a DDoS Mitigation VNF in conjunction with a firewall VNF, where trusted flows have guaranteed quality of service, while suspicious flows are handled based on the availability of resources (FUNG; MCCORMICK, 2015). Rashidi and Fung (RASHIDI; FUNG, 2016) present a colaborative DDoS mitigation solution called *CoFence*, where cloud NFV-enabled domains can direct excessive traffic to other trusted external domains for DDoS filtering.

Another common approach is the combination of SDN and NFV, using SDN to adapt traffic forwarding and NFV to dynamically instantiate security VNFs, enabling network operators to react to DDoS attacks in SDN+NFV scenarios quickly. As such, Lin *et al.* (LIN et al., 2017) explored shuffling algorithms for user traffic redirection, dynamically redirecting HTTP DDoS attacks to defense VNFs. Likewise, Fayaz *et al.* (FAYAZ et al., 2015) propose Bohatei, using SDN and NFV capabilities to design resource allocation and control/data plane mechanisms to avoid bottlenecks for DDoS defense in ISP backbones. Finally, Fallon *et al.* (FALLON et al., 2017) designed and analyzed an autonomic control loop for DDoS detection and mitigation in mobile networks, using NFV to deploy all network elements as VNFs and OpenFlow switches to determine the action taken on each flow under analysis.

Despite the proven effectiveness in using NFV-based solutions as well as its combination with SDN to overcome security problems, with special attention to DDoS attacks, some threats need more complex mechanisms to be overcome. As such, the combination of security VNFs started to be explored, composing security SFCs to improve network security as well as deal with different security threats. Thus, Sendi *et al.* designed a solution to improve the placement of security SFCs composed of different security-based VNFs (SENDI et al., 2017). Their solution considers predefined security constraints and improves scalability by optimizing both network and computer resource allocation. Along the same lines, Park *et al.* presented a SFC solution focused on resource consumption optimization while performing valid intrusion detection functions (PARK et al., 2017).

Both single VNFs and their combination into complex security SFCs have proven effectiveness in protecting resources and network traffic. However, the security and protection of users also deserve attention. Thus, NFV-based solutions have emerged covering threats related to the Access domain, such as the work of Lin *et al.* (LIN et al., 2016), in which the authors use NFV and SDN to reduce communication overhead during Deep Packet Filtering (DPF) process for preserving user privacy. Similarly, Han *et al.* (HAN et al., 2017) proposed a Security Trust Zone for safe access management between central cloud and edge cloud sites in 5G networks. The proposed solution uses NFV elements to reduce risks during authentication, authorization, and accounting (AAA) operations in 5G networks. Using both SDN and NFV capabilities, Park *et al.* (PARK et al., 2017) designed a lightweight solution for intrusion detection that creates SFCs to analyze network traffic, using fragmented DPIs implemented as VNFs and orchestrated by an SDN controller, inspecting different protocols with low computational load.

NFV-based solutions targeting multiple threats are also found. In the context of LTE networks, Liyanage *et al.* present a security architecture based on SDN and NFV for automated security management, dynamic attack mitigation, resource optimization, among other functionalities for network safety (LIYANAGE et al., 2015; LIYANAGE et al., 2017). In the work of Luo *et al.* (LUO et al., 2016a), an extension to the Software-Defined Home Networks (SDHN) architecture is presented, proposing a multi-stage attack mitigation scheme using SDN and NFV where security functions are deployed on demand to mitigate different types of attacks. Similarly, Chou *et al.* designed an architecture for on-demand security services deployment combining SDN and NFV, in which security SFCs are autonomously created with security VNFs for particular users, customized to deal with specific security threats (CHOU et al., 2016).

There are still many other security applicabilities of NFV concepts, such as policy enforcement (WANG et al., 2016a), IoT security (IQBAL et al., 2017) and VM sniffing/hijacking detection (AGUADO et al., 2017). In this subsection, the focus is on the most explored NFV-based solutions for different vulnerabilities and threats found in existent networking environments. In the next subsection, the focus changes to solutions designed to protect emerging NFV scenarios as well as the vulnerabilities and threats involved in such environments.

3.2.2 Security Solutions to Protect NFV Environments

As the deployment of NFV-based environments increased, solutions to overcome the vulnerabilities of NFV elements in different scenarios started to appear. Schöller *et al.* present an architecture for VNF orchestration focused on resilient deployment of VNF, proposing the deployment of redundant VNFs in different zones of the network while guaranteeing suitable communication delay when switching from a compromised VNFs to the redundant equivalent (SCHöLLER et al., 2013). In the work of Lal *et al.* (LAL et al., 2017), the authors propose a mechanism to verify the integrity of VNF images during the instantiation process in NFV-based telecommunication networks, analyzing the existence of potential malicious VNFs before they get fully functional in the environment. Such solutions provide a certain level of resiliency to individual VNFs, but there is no concern about the security of VNF composition in SFCs.

Regarding SFC security, Liu *et al.* consider security-related VNFs to introduce an architecture for security SFC composition and provisioning, using SDN to forward traffic to the respective VNF in the SFC (LIU et al., 2016b). Similarly, Nguyen *et al.* (NGUYEN et al., 2017) considered encryption-based security for SFCs by analyzing the performance of three different encryption algorithms applied to the Network Service Header (NSH) proposed by IETF SFCWG (QUINN; ELZUR, 2016), comparing the latency of each algorithm for encrypting/decrypting encapsulated SFC headers according to IETF's NSH. Taking into account resilient SFC provisioning, Hmaity *et al.* proposed a VNF placement scheme considering single-link and single-node failures (HMAITY et al., 2016).

Taking into account NFVI security in different scenarios, Liu *et al.* (LIU et al., 2016a) performed a study regarding the reliability of NFV mobile environments, aiming to help network operators to evaluate the robustness of their environments. The authors proposed four different mechanisms to identify the minimal number of physical or virtual NFV elements whose malfunction or removal from the NFVI will lead to the failure of the NFV environment. Complementarily, Ravidas *et al.* (RAVIDAS et al., 2017) analyzed the importance of incorporating trust in NFV telecommunication clouds, presenting mechanisms for VNF integrity verification and VNF binding, and a policy-based fault tolerance mechanism to fulfill resource management and QoS requirements of VNFs.

Concerned with policy-based NFV protection, Sauvanaud *et al.* (SAUVANAUD et al., 2016) proposed an SLA violation detection mechanism, analyzing VNFs VM monitored data to identify anomalous VMs in a predictive way before they interfere in the network operation. In the work of Durante *et al.* (DURANTE et al., 2017), the authors presented a formal model to verify security policies applied to guarantee the proper operation of individual VNFs and their combination in SFCs. Additionally, Banse and Schuette (BANSE; SCHUETTE, 2017) proposed an approach to define taxonomy-based security policies for SDN and NFV-based networks. Such an approach is based on deriving predicates from the network used in a security policy description language to determine fine-grained rules to enforce the policies to VNFs and users.

As discussed previously, misconfigurations in NFV execution elements may present a threat to the network environment, even when security strategies and policy enforcement mechanisms to protect VNFs, SFCs, and NFVI (as mentioned before) are present in the NFV environment. Thus, Shin *et al.* (SHIN et al., 2015) proposed a framework to verify and debug NFV-related elements, considering dependencies of SFC components, loop-free SFCs, load balancing among VNF instances, state consistency, among other properties to guarantee the proper operation of NFV elements. Considering the internal execution states of NFV elements, Shih *et al.* (SHIH et al., 2016) presented a new protection scheme called S-NFV, that isolates states of NFV elements to prevent them from entering in inconsistent states and become malicious hosts to the NFV environment. In addition, Qing, Weifei, and Julong (QING; WEIFEI; JULONG, 2017) dealt with the VNF fault protection problem, proposing an algorithm able to add redundant VNFs and select the most suitable ones in case of VNF failures, considering minimization of resource consumption and ensuring SFC reliability.

With regards to user-related vulnerabilities, Pattaranantakul *et al.* (PATTARANANTAKUL et al., 2016) proposed a framework called SecMANO, designing an access control solution that enables the dynamic creation of access control models and policies according to network operators' needs, as well as protecting NFV elements from unauthorized access or privileges. The authors later extended their work to create fine-grained access control rules to block illegal access by grouping or isolating protected network resources from unauthorized users (PATTARANANTAKUL et al., 2017). In the context of cloud computing, Coughlin, Keller, and Wustrow (COUGHLIN; KELLER; WUSTROW, 2017) designed an architecture to increase user privacy in NFV services without data overhead, protecting private data through the establishment of secure channels between protected areas of execution in VNF memory and the rest of the network.

In summary, many of the NFV security domains defined in this thesis have specific solutions to overcome threats that may affect each domain. In this section, some important works related to the most common vulnerabilities are presented. However, additional works that fit in the classification presented in this work are found, such as security orchestrators (JAEGER, 2015), user-centered security approaches (MONTERO et al., 2015), integrated policy enforcement (BASILE et al., 2015; OH et al., 2017), resource-based anomaly detection (KOURTIS et al., 2016), and anti-viruses (KAO et al., 2015). The works presented in this section and additional investigations are summarized in Table 3.1, providing an overview of NFV security threats covered available solutions.

As can be seen in Table 3.1, specific threats have been more heavily studied than others. Among them, NFV-based solutions for DoS/DDoS detection and mitigation have received more attention. The popularity of such approaches can be credited to the flexibility provided by NFV to scale up or down network services. In addition, multi-objective approaches also have been gathering attention from the research community, with different proposals to detect and mitigate traffic-related threats (*e.g.,* redirection and duplication) and to guarantee VM protection, avoiding VM sniffing and hijacking. In the same way, the deployment of security functions like DPIs for the detection of malicious users in the network is also facilitated by NFV, explaining the existence of a considerable number of works on this subject.

Despite the increasing interest in improving NFV security, some vulnerabilities still need further investigation. As NFV brings innovative software-based network elements, software-related threats such as data injection, malware, and misconfiguration have potentially more impact in the network environment. Thus, NFV environments are susceptible to such threats and, therefore, this is a likely candidate for further research. In the same vein, the protection of NFV element APIs (*e.g.,* south/northbound APIs of NFVOs, VNFMs, and VIMs) deserves attention. For instance, an unprotected NFVO northbound API can be exploited to take malicious control over the services life-cycle management by unauthorized users. Finally, legacy system exploits brought by the interaction between traditional PNFs and new VNFs through gradual NFV adoption still need further investigation.

Complimentary to Table 3.1, we provide an overview of the coverage of each solution designed with multiple objectives for emerging NFV environments, which is the context of this thesis. The overview is summarized in Table 3.2.

Table 3.1: Security threats in NFV and respective solutions that use NFV concepts or are designed for NFV environments

Threat	**NFV-based solutions for existing environments**	**Solutions for emerging NFV environments**
Application-level malware	–	(KAO et al., 2015)
Resource starvation	(PARK et al., 2016)	(LIU et al., 2016a) (RAVIDAS et al., 2017) (KOURTIS et al., 2016)
Traffic redirection/duplication	(covered by some Multi-objective solutions - see Table 3.2)	(KOURTIS et al., 2016) (HOLLICK et al., 2017) (BASILE et al., 2016)
DoS/DDoS	(MAURICIO; RUBINSTEIN; DUARTE, 2016) (GRAY et al., 2017) (JAKARIA et al., 2017) (GUENANE; NOGUEIRA; SERHROUCHNI, 2015) (FUNG; MCCORMICK, 2015) (RASHIDI; FUNG, 2016) (LIN et al., 2017) (FAYAZ et al., 2015) (FALLON et al., 2017) (SENDI et al., 2017) (MIGAULT et al., 2017) (FAN et al., 2015) (ALJUHANI; ALHARBI; LIU, 2017) (SAHAY et al., 2017)	(SCHöLLER et al., 2013) (HMAITY et al., 2016) (WANG et al., 2018)
Misconfiguration/reconfiguration	–	(LAL et al., 2017) (SHIN et al., 2015) (SHIH et al., 2016) (QING; WEIFEI; JULONG, 2017) (MONACO; TSANKOV; KELLER, 2016) (BLAISE; WONG; AGHVAMI, 2018) (LI et al., 2018)
VM sniffing/hijacking	(AGUADO et al., 2017)	(covered by some Multi-objective solutions - see Table 3.2)
Conflicting policies exploits	(WANG et al., 2016a) (LORENZ et al., 2017)	(SAUVANAUD et al., 2016) (DURANTE et al., 2017) (BANSE; SCHUETTE, 2017) (BASILE et al., 2015) (OH et al., 2017)
VM side channel	–	(ZHANG et al., 2011) (NEZARAT, 2017)
MitM attack	(LAI; FU, 2016)	–
Unauthorized access/privilege	(HAN et al., 2017)	(PATTARANANTAKUL et al., 2016) (PATTARANANTAKUL et al., 2017) (COUGHLIN; KELLER; WUSTROW, 2017) (SALMAN et al., 2017)
Information leakage	(LIN et al., 2016)	(NGUYEN et al., 2017)
Malicious insider/intrusion	(PARK et al., 2017) (LUO et al., 2016b) (BOUET; LEGUAY; CONAN, 2015) (LIN; WU; SHIH, 2017)	(MONTERO et al., 2015) (MATIAS et al., 2016)
User data sniffing	(LIN et al., 2016) (AGUADO et al., 2018)	(NGUYEN et al., 2017)
Multi-objective	(LIYANAGE et al., 2015) (VASSILAKIS et al., 2017) (LIYANAGE et al., 2017) (LUO et al., 2016a) (CHOU et al., 2016) (IQBAL et al., 2017) (FYSARAKIS et al., 2017) (HU; YIN, 2017) (GARDIKIS et al., 2017) (MASSONET et al., 2016) (PASTOR et al., 2018) (LOPEZ et al., 2018)	(LIU et al., 2016b) (BONDAN et al., 2017b) (JAEGER, 2015) (BASILE et al., 2015) (WENDLAND; BANSE, 2018)

Focusing on the execution domain, Liu *et al.* (LIU et al., 2016b) present a process of booting a protected VM by allocating security VNFs to cover the VM to be protected. Such VNFs can theoretically include any security functions, such as firewalls, IDSs, and DPI. However, these functions are still suggestible to threats, without a security mechanism for its information or configuration. In turn, Jaeger (JAEGER, 2015) proposes a security orchestrator extending the ETSI NFV architectural framework, keeping the original

Table 3.2: Multi-objective proposals coverage of security domains ('*' indicates partial coverage)

Work	NFVI	Execution	MANO	Access
(LIU et al., 2016b)		X		
(JAEGER, 2015)		X		
(BASILE et al., 2015)		X*	X*	
(WENDLAND; BANSE, 2018)	X	X*		
This thesis	X*	X	X*	X

elements of such architecture intact. The proposed security orchestrator aims to ensure the security of execution elements, such as VNFs, PNFs, EMSs, and virtual machines. Basile *et al.* (BASILE et al., 2015) propose the introduction of a Policy Manager in the NFV architecture, allowing users to specify their security requirements to the correct deployment and configuration of security functions. Such approach partially covers the MANO domain through the autonomic translation of policies into VNFs configurations; and the execution domain, being limited to a predefined set of filtering functions, such as packet filters, stateful firewalls, L7 filters, and some basic content inspection functions. Moreover, the proposed Policy Manager does not consider the security of SFCs. Similarly to Jaeger, Wendland and Banse (WENDLAND; BANSE, 2018) propose an enhanced NFV orchestration approach to guarantee data retention, isolation, and security-focused placement of VNFs, restricting the locations were VNFs can be deployed to ensure both the integrity of both VNFs information and the proper resource allocation.

Although this thesis covers all the NFV security domains defined, some specific targets are not covered by the proposed solution in two specific domains (indicated by an asterisk in Table 3.2. On the NFVI domain, hypervisor/container engine protection threats such as malware on the application level are not directly detected using NSM. However, their effects can be observed when anomalies are identified in the NFV environment. On the MANO domain, malware injection is another example of non-direct detection by the anomaly detection mechanism proposed. However, the proposed NSM is the first solution to be present in all domains, with direct or indirect detection, *i.e.,* detection of the specific threat or the effects of unknown threats (*e.g.,* zero-day threats).

3.3 Motivation and Problem Statement

Despite the advances in different security-related areas presented and discussed in Section 3.2, there is still a lack of proposals dealing with security challenges in the context of NFV (BRISCOE et al., 2014). Anomalies in NFV elements can indicate a series of different threats for the SFC operation overall. For example, missing elements, misconfiguration, and redirection can lead to the interruption of service delivery and, in some extreme cases, can indicate attacks with enough power to compromise the entire network operation. As such, the detection of such violations is a challenge to be overcome regarding the operation of NFV elements.

Anomalies in the network should be detected as soon as possible, enabling network operators and mitigation mechanism to apply countermeasures quickly, avoiding major injuries to service delivery for customers. Moreover, considering the intrinsic relation between SFC and NFV, the integrity of SFC elements is fundamental for NFV environments using SFCs for service delivery. For example, a compromised VNF may change received packets and forward them to the next element in the SFC with malicious code, disturbing or even interrupting the service delivery. Similarly, a compromised VNF can also be changed to spoof traffic, aiming to obtain sensitive information regarding the SFCs containing the compromised VNF.

Motivated by (*i*) the lack of solutions for virtualization elements integrity, (*ii*) the potential vulnerabilities of NFV environments, and (*iii*) the valuable results obtained by anomaly detection mechanisms, we have investigated and proposed an architectural framework to allow the design and implementation of anomaly detection mechanisms for NFV environments (BONDAN et al., 2017b). As shown by Chandola *et al.* (CHANDOLA; BANERJEE; KUMAR, 2009), many anomaly detection techniques have been proposed to identify abnormal behaviors on the network, playing an important role to identify threats in different networks security contexts.

The hypothesis considered in this work is that anomaly detection mechanisms can properly keep the integrity of different NFV different environments. The main objective is not only to propose the framework but also to validate and to evaluate anomaly detection mechanisms in NFV environments. Therefore, the main contributions of NSM are threefold: (*i*) the proposal of a framework based on the addition of an NFV Security Module (NSM) to the NFV MANO architecture, designed to ease the deployment of anomaly detection mechanisms in NFV environments; (*ii*) the design and validation of an SFC anomaly detection mechanism elaborated using the SIM framework, evaluating its operation under different network setups; and (*iii*) the integration of anomaly detection solutions implemented through SIM to the NFV MANO architecture, using an information model based on the NFV MANO network service registers and catalogs. Such contributions support the answer to the RQ III through the design of NSM, presented in details in the next chapter.

4 NFV SECURITY MODULE

This chapter presents the NSM architectural framework proposed in this thesis to answer RQ III. In Section 4.1, the NSM architecture is presented, providing details about all functional blocks and components composing the architecture, their functionality, and their role in the whole anomaly detection process. Then, in Section 4.2, two different operational modes of NSM are introduced, highlighting their pros and cons.

4.1 NSM Functional Blocks and Components

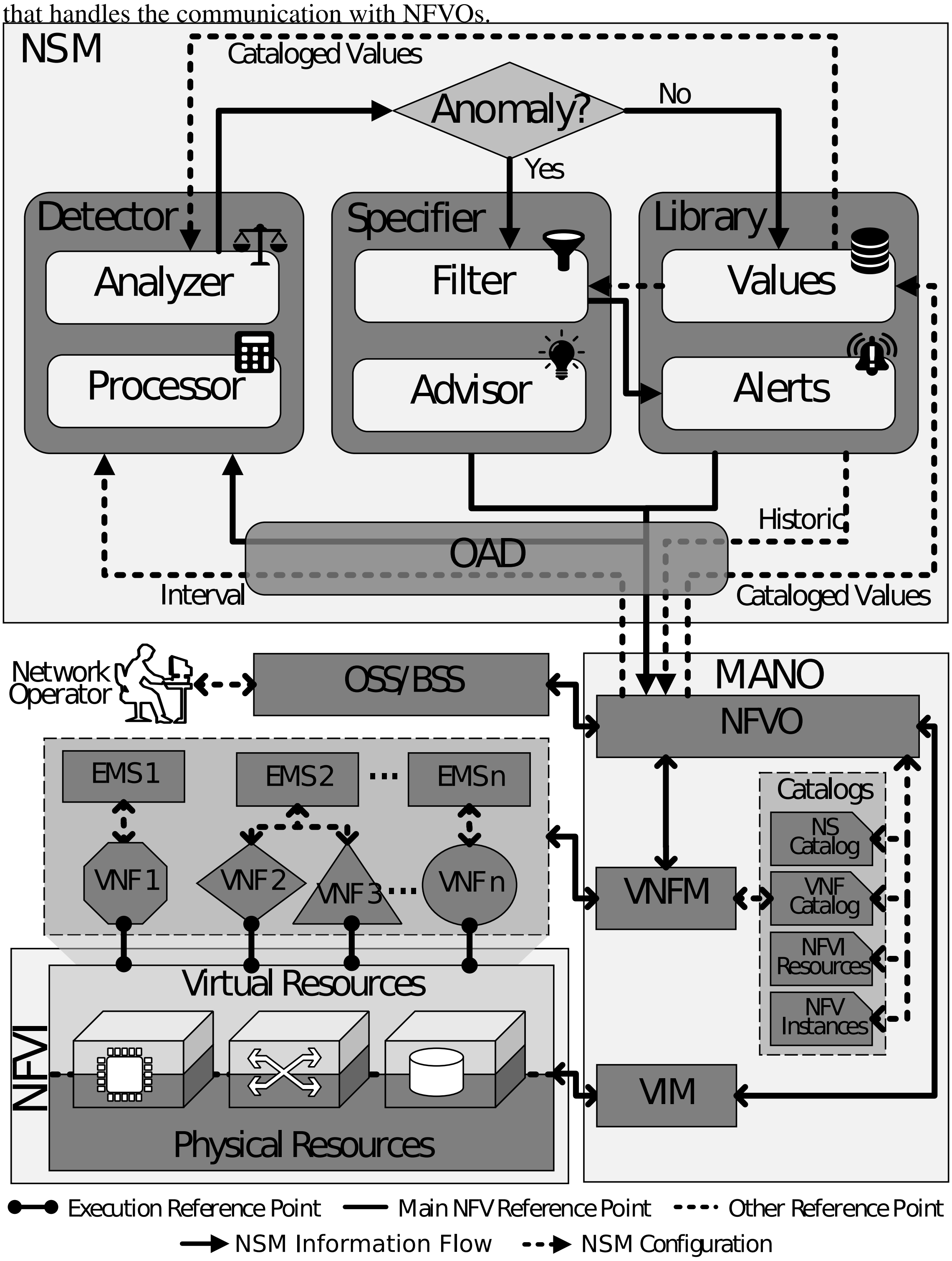
To overcome security-related issues in NFV environments, four basic steps are defined: monitoring, analysis, planning, and acting (or executing). These steps are derived from the MAPE-K loop used for autonomic computing (IBM, 2005). The MAPE-K loop presents Monitoring, Analysis, Planning, and Execution steps, with proposed NSM fitting in the analysis step (with entropy calculation and anomaly filtering) and in the planning step (with the suggestion of actions to be taken to overcome possible threats). Both monitoring and execution steps are responsibilities of NFVOs since the NSM operates apart from the standard NFV elements defined by ETSI, leaving to the NFVO with both information monitoring and execution of the suggested actions.

NSM relies on NFVOs monitoring capabilities to acquire and analyze the behavior of NFV elements and identify potential security threats. Then, NSM suggests actions to be performed by NFVOs to overcome the identified threats. As such, the operation of NSM with NFVOs is crucial to guarantee not only VNFs integrity but also their availability and confidentiality (AVIZIENIS et al., 2004).

NSM operation is based on the addition of a new module to the NFV MANO architecture, directly communicating with NFVOs to request information regarding NFV element operation and also forwarding the results of the anomaly detection analysis. NSM is designed detached from NFVO to turn it independent of the NFVO implementation. Therefore, NSM is able to operate with any NFVO using their standard northbound APIs.

NSM is realized as a modular architecture, providing flexibility for implementing different anomaly detection mechanisms. Network operators can directly configure NSM. However, the most suitable approach is controlling and configuring NSM through NFVO, taking advantage of management interfaces already provided by NFVO. In Figure 4.1, all NSM functional blocks and components are presented, as well as their interaction with the NFV architecture[1].

Figure 4.1: Detailed NSM architecture with internal functional blocks and components. NSM is integrated into the ETSI NFV architecture through a dedicated OAD component that handles the communication with NFVOs.



[1] Early versions of NSM were published at NetSoft 2017 and AIMS 2017. More details regarding these papers can be found in Annex D and Annex C, respectively.

In an NFV data center, network operators configure services through Operations and Business Support Systems (OSS/BSS) and NFVOs, as well as SFCs and VNFs responsible for service delivery to customers. According to the NFV architecture, the NFVO must deal with all responsibilities regarding services' life-cycle management (QUITTEK et al., 2014). To do so, NFVOs manage services, SFCs, and VNFs available through a catalog with information regarding their operation. For deploying a new VNF, the network operator should first catalog it. Once deployed, the VNFs' operation should be monitored and registered by NFVOs. Decoupled from NFVO operation, NSM is composed of four functional blocks as follows.

Orchestrator Abstraction Driver (OAD): Responsible for handling all communication between NFVO and NSM. Since NSM was designed to operate with any NFVO, NSM should be able to adapt its communication to fit their northbound APIs. OAD hosts the communication functions of the NFVO being used. To change the NFVO or communicate with multiple NFVOs, only the OAD block needs to be modified, avoiding changes and bringing flexibility to NSM operation. In other words, OAD works as an abstraction layer, translating information sent from the NFVO in a format understandable by NSM, and also forwarding information from NSM to NFVO through its northbound API calls. OAD is crucial to provide the separation and flexibility aimed for NSM, with its operation being apart from the standard ETSI's modules.

Detector: Requests and receives information regarding SFC and VNF operation to/from NFVO, and performs the implemented anomaly detection mechanism. This block can be configured into two different ways: oriented (*i*) by *polling*, in which NSM periodically looks for anomalies based on a predefined time interval; and (*ii*) by *events*, where NSM requests and analyzes SFCs and VNFs information only when NFVO signals a new event related to these elements. More details regarding the operation modes are presented and discussed in the next section. The Detector block consists of two components. The first one is the *Processor* component, responsible for processing the information acquired from NFVO and formatting it in a format suitable for interpretation by the anomaly mechanisms (*e.g.,* creating information lists for each SFC).

The second component is called *Analyzer*, and it uses the processed information to identify potential anomalies based on cataloged values. If an anomaly is detected, results are forwarded to the Specifier block. Otherwise, the Library block stores the results and the NFVO is notified about the absence of anomalies, since NFVOs may expect a positive report to adapt the operation of a specific SFC, for example. Since the hypothesis defined in this thesis considers that anomaly detection techniques can provide security to NFV environments, the Detector is a key part of the architecture, enabling different anomaly detection techniques to be designed and employed in the NFV environment.

Specifier: Identifies the anomalous elements and selects the most appropriate action to be taken. The main reason for separating the anomaly detection from its specification is to save time and computational resources since the filtering process will only be performed when an anomaly is detected among the monitored elements. In other words, the Detector block informs that something is out of the ordinary, while the Specifier identifies which NFV element is presenting the anomalous behavior. To do so, a *Filter* component is defined for filtering the anomalies from the list of monitored elements, which is done by comparing monitored and cataloged element lists or using predefined thresholds for each NFV element operation.

After identifying the anomalous elements, the *Advisor* component evaluates which is the most appropriate action to be taken to overcome the anomalies and sends an alert message to the NFVO. For example, if an additional unregistered VNF is detected, the Advisor may suggest NFVO to shut down of such an element. The suggestion can be based on both predefined sets of actions and learning mechanisms, depending on the implementation given to the Advisor component.

Anomalies, however, may reflect both known and unknown threats. When the threat is known, the Advisor suggests one of the predefined actions to the NFVO. When the anomaly does not reflect a known threat, the Advisor reports to the NFVO that there is an anomaly that may represent a potential unknown threat. Likewise, it suggests a set of actions to be taken over the anomalous element. For example, in the case of an unknown anomaly in a VM, the Advisor suggests VM-related operations such as restarting, reconfiguring, or even removing the VM. After selecting the action, the Specifier block sends a notification containing the information regarding the anomalous elements to the Library. The final choice of whether to apply the suggested actions or not and possible impacts of such actions lie with NFVO. The Specifier is an important element related to the goal of identifying threats based on anomalies since it is responsible for identifying which threat might be affecting the NFV environment and which element is presenting the anomaly related to the corresponding threat. Moreover, the suggestions given by the Advisor are fundamental to overcome the identified threats.

Library: Stores the anomaly detection results and forwards them to the NFVO when queried. The *Alerts* component handles information regarding alerts generated by the Specifier, which can also be used by the network operator to generate reports regarding the historical occurrence of anomalies in the data center. Likewise, the *Values* component handles the results of analyses that did not detect any anomaly. These values can be used as the baseline for further analysis depending on the anomaly detection mechanism implemented in the Detector block, or they can be re-evaluated when new anomaly detection mechanisms are implemented, enabling the detection of previously undetected anomalies (CHANDOLA; BANERJEE; KUMAR, 2009). The Library has an important role in the anomaly detection process since it is responsible for generating the messages to NFVOs with both standard and anomalous information to be used to overcome the threats.

4.2 Detector Operational Modes

The first operational mode of the Detector block is the *polling-based* mode, in which NSM periodically looks for anomalies based on a predefined time interval. When the time comes, NSM requests to NFVO the information regarding the current status of SFCs and VNFs in execution. Then, NSM executes the anomaly detection mechanism based on the operational status (monitored information) and the cataloged information. The time interval may be configured to coincide with the periodicity at which NFVO collects information from the network.

In the *event-based* operational mode, NSM requests and analyzes SFC and VNF information only when a new event related to these elements occurs. Some examples are the deployment of a new SFC or changes in the configuration of a VNF. NFVO must notify NSM about these changes in the cataloged elements. Then, NSM analyzes the elements currently instantiated based on the monitored information acquired from NFVO, regarding both their operational and cataloged information (*i.e.,* expected behavior), notifying NFVO in case an anomaly is detected.

The *polling-based* mode has the advantage of being independent of the NFVO monitoring activity, which may enable NSM to detect anomalies quickly when configured with short analysis intervals. However, short intervals may overload both NFVO and the network with information requests. In turn, the *event-based* mode, only requests information when NFVO modifies its catalogs, relieving both network and NFVO from excessive requests. However, anomalies may take more time to be detected. The effectiveness of the *event-based* mode is highly dependent on the number of events. An analysis of both operational modes is provided in Section 6.6.

4.3 Compatibility with NFV Standardization Efforts

As stated earlier in this thesis, two main standardization efforts are heading NFV adoption. ETSI conducts the first one through its NFV ISG (CHIOSI et al., 2012), which defines basic NFV concepts and dedicates a group exclusively to NFV MANO (QUITTEK et al., 2014). NFV MANO presents a common management and orchestration architecture for NFV elements, based on an information model designed to handle all information regarding the registration and operation of such elements in the NFVI. Such information model follows a hierarchical organization subdivided into four different classes, as can be seen on the right side of Figure 2.1 and detailed as follows.

1. *Service catalog*: contains the description of all network services available in the network, supporting the creation and management of service deployment templates usually composed of network services, virtual links, and SFC descriptors.
2. *VNF catalog*: stores information regarding all VNFs available for deployment, including VNF descriptors, software images (*e.g.,* virtual machines or containers templates), and network requirements for their deployment. This catalog can be accessed by both NFVO and VNFMs to perform operations such as VNF validation and deployment feasibility.
3. *NFV instances*: holds information of all operational VNFs and services in the network through VNF and NS records. Such records are constantly updated by both NFVO and VIMs during the life-cycle of the respective elements, aiming to keep VNFs and NSs operational status information up-to-date.
4. *NFVI resources*: stores information handled by VIMs regarding the available, reserved or allocated resources in the operator's NFVI. Such information is important for reservation, allocation, and monitoring operations performed by NFVOs, since orchestration strategies usually consider NFVI resources to trace deployment plans.

Despite that ETSI NFV MANO does not deal with the NFV standardization, the information model proposed plays an important role in the integration of different solutions in the same NFV environment. For example, the cooperation between two different NFVOs is easier when both solutions use the same information format, *i.e.,* a common pattern for storing and processing data regarding the operation of NFV elements. For this reason, the information model proposed by ETSI is considered in the design of the NSM architecture, allowing it to integrate with any orchestration solution based on this model.

The second standardization effort comes from the Internet Society, especially from two groups: the NFV Research Group (NFVRG) under the Internet Research Task Force (IRTF) supervision[2], and the SFCWG under IETF supervision[3]. The NFVRG covers NFV-related topics in different areas, such as VNF orchestration, services verification, and multi-domain virtualization. In turn, SFCWG keeps focused on the operations related to the establishment and maintenance of SFCs, proposing an architecture to support their operation (HALPERN; PIGNATARO, 2015). Despite that NFV is not mandatory for SFC deployment, SFCs can benefit from the NFV MANO functions, more specifically from the centralized orchestration provided by NFVOs, since NVFOs based on the ETSI NFV architecture have access to all important information for SFC life-cycle management. Moreover, SFCWG states that all entities composing SFCs should have its integrity maintained against different types of anomalies, but SFCWG does not propose any mechanism to enable it. Thus, although foreseen, the design of SFC integrity solutions is not covered by SFCWG.

Considering both the information model proposed by ETSI NFV MANO and the definitions for SFC composition presented by SFCWG, the NSM is based on ETSI MANO information model to enable the integration with MANO-based solutions, thus providing the integrity required by SFCWG for SFC operation. Moreover, using the ETSI NFV MANO information model, it is possible to easily manage monitored information since its classes properly cover the data needed to be processed by anomaly detection mechanisms. Such an information model is sufficient to answer the RQ II. More details regarding the monitored information, as well as the data sets used to validate NSM are provided in the next chapter.

[2] https://irtf.org/nfvrg

[3] https://datatracker.ietf.org/wg/sfc/about/

5 CASE STUDY

In this chapter, the mechanisms considered to perform anomaly detection using NSM are introduced. First, the scenarios considered for NSM evaluation are detailed in Section 5.1. Then, the threats aimed to be detected and their main characteristics are described in Section 5.2. Finally, in Section 5.3, the anomaly detection mechanisms designed to validate NSM operation are presented and discussed.

5.1 Scenarios

To define the validation scenarios for NSM, some conditions for NSM operation were assumed. The main conditions are summarized in the following assumptions:

- *Bug-free VNFs and SFCs*: VNFs passed through code verification process prior their registration and deployment;
- *Dedicated management communication channel*: no concurrence with users' traffic in the communication between NFVO and NSM;
- *Stable network connection*: problems related to the communication are solved by the network itself without impact for transmitters and receivers (*i.e.,* NFVO and NSM);
- *Human-free operation*: no direct interference of human operators in the anomaly detection process, avoiding the occurrence of human-based errors that may reflect into false-positive anomalies;
- *Sufficient NFVI resources*: all resources needed to instantiate VNFs and to create SFCs are available, so errors regarding resources scarcity are not considered.

Considering the assumptions above, the following aspects are outside of the scope of NSM validation:

- Threats directly related to human error in the network operation;
- VNFs and SFCs verification and validation;
- Infrastructure errors (network and resources);
- Dependability related attributes: reliability and maintainability.

The best way to validate the NSM is by using network scenarios applicable in realistic production environments. Therefore, the network scenarios considered in this thesis follow the definitions presented by ETSI NFV-SEC (BRISCOE et al., 2014). Such scenarios were selected considering their wide adoption in NFV-related areas. More precisely, the (*i*) monolithic and the (*ii*) hosted virtual network operators scenarios are used. In the former, the organization (*i.e.,* NFVI provider/operator) which handles the NFVI resources is the same that operates the VNFs, as well as the resources required by them. In the latter, along with its own VNFs, the NFVI operator also hosts other virtual network service providers. Thus, the first scenario is also present in the second one, with the NFVI operator figuring itself both as a customer and as an NFVI resource provider. A simplified example of such a scenario is depicted in Figure 5.1.

Figure 5.1: Example of the hosted virtual network operators scenario. Customers receive their services through different SFCs, which may share VNFs along their paths.

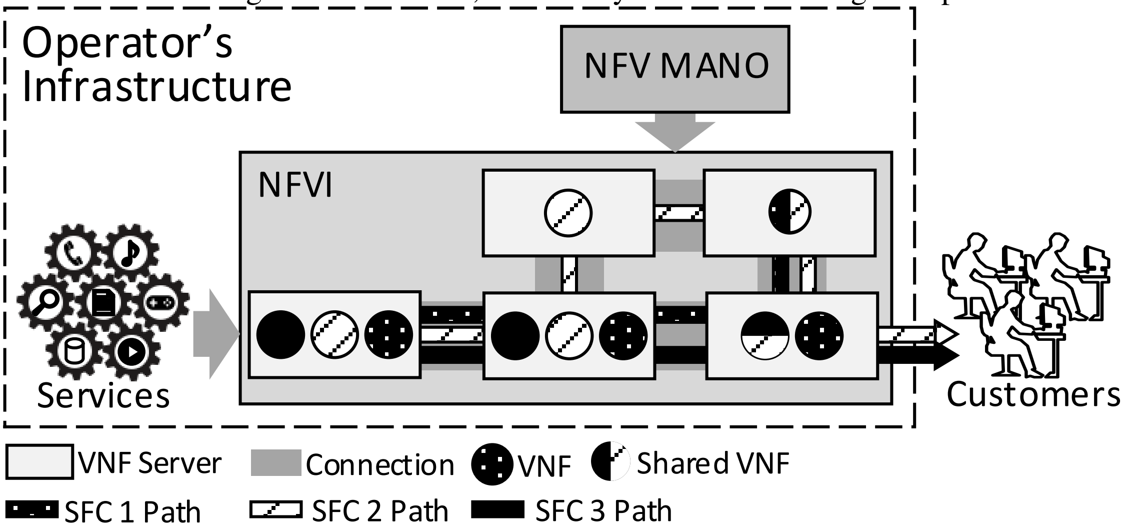


Figure 5.1 depicts three different SFCs, along with their paths through the VNF servers that compose the operator's NFVI. VNFs can belong to both the NFVI operator or customers, but regardless of the VNF owner, all SFC are handled by the NFVI operator through the NFV MANO plane. SFC 1 has its endpoint inside the NFVI, which indicated the service is consumed by the NFVI provider itself, *e.g.,* performing predictive caching for content delivery networks. In this case, another SFC can be instantiated to deliver the service to a given customer when requested.

5.2 Threats

Threat characteristics reflect the type of anomalies occurring in the network, determine the type of information to be monitored, and the effectiveness of the anomaly detection mechanism. Moreover, some characteristics are not exclusive to a specific threat, which may indicate the occurrence of other types of anomalies. Thus, monitoring NFV elements looking for characteristics of well-known threats can also lead to the detection of new types of anomalies, which may indicate the occurrence of zero-day exploits. The anomalies selected for detection in this work and their respective characteristics are based on the threats classified in Chapter 3, summarized in Table 5.1. The threat model used to evaluate NSM is based on the insertion and/or modification of monitored information to reflect the anomalies presented in Table 5.1.

Table 5.1: Threat Characteristics

Threat	Security Attribute	Characteristics	Anomalies
DoS	Availability	Service stops working or not working properly	Missing SFC element
Flow duplication	Confidentiality/Integrity	Information leaked from a flow to unauthorized users or attackers	Uncataloged/modified VNF and virtual link
Unauthorized access	Confidentiality/Integrity	Unauthorized users accessing SFC elements	Uncataloged/modified connection point
Unauthorized privilege	Integrity	User receiving privileges above stipulated	Uncataloged/modified VNF, virtual link, connection point, unauthorized bandwidth growth

In this evaluation, 4 common network threats feasible to occur in NFV environments were selected to evaluate the efficiency of the anomaly detection mechanism used, each one related to specific security attributes (AVIZIENIS et al., 2004): (*i*) **DoS**, when some element of an SFC is missing or presents a different behavior than the cataloged one; (*ii*) **flow duplication**, which may indicate malicious information leaks; (*iii*) **unauthorized access**, when NFV elements (such as VNFs and connection points) are inserted or modified in the chain for providing access to malicious users; and (*iv*) **unauthorized privilege**, where uncatalogued or modified elements changes SFCs operation to privilege some users or customers.

The mentioned anomalies used for detecting threats may not directly relate only to these threats. For example, detecting an uncatalogued connection point could not only indicate unauthorized access and privileges but also lead to the detection of new threats not yet addressed or registered by academia and industry. However, even in the case where an anomaly is related to a different threat, it still needs to be handled. Thus, the choice for an anomaly detection mechanism able to detect different types of anomalies more generically is crucial to keep the SFCs integrity and, consequently, NFV services delivery. Although known threats are used to validate the system, we argue that anomalies may be related to zero-day threats, which do not have a clear description, characteristics, or mitigation strategies. Even though NSM can detect such anomalies, but without indicating which kind of threat is affecting the network, since its an unknown threat. The anomaly detection technique must be selected based on the characteristics of the NFV environment. The next section provides details regarding the selection of the anomaly detection technique.

5.3 Anomaly Detection Mechanisms

There are several anomaly detection mechanisms in literature, each one suitable for different network scenarios and monitored information patterns (CHANDOLA; BANERJEE; KUMAR, 2009). Techniques which require supervised training or statistical modeling regarding network operation may not be suitable for NFV scenarios due to their dynamic behavior. For instance, supervised training techniques require well-defined training data sets (containing anomalous and regular traces of the information monitored. Since anomalies might not be known in their first occurrence, such techniques may not fit well in an NFV environment. Similarly, statistical modeling requires a well-defined behavior of anomalous and regular information, missing the capability to detect zero-day threats.

Despite their high accuracy, spectral theory-based techniques also require normal and anomalous instances to be separable in the lower dimensional embedding of the data, *i.e.,* the variation between anomalous and regular information must be high enough to separate them using reduction algorithms. Information theory-based techniques, however, require neither training data set nor statistical models to operate, as classification and statistics based techniques do. Moreover, information theory-based techniques are less complex than spectral theory-based techniques and more sensitive to data sets with small variations, demanding less processing capabilities to run in an acceptable time.

In this work, Shannon's Information Entropy-based anomaly detection techniques are designed into the Detector block, based on the type of information monitored and the proven effectiveness of using entropy for detecting anomalies on network environments (SILVA et al., 2016). Results obtained in previous investigations show that entropy-based detection is a good candidate to detect anomalies considering data sets following the ETSI

NFV MANO information model format (BONDAN et al., 2017a). In addition, using an entropy-based detector, regular operation will not affect the resulting entropy. Even in case of changes, the second level analysis of NSM will verify the real existence of threats.

Entropy-based anomaly detection works by tracking disorders in the data set of monitored elements handled by NFVO, which indicate anomalies in the operation of NFV elements. Assuming X as the data set under analysis with length n, Shannon's entropy $H(X)$ is given by the equation:

$$H(X) = \sum_{i=1}^{n} p(x_i) \log \frac{1}{p(x_i)} \tag{5.1}$$

where $p(x_i)$ is the probability of the element x_i to occur in X and $\log \frac{1}{p(x_i)}$ is the uncertainty related to x_i. The higher the probability of x_i to occur, the lower its uncertainty. As the number of elements with low probability increases in the list, *i.e.,* highly uncertain elements, the entropy will change, indicating a disorder in the monitored elements. Only if the entropy changes, the filtering process will be started to identify where the anomaly occurs. The two-level approach of NSM (detection and filtering) also improves the accuracy and avoids false negatives alarms, two known issues of entropy-based detection mechanisms (BEREZINSKI; JASIUL; SZPYRKA, 2015). The filtering process designed to validate our approach consists of analyzing element-by-element in the data set, looking for the specific monitored information differing from the original configuration cataloged for the anomalous element. Once specified where the anomaly occurs, the Advisor module looks in a predefined set of suggestions with is the most suitable action to be taken for overcome the potential threat(s) identified, similar to a cause-effect table.

Regarding scalability, the computational cost of calculating the entropy is smaller than comparing element by element. Moreover, the entropy of cataloged information does not need to be calculated every time, but only when new elements are registered in the catalogs. As such, the entropy-based anomaly detection mechanism is scalable and has low computational cost, avoiding constant entropy recalculation. Additionally, fast entropy calculation approaches can reduce even more the entropy calculation complexity (PAN et al., 2011).

To validate NSM, different entropy-based anomaly detection mechanisms were implemented to prove the implementation flexibility of NSM. Such mechanisms were designed based on the two types of information available: (*i*) *qualitative* information, which is interpreted as characteristics and descriptors (*i.e.,* textual information), such as identifiers, IP addresses, member VNFs; and (*ii*) *quantitative*, analyzed and processed as numerical values, such as the customers' bandwidth. Both types of information should be analyzed separately due to their different natures. While quantitative information can present small variations and still be consistent (*i.e.,* do not correspond to an anomaly), any variation in qualitative information may represent an inconsistency and a potential anomaly. The qualitative detectors are referred as *Single Entropy-based Detector* (SED)[1] and *Merged Entropy-based Detector* (MED)[2], and the quantitative detector as *Numerical Entropy-based Detector* (NED).

5.3.1 Qualitative Anomaly Detection Algorithms

SED was designed to compare the entropy of monitored information with cataloged information entropy. Despite its proven efficiency and fast execution time (BONDAN et al., 2017b), the monitored entropy value may remain unchanged in comparison with cataloged entropy even when anomalies are known to occur. Although rare, such situations may occur when, for example, the number of missing elements is equal to the number of unregistered elements in the monitored information. To avoid such false-negative results, the entropy calculation for qualitative information was refined by merging monitored and cataloged information. Thus, if any missing or unregistered element occurs in the monitored list, the merged entropy will differ from the cataloged entropy, indicating an anomaly. Algorithm 1 presents the pseudo-code of the anomaly detection mechanism for qualitative information, which also represents the flow of information handled by NSM.

First, NSM requests information regarding the current status of instantiated elements from NFVO through OAD (line 1). Then, the Processor appends the received information into a merged information list (line 2). The Analyzer then verifies the configured operational mode: if NSM is configured to operate in *event-based* mode (line 3), updated cataloged information is requested to the NFVO (line 4), the newly cataloged entropy is calculated (line 5), and both cataloged information and entropy values are stored in the Library (line 6). Otherwise (*polling* mode), the Detector retrieves the last cataloged entropies from the Library (line 8). Next, the Analyzer appends the cataloged information into the merged information list (line 10) and calculates its entropy (line 11). Both

[1] SED was published as a short paper at NetSoft 2017. More details in the Annex D.

[2] MED was published on the Ph.D. track of AIMS 2017. More details in the Annex C.

merged and cataloged entropies are then compared (line 12). If the difference remains unchanged, there is no indication of anomalies, so the current values are stored in the Library for future queries (line 13), and a report message is build to NFVO, informing that no anomaly was found (line 14). If the merged entropy differs from the cataloged entropy, the Specifier filters the anomaly using the latest cataloged values. If NSM is configured in *polling* mode, it retrieves the cataloged information from the library (line 17). Otherwise (*event-based* mode), it was already requested by the Analyzer (line 4). The filter implemented in this work compares monitored and cataloged information. An alarm message is generated with the filtering results (line 19), and it is stored in the Library (line 20). Finally, the Advisor looks for the most appropriate action to be taken based on the identified anomaly (line 21) and a predefined set of actions, sending a report message to NFVO with the anomalies and the suggested actions (line 23).

Algorithm 1 NSM pseudo-code: Merged entropy-based Detector (MED)

```
 1: $m\_list[customers] \leftarrow OADReqMonitoredInfo()$
 2: $merged\_info[customers].append(m\_list)$        # DETECTOR: Processor
 3: if $MODE = EVENTS$ then                         # DETECTOR: Analyzer
 4:     $c\_list[customers] \leftarrow OADReqCatalogedInfo()$
 5:     $c\_ent[customers] \leftarrow calcEnt(c\_list)$
 6:     $libStoreValues(c\_list, c\_ent)$
 7: else # $MODE = POLLING$
 8:     $c\_ent[customers] \leftarrow libReqValues(c\_ent)$        # LIBRARY: Values
 9: end if
10: $merged\_info[customers].append(c\_list)$
11: $merged\_ent \leftarrow calcEnt(merged\_info[customers])$
12: if $merged\_ent[customers] = c\_ent[customers]$ then
13:     $libStoreValues(m\_list)$                          # LIBRARY: Values
14:     $rep\_msg \leftarrow “NO\_ANOMALY''$
15: else                                                   # SPECIFIER: Filter
16:     if $MODE = POLLING$ then
17:         $c\_list[customers] \leftarrow libReqValues(c\_list)$
18:     end if
19:     $alarm \leftarrow filterAnomaly(m\_list[customers], c\_list[customers])$
20:     $libStoreAlarm(alarm)$                             # LIBRARY: Alarms
21:     $rep\_msg \leftarrow alarm + advSuggAction(alarm)$ # SPECIFIER: Advisor
22: end if
23: $OADReportMessage(rep\_msg)$
```

The main difference from SED to MED resides in the entropy calculation. The merged entropy-based mechanism creates the merged list (lines 2 and 10) with information from both cataloged and monitored information, and calculates the merged list entropy to compare with the cataloged information entropy (lines 11 and 12). In turn, the original entropy-based mechanism uses monitored information entropy instead of the merged information entropy.

5.3.2 Quantitative Anomaly Detection Algorithm

Different from the qualitative versions, there is no need for merging monitored and cataloged information for quantitative values since their discrete nature makes it virtually impossible to have the same entropy variations in two different analysis. However, it may result in different entropy values in every evaluation, turning direct entropy comparison ineffective. To overcome this issue, the algorithm analyzes the monitored entropy in the face of historical cataloged entropies. Da Silva *et al.* proved the effectiveness in detecting anomalies for quantitative information by analyzing if the monitored entropy fits into the interval composed by the mean of historical entropy values plus/minus their standard deviation (SILVA et al., 2016). Complementarity, an additional parameter (β) is defined to adjust the size of such interval. The higher the value of β, the bigger the interval, with $\beta = 1$ representing no change in the interval size.

The Library should have valid entropy samples for each monitored parameter, used to compare with the currently monitored entropy. Such samples can be inserted in the Library in two different ways: manually defined by the network operator or composed of monitored samples in a trusted interval (*i.e.,* when the network is known to be operating without anomalies). The modifications made for the quantitative anomaly detection mechanism are highlighted in Algorithm 2. The limited amount of modifications reflects the modular architecture of NSM, one of the main characteristics aimed for during its design, highlighting NSM's flexibility in implementing different anomaly detection mechanisms.

Algorithm 2 NSM pseudo-code: changes for quantitative analysis (NED)

```
     ...
 2:  $m\_ent \leftarrow calcEnt(m\_list[customers])$
     ...
10:  $mean\_ent[customers] \leftarrow mean(libReqValues(c\_ent))$
11:  $stdv\_ent[customers] \leftarrow stdv(libReqValues(c\_ent))$
12:  **if** $mean\_ent - \beta * stdv\_ent < m\_ent[customers] < mean\_ent + \beta * stdv\_ent$ **then**
         ...
22:  **end if**
     ...
```

The first modification occurs in line 2, where instead of composing the merged information list to calculate its entropy further, the algorithm calculates the entropy of the monitored information received from NFVO. Further, the pseudo-code has been adapted to calculate the mean (line 10) and the standard deviation (line 11), verifying if the monitored information belongs to the interval composed of these values (line 12).

6 EVALUATION AND RESULTS

In this chapter, the details of the NSM evaluation are presented based on the definitions presented in the previous sections. First, the data set used in the evaluation of NSM as well as the parameters involved in the experiments are presented in Section 6.1. Then, the results obtained through the experimental evaluation are presented and discussed, considering their accuracy in Section 6.2. Note that by accuracy we mean the correct detection of positive anomalies, *i.e.,* true positives. In Section 6.3, the accuracy of the qualitative anomaly detection algorithms is compared for types of anomalies. Then, in Section 6.4, the accuracy evaluation of NED is presented, considering different sample sizes. Next, Section 6.5 presents a time comparison among MED, NED, and a standard element-to-element comparison mechanism. Finally, the detection times of the Detector block operational modes are analyzed in Section 6.6.

6.1 Evaluation Parameters

To define the network scenarios exemplified in Subsection 5.1 as well as to answer the RQ II, we should consider the particular characteristics and information of such scenarios. In the defined scenarios, such characteristics are reflected in the number of NFV elements present in the network environment, *i.e.,* VNFs, SFCs, and their respective configurations. According to Sherry *et al.* (SHERRY et al., 2012), the number of network functions composing SFCs on large scale enterprise networks is around 100, where each SFC is composed of 2 to 7 VNFs. Considering the analysis of Sherry *et al.*, Rankothge *et al.* (RANKOTHGE et al., 2017a) have defined an algorithm to characterize such environments, where the number of VNFs for a given customer follows a truncated power-low distribution with exponent 2, minimum 2 and maximum 7. For data set generation, an algorithm based on Rankothge *et al.* equation for SFC composition in large-scale enterprises (RANKOTHGE et al., 2017a) is used. The original algorithm was adapted to include information regarding VNF characteristics, *i.e.,* members VDUs, connection points, and virtual links. The data set fields are summarized in Table 6.1.

Generated data sets include information regarding registered customers and respective SFCs, as well as VNF composition and traffic passing through each SFC (customers' bandwidth). SFCs and VNFs have a unique identifier (ID) and a set of connection points, represented by source and destination IPs for SFCs and virtual network interfaces

Table 6.1: Fields and examples of values from the data set used in evaluations

SFC fields		**VNF fields**	
Field	*Example*	*Field*	*Example*
ID	1	ID	1
Customer	C1	Connection points	eth0,eth1
Bandwidth	27	Virtual links	1,2
Source-Destination IPs	10.0.0.1-10.0.0.2	Number of VDUs	3
VNFs	1 1 3 1 1 5	–	–

for VNFs. VNFs also have a set of virtual links connecting them to the others VNFs in the chain. By monitoring connection points and virtual links, it is possible to identify duplication of flows, *i.e.,* when an abnormal connection point or a virtual link was created for duplicating incoming or outgoing flows of an SFC or VNF. Moreover, each SFC also contains information regarding the customer they are supplying, as well as the bandwidth assigned to that customer. Variations in the bandwidth configured may indicate, among other threats, an unauthorized privilege for a given customer. Finally, SFCs are composed of one or more VNFs, while VNFs are composed of one or more VDUs. Monitoring VNFs and VDUs is important to identify situations when unauthorized elements are inserted in the chain for malicious purposes, such as copying messages or changing the SFC functionality. The evaluation parameters used in each experiment are summarized in Table 6.2 and explained in details in their respective sections.

Table 6.2: Parameters used in each experiment

Parameter	**Value**
Number of SFCs	from 1 to 192
VNFs per SFC	3
VDUs per VNF	3
Number of customers	from 5 to 40
MED polling interval	from 1 to 60 minutes
Catalogued information events	from 1 to 10 per hour
NED number of samples	from 5 to 100
NED β values	0.5, 1 and 2 (first experiment); 1 (other experiments)
Anomalies considered	Unregistered SFCs, missing SFCs, unauthorized changes in the SFC, unauthorized changes in the customers' bandwidth
Anomalies likelihood	60% (single or multiple)
Machine used	Intel i5 2,5GHz; 8GB RAM.
Confidence level	99%

All experiments have been repeated until a confidence level of 99% is achieved, using an Intel i5 $2,5$GHz computer with 8GB of RAM. Following enterprise reports, anomalies are injected in the data set with a likelihood of 60% (ANSTEE et al., 2017) during the experiments. Moreover, to evaluate the *event-based* operational mode of the

Detector block, we also randomized the creation of new cataloged information, *i.e.,* the registration of new NFV elements, such as SFCs and VNFs. Four anomaly types are considered: (*i*) unregistered SFCs; (*ii*) missing SFCs; (*iii*) unauthorized changes in the SFC, such as additional, missing, and switched VNFs; and (*iv*) unauthorized changes in the customers' bandwidth. The evaluations performed in the upcoming sections support the use of this data set to answer the RQ II: what information should be analyzed to keep the NFV environment safe and how such information should be acquired.

6.2 Overall Detection Accuracy

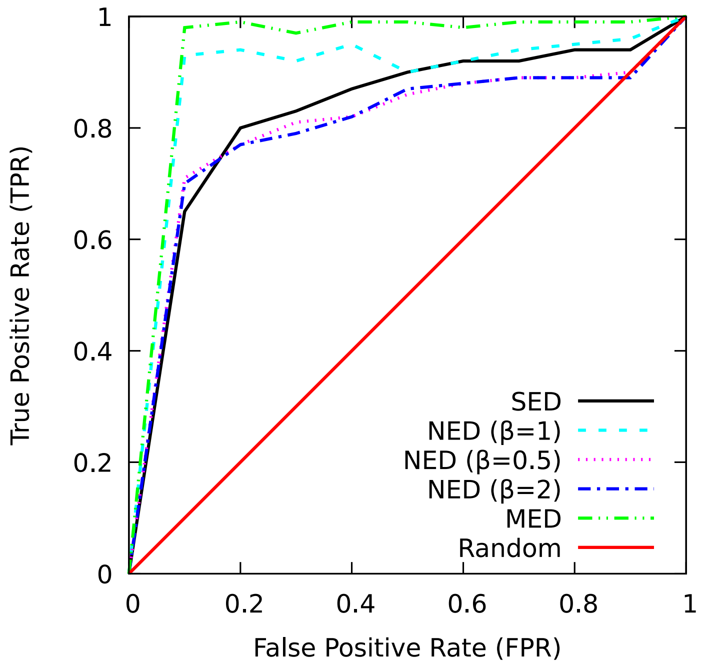
This experiment presents the accuracy of the anomaly detection mechanisms, as well as the false-negative rate of each mechanism. The number of customers currently registered in the network varies from 5 to 40, each one following the specifications for SFC deployments in large enterprises discussed in Section 5.1. Additionally, NED starts its operation with 100 samples, and the value of β was changed three times: 1 (no change in the interval size), 0.5 (half-sized interval), and 2 (double-sized interval). The results obtained are depicted in Figure 6.1 through a Receiver Operating Characteristic curve (ROC curve) (FAWCETT, 2006).

It is important to emphasize that each detector is evaluated analyzing the information they are designed for, *i.e.,* qualitative information for SED and MED, and quantitative for the NED, since they could show high false-positive rates when out of their scope. Therefore, anomalies in quantitative information were not considered in the evaluation of SED and MED, while qualitative information was not considered in NED experiments.

In general, MED presents higher accuracies in all cases investigated, followed by NED, and SED, which presents the lowest accuracy. The best results achieved by MED can be credited to the composition of the merged list, which reduces the occurrence of false-negatives by merging cataloged and monitored information to calculate one entropy.

NED with $\beta = 1$ presents an accuracy slightly smaller than the MED (95% on average). Such behavior can be assigned to scenarios where small changes in the bandwidth may not be detected, especially during the beginning of its execution, when there are still few monitored values to calculate the mean and standard deviation that better characterize the entropy of the monitored elements. Thus, NED can present false detections until it has enough entropy calculations (or samples) to properly characterize the monitored information pattern (100 samples according to da Silva *et al.* (SILVA et al., 2016)). An in-depth

Figure 6.1: ROC curve comparing the accuracy results of the Single Entropy-based Detector (SED), the Merged Entropy-based Detector (MED), and three β configurations for the Numerical Entropy-based Detector (NED).



evaluation of NED accuracy and the number of samples used is provided in Section 6.4.

The size of the interval used as threshold detection by NED was modified to evaluate its accuracy using the β parameter. With a bigger β (2), NED accuracy decreased. Such behavior may occur given the higher tolerance when using a bigger β value (*i.e.,* NED might consider greater changes in the entropy as normal when they might be anomalies). Same way, using a smaller β value may restrict too much the analyzed samples, and normal information might be considered anomalies.

The sub-optimal results for lower numbers of customers presented by SED are directly related to the type of anomalies that occur in the monitored information. More details on the influence of specific anomaly types on the accuracy of SED and MED are provided in the next section.

6.3 Influence of Anomaly Type

This evaluation regards the accuracy of the qualitative detectors implemented in the face of different anomaly types. Since SED may present different accuracies for specific types of anomalies, isolating such anomalies is important to identify which threats each detector is suitable. Two different types of anomalies can occur with monitored information: (*i*) changes in values configured for NFV elements, and (*ii*) unregistered/missing NFV elements in the monitored information. These anomalies types are individually analyzed and the accuracy results are presented in Figures 6.2 and 6.3.

Figure 6.2: Accuracy of the Single Entropy-based Detector (SED) and the Merged Entropy-based Detector (MED) in face of changes in monitored element information

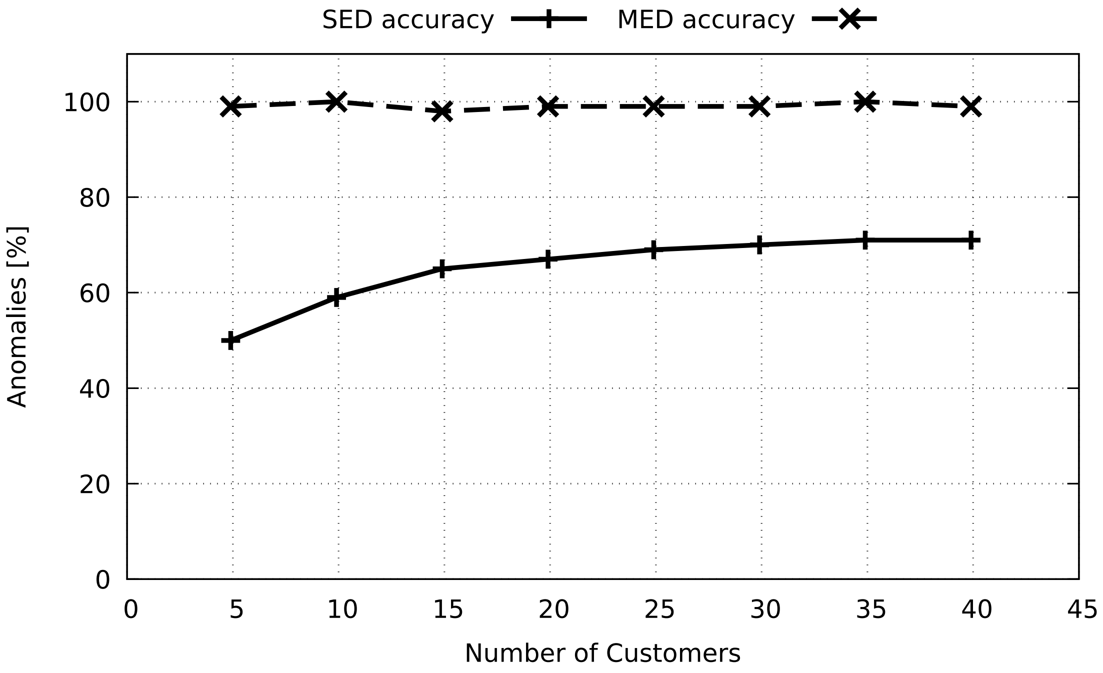
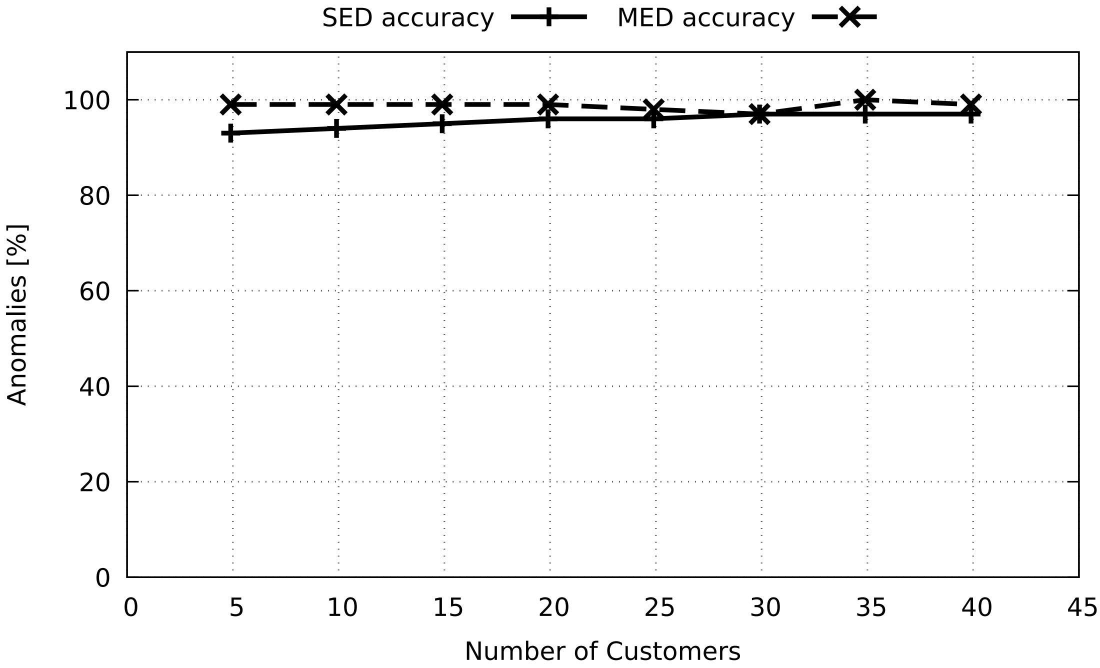


Figure 6.2 shows the accuracy results of SED and MED in the face of anomalies of type (*i*) only. In such a scenario, SED presents an accuracy of 50% for 5 customers, which increases as the number of customers increases until reaching 70%. These values can be explained due to the occurrence of changes in the monitored information that cause the same impact in the entropy value, but with opposite signs. MED presents accuracies around 99% for all number of customers. In turn, when considering anomalies only of type (*ii*) (Figure 6.3), both detectors present accuracies higher than 90%, until achieving 97% of accuracy for 40 customers, while MED keeps its accuracy around 99% for all number of customers. The explanation of such improvements lies in the presence of anomalies only related to unregistered/missing elements, which usually do not present the same impact in the entropy values, as discussed in our previous work (BONDAN et al., 2017b). Thus, SED is not the preferred option to detect threats related to changes in the configuration of the monitored elements, such as flow duplication, unauthorized access, and unauthorized privileges (see Table 5.1). However, SED is still an option to

Figure 6.3: Accuracy of the Single Entropy-based Detector (SED) and the Merged Entropy-based Detector (MED) in face of unregistered/missing monitored elements



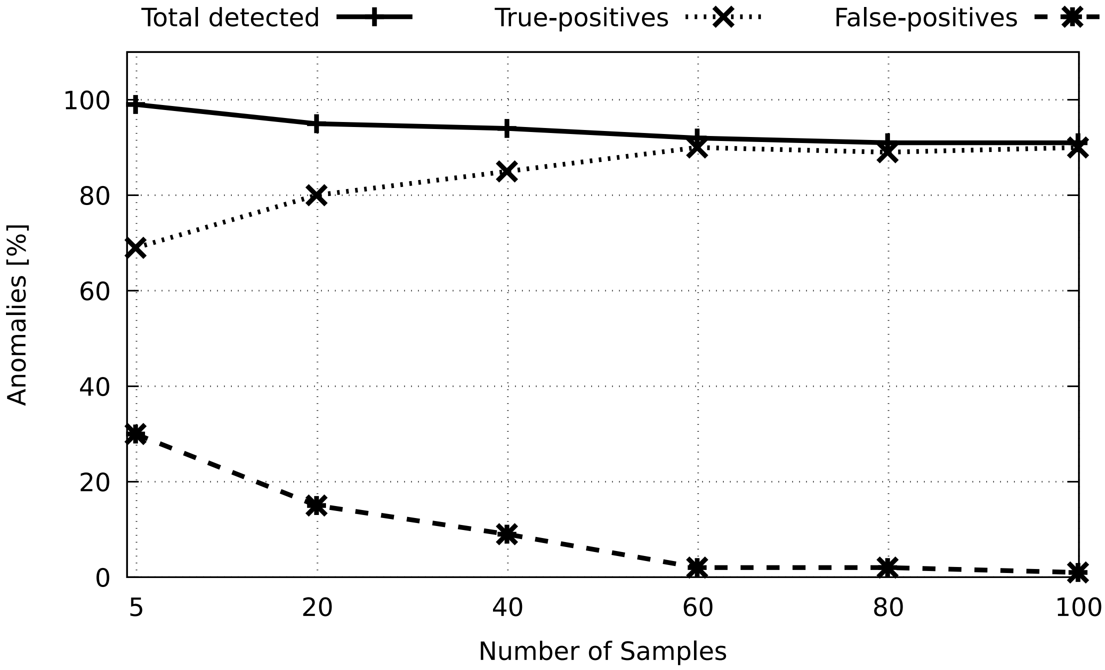
detect the occurrence of DoS when NFV elements are missing or new ones are inserted in an SFC to deny service delivery.

6.4 Influence of Sample Size

To evaluate the impact of the sample size in the accuracy of the quantitative detector, the number of samples used by NED is varied from 5 to 100, considering 40 customers. Such samples are obtained by executing the entropy calculation of monitored information during a trusted interval, *i.e.,* without the occurrence of anomalies (a cold-start phase). The parameter monitored in this evaluation is the bandwidth configured for each customer. The results of this evaluation are summarized in Figure 6.4.

With a few samples, NED presents high relatively false-positives rates. Such behavior occurs due to the limited amount of information provided to NED for composing the numeric interval in which entropies are not considered anomalies. With few samples, this interval tends to be small or inaccurate, so small variations in the entropy will be classified as anomalies even when they are not. With 5 samples, the number of anomalies detected is close to 100%, but above 75% are real anomalies (true-positives). This behavior occurs because the total number of detected anomalies is composed of actually

Figure 6.4: True and false-positive results of the Numerical Entropy-based Detector (NED) varying the number of network samples (cold-start).



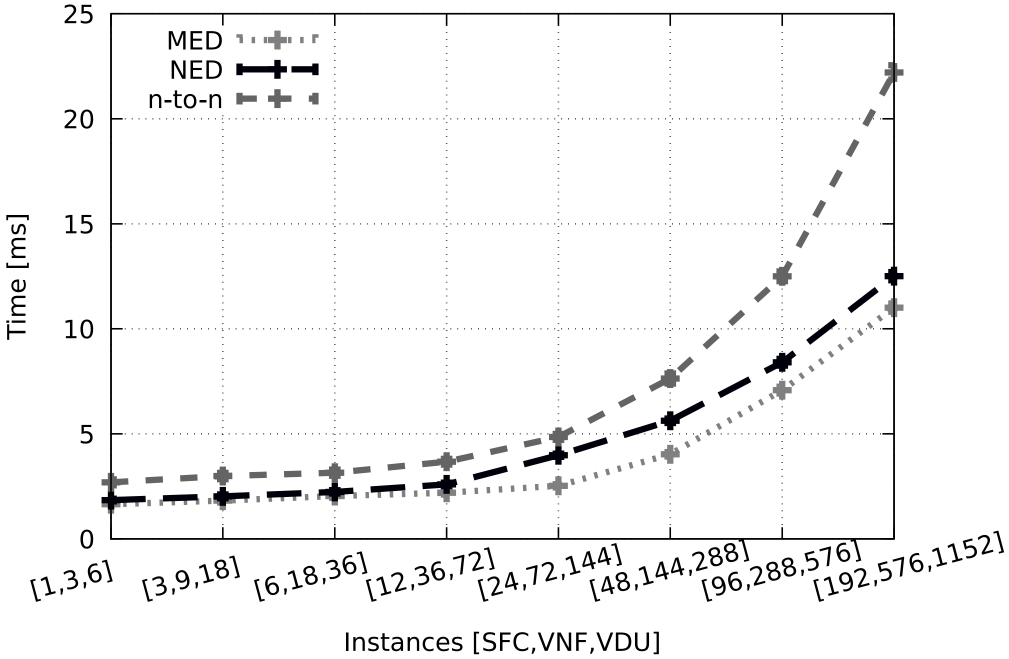
detected anomalies as well as false-positives. As the number of samples increases, the false-positive rate decreases, as well as the total amount of detected anomalies. Even with a few samples, the two-leveled approach of NSM (detection and filtering) properly handles the occurrence of false-positives in the filtering process. However, false-positives should be kept as few as possible to avoid the activation of time-expensive filters.

6.5 Execution Time

The time spent by anomaly detection mechanisms to analyze the data set was evaluated for (*i*) MED, (*ii*) NED, and (*iii*) a direct comparison of monitored and cataloged information, comparing element by element on such lists (n-to-n). The number of instantiated SFC varied from 1 to 192, each one composed of 3 VNFs and up to 3 VDUs. The results are depicted in Figure 6.5.

Entropy-based solutions are considered light-weight anomaly detection mechanisms, suitable for performing the first evaluation of the monitored information to detect anomalies. In the execution time evaluation, both entropy-based mechanisms were faster than comparing element by element from the monitored and cataloged information lists for all number of instances (deployed elements) evaluated. As can be observed in Fig-

Figure 6.5: Execution time comparison of MED, NED, and element-to-element (n-to-n) versions of the anomaly detection mechanism.



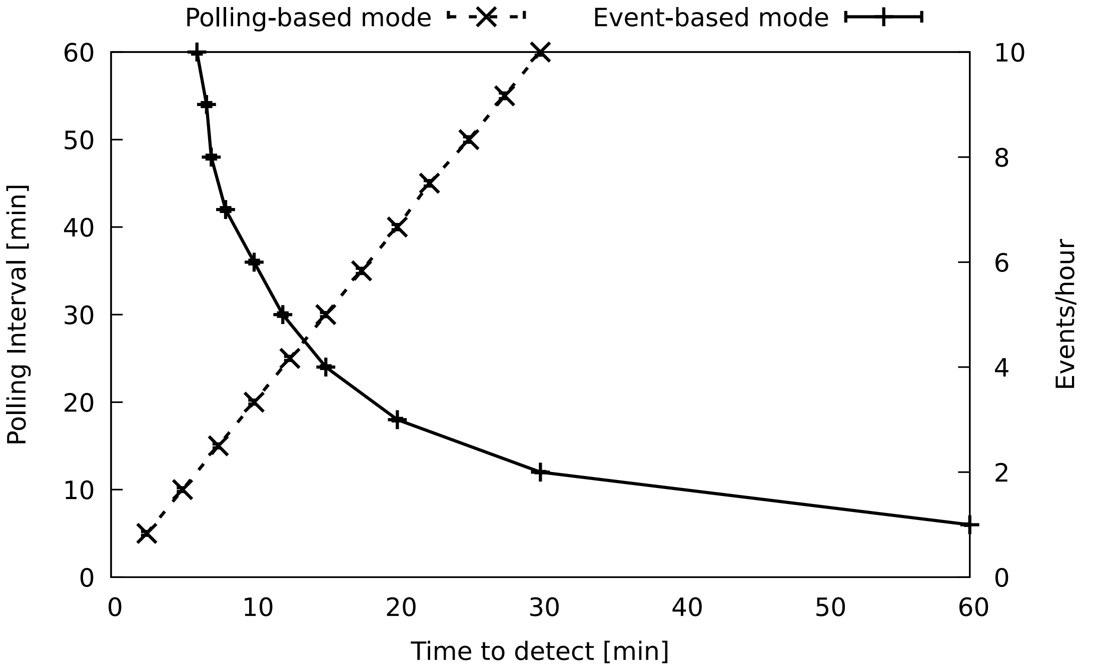
ure 6.5, although all execution times are close to each other for small deployments (*e.g.,* above 6 SFCs, 18 VNFs, 36 VDUS), as the number of instances increases, the entropy-based mechanisms present a less accentuated growth.

Among the implemented entropy-based mechanisms, MED was faster than NED in all evaluations. Such results can be credited to the preprocessing needed for MED, which needs to compose a merged list of monitored and cataloged information. SED, in turn, requires the calculation of the mean and standard deviation to compare the monitored entropy but using Welford's method the list of entropies should be accessed just once, saving time in this calculation. Additionally, incremental mean calculation algorithms can be implemented to save more time in each analysis of the quantitative mechanism.

6.6 Influence of Operational Mode

This evaluation compares the average time before an anomaly is detected in each operational mode. In this experiment, MED is used due to its higher accuracy for qualitative information. The occurrence of anomalies is simulated in an interval of 60 minutes, with a likelihood of 60% to occur every minute, which may present single or multiple anomalies in each occurrence. The *event-based* mode is configured to execute every time a new event related to the cataloged information occurs, such as the registration of new VNFs or configuration changes in the existing ones. The occurrence of such events (anomalous or not) varies from 1 to 10 per hour in an interval of 60 minutes. The *polling-based* operational mode is configured to analyze the monitored information in intervals from 1 to 60 minutes. The results obtained are depicted in Figure 6.6. The order of magnitude for the detection times of both operational modes is directly related to the order of magnitude configured in the experiment. Wide intervals (from minutes to hours) are considered in this experiment, but in high-sensitive network scenarios, such intervals can be reduced to seconds or milliseconds to guarantee faster detection times.

Figure 6.6: Comparison of the detection time on both Detector operational modes



As the polling interval increases, the time needed to detect anomalies increases linearly in the *polling-based* mode. On average, the detection time takes half of the polling interval value to detect anomalies in the *polling-based* mode. In turn, the *event-based* mode presents an exponential decreasing time to detect anomalies as the number of events per hour increases linearly. There is an intersection point where both *event-based* and *polling-based* modes present the same detection time, at 12 min when 5 events per hour and a polling interval at 24 min are used. In highly dynamic scenarios, where information regarding cataloged NFV elements changes often, the *event-based* mode tends to be the best option since it will be able to quickly detect anomalies without performing unnecessary analysis (which may occur in the *polling-based* mode).

The effectiveness of the *polling-based* mode is directly related to the time interval configured. Short polling intervals are the best way to detect anomalies quickly but imply in more NSM executions, which may overload the NFVO with NFVO-to-NSM communication. Higher polling intervals imply less anomaly detection executions, but anomalies will take longer to be detected. Moreover, different strategies can be applied to improve the *event-based* detection, such as performing the anomaly detection every time the NFVO acquires information from the NFV elements (monitoring events). However, such strategy may increase the processing time of both NSM and NFVO, and NFVOs should be able to perform many operations over monitored information in parallel, *i.e.,* acquire monitored information, send it to NSM, receive the anomaly detection results and apply the suggested actions.

7 CONCLUSIONS

This thesis has presented and discussed the advancements in NFV, from its definition to the most recent proposals regarding the integrity and security of NFV elements in emerging network environments. A classification for NFV threats was presented, introducing security domains where the main threats for different NFV elements are organized. In addition, an NFV Security Module (NSM) was proposed as an additional module to the ETSI NFV architectural framework, where anomaly detection mechanisms communicate directly with NFV orchestrators, analyzing the operation of NFV elements operating under their control and providing hints to network operators to overcome threats in their NFV environments.

Considering the lack of solutions designed to guarantee the integrity of NFV element in emerging NFV-environments, as well as existing solutions designed to detect anomalous behaviors in different scenarios, through extensive research, prototypical development, and experimentation, this thesis aimed to verify the following hypothesis:

***Hypothesis*: the employment of anomaly detection mechanisms in conjunction with network orchestrators can properly identify anomalous behaviors related to security threats to NFV virtualization elements in different networking environments.**

The investigations conducted and the results presented in this thesis set a clear path towards supporting the proposed hypothesis. The experiments conducted have demonstrated the effectiveness of anomaly detection mechanisms applied to identify potential threats in NFV environments, considering a case study encompassing two network scenarios: monolithic and hosted virtual network operators. Through realistic data sets, different entropy-based anomaly detection mechanisms have been designed to validate NSM in such scenarios.

The accuracies obtained using different solutions have been evaluated, varying specific parameters of each mechanism to analyze their impact on the final accuracy obtained as well as the mechanism more suitable for each type of threat. Additionally, the false-negative rate of NED was analyzed, decreasing as the number of samples increases until achieving 1% using 100 samples. Moreover, NSM operational modes have been analyzed in terms of the average time needed to detect anomalies. Accuracies above 90% have been achieved using MED and NED, while SED has presented 81% of accuracy at its best, considering its limitations. Therefore, entropy can be used as a general anomaly detector, operating in conjunction with refined filtering processes to specify and mitigate threats in the NFV environment.

Based on the work presented in this thesis, it is possible to identify evidences to answer the research questions (RQs) associated with the hypothesis that have been proposed to guide this study. The answers to each question are detailed as follows.

RQ I. *What are the threats that may affect NFV environments?*

Answer: Security in NFV is an emerging subject, receiving increasing attention from both industry and academia. Although NFV has been explored to provide solutions to overcome traditional networking security vulnerabilities, NFV itself presents specific threats resulting from the combination of generic networking and virtualization threats. In this thesis, through an in-depth investigation of both NFV-specific threats from the intersection of virtualization and networking threats, NFV security threats have been organized accordingly to the NFV elements and respective vulnerabilities of each element.

The classification presented considers security threats that may affect the main NFV elements, organizing them according to different NFV security domains. The main research avenues and solutions to threats related to each NFV security domain were presented, divided into NFV-based solutions for existing network environments and proposals to improve the security of NFV emerging environments. By presenting an overview of solutions available to overcome the identified threats, network operators can decide which approach is the most suitable to cover the vulnerabilities of their environments.

RQ II. *What information should be analyzed to keep the NFV environment safe and how such information should be acquired?*

Answer: In NFV environments compliant to the ETSI architectural framework, every element (*i.e.,* virtual machine, VNF, SFC, etc.) should be cataloged before being used. Once deployed, such elements are monitored by NFVOs to adjust their operation according to demand, shutdown elements when they are not needed anymore, migrate elements to meet QoS metrics, among other reasons. As such, not only operational information but also cataloged information must be analyzed to keep the integrity of NFV elements in the network.

As cataloged information remains stored in the catalogs, it can be acquired directly from them. In addition, operational information is handled by NFVOs. As such, the best way to acquire and analyze operational information is taking advantage of NVFOs' control over NFV elements to request operational information to it. Moreover, NFV information is found in two different natures: qualitative (*e.g.,* identifiers, IP addresses, member VNFs) and quantitative (*e.g.,* customers' bandwidth), requiring specific analyses for each nature. For this reason, different anomaly detection mechanisms were designed for the two types of information available, obtaining accuracies above 90% on average, using qualitative (MED) and quantitative (NED) detectors.

RQ III. *How to provide a flexible way to analyze different threats in NFV environments?*

Answer: ETSI NFV architectural framework does not provide any direction in how to protect NFV environments against attacks, arguing that such mechanisms should be provided apart from their framework. After analyzing different information types of NFV elements and how any disturbance in such information may indicate a threat to the NFV environment, anomaly detection mechanisms appeared as the most suitable solutions to keep NFV elements safe. Anomaly detection solutions are used in different network scenarios, given its flexibility to operate with almost any kind of information.

This thesis has shown the effectiveness in applying anomaly detection mechanisms to guarantee the safety of NFV elements by adding a new module to the standard ETSI NFV architectural framework. Such a module does not interfere in the operation of the original modules presented by ETSI since it communicates with NFVOs using their standard communication APIs. By adding the proposed NSM, it is possible to implement different anomaly detection mechanisms for information of different natures (*i.e.,* qualitative and quantitative), with proven effectiveness in detecting disturbances in NFV elements' operation.

7.1 Future Work

Challenges and open issues that still need further investigations were also identified through the studies conducted, which can be subject to future work. The employment of autonomic verification mechanisms to validate VNF and SFC operation, AAA solutions to avoid user authentication flaws, trustable intra-API communication, are some examples of security-research topics identified in this thesis that deserve attention from the NFV community. As future research, we aim to extend NSM detection to consider real-time information regarding resource consumption by virtual machines and container engines (*e.g.,* CPU, RAM, disk).

Analyzing NSM operation in production networks is another important step to extend our research, since production networks may present unpredicted behaviors, such as communication problems between NVFOs and other network elements. To do so, we plan to employ NSM in the FENDE platform (BONDAN et al., 2018; BONDAN et al., 2019), which provides a marketplace for acquiring and executing VNFs and SFCs while fully compliant with ETSI's NFV architectural framework[1]. Furthermore, machine learning mechanisms can be investigated to improve both detection and filtering process by analyzing past NSM executions.

[1] More details about FENDE project can be found in the demonstration presented at SIGCOMM 2018 (Annex B) and the article published in IEEE COMMAG (Annex A)

REFERENCES

AGUADO, A. et al. Secure NFV Orchestration Over an SDN-Controlled Optical Network With Time-Shared Quantum Key Distribution Resources. **Journal of Lightwave Technology**, v. 35, n. 8, p. 1357–1362, Apr. 2017.

AGUADO, A. et al. Virtual network function deployment and service automation to provide end-to-end quantum encryption. **IEEE/OSA Journal of Optical Communications and Networking**, v. 10, n. 4, p. 421–430, Apr. 2018.

ALHARBI, T.; ALJUHANI, A.; LIU, H. Holistic DDoS Mitigation Using NFV. In: **Proceedings... IEEE Annual Computing and Communication Workshop and Conference (CCWC)**. Las Vegas, NV, USA: IEEE, 2017. p. 1–4.

ALJUHANI, A.; ALHARBI, T.; LIU, H. XFirewall: A Dynamic and Additional Mitigation Against DDoS Storm. In: **Proceedings... International Conference on Compute and Data Analysis (ICCDA)**. New York, NY, USA: ACM, 2017. p. 1–5.

ANSTEE, D. et al. **Worldwide Infrastructure Security Report**. White Paper, 2017. Available at: https://www.arbornetworks.com/insight-into-the-global-threat-landscape. Accessed on January, 2019.

ASHKTORAB, V.; TAGHIZADEH, S. Security Threats and Countermeasures in Cloud Computing. **International Journal of Applications and Innovation in Engineering and Management**, v. 1, n. 2, p. 234–245, Oct. 2012.

AVIZIENIS, A. et al. Basic Concepts and Taxonomy of Dependable and Secure Computing. **IEEE Transactions on Dependable and Secure Computing**, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 1, n. 1, p. 11–33, jan. 2004.

BANSE, C.; SCHUETTE, J. A Taxonomy-based Approach for Security in Software-Defined Networking. In: **Proceedings... IEEE International Conference on Communications (ICC)**. Paris, France: IEEE, 2017. p. 1–6.

BARI, F. et al. Orchestrating Virtualized Network Functions. **IEEE Transactions on Network and Service Management**, v. 13, n. 4, p. 725–739, Dec. 2016.

BASILE, C. et al. Inter-function Anomaly Analysis for Correct SDN/NFV Deployment. **International Journal of Network Management**, Wiley-Interscience, New York, NY, USA, v. 26, n. 1, p. 25–43, jan. 2016.

BASILE, C. et al. A Novel Approach for Integrating Security Policy Enforcement With Dynamic Network Virtualization. In: **Proceedings... IEEE Conference on Network Softwarization (NetSoft)**. London, UK: IEEE, 2015. p. 1–5.

BELLAVISTA, P. et al. Extensible Orchestration of Elastic IP Multimedia Subsystem as a Service Using Open Baton. In: **Proceedings... IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)**. San Francisco, USA: IEEE, 2017. p. 88–95.

BEREZINSKI, P.; JASIUL, B.; SZPYRKA, M. An Entropy-Based Network Anomaly Detection Method. **Entropy**, v. 17, n. 4, p. 2367–2408, 2015.

BERNINI, G. et al. **VNF Pool Orchestration For Automated Resiliency in Service Chains**. Online, 2017. Available at: https://datatracker.ietf.org/doc/html/draft-bernini-nfvrg-vnf-orchestration-04. Accessed on September, 2019.

BLAISE, A.; WONG, S.; AGHVAMI, A. H. Virtual Network Function Service Chaining Anomaly Detection. In: **Proceedings... International Conference on Telecommunications (ICT)**. [S.l.: s.n.], 2018. p. 411–415.

BONDAN, L. et al. FENDE: Marketplace-Based Distribution, Execution, and Life Cycle Management of VNFs. **IEEE Communications Magazine**, v. 57, n. 1, p. 13–19, Jan. 2019.

BONDAN, L. et al. FENDE: Marketplace and Federated Ecosystem for the Distribution and Execution of VNFs. In: **Proceedings... ACM SIGCOMM Conference on Posters and Demos**. [S.l.]: ACM, 2018. p. 135–137.

BONDAN, L. et al. A Framework for SFC Integrity in NFV Environments. In: **Proceedings... Security of Networks and Services in an All-Connected World**. Zürich, Switzerland: Springer International Publishing, 2017. p. 179–184.

BONDAN, L. et al. Anomaly Detection Framework for SFC Integrity in NFV Environments. In: **Proceedings... IEEE Conference on Network Softwarization (NetSoft)**. Bologna, Italy: IEEE, 2017. p. 1–5.

BOUET, M.; LEGUAY, J.; CONAN, V. Cost-based Placement of vDPI Functions in NFV Infrastructures. In: **Proceedings... IEEE Conference on Network Softwarization (NetSoft)**. London, UK: IEEE, 2015. p. 1–9.

BRISCOE, B. et al. **Network Functions Virtualisation (NFV) - NFV Security: Problem Statement**. Online, 2014. v. 1, 1–50 p. Available at: https://www.etsi.org/deliver/etsi_gs/NFV-SEC/001_099/001/01.01.01_60/gs_NFV-SEC001v010101p.pdf. Accessed on September, 2018.

CARELLA, G. A. et al. Prototyping NFV-based Multi-access Edge Computing in 5G Ready Networks With Open Baton. In: **Proceedings... IEEE Conference on Network Softwarization (NetSoft)**. Bologna, Italy: IEEE, 2017. p. 1–4.

CERRATO, I.; ANNARUMMA, M.; RISSO, F. Supporting Fine-Grained Network Functions Through Intel DPDK. In: **Proceedings... European Workshop on Software Defined Networks (EWSDN)**. Washington, DC, USA: IEEE Computer Society, 2014. p. 1–6.

CHANDOLA, V.; BANERJEE, A.; KUMAR, V. Anomaly Detection: A Survey. **ACM Computing Surveys**, ACM, New York, NY, USA, v. 41, n. 3, p. 15:1–15:58, Jul. 2009.

CHEN, J. et al. Implementing NFV system with OpenStack. In: **Proceedings... IEEE Conference on Dependable and Secure Computing (DESEC)**. Taipei, Taiwan: IEEE, 2017. p. 188–194.

CHEN, J. et al. Toward Discovering and Exploiting Private Server-Side Web APIs. In: **Proceedings... IEEE International Conference on Web Services (ICWS)**. San Francisco, CA, USA: IEEE, 2016. p. 420–427.

CHIOSI, M. et al. **Network Functions Virtualisation (NFV)**. Online, 2012. 1–16 p. Available at: https://portal.etsi.org/NFV/NFV_White_Paper.pdf. Accessed on July, 2018.

CHIOSI, M. et al. **Network Functions Virtualisation (NFV) - Architectural Framework**. [S.l.], 2013. v. 1, 1–21 p. Available at: https://www.etsi.org/deliver/etsi_gs/nfv/001_099/002/01.01.01_60/gs_nfv002v010101p.pdf. Accesed on July, 2018.

CHIOSI, M. et al. **Network Functions Virtualisation (NFV) - Use Cases**. Online, 2013. v. 1, 1–50 p. Available at: https://www.etsi.org/deliver/etsi_gs/nfv/001_099/001/01.01.01_60/gs_nfv001v010101p.pdf. Accessed on July, 2018.

CHOU, L. D. et al. A Security Service on-demand Architecture in SDN. In: **Proceedings... International Conference on Information and Communication Technology Convergence (ICTC)**. Jeju Island, Korea: IEEE, 2016. p. 287–291.

COMBE, T.; MARTIN, A.; PIETRO, R. D. To Docker or Not to Docker: A Security Perspective. **IEEE Cloud Computing**, IEEE, Singapore, v. 3, n. 5, p. 54–62, Sep. 2016.

COUGHLIN, M.; KELLER, E.; WUSTROW, E. Trusted Click: Overcoming Security Issues of NFV in the Cloud. In: **Proceedings... ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization (SDN-NFVSec)**. New York, NY, USA: ACM, 2017. p. 31–36.

DALLA-COSTA, A. G. et al. Maestro: An NFV Orchestrator for Wireless Environments Aware of VNF Internal Compositions. In: **Proceedings... IEEE International Conference on Advanced Information Networking and Applications (AINA)**. Tamkang University, Taipei, Taiwan: IEEE, 2017. p. 484 – 491.

DUNCAN, A. et al. Cloud Computing: Insider Attacks on Virtual Machines during Migration. In: **Proceesdings... IEEE International Conference on Trust, Security and Privacy in Computing and Communications**. New York, USA: IEEE, 2013. p. 493–500.

DURANTE, L. et al. A Model for the Analysis of Security Policies in Service Function Chains. In: **Proceedings... IEEE Conference on Network Softwarization (NetSoft)**. Bologna, Italy: IEEE, 2017. p. 1–6.

ESTEVES, R. P.; GRANVILLE, L. Z.; BOUTABA, R. On the Management of Virtual Networks. **IEEE Communications Magazine**, v. 51, n. 7, p. 80–88, Jul. 2013.

FALLON, L. et al. Using the COMPA Autonomous Architecture for Mobile Network Security. In: **Proceedings... IFIP/IEEE Symposium on Integrated Network and Service Management (IM)**. Lisbon, Portugal: IEEE, 2017. p. 747–753.

FAN, J. et al. GREP: Guaranteeing Reliability with Enhanced Protection in NFV. In: **Proceedings... ACM SIGCOMM Workshop on Hot Topics in Middleboxes and Network Function Virtualization (HotMiddlebox)**. New York, NY, USA: ACM, 2015. p. 13–18.

FAWCETT, T. An Introduction to ROC Analysis. **Pattern Recognition Letters**, v. 27, n. 8, p. 861 – 874, 2006.

FAYAZ, S. K. et al. Bohatei: Flexible and Elastic DDoS Defense. In: **Proceedings... USENIX Security Symposium (USENIX Security)**. Washington, D.C.: USENIX Association, 2015. p. 817–832.

FIREEYE. **Zero-Day Danger: A Survey of Zero-Day Attacks and What They Say About the Traditional Security Model**. Online, 2015. Available at: https://www.fireeye.com/current-threats/recent-zero-day-attacks/wp-zero-day-danger.html. Accessed on September, 2017.

FUNG, C. J.; MCCORMICK, B. VGuard: A distributed Denial of Service Attack Mitigation Method Using Network Function Virtualization. In: **Proceedings... International Conference on Network and Service Management (CNSM)**. Barcelona, Spain: IFIP, 2015. p. 64–70.

FYSARAKIS, K. et al. A Reactive Security Framework for Operational Wind Parks Using Service Function Chaining. In: **Proceedings... IEEE Symposium on Computers and Communications (ISCC)**. Heraklion, Crete, Greece: IEEE, 2017. p. 663–668.

GARDIKIS, G. et al. SHIELD: A Novel NFV-based Cybersecurity Framework. In: **Proceedings... IEEE Conference on Network Softwarization (NetSoft)**. Bologna, Italy: IEEE, 2017. p. 1–6.

GIANNOULAKIS, I. et al. On the Applications of Efficient NFV Management Towards 5G Networking. In: **Proceedings... International Conference on 5G for Ubiquitous Connectivity**. Akaslompolo, Finland: IEEE, 2014. p. 1–5.

GIOTIS, K.; ANDROULIDAKIS, G.; MAGLARIS, B. S. A Scalable Anomaly Detection and Mitigation Architecture for Legacy Networks via an OpenFlow Middlebox. **Security and Communication Networks**, v. 9, p. 1958–1970, Oct. 2015.

GRAY, N. et al. A Priori State Synchronization for Fast Failover of Stateful Firewall VNFs. In: **Proceedings... International Conference on Networked Systems (NetSys)**. Göttingen, Germany: IEEE, 2017. p. 1–6.

GUENANE, F.; NOGUEIRA, M.; SERHROUCHNI, A. DDoS Mitigation Cloud-Based Service. In: **Proceedings... IEEE Trustcom/BigDataSE/ISPA**. Helsinki, Finland: IEEE, 2015. v. 1, p. 1363–1368.

HALPERN, J. M.; PIGNATARO, C. **Service Function Chaining (SFC) Architecture**. Online, 2015. Available at: https://rfc-editor.org/rfc/rfc7665.txt. Accessed on January, 2019.

HAN, B. et al. Security Trust Zone in 5G Networks. In: **Proceedings... International Conference on Telecommunications (ICT)**. Limassol, Cyprus: IEEE, 2017. p. 1–5.

HAWILO, H. et al. NFV: State of the Art, Challenges, and Implementation in Next Generation Mobile Networks (vEPC). **IEEE Network**, v. 28, n. 6, p. 18–26, Nov. 2014.

HERBAUT, N. et al. Service Chain Modeling and Embedding for NFV-based Content Delivery. In: **Proceedings... IEEE International Conference on Communications (ICC)**. [S.l.: s.n.], 2017. p. 1–7.

HERRERA, J. G.; BOTERO, J. F. Resource Allocation in NFV: A Comprehensive Survey. **IEEE Transactions on Network and Service Management**, v. 13, n. 3, p. 518–532, Sep. 2016.

HMAITY, A. et al. Virtual Network Function Placement for Resilient Service Chain Provisioning. In: **Proceedings... International Workshop on Resilient Networks Design and Modeling (RNDM)**. Halmstad, Sweden: IEEE, 2016. p. 245–252.

HOLLICK, M. et al. Toward a Taxonomy and Attacker Model for Secure Routing Protocols. **SIGCOMM Computer Communications Review**, ACM, New York, NY, USA, v. 47, n. 1, p. 43–48, Jan. 2017.

HU, Z.; YIN, Y. A Framework for Security on Demand. In: **Proceedings... International Wireless Communications and Mobile Computing Conference (IWCMC)**. Valencia, Spain: IEEE, 2017. p. 378–383.

IBM. **An architectural blueprint for autonomic computing**. [S.l.], 2005. Available at: https://www-03.ibm.com/autonomic/pdfs/AC%20Blueprint%20White%20Paper%20V7.pdf. Accessed on March, 2019.

IQBAL, H. et al. Augmenting Security of Internet-of-Things Using Programmable Network-Centric Approaches: A Position Paper. In: **Proceedings... International Conference on Computer Communication and Networks (ICCCN)**. Vancouver, Canada: IEEE, 2017. p. 1–6.

JAEGER, B. Security Orchestrator: Introducing a Security Orchestrator in the Context of the ETSI NFV Reference Architecture. In: **Proceedings... IEEE Trustcom/BigDataSE/ISPA**. Helsinki, Finland: IEEE, 2015. v. 1, p. 1255–1260.

JAKARIA, A. et al. Dynamic DDoS Defense Resource Allocation Using Network Function Virtualization. In: **Proceedings... ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization (SDN-NFVSec)**. New York, NY, USA: ACM, 2017. p. 37–42.

KAO, C.-N. et al. Fast Proxyless Stream-based Anti-virus for Network Function Virtualization. In: **Proceedings... IEEE Conference on Network Softwarization (NetSoft)**. London, UK: IEEE, 2015. p. 1–5.

KITCHENHAM, B. **Procedures for Performing Systematic Reviews**. [S.l.], 2004. Available at: http://www.inf.ufsc.br/ aldo.vw/kitchenham.pdf. Accessed on March, 2019.

KOURTIS, M. A. et al. Statistical-based Anomaly Detection for NFV Services. In: **Proceedings... IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)**. Palo Alto, CA, USA: IEEE, 2016. p. 161–166.

KOURTIS, M. A. et al. Enhancing VNF Performance by Exploiting SR-IOV and DPDK Packet Processing Acceleration. In: **Proceedings... IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN)**. San Francisco, USA: IEEE, 2015. p. 74–78.

LAI, J.; FU, Q. Man-In-the-Middle Anycast (MIMA): CDN User-Server Assignment Becomes Flexible. In: **Proceedings... IEEE Conference on Local Computer Networks (LCN)**. Dubai, UAE: IEEE, 2016. p. 451–459.

LAL, S. et al. Assuring Virtual Network Function Image Integrity and Host Sealing in Telco Cloud. In: **Proceedings... IEEE International Conference on Communications (ICC)**. Paris, France: IEEE, 2017. p. 1–6.

LI, G. et al. Rule Anomaly-Free Mechanism of Security Function Chaining in 5G. **IEEE Access**, v. 6, Mar. 2018.

LI, X.; QIAN, C. A Survey of Network Function Placement. In: **Proceedings... IEEE Consumer Communications Networking Conference (CCNC)**. Las Vegas, USA: IEEE, 2016. p. 948–953.

LIN, P.; WU, C.; SHIH, P. Optimal Placement of Network Security Monitoring Functions in NFV-Enabled Data Centers. In: **Proceedings... International Symposium on Cloud and Service Computing (SC2)**. [S.l.: s.n.], 2017. p. 9–16.

LIN, Y. H. et al. A Cost-effective Shuffling-based Defense Against HTTP DDoS Attacks with SDN/NFV. In: **Proceedings... IEEE International Conference on Communications (ICC)**. Paris, France: IEEE, 2017. p. 1–7.

LIN, Y. H. et al. Privacy-preserving Deep Packet Filtering over Encrypted Traffic in Software-Defined Networks. In: **Proceedings... IEEE International Conference on Communications (ICC)**. Kuala Lumpur, Malaysia: IEEE, 2016. p. 1–7.

LIU, J. et al. Reliability Evaluation for NFV Deployment of Future Mobile Broadband Networks. **IEEE Wireless Communications**, v. 23, n. 3, p. 90–96, Jun. 2016.

LIU, Y. et al. To Achieve a Security Service Chain by Integration of NFV and SDN. In: **Proceedings... International Conference on Instrumentation Measurement, Computer, Communication and Control (IMCCC)**. Heilongjiang, China: IEEE, 2016. p. 974–977.

LIYANAGE, M. et al. Leveraging LTE Security with SDN and NFV. In: **Proceedings... IEEE International Conference on Industrial and Information Systems (ICIIS)**. Sri Lanka: IEEE, 2015. p. 220–225.

LIYANAGE, M. et al. Enhancing Security of Software Defined Mobile Networks. **IEEE Access**, v. 5, p. 9422–9438, 2017.

LOPEZ, M. A. et al. An Evaluation of a Virtual Network Function for Real-time Threat Detection Using Stream Processing. In: **Proceedings... International Conference on Mobile and Secure Services (MobiSecServ)**. [S.l.: s.n.], 2018. p. 1–5.

LORENZ, C. et al. An SDN/NFV-Enabled Enterprise Network Architecture Offering Fine-Grained Security Policy Enforcement. **IEEE Communications Magazine**, v. 55, n. 3, p. 217–223, March 2017.

LUO, S. et al. How to Defend against Sophisticated Intrusions in Home Networks Using SDN and NFV. In: **Proceedings... IEEE Vehicular Technology Conference (VTC Spring)**. Montréal, Canada: IEEE, 2016. p. 1–5.

LUO, S. et al. A Multi-stage Attack Mitigation Mechanism for Software-Defined Home Networks. **IEEE Transactions on Consumer Electronics**, v. 62, n. 2, p. 200–207, May 2016.

MANSFIELD-DEVINE, S. Fileless Attacks: Compromising Targets Without Malware. **Network Security**, v. 2017, n. 4, p. 7 – 11, 2017.

MASSONET, P. et al. An Architecture for Securing Federated Cloud Networks with Service Function Chaining. In: **Proceedings... IEEE Symposium on Computers and Communication (ISCC)**. Messina, Italy: IEEE, 2016. p. 38–43.

MATIAS, J. et al. FlowSNAC: Improving FlowNAC with Secure Scaling and Resiliency. In: **Proceedings... European Workshop on Software-Defined Networks (EWSDN)**. The Hague, Netherlands: IEEE, 2016. p. 59–61.

MAURICIO, L. A. F.; RUBINSTEIN, M. G.; DUARTE, O. C. M. B. Proposing and Evaluating the Performance of a Firewall Implemented as a Virtualized Network Function. In: **Proceedings... International Conference on the Network of the Future (NOF)**. Búzios, RJ, Brazil: IEEE, 2016. p. 1–3.

MCGRATH, M. J. et al. Performant Deployment of a Virtualised Network Functions in a Data Center Environment Using Resource Aware Scheduling. In: **Proceedings... IFIP/IEEE International Symposium on Integrated Network Management (IM)**. Ottawa, Canada: IEEE, 2015. p. 1131–1132.

MECHTRI, M. et al. NFV Orchestration Framework Addressing SFC Challenges. **IEEE Communications Magazine**, v. 55, n. 6, p. 16–23, 2017.

MEDHAT, A. M. et al. Resilient Orchestration of Service Functions Chains in a NFV Environment. In: **Proceedings... IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)**. Palo Alto, CA, USA: IEEE, 2016. p. 7–12.

MIGAULT, D. et al. A Framework for Enabling Security Services Collaboration Across Multiple Domains. In: **Proceedings... IEEE International Conference on Distributed Computing Systems (ICDCS)**. Atlanta, GA, USA: IEEE, 2017. p. 999–1010.

MIJUMBI, R. et al. Management and Orchestration Challenges in Network Functions Virtualization. **IEEE Communications Magazine**, v. 54, n. 1, p. 98–105, Jan. 2016.

MIJUMBI, R. et al. Network Function Virtualization: State-of-the-Art and Research Challenges. **IEEE Communications Surveys & Tutorials**, v. 18, n. 1, p. 236–262, Mar. 2016.

MONACO, M.; TSANKOV, A.; KELLER, E. Taking the Surprise out of Changes to a Bro Setup. In: **Proceedings... ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization (SDN-NFV Security)**. New York, NY, USA: ACM, 2016. p. 49–52.

MONTERO, D. et al. Virtualized Security at the Network Edge: a User-centric Approach. **IEEE Communications Magazine**, v. 53, n. 4, p. 176–186, Apr. 2015.

NEZARAT, A. A Game Theoretic Method for VM-to-Hypervisor Attacks Detection in Cloud Environment. In: **Proceedings... IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)**. Madrid, Spain: IEEE, 2017. p. 1127–1132.

NGUYEN, V. C. et al. An Experimental Study of Security for Service Function Chaining. In: **Proceedings... International Conference on Ubiquitous and Future Networks (ICUFN)**. Milan, Italy: IEEE, 2017. p. 797–799.

NUNES, B. A. A. et al. A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks. **IEEE Communications Surveys Tutorials**, v. 16, n. 3, p. 1617–1634, Dec. 2014.

OH, S. et al. A Flexible Architecture for Orchestrating Network Security Functions to Support High-level Security Policies. In: **Proceedings... International Conference on Ubiquitous Information Management and Communication (IMCOM)**. New York, NY, USA: ACM, 2017. p. 44:1–44:5.

PAGLIERANI, P. High Performance Computing and Network Function Virtualization: A Major Challenge Towards Network Programmability. In: **Proceedings... IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)**. Constanta, Romania: IEEE, 2015. p. 137–141.

PAN, Y.-H. et al. Fast Computation of Sample Entropy and Approximate Entropy in Biomedicine. **Computer Methods and Programs in Biomedicine**, v. 104, n. 3, p. 382 – 396, 2011.

PARK, T. et al. QoSE: Quality of Security a Network Security Framework with Distributed NFV. In: **Proceedings... IEEE International Conference on Communications (ICC)**. Kuala Lumpur, Malaysia: IEEE, 2016. p. 1–6.

PARK, Y. et al. Dynamic Defense Provision via Network Functions Virtualization. In: **Proceedings... ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization (SDN-NFVSec)**. New York, NY, USA: ACM, 2017. p. 43–46.

PASTOR, A. et al. The Mouseworld, a Security Traffic Analysis Lab Based on NFV/SDN. In: **International Conference on Availability, Reliability and Security (ARES)**. [S.l.: s.n.], 2018. (2018), p. 1–6.

PATTARANANTAKUL, M. et al. SecMANO: Towards Network Functions Virtualization (NFV) Based Security MANagement and Orchestration. In: **Proceedings... IEEE Trustcom/BigDataSE/ISPA**. Tianjin, China: IEEE, 2016. p. 598–605.

PATTARANANTAKUL, M. et al. A First Step Towards Security Extension for NFV Orchestrator. In: **Proceedings... ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization (SDN-NFVSec)**. New York, NY, USA: ACM, 2017. p. 25–30.

PENG, J.; CHOO, K.-K. R.; ASHMAN, H. User Profiling in Intrusion Detection: A Review. **Journal of Network and Computer Applications**, v. 72, n. Supplement C, p. 14 – 27, 2016.

QING, H.; WEIFEI, Z.; JULONG, L. Virtual Network Protection Strategy to Ensure the Reliability of SFC in NFV. In: **Proceedings... International Conference on Information Engineering (ICIE)**. New York, NY, USA: ACM, 2017. p. 17:1–17:5.

QUINN, P.; ELZUR, U. **Network Service Header**. Online, 2016. Available at: https://tools.ietf.org/html/draft-ietf-sfc-nsh-10. Accessed on January, 2017.

QUITTEK, J. et al. **Network Functions Virtualisation (NFV) - Management and Orchestration**. Online, 2014. Available at: https://www.etsi.org/deliver/etsi_gs/nfv-man/001_099/001/01.01.01_60/gs_nfv-man001v010101p.pdf. Accessed on April, 2019.

RANKOTHGE, W. et al. Data Modelling for the Evaluation of Virtualized Network Functions Resource Allocation Algorithms. **Computing Research Repository (CoRR)**, Online, 2017. Available at: http://arxiv.org/abs/1702.00369. Accessed on September, 2017.

RANKOTHGE, W. et al. Optimizing Resources Allocation for Virtualized Network Functions in a Cloud Center using Genetic Algorithms. **IEEE Transactions on Network and Service Management**, PP, n. 99, p. 1–1, 2017.

RASHIDI, B.; FUNG, C. CoFence: A Collaborative DDoS Defence Using Network Function Virtualization. In: **Proceedings... International Conference on Network and Service Management (CNSM)**. Motreal, Quebec, Canada: IEEE, 2016. p. 160–166.

RASHIDI, B.; FUNG, C.; BERTINO, E. A Collaborative DDoS Defence Framework Using Network Function Virtualization. **IEEE Transactions on Information Forensics and Security**, v. 12, n. 10, p. 2483–2497, Oct. 2017.

RAVIDAS, S. et al. Incorporating Trust in NFV: Addressing the Challenges. In: **Proceedings... Conference on Innovations in Clouds, Internet and Networks (ICIN)**. Paris, France: IEEE, 2017. p. 87–91.

RAZAVI, K. et al. Flip Feng Shui: Hammering a Needle in the Software Stack. In: **Proceedings... USENIX Security Symposium (USENIX Security)**. Austin, TX: USENIX Association, 2016. p. 1–18.

RISTENPART, T. et al. Hey, You, Get off of My Cloud: Exploring Information Leakage in Third-party Compute Clouds. In: **Proceedings... ACM Conference on Computer and Communications Security (CCS)**. New York, NY, USA: ACM, 2009. p. 199–212.

SAHAY, R. et al. ArOMA: An SDN based autonomic DDoS mitigation framework. **Computers & Security**, v. 70, n. Supplement C, p. 482 – 499, 2017.

SAHHAF, S. et al. Scalable Architecture for Service Function Chain Orchestration. In: **Proceedings... European Workshop on Software Defined Networks (EWSDN)**. Bilbao, Spain: IEEE, 2015. p. 19–24.

SAHHAF, S. et al. Network Service Chaining with Optimized Network Function Embedding Supporting Service Decompositions. **Computer Networks**, Elsevier North-Holland, Inc., New York, NY, USA, v. 93, n. P3, p. 492–505, Dec. 2015.

SALMAN, O. et al. Software Defined IoT Security Framework. In: **Proceedings... International Conference on Software Defined Systems (SDS)**. Valencia, Spain: IEEE, 2017. p. 75–80.

SALVADOR, P.; NOGUEIRA, A. Customer-side detection of internet-scale traffic redirection. In: **Proceedings... International Telecommunications Network Strategy and Planning Symposium (Networks)**. Madeira Island, Portugal: IEEE, 2014. p. 1–5.

SAUVANAUD, C. et al. Anomaly Detection and Root Cause Localization in Virtual Network Functions. In: **Proceedings... IEEE International Symposium on Software Reliability Engineering (ISSRE)**. Ottawa, Canada: IEEE, 2016. p. 196–206.

SCHAFFER, G. P. Worms and Viruses and Botnets, oh my! Rational Responses to Emerging Internet Threats. **IEEE Security Privacy**, v. 4, n. 3, p. 52–58, May 2006.

SCHöLLER, M. et al. Resilient Deployment of Virtual Network Functions. In: **Proceedings... International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)**. Almaty, Kazakhstan: IEEE, 2013. p. 208–214.

SENDI, A. S. et al. Efficient Provisioning of Security Service Function Chaining Using Network Security Defense Patterns. **IEEE Transactions on Services Computing**, PP, n. 99, p. 1–1, 2017.

SHANNON, C. E. A Mathematical Theory of Communication. **The Bell System Technical Journal**, v. 27, n. 3, p. 379–423, July 1948.

SHERRY, J. et al. Making Middleboxes Someone else's Problem: Network Processing As a Cloud Service. In: **Proceedings... ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication**. Helsinki, Finland: ACM, 2012. p. 13–24.

SHIH, M.-W. et al. S-NFV: Securing NFV States by Using SGX. In: **Proceedings... ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization (SDN-NFV Security)**. New York, NY, USA: ACM, 2016. p. 45–48.

SHIN, M. K. et al. Verification for NFV-enabled Network Services. In: **Proceedings... International Conference on Information and Communication Technology Convergence (ICTC)**. Jeju Island, Korea: IEEE, 2015. p. 810–815.

SHIN, M.-K. et al. **Verification of NFV Services : Problem Statement and Challenges**. Online, 2017. Available at: https://datatracker.ietf.org/doc/html/draft-irtf-nfvrg-service-verification-04. Accessed on September, 2017.

SILVA, A. S. da et al. ATLANTIC: A Framework for Anomaly Traffic Detection, Classification, and Mitigation in SDN. In: **Proceedings... IEEE/IFIP Network Operations and Management Symposium (NOMS)**. Instanbul, Turkey: IEEE, 2016. p. 27–35.

SULATYCKI, R.; FERNANDEZ, E. B. Two Threat Patterns That Exploit "Security Misconfiguration" and "Sensitive Data Exposure" Vulnerabilities. In: **Proceedings...**

European Conference on Pattern Languages of Programs. New York, NY, USA: ACM, 2015. (EuroPLoP), p. 46:1–46:11.

THING, V. L. L.; LEE, H. C. J.; SLOMAN, M. Traffic Redirection Attack Protection System (TRAPS). In: **Proceedings... IFIP International Information Security Conference**. Chiba, Japan: Springer US, 2005. p. 309–325.

THONGTHUA, A.; NGAMSURIYAROJ, S. Assessment of Hypervisor Vulnerabilities. In: **Proceedings... International Conference on Cloud Computing Research and Innovations (ICCCRI)**. [S.l.: s.n.], 2016. p. 71–77.

VACCA, J. R. **Cloud Computing Security: Foundations and Challenges**. Boca Raton, FL, USA: CRC Press, Inc., 2016.

VASSILAKIS, V. G. et al. Security Requirements Modelling for Virtualized 5G Small Cell Networks. In: **Proceedings... International Conference on Telecommunications (ICT)**. Limassol, Cyprus: IEEE, 2017. p. 1–5.

VAUGHAN-NICHOLS, S. J. Virtualization Sparks Security Concerns. **Computer**, v. 41, n. 8, p. 13–15, Aug 2008.

WANG, J. et al. Challenges Towards Protecting VNF With SGX. In: **Proceedings... ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization (SDN-NFV Sec)**. [S.l.: s.n.], 2018. p. 39–42.

WANG, X. et al. Efficient Network Security Policy Enforcement With Policy Space Analysis. **Proceedings... IEEE/ACM Transactions on Networking**, v. 24, n. 5, p. 2926–2938, Oct. 2016.

WANG, Z. et al. A Shared Memory Based Cross-VM Side Channel Attacks in IaaS Cloud. In: **Proceedings... ACM Conference on Computer Communications Workshops (INFOCOM WKSHPS)**. Orlando, FL, USA: ACM, 2016. p. 181–186.

WENDLAND, F.; BANSE, C. Enhancing NFV Orchestration with Security Policies. In: **Proceedings... International Conference on Availability, Reliability and Security (ARES)**. [S.l.: s.n.], 2018. (2018), p. 1–6.

WICKBOLDT, J. A. et al. Resource Management in IaaS Cloud Platforms Made Flexible Through Programmability. **Computer Networks**, v. 68, p. 54 – 70, 2014.

WICKBOLDT, J. A. et al. Software-Defined Networking: Management Requirements and Challenges. **IEEE Communications Magazine**, v. 53, n. 1, p. 278–285, Jan. 2015.

WRIGHT, S.; HU, Y. C.; REID, A. **Network Functions Virtualisation (NFV) - Infrastructure Overview**. Online, 2015. v. 1, 1–59 p. Available at: https://www.etsi.org/deliver/etsi_gs/NFV-INF/001_099/001/01.01.01_60/gs_nfv-inf001v010101p.pdf. Accessed on January, 2017.

XILOURIS, G. et al. T-NOVA: A Marketplace for Virtualized Network Functions. In: **Proceedings... European Conference on Networks and Communications (EuCNC)**. Bologna, Italy: IEEE, 2014. p. 1–5.

XIONG, D.; ZOU, P.; CAI, J. A Study on Policy Conflicts Analysis of Multi-domain Access Control in Clouds. In: **Proceedings... IEEE International Conference on Software Engineering and Service Science (ICSESS)**. Beijing, China: IEEE, 2015. p. 271–274.

YANG, W.; FUNG, C. A Survey on Security in Network Functions Virtualization. In: **Proceedings... IEEE NetSoft Conference and Workshops (NetSoft)**. Seoul, South Korea: IEEE, 2016. p. 15–19.

YUSOP, Z. M.; ABAWAJY, J. Analysis of Insiders Attack Mitigation Strategies. In: **Proceedings... International Conference on Innovation, Management and Technology Research (ICIMTR)**. Malaysia: Springer, 2014. p. 581 – 591.

ZHANG, Y. et al. HomeAlone: Co-residency Detection in the Cloud via Side-Channel Analysis. In: **Proceedings... IEEE Symposium on Security and Privacy**. Oakland, CA. USA: IEEE, 2011. p. 313–328.

ANNEX A PUBLISHED ARTICLE – COMMAG

This article was developed under the Working Group FENDE: Federated Ecosystem for Offering, Distribution, and Execution of Virtual Network Functions, supported by the Brazilian National Research Network (RNP). This article presents the ecosystem developed within the project, which provides a marketplace and ecosystem reference architecture for the distribution and execution of VNFs and composition of Service Function Chains (SFCs).

- **Title:**
  FENDE: Marketplace-based Distribution, Execution, and Lifecycle Management of VNFs
- **Journal:**
  IEEE Communications Magazine: Network & Service Management Series
- **URL:**
  <https://ieeexplore.ieee.org/document/8613268>
- **Date:**
  Published on Jan, 2019

FENDE: Marketplace-Based Distribution, Execution, and Life Cycle Management of VNFs

Lucas Bondan, Muriel F. Franco, Leonardo Marcuzzo, Giovanni Venancio, Ricardo L. Santos, Ricardo J. Pfitscher, Eder J. Scheid, Burkhard Stiller, Filip De Turck, Elias P. Duarte Jr., Alberto E. Schaeffer-Filho, Carlos R. P. dos Santos, and Lisandro Z. Granville

The authors review the historical perspective of networking paradigms and technologies to propose FENDE, a marketplace and ecosystem for the distribution and execution of VNFs and composition of service function chains. Major challenges that must be overcome to promote the adoption of marketplaces in emerging NFV-based networks are investigated and discussed.

Abstract

The emergence of NFV has drawn the attention of academia, standardization bodies, and industry, because of the possibility of reducing capital and operational costs while introducing innovation in computer networks. To enable developers to independently publish and distribute VNFs, marketplaces akin to online application stores are essential. Research efforts in several aspects are necessary to enable wider adoption of such online application stores in emerging NFV-based computer networks. This article reviews the historical perspective of networking paradigms and technologies to propose FENDE, a marketplace and ecosystem for the distribution and execution of VNFs and composition of service function chains. Major challenges that must be overcome to promote the adoption of marketplaces in emerging NFV-based networks are investigated and discussed.

Introduction

Computer networking technologies have evolved in many different ways to support the development and adoption of innovative services. Programmable virtual networking (PVN), software-defined networking (SDN), and network functions virtualization (NFV) are examples of disruptive concepts that have been exploited to offer advanced networking environments which foster innovation. Recently, special attention has been given to NFV and its capability to develop, deploy, manage, and integrate virtualized network functions (VNFs). NFV has begun to be widely adopted by both industry and academia, thus becoming fundamental for providing flexible network services.

As NFV adoption grows, the number of available VNFs also increases, leading to the need for proper solutions to offer and distribute these functions to network operators. We advocate that NFV can benefit from a software offering and distribution model, which has proven to be effective for other technologies. More specifically, following the trend initiated by the Google Play Store and Apple App Store, which popularized the business model where third-party developers are able to offer applications to users of mobile devices, marketplace solutions for NFV have been proposed [1, 2]. However, the available NFV marketplaces are designed for specific scenarios and to fulfill specific demands without considering its adoption in different network scenarios, such as multi-vendor VNF acquisition and service function chaining (SFC) composition. Moreover, such solutions usually provide VNFs' source code for download but do not offer adequate management tools nor the NFV infrastructure (NFVI) to execute VNFs. We argue, on the other hand, that the design of NFV marketplaces should consider three fundamental aspects: VNF offering, life cycle management, and infrastructure management.

Based on fundamental aspects of NFV marketplaces, in this article we propose FENDE (https://gt-fende.inf.ufrgs.br), a marketplace and federated ecosystem for the distribution and execution of VNFs [3]. In FENDE, developers are able to offer their VNF solutions, while customers can acquire them and choose whether to use public or private infrastructures to instantiate the acquired VNFs. In addition, FENDE provides infrastructure support for VNF instantiation, so customers can acquire and execute VNFs through a unified interface. FENDE also delivers all VNF life cycle management operations and SFC capabilities, in which customers can compose chains with the acquired VNFs to deliver network services. FENDE is the first NFV ecosystem that provides a marketplace for VNF offering together with VNF and SFC creation and life cycle management, as well as the infrastructure support needed for VNF and SFC instantiation.

Marketplaces: Historical Perspective

Online marketplaces have evolved alongside the emergence of new technologies. The popularization of smartphones, for example, created a market for apps, which led the main mobile vendors to deploy their marketplaces as a way to offer applications to end users. A historical perspective of online marketplaces is depicted in Fig. 1. Each paradigm corresponds to a horizontal line parallel to the main timeline, on which technologies and marketplaces are plotted. These marketplaces are those that have/had market dominance, regard-

Lucas Bondan, Ricardo L. Santos, Ricardo J. Pfitscher, Alberto E. Schaeffer-Filho, and Lisandro Z. Granville are with UFRGS; Leonardo Marcuzzo and Carlos R. P. dos Santos are with UFSM; Giovanni Venancio and Elias P. Duarte Jr. are with UFPR; Muriel F. Franco, Eder J. Scheid, and Burkhard Stiller are with UZH; Filip De Turck is with UGent-imec.

Digital Object Identifier:
10.1109/MCOM.2018.1800507

0163-6804/19/$25.00 © 2019 IEEE

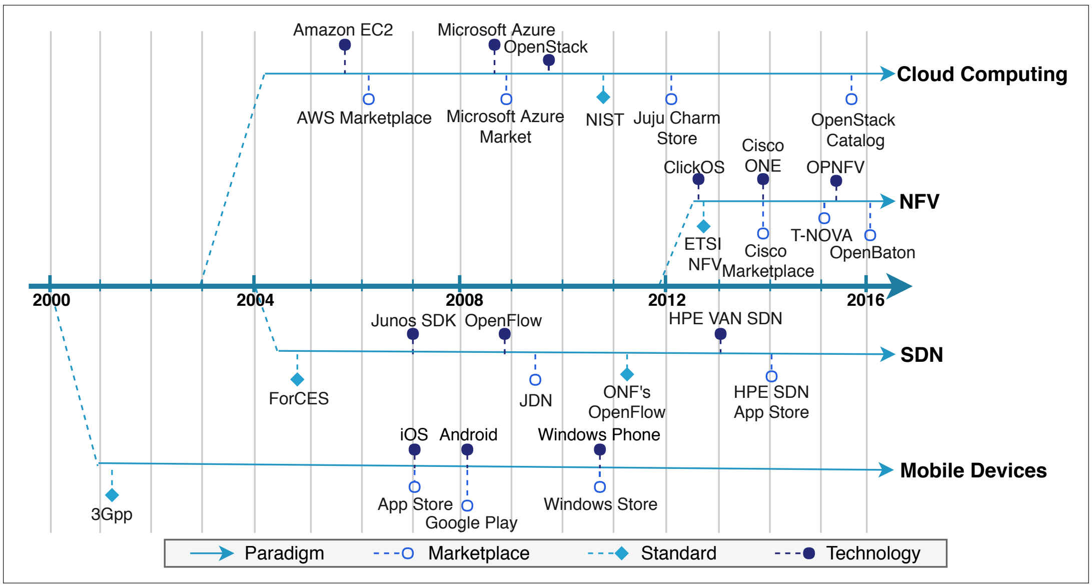


Figure 1. Technologies and marketplace timeline.

Online marketplaces have evolved alongside the emergence of new technologies. The popularization of smartphones, for example, created a market for apps, which led the main mobile vendors to deploy their marketplaces as a way to offer applications to end-users.

ing the amount of customers and services available, in every technology and paradigm that we considered.

The Third Generation Partnership Project (3GPP) released the 3G Networks specifications in the early 2000s, representing a milestone in mobile networks. Later, in the mid-2000s, the popularization of smartphones boosted the mobile market. Although several mobile platforms were developed (e.g., BlackBerry, Symbian, and Windows Mobile), Apple and Google opened up the market of apps by introducing, in 2007 and 2008, respectively, whole ecosystems composed of operating systems (i.e., iOS and Android) and marketplaces (e.g., Apple's App Store and Google Play) to provide apps to smartphone users. In 2010, Microsoft also adopted this strategy by making available Windows Phone and Windows Store. As a result, the combination of mobile platforms and marketplaces allowed third-party developers to offer a broad range of apps. As an illustration, in 2017, Google Play offered around 3.1 million distinct apps, and its revenue is forecasted to add nearly US$10 billion to the world economy [4].

Although cloud computing emerged at the beginning of the 2000s, its consolidation occurred from 2006 onward, when providers started to offer on-demand services over the cloud. In 2006, Amazon announced an on-demand computing platform called Elastic Compute Cloud (EC2), and in 2008 Microsoft entered into this market by launching the Microsoft Azure cloud computing platform. Later, in 2010, OpenStack was also introduced as an open source cloud enabler. Because of the wide adoption of this paradigm, in 2011, the National Institute of Standards and Technology published the NIST definition of cloud computing [5]. Meanwhile, several companies invested in marketplaces to offer an easy way to distribute applications and services over the cloud. Among them, four marketplaces are worth highlighting: Amazon Web Service (AWS) Marketplace, Microsoft Azure Market, Juju Charm Store, and OpenStack Catalog. AWS Marketplace contains a collection of cloud computing services for EC2. Azure Market provides a collection of integrated cloud services with solutions for data storage, database management, mobile services, and networking. Juju Charm is a project under the auspices of Canonical, which consists of a marketplace to enable applications and services modeling for clouds. Finally, OpenStack Catalog hosts ready-to-use applications that customers can deploy within OpenStack clouds.

In computer networks, SDN is a paradigm characterized by decoupling the network control (control plane) from the forwarding functions (data plane) [6]. One of the first relevant attempts to standardize SDN was ForCES, which defined an architectural framework and associated protocols for the communication between control and forwarding elements. In 2007, Juniper released the Junos service development kit (SDK) to allow the development of applications in the Junos OS, and since 2009 Juniper maintains the Juniper Developer Network (JDN) and Juniper Professional Services Marketplace (JPSM) to foster a community of network application developers. At the end of 2008, OpenFlow established itself as the most important SDN implementation. Shortly after the first specification of OpenFlow, the Open Network Foundation (ONF) became responsible for the standardization efforts. In turn, Hewlett-Packard Enterprise (HPE) released the HPE VAN SDN controller at the beginning of 2013. Right after, in 2014, HPE introduced a marketplace for SDN applications (HPE SDN App Store), allowing network end users to deploy services by aligning the network with business needs.

With respect to the virtualization of network functions, between 2012–2013, the European Telecommunications Standards Institute (ETSI) published the NFV architectural framework [7], running in virtual machines (VMs) hosted on commercial off-the-shelf servers. From 2012 onward, several solutions were proposed to accelerate the adoption of NFV, such as open platforms for NFV (e.g., ClickOS in 2012 and OPNFV in 2014) and frameworks that simplify the development of VNFs (e.g., Cisco Open Network Environment [ONE] in 2013). Also, important players have

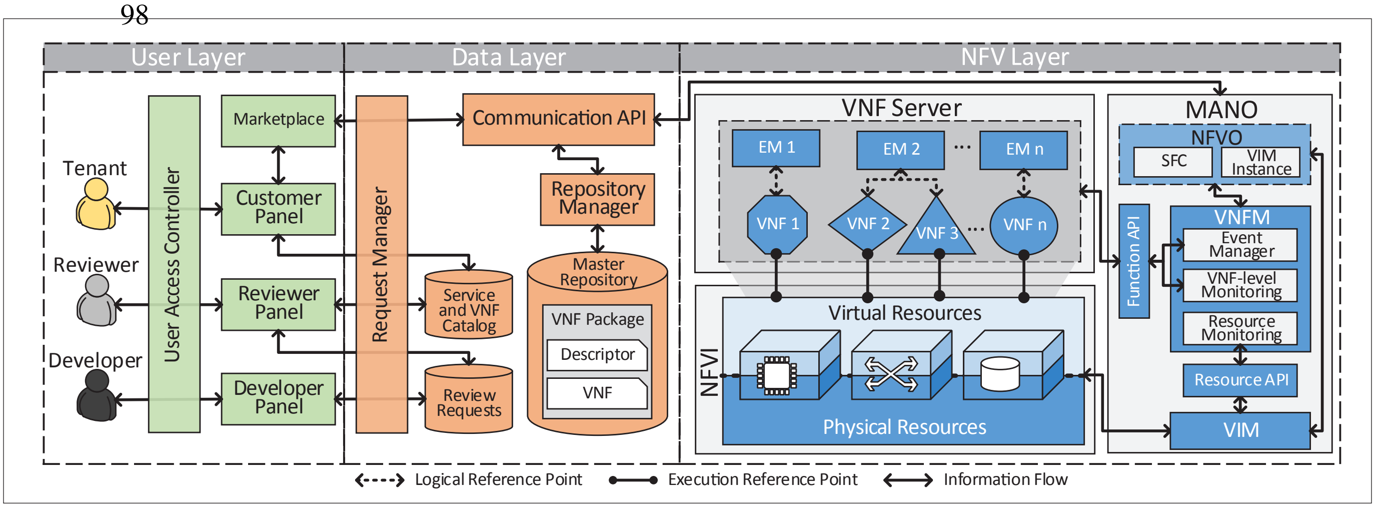


Figure 2. FENDE architecture.

spent efforts to facilitate the distribution of VNFs: Cisco Marketplace (released in 2014) offers applications and hardware solutions from Cisco itself as well as from partner companies, whereas the T-NOVA project (started in 2015) proposes a marketplace that enables network end users to purchase and deploy VNFs according to their demands. In 2016, the Open Baton project made available a marketplace for downloading VNFs compatible with the Open Baton NFV Orchestrator and VNF Managers.

By analyzing both NFV characteristics and general marketplace features, we identified four main requirements for the development of NFV marketplaces: offering, execution, accounting, and management. Today, NFV marketplaces partially cover these requirements. Cisco Marketplace, T-NOVA, and Open Baton offer VNFs and services as well as tools for their configuration. However, only the first offers physical resources for their execution. Cisco and T-NOVA provide a business model where network end users can purchase and deploy VNFs according to their demands. Such a business model, however, does not consider third-party developers offering their solutions in the marketplace. Moreover, there is no possibility of integration with institutions belonging to federated infrastructures, widely disseminated infrastructures/testbeds, or even the usage of private infrastructures for VNF execution. We capitalize on these previous efforts to investigate and present all functionalities needed in a full marketplace ecosystem for VNF offering, execution, accounting, and management.

FENDE: Marketplace and Federated Ecosystem for VNFs

FENDE was designed considering three users: developers, reviewers, and customers. Each interacts with FENDE through dedicated access and management panels that provide all operations needed for marketplace operations. Also, FENDE provides infrastructure support to execute and monitor virtualized functions. Thus, FENDE places itself as the first NFV marketplace solution that provides all functionalities needed for customers to acquire and execute VNFs and SFCs, thus combining marketplace, management, and infrastructure capabilities in one solution.

FENDE Architecture

FENDE is based on the NFV architectural framework defined by ETSI, as illustrated in Fig. 2. The FENDE architecture is divided into three layers, each layer with specific modules for different operational levels.

User Layer: The user layer contains the elements responsible for the interaction between different users with the platform. Developers fill a registration form for VNFs they want to offer, containing information regarding VNF characteristics (e.g., source code and virtualization requirements). Then reviewers analyze registration requests sent by developers before VNFs become available in the marketplace. Once approved, customers can access the marketplace and select the VNFs they want. Acquired VNFs are available in the customer's library, where instances of each VNF can be created. In addition, customers are also able to perform life cycle management operations as well as create SFCs with VNFs acquired.

Data Layer: The data layer handles all information regarding VNFs, SFCs, and FENDE's users. As such, three main databases are designed: Review Requests, containing all developers' requests for VNF registration in the marketplace; Service and VNF Catalog, containing all VNFs and SFCs available for acquisition by customers in the marketplace; and Master Repository, where all VNF descriptors and information on running instances are stored. Once VNFs are accepted and/or instantiated, a series of events must occur in the platform so that other modules can use the information synchronously. To do so, three modules in this layer integrate the user layer with the NFV layer.

Request Manager: Controls the repository of submissions in the Review Requests database and also performs the migration to the catalog when a VNF is accepted.

Communication API: Provides communication between the user and NFV layers. Its main functions are: 1) requesting the creation or update of VNFs' repositories and 2) requesting VNFs' descriptors for instantiation. All modules that need information belonging to the Repository Manager can forward the request through the Communication application programming interface (API).

Repository Manager: Creates, maintains, and manages VNFs' descriptors available in the Catalog. For example, when a repository is accepted, the Repository Manager clones and maintains a local version of that repository.

NFV Layer: This layer brings together the main NFV elements proposed by ETSI, divided into three sublayers.

NFV Management and Orchestration (MANO): Designed to handle operations related to services' and functions' life cycle management, as well as resource sharing among virtual elements. It has three main components.

VNF Manager (VNFM): Responsible for VNF life cycle management operations, such as instantiating, removing and updating VNFs, as well as creating SFCs. To enable both hardware and software-level VNF management, FENDE has three main modules:

- Events Manager, responsible for receiving requests from the user layer and performing VNF life cycle management, activating the two other modules accordingly
- Resource Monitoring, which monitors metrics related to physical resources assigned for each VNF such as CPU, memory, and storage
- VNF-Level Monitoring, which monitors the function of each VNF, collecting metrics related to the function usage, such as number of processed packets and operations latency

Virtualized Infrastructure Manager (VIM): Controls all resources available in the NFVI. FENDE supports different VIMs, using its communication API to abstract technology-specific commands. Thus, FENDE supports the composition of heterogeneous infrastructures, such as local interconnected infrastructures based on the CloudStack and OpenStack platforms;

NFV Orchestrator: Brings intelligence to service provisioning and composition processes, directly interacting with VNFMs for managing VNF operation life cycle. Likewise, NFVI virtual and physical resource sharing orchestration among different virtualized elements is performed by NFVOs through VIMs.

VNF Server: Supports the execution and control of VNFs' operation locally. In virtualized environments, resources in the underlying infrastructure must be abstracted, for example, through hypervisor-based or container-based virtualization. In addition, element managers (EMs) are designed to handle technology-specific information, such as fault, configuration, accounting, performance, and security (FCAPS) parameters. EMs must be co-located with VNFs and retrieve information regarding VNFs' execution, sending such information to the VNFM to control VNFs' operation.

NFV Infrastructure: Corresponds to physical and virtual resources available for VNF deployment, that is, computing, memory, storage, and networking. FENDE supports multiple network domains to compose the NFVI, connecting them through VPNs, so communication among VNFs is possible and instances run on the same subnet, allowing the use of SFCs spanning over multiple domains. Although no elasticity mechanism is currently implemented, FENDE supports the definition of placement optimization mechanisms, which can be used for automated horizontal and vertical scaling [8].

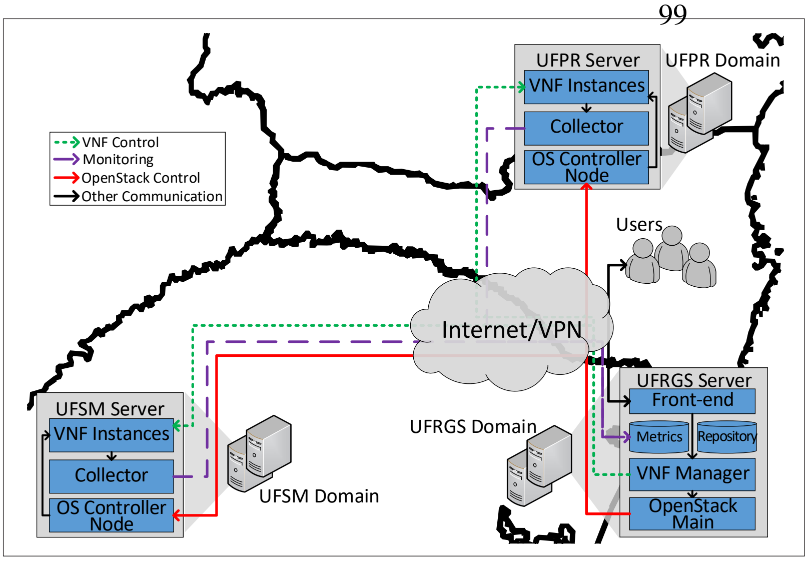


Figure 3. FENDE platform deployment scenario.

FENDE Prototype

The FENDE prototype provides a web interface to enable management and interaction with different users. Each user interface provides resources that enable a set of operations within the ecosystem. Developers must submit a valid Git repository with the VNF source code to be evaluated by reviewers. The current review process is manual, with reviewers analyzing if submitted VNFs perform as described by developers, and checking for the absence of malicious code. However, autonomic revision mechanisms (e.g., bots as used in Google Play) are an interesting research topic to be further explored. Once approved, the Git repository is cloned to the local marketplace repositories, and customers are then able to acquire, instantiate, and manage VNFs and create their own SFCs. In the NFV layer, OpenStack and Tacker were used for management at the hardware level. For software-level management, Click-On-OSv [9] is used. In addition, a VNFM submodule was developed to consume both APIs and to fully manage VNFs' life cycle.

Federated Testbed

In our testbed, the Brazilian National Research Network (RNP) is the marketplace regulator, while the FENDE project is in charge of maintaining the marketplace, that is, responsible for its operation and for the VNF review process. Developers are from both industry and academia (Brazilian universities), and can register their own VNF solution and acquire VNFs for instantiation. On the NFVI layer, cloud infrastructure providers such as Amazon EC2 and Windows Azure could be registered along with the infrastructure provided by FENDE.

FENDE was deployed in three network domains across two different Brazilian states, as depicted in Fig. 3: the UFRGS and UFSM domains in the state of Rio Grande do Sul, and the UFPR domain in the state of Paraná. The marketplace

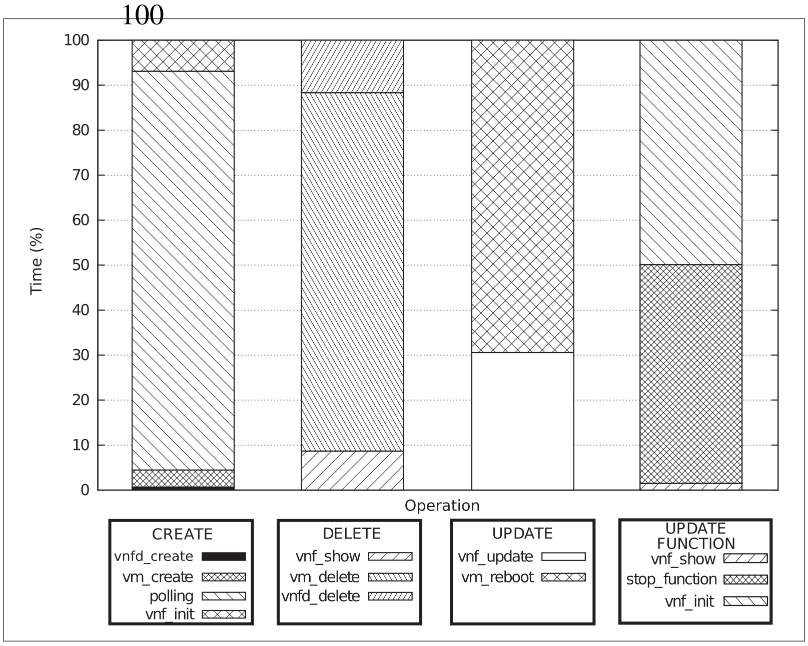


Figure 4. FENDE platform deployment scenario.

and VNF management front-end are instantiated at the UFRGS domain, while the other domains are used as NFVI, hosting VNFs, and SFCs. The prototype supports operations performed by each actor, such as VNF submission, VNF review process, and all operations regarding VNF life cycle management by customers, including statistics retrieval from VNF operation.

Experimental Results

Since VNFs can be instantiated, configured, and dynamically scaled, the execution time of these operations is a relevant metric to analyze, as delays in such operations can negatively impact a VNF's execution. This way, the execution time of each VNF management operation and respective sub-operations that constitute them was evaluated, showing the cost of FENDE's VNF management calls. This metric is important to quantify the overhead of FENDE in terms of VNF management. These results are shown in Fig. 4.

The *Create* operation is composed of four sub-operations, with the *polling* sub-operation consuming around 90 percent of the total execution time. This occurs because, when sending a request to VM creation (sub-operation *vm_create*), the *polling* sub-operation should periodically check and wait for the infrastructure to finish the instantiation process. Then the function can be configured in the next sub-operation (*vnf_init*). The *Update Function* operation needs to upload a new network function (VNF software) and wait for the VNF to restart. The *Update* operation, after updating the VM description, waits for it to restart, while the *Delete* operation sends commands for the VM to be removed.

Research Challenges and Directions

In the context of NFV, network marketplaces must deal with specific challenges. While typical marketplaces (e.g., Google Play and Apple App Store) are concerned mostly with publishing and deployment, NFV marketplaces need to address aspects such as placement, auditing, and VNF life cycle management. Although FENDE is concerned to some extent with these aspects, it can benefit from improvements in several areas of computer science that can be applied to NFV marketplaces. Based on our experience in developing the FENDE ecosystem in a national backbone network, we identified fundamental challenges, which are detailed next.

Business Model

Business models are critical for the wide adoption of network marketplaces. Nowadays, two major methods are used for application acquisition: *fixed-price* and *pay-as-you-go*. In the former, customers must pay a predefined price for each VNF and can use them without restrictions (e.g., time or size). Google Play, JDN, and Cisco Market are examples of marketplaces that use this method of offering/acquisition. The latter considers different aspects to define the value to be paid (pre/post) to use the application. For example, a developer can offer a multicast-based application and charge the customers based on the number of concurrent flows. AWS and Microsoft Azure, for example, employ this method for specific cases.

In our view, future NVF marketplaces can support the previous cited business models and other innovative methods according to business requirements. We can cite two other interesting methods for network service acquisition: *auction-based* and *custom-built*. An auction-based method can benefit both third-party developers and customers because of its capacity to enable the competition to provide the best product regarding cost and performance. In the custom-built method, there is a negotiation between third-party developers and customers to develop a VNF for specific needs. This negotiation considers the final price, requirements, service level agreements (SLAs), deadlines, and desired features of the VNF to be developed.

These business models have advantages and disadvantages. Fixed-price is the simplest, but does not support any additional customization. Pay-as-you-go and auction-based models are able to adapt to customers' demands, but may be complex to deploy (e.g., need to define trustful monitoring for accounting and a reliable auction system). Finally, the custom-built method provides freedom for the customers to order customized VNFs, but implies challenges to guarantee that customers will describe requirements correctly.

Auditing

Network end users should be able to verify if the deployed VNFs are providing the advertised functionalities. Therefore, network marketplaces must apply auditing mechanisms to gather information about the execution of VNFs. For example, an end user may deploy a network service for distributed denial of service (DDoS) prevention. Upon request, the marketplace must provide reports to the end user showing that the contracted VNF is preventing DDoS attacks according to the previously defined SLAs.

Auditing reports must consider not only if a VNF meets the established SLA, but also how it affects the environment in which it runs. Thus, research efforts should be devoted to designing mechanisms that combine monitoring information (e.g., traffic pattern and resource usage) and diagnosis models (e.g., classification and machine learning approaches) to generate comprehensive reports. Although a comprehensive auditing approach is an open research challenge, there are current efforts that deal with specific parts of it. In the NFV context, Bless and Flittner [10] propose an auditing mechanism to allow customers to verify if the resources allocated by service providers are in accordance with the established SLAs.

VNF Recommendation

As the NFV market grows, the number of VNFs developed is also expected to increase proportionally. Reports indicate that by 2024 the NFV market will be valued at US$70 billion [11]. Although we cannot precisely estimate how many VNFs will be available, we can consider the number of middleboxes present in current network infrastructures as a baseline [12]. In such a direction, an open research challenge is to provide means to distinguish (or compose) the available VNFs to meet specific requirements. For example, security-related VNFs can offer security capabilities at distinct levels, such as inspection firewalls for L3 packets and intrusion prevention systems that detect malicious traffic patterns. The challenge is how to define which VNFs must be selected by network end users to meet their target requirements.

Clustering techniques can be applied to address the recommendation of applications and products. Similarly, VNFs could be grouped into clusters in a multi-dimensional plane considering distinct levels of, for example, security and performance requirements. This could help identify VNFs that provide a high level of security but a low level of performance; and VNFs that provide a low level of security but a high level of performance. However, a reliable recommendation mechanism for VNFs must address several challenges regarding: classification mechanism, number of VNFs in each cluster, order of VNFs through which the flows will pass, classification accuracy, and affinity and anti-affinity relations among VNFs.

In our ongoing research, we are currently addressing these aforementioned recommendation challenges. In a recent study [13], we proposed an intent refinement process that clusters VNFs according to user-defined contexts. In another study [14], we introduced a mechanism to compute the affinity score for each pair of VNFs in a service function chain. During the development of these works, we identified challenges to the recommendation of VNFs due to the limited knowledge about the behavior of VNFs and customers. Besides, the current solutions are not able to deal with all issues related to validation, verification, and performance analysis of the recommended VNFs. Thus, research efforts are still necessary to fully integrate these aspects into network marketplaces.

Placement

Finding the best placement of VNFs over the substrate infrastructure is difficult because each network end user may have different priorities and goals. Also, some VNFs may require a specific location for execution. For example, while firewalls are better placed in the network edge (i.e., close to the external link), an IP media transcoder should stay close to content servers. We observe that several efforts are focusing on optimizing the placement task in NFV-enabled networks. Placement mechanisms must take into account predefined criteria, and provide automated and manual mechanisms to define optimal locations for VNFs. The placement criteria can include minimal network delay, energy saving, deployment cost, and resource utilization.

The placement problem in software-based networks has been widely investigated for years. For instance, Moens and Turck [8] aimed to optimally place VNFs and network services according to established policies in the context of NFV. However, network marketplaces for NFV must also enable the easy and flexible placement of VNFs regarding both distinct technologies and concurrent or conflicting placement criteria. Further, marketplaces must be able to deal with custom infrastructures provided by end users.

Security

We expect that VNFs will be developed and published by distinct third-party developers and that different environments will deploy these VNFs. For these reasons, the marketplace must employ security mechanisms to prevent the environment from becoming a target of malicious attacks. For instance, if a network end user acquires a VNF for energy saving in his/her network, the marketplace needs to ensure the integrity of the VNF, and also provide a secure communication channel to deploy and send management commands to the VNF. This would prevent malicious users from interfering with the communication (e.g., man in the middle attacks to steal sensitive data) or sending commands to perform undesired actions (e.g., installing malicious software or stopping services).

Malicious users could also develop VNFs to be the source of attacks against third-party environments. In view of this, marketplaces should employ tools to guarantee the integrity of VNFs and SFCs [15]. Much can be learned from the two most successful mobile marketplaces: Google's Play Store and Apple's App Store. Apple developers need to go through a rigorous enrollment process and adhere to a stringent review process in order to publish their apps. Despite being less restrictive with submissions, Google imposes security mechanisms for submitted apps, automatically scanning them for potentially malicious code before acceptance. This way, strict contracts and autonomic VNF check mechanisms would play an important role to guarantee both marketplace and customer safety.

Conclusion

As NFV becomes more popular, a sharp increase in the number of VNFs available in the market is expected. As such, NFV marketplaces can provide the environment where VNF developers and customers can negotiate solutions. In this article we introduce FENDE, an NFV architecture for marketplace-based distribution and execution of VNFs. FENDE provides a VNF marketplace together with all life cycle management functionalities needed to instantiate and control VNFs' operation, as well as the composition of SFCs. In addition, FENDE provides the infrastructure for VNF and

We expect that VNFs will be developed and published by distinct third-party developers and that different environments will deploy these VNFs. For these reasons, the marketplace must employ security mechanisms to prevent the environment from becoming a target of malicious attacks.

We find that issues regarding auditing, recommendation, and placement still require considerable research efforts. Also, significant research efforts are needed to integrate the distinct technologies to provide flexible and useful NFV ecosystems.

SFC instantiation, placing itself as the first end-to-end NFV marketplace ecosystem.

Research challenges regarding the adoption of NFV marketplaces are also investigated. We find that issues regarding auditing, recommendation, and placement still require considerable research efforts. Also, significant research efforts are needed to integrate the distinct technologies to provide flexible and useful NFV ecosystems. As future research, security implications of the VNFs published in marketplaces must be investigated. For example, mechanisms to keep the integrity of NFV elements throughout their lifetimes are needed to avoid security breaches or service unavailability. Moreover, auditing mechanisms can help point out responsibilities when something goes wrong in any part of the system.

References

[1] G. Xilouris *et al.*, "T-NOVA: A Marketplace for Virtualized Network Functions," *Proc. Euro. Conf. Networks and Commun.*, 2014, pp. 1–5.

[2] OpenBaton; https://openbaton.github.io/; accessed 15 June, 2018.

[3] L. Bondan *et al.*, "FENDE: Marketplace and Federated Ecosystem for the Distribution and Execution of VNFs," *Proc. ACM SIGCOMM — Posters and Demos*, 2018, pp. 135–37.

[4] S. T. S. Portal, "App Stores — Statistics & Facts," 2017; https://www.statista.com/topics/1729/app-stores/, accessed 15 June, 2018.

[5] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," White Paper, NIST, 2011; https://csrc.nist.gov/publications/detail/sp/800-145/final, accessed 15 June, 2018.

[6] J. A. Wickboldt *et al.*, "Software-Defined Networking: Management Requirements and Challenges," *IEEE Commun. Mag.*, vol. 53, no. 1, Jan. 2015, pp. 278–85.

[7] M. Chiosi *et al.*, "Network Functions Virtualisation (NFV)," ETSI NFV ISG, White Paper 1, 2012; https://portal.etsi.org/NFV/NFV White Paper.pdf, accessed 15 June, 2018.

[8] H. Moens and F. De Turck, "VNF-P : A Model for Efficient Placement of Virtualized Network Functions," *Proc. Int'l. Conf. Network and Service Management*, Nov 2014, pp. 418–23.

[9] L. da Cruz Marcuzzo *et al.*, "Click-on-OSv: A Platform for Running Click-Based Middleboxes," *Proc. IFIP/IEEE Symp. Integrated Network and Service Management*, May 2017, pp. 885–86.

[10] R. Bless and M. Flittner, "Towards Corporate Confidentiality Preserving Auditing Mechanisms for Clouds," *Proc. IEEE Int'l. Conf. Cloud Networking*, Oct. 2014, pp. 381–87.

[11] "Network Function Virtualization (NFV) Market to See 42% Growth to 2024"; https://markets.businessinsider.com/news/stocks/networkfunction-virtualization-nfv-market-to-see-42-growth-to-2024- 1027473777, accessed 15 Nov. 2018.

[12] J. Sherry *et al.*, "Making Middleboxes Someone Else's Problem: Network Processing as a Cloud Service," *Proc. ACM SIGCOMM Conf. Applications, Technologies, Architectures, and Protocols for Computer Commun.*, 2012, pp. 13–24.

[13] E. J. Scheid *et al.*, "INSpIRE: Integrated NFV-Based Intent Refinement Environment," *Proc. IFIP/IEEE Symp. Integrated Network and Service Management*, May 2017, pp. 186–94.

[14] A. S. Jacobs *et al.*, "Affinity Measurement for NFV-Enabled Networks: A Criteria-Based Approach," *Proc. IFIP/IEEE Symp. Integrated Network and Service Management*, May 2017, pp. 125–33.

[15] L. Bondan *et al.*, "Anomaly Detection Framework for SFC Integrity in NFV Environments," *Proc. IEEE Conf. Network Softwarization*, July 2017, pp. 1–5.

Biographies

Lucas Bondan is a Ph.D. student at UFRGS, Brazil. His research interests include NFV, network management and orchestration, and SFC.

Muriel F. Franco is a Ph.D. student at UZH, Switzerland. His research interests include NFV, network management, information visualization, and blockchain.

Leonardo Marcuzzo is an M.Sc. student at UFSM, Brazil. His research interests include NFV and operating systems.

Giovanni Venancio is a Ph.D. student at UFPR, Brazil. His research interests include NFV and fault-tolerant distributed systems.

Ricardo L. dos Santos is a Ph.D. student at UFRGS, Brazil. His research interests include network programmability and network virtualization.

Ricardo J. Pfitscher is a Ph.D. student at UFRGS, Brazil. His research interests include network virtualization, VNF monitoring, and network management.

Eder J. Scheid is a Ph.D. student at UZH, Switzerland. His research interests include NFV, PBNM, and blockchain.

Burkhard Stiller is a full professor at UZH, Switzerland. His research interests include Internet services, decentralized systems, and network and service management.

Filip De Turck is a full professor at the University of Gent, Belgium. His research interests include network and service management, IoT, and multimedia delivery systems.

Elias P. Duarte, Jr. is a full professor at UFPR, Brazil. His research interests include network management, distributed systems, and algorithms.

Alberto E. Schaeffer-Filho is an associate professor at UFRGS, Brazil. His areas of expertise are network/service management, network resilience, and programmable networks.

Carlos R. P. dos Santos is an adjunct professor at UFSM, Brazil. His research interests include network virtualization, network programmability, and QoS management.

Lisandro Z. Granville is an associate professor at UFRGS, Brazil. His research interests include network management, Intent-based networking, and network programmability.

ANNEX B ACCEPTED PAPER – SIGCOMM 2018 - DEMO

This paper was accepted for presentation as a demonstration in the ACM SIGCOMM conference. The demonstration show the ecosystem developed in the FENDE working group of the Brazilian National Research Network (RNP), which provides a marketplace and lifecycle management ecosystem for the distribution and execution of VNFs and composition of Service Function Chains (SFCs).

- **Title:**
  FENDE: Marketplace and Federated Ecosystem for the Distribution and Execution of VNFs
- **Conference:**
  ACM SIGCOMM 2018 Conference Posters and Demos
- **URL:**
  <http://conferences.sigcomm.org/sigcomm/2018/>
- **Date:**
  20-25 August, 2018
- **Held at:**
  Budapest, Hungary
- **Digital Object Identifier (DOI):**
  <10.1145/3234200.3234235>

FENDE: Marketplace and Federated Ecosystem for the Distribution and Execution of VNFs

Lucas Bondan, Muriel F. Franco, Alberto E. Schaeffer-Filho, Lisandro Z. Granville
UFRGS - Brazil
{lbondan,mffranco,alberto, granville}@inf.ufrgs.br

Leonardo Marcuzzo, Cassiano A. D. S. Schneider, Carlos R. P. dos Santos
UFSM - Brazil
{lmarcuzzo,cschneider,csantos}@ inf.ufsm.br

Giovanni Venâncio, Elias P. Duarte Jr.
UFPR - Brazil
{gvsouza,elias}@inf.ufpr.br

ABSTRACT

In this demo, we present FENDE: Marketplace and Federated Ecosystem for the Distribution and Execution of Virtualized Network Functions (VNFs) and for the creation of Service Function Chains (SFCs). The FENDE ecosystem enables the distribution of both network functions and services in a manner that is akin to marketplaces found in mobile platforms (*e.g.,* Google Play and Apple Store), leveraging a federated infrastructure and testbed that spans three research institutions that are part of the Brazilian Research Backbone.

1 INTRODUCTION

Network Functions Virtualization (NFV) promotes the design, deployment, management, and integration of Virtualized Network Functions (VNF)s through an architectural framework proposed by the European Telecommunications Standards Institute (ETSI) [1]. Given its capability of VNF lifecycle management and integration, the NFV paradigm has started to be adopted by both industry and academia, becoming an enabler for flexible network service provisioning. In addition, NFV eases the provision of services through the composition of functions into Service Function Chains (SFC)s [5]. Given the indisputable advantages of NFV, solutions for different elements of its architecture have emerged, specially to design and deploy innovative VNFs [2, 3].

As the number of VNFs designed started to grow, solutions have been proposed to offer these functions to customers through NFV marketplaces [4, 7]. However, such solutions provide VNFs' source code for download but do not offer management tools or the NFV Infrastructure (NFVI) to execute VNFs. When VNF management tools are provided, providers tend to require users to pay for them or register their infrastructures to be able to use their VNFs. Shieldbox [6] provides a framework for middlebox instantiation focused in security deployment over untrusted commodity servers, using Docker Hub[1] as a catalog of code repositories to build images and stores manually pushed images for local deployment. Despite the VNF management tools provided by Shieldbox, it does not support the required mechanisms to enable offering and execution of VNFs in both public and private infrastructures. Moreover, there is no well-defined roles to define a marketplace, such as developers offering network functions and customers interested in acquiring such functions.

In this paper, FENDE is presented: a Marketplace and Federated Ecosystem for the Distribution and Execution of VNFs. The FENDE ecosystem allows the distribution of both network functions and services, also encompassing a platform for executing VNFs in federated infrastructures. FENDE provides a platform for developers, network operators, and network researchers to create, offer, and distribute VNFs by using an ecosystem similar to mobile distribution applications platforms. Moreover, FENDE allows not only the management of the virtualized environment (including VNF and SFC lifecycle management), but also assists the creation of network functions and the composition of services to supply specific demands.

2 SYSTEM DESIGN

The prototype was designed to reflect a real VNF marketplace scenario. FENDE architecture is based on basic elements of the NFV architectural framework defined by ETSI. The prototype's architecture is illustrated in Figure 1, divided

SIGCOMM Posters and Demos '18, August 20–25, 2018, Budapest, Hungary
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5915-3/18/08…$15.00
https://doi.org/10.1145/3234200.3234235

[1] https://hub.docker.com/

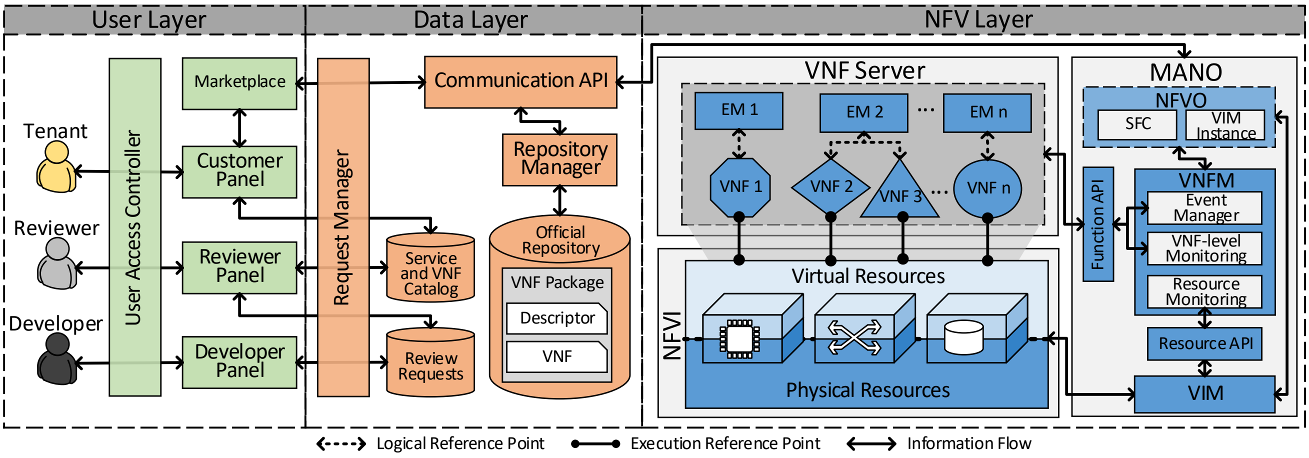


Figure 1: FENDE architecture

into three different layers, each layer with specific modules for different operational levels, as described in the following.

2.1 User Layer

Contains elements responsible for the interaction between different actors with the platform. Three different actors were considered: *(i)* Developers, who may request the insertion of their VNFs in the marketplace; *(ii)* Reviewers, responsible for accepting or rejecting developers’ requests; and *(iii)* Customers, who may acquire and deploy VNFs available in the marketplace. For management and interaction with different users, a Web interface was designed.

2.2 Data Layer

Once VNFs are reviewed, a series of events must occur in the platform so that other modules can use the information synchronously. To do so, three modules were designed in the Data layer to integrate the User layer with NFV layer:

- **Request Manager**: Controls the repository of submissions in the Review Requests database and performs the migration to the catalog when the repository is accepted. Developers must submit a Git repository with the VNF source code to be evaluated by reviewers;
- **Communication API**: Provides communication between User and NFV layers. Its main functions are: *(i)* to request the creation or update of VNFs’ repositories and *(ii)* to request VNFs’ descriptors for instantiation. All modules should forward requests to the Repository Manager through the Communication API;
- **Repository Manager**: Creates and manages VNFs’ descriptors available in the Catalog. For example, when a repository is accepted, the Repository Manager clones and maintains a local version of that Git repository.

2.3 NFV Layer

This layer brings together the main NFV elements proposed by ETSI, divided into three sublayers: *(i)* NFV Management and Orchestration (MANO), with components responsible for VNF and service management; *(ii)* VNF Instances, responsible for VNF execution; and *(iii)* NFVI, which provides the resources to execute VNFs. The modules developed for each sublayer are described below.

- **VNF Manager (VNFM)**: Performs VNF and SFC lifecycle management operations at two levels: at the hardware level, virtual machine characteristics are adjusted (*e.g.*, memory and CPU). At the software level, the function can be configured, initialized, updated, and terminated;
- **Virtualized Infrastructure Manager (VIM)**: Controls all resources available in the NFV infrastructure. FENDE currently uses OpenStack as VIM, due to its extensive documentation, performance, and especially its wide adoption by the community;
- **NFV Infrastructure (NFVI)**: Disposes physical and virtual resources available for VNF deployment, as well as virtualization and networking tools.

The FENDE front-end is accessed to perform operations of each different actor, such as developers requesting the insertion of their VNFs in the marketplace, the VNF review process, and all operations regarding VNF lifecycle management by customers, including interactive statistical information about VNFs operation. FENDE offers a marketplace for VNF distribution between developers and customers, together with the execution, management, and monitoring environment for VNFs and SFCs through an intuitive front-end. FENDE is available in the following link: https://gt-fende.inf.ufrgs.br/marketplace/.

REFERENCES

[1] Margaret Chiosi et al. 2012. *Network Functions Virtualisation (NFV).* White Paper 1. ETSI NFV ISG. 1–16 pages.

[2] Jinho Hwang, K.K. Ramakrishnan, and T. Wood. 2015. NetVM: High Performance and Flexible Networking Using Virtualization on Commodity Platforms. *IEEE Transactions on Network and Service Management* 12, 1 (March 2015), 34–47. https://doi.org/10.1109/TNSM.2015.2401568

[3] Joao Martins, Mohamed Ahmed, Costin Raiciu, Vladimir Olteanu, Michio Honda, Roberto Bifulco, and Felipe Huici. 2014. ClickOS and the Art of Network Function Virtualization. In *USENIX Symposium on Networked Systems Design and Implementation (NSDI).* USENIX Association, Seattle, WA, 459–473.

[4] OpenBaton. [n. d.]. https://openbaton.github.io/. ([n. d.]). Accessed: 18 May, 2018.

[5] Paul Quinn and Uri Elzur. 2016. *Network Service Header.* Internet-Draft draft-ietf-sfc-nsh-10. Internet Engineering Task Force. https://tools.ietf.org/html/draft-ietf-sfc-nsh-10 Work in Progress.

[6] Bohdan Trach, Alfred Krohmer, Franz Gregor, Sergei Arnautov, Pramod Bhatotia, and Christof Fetzer. 2018. ShieldBox: Secure Middleboxes Using Shielded Execution. In *Proceedings of the Symposium on SDN Research (SOSR ’18).* ACM, New York, NY, USA, Article 2, 14 pages. https://doi.org/10.1145/3185467.3185469

[7] Georgios Xilouris, E Trouva, F Lobillo, JM Soares, J Carapinha, Michael J McGrath, George Gardikis, P Paglierani, Evangelos Pallis, L Zuccaro, et al. 2014. T-NOVA: A marketplace for virtualized network functions. In *European Conference on Networks and Communications (EuCNC).* 1–5.

ANNEX C PUBLISHED PAPER – AIMS 2017 - PH.D. PROPOSAL

This paper was accepted for presentation in the Ph.D. Research track, where I presented the advancements of our work. Such advancements include a new entropy-based anomaly detection mechanism and the validation using realistic SFC data sets.

- **Title:**
  A Framework for SFC Integrity in NFV Environments
- **Conference:**
  International Conference on Autonomous Infrastructure Management and Security (AIMS)
- **URL:**
  <http://www.aims-conference.org/2017/>
- **Date:**
  10-14 July, 2017
- **Held at:**
  Zurich, Switzerland
- **Digital Object Identifier (DOI):**
  <10.1109/NETSOFT.2017.8004204>

A Framework for SFC Integrity in NFV Environments

Lucas Bondan[12], Tim Wauters[2], Bruno Volckaert[2], Filip De Turck[2], and Lisandro Zambenedetti Granville[1]

[1] Institute of Informatics (INF) – Federal University of Rio Grande do Sul – Brazil
[2] Department of Information Technology (INTEC) – Ghent University – Belgium
Email: {lbondan, granville}@inf.ufrgs.br, {tim.wauters, bruno.volckaert, filip.deturck}@intec.ugent.be

Abstract. Industry and academia have increased the deployment of Network Functions Virtualization (NFV) on their environments, either for reducing expenditures or taking advantage of NFV flexibility for service provisioning. In NFV, Service Function Chainings (SFC) composed of Virtualized Network Functions (VNF) are defined to deliver services to different customers. Despite the advancements in SFC composition for service provisioning, there is still a lack of proposals for ensuring the integrity of NFV service delivery, *i.e.,* detecting anomalies in SFC operation. Such anomalies could indicate a series of different threats, such as DDoS attacks, information leakage, and unauthorized access. In this PhD, we propose a framework composed of an SFC Integrity Module (SIM) for the standard NFV architecture, providing the integration of anomaly detection mechanisms to NFV orchestrators. We present recent results of this PhD regarding the implementation of an entropy-based anomaly detection mechanism using the SIM framework. The results presented in this paper are based on the execution of the proposed mechanism using a realistic SFC data set.

Keywords: Service Function Chaining, Network Functions Virtualization, Anomaly Detection

1 Introduction

Network Functions Virtualization (NFV) was proposed to deal with the virtualization of network functions usually performed by dedicated hardware devices (*e.g.,* firewalls, session border controllers, load balancers) [1]. In NFV, Virtual Network Functions (VNF) are connected to each other, composing Service Function Chainings (SFC) for service delivery. Any anomaly in SFC operation, such as missing elements, misconfiguration, and redirection, could lead to the interruption of the service delivery and, in some cases, could indicate attacks to the network. For this reason, in this PhD, we propose an additional SFC Integrity Module (SIM) to the NFV architecture [2]. SIM is a framework that allows the implementation of different anomaly detection mechanisms and the integration of such mechanisms into any NFV network under the control of NFV Orchestrators (NFVO). In this PhD, our focus resides in: (*i*) the applicability of existent and new anomaly detection mechanisms for SFC integrity in NFV environments, (*ii*) how to integrate such mechanisms to the NFV Management and Orchestration (MANO) architecture [3], and (*iii*) the evaluation of anomaly detection solutions in realistic NFV scenarios using the proposed SIM framework.

1.1 Motivation

In virtualized environments, vulnerabilities and exploits can lead to different SFC threats, since virtualization elements of NFV environments are susceptible to exploits. Examples of exploitable elements are container engines [4], hypervisors [5], and virtual machines [6]. Therefore, solutions have been proposed to detect anomalies in different NFV elements, such as VNFs [7], NFV services [8], and SLA violations [9]. However, there is still a lack of proposals dealing with security and integrity issues in the context of SFC [10]. In this PhD, we consider both the lack of solutions for SFC integrity and the potential vulnerabilities of NFV environments as research opportunities to be properly explored. To do so, we first investigated and proposed a framework that allows the implementation of anomaly detection techniques based on the NFV MANO information model.

2 SFC Integrity Framework

The NFV MANO architecture does not consider security-related tasks to protect functions and services. In this PhD research, we seek to guarantee the integrity of SFC operation for service delivery. Our proposal is designed to operate in NFV networks ruled by NFVOs according to the standard NFV MANO architecture.

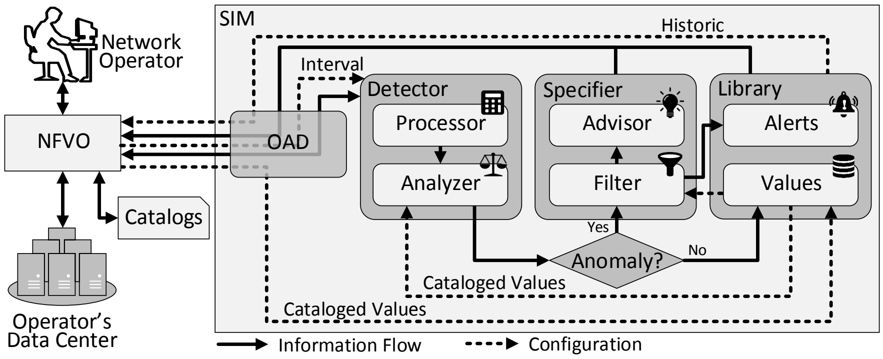


Fig. 1. Detailed SIM architecture [2] – The SIM communicates directly with NFVOs, using standard northbound APIs for requesting information regarding NFV elements operation and also to forward the results of the anomaly detection analysis.

2.1 Proposed Approach

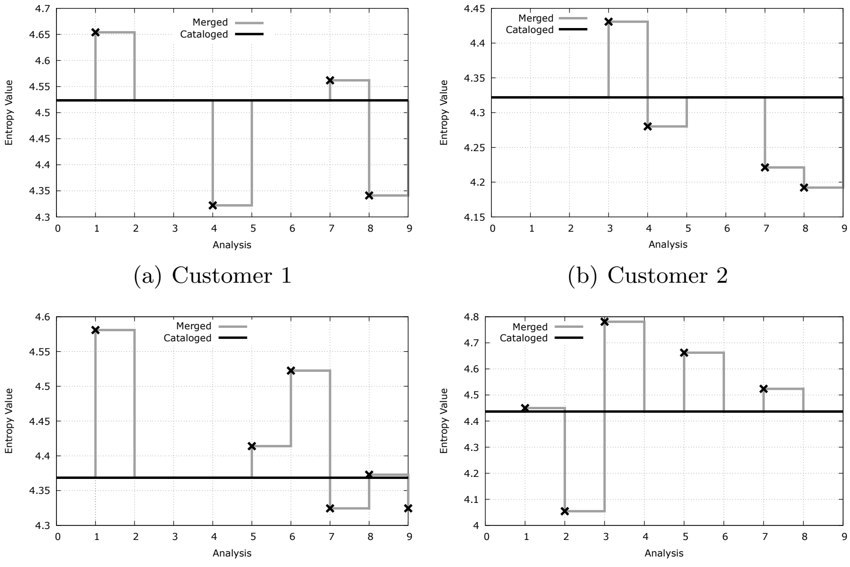
The NFVO sends cataloged and monitored information to an Orchestrator Abstraction Driver (OAD), depicted in Fig. 1 along with all SIM internal components. The information is then processed and analyzed according to the anomaly detection mechanisms implemented in the Detector component. If no anomalies are detected, the results are stored in the Library for further access. Otherwise, the results are filtered using the Filter module to specify the sources of such anomalies. Once identified, SIM stores it in the Library and forwards a report message to NFVO with the filtered results and suggestions from the Advisor module for overcoming such anomalies, *e.g.,* turn off unregistered VNFs.

2.2 Methodology

SIM was designed with specific elements for processing, analyzing, and filtering, enabling the design and implementation of different anomaly detection mechanisms. In this paper, we advance our first investigation using entropy-based anomaly detection [2] in two ways: (*i*) evaluating our solution using realistic NFV data sets [11] and (*ii*) improving the entropy-based anomaly detection mechanism to work with the current data set. These improvements enabled us to analyze each customer individually, increasing the accuracy of the anomaly detection mechanism. The data set was generated based on realistic information regarding the number of network functions composing SFCs on lager scale enterprise networks (with around 100 VNFs) [11]: 2 to 7 VNFs per SFC, mostly 2 to 5 [12]. So the number of VNFs for a given customer follows a truncated power-low distribution with exponent 2, minimum 2 and maximum 7. Following enterprise reports, anomalies were injected in the data set with a likelihood of 60% [13]. We considered three anomaly types: (*i*) unregistered SFCs, (*ii*) missing SFCs, and (*iii*) unauthorized changes in the SFC, such as additional or missing VNFs.

2.3 Results Obtained

Fig. 2 shows the entropy results of the anomaly detection mechanism considering 4 customers with different sets of SFCs. The detector creates a merged list with cataloged and monitored information. As the number of elements with low probability increases in the list, *i.e.,* highly uncertain elements, the merged entropy changes, indicating a disorder in the monitored elements. The merged entropy varies according to the number and type of anomalies detected (represented by markers). In our experiments, anomalies of type (*i*) and (*ii*) decreased the entropy value, since they involve adding or subtracting information, while anomalies of type (*iii*) (changes in existing values) increased the entropy value. It may lead to situations where anomalies of type (*i*) and (*ii*) cancel the entropy variations caused by anomalies of type (*iii*) and vice-versa. Despite rare to occur, this problem should be properly addressed to avoid false negatives. With the two-level approach of SIM (detection and filtering) it is possible to avoid false negatives with fine-grained filters comparing monitored and cataloged information. After each analysis the entropy values go back to normal (cataloged).



(a) Customer 1

(b) Customer 2

(c) Customer 3

(d) Customer 4

Fig. 2. Entropy results per customer. When anomalies occur (represented by markers), the entropy values varies, according to the amount of anomalies and their type.

3 Conclusions and Future Work

This PhD aims to propose efficient solutions for maintaining the integrity of service delivery in NFV environments. As first step, we proposed a SIM framework that allows the implementation of different anomaly detection mechanisms to analyze the network operation. The SIM modular architecture has the ability to operate with different NFVOs, requiring only to adapt one specific block. For future research, we foresee the following topics as good directions to follow.

Detection on different information levels. SIM was designed to operate at different levels of information. In this way, we foresee the possibility to analyze information regarding real-time resource consumption by virtual machines (*e.g.,* CPU, RAM, disk) and network information (*e.g.,* SFC traffic flows, bandwidth).

Evaluation of different detection mechanisms and network scenarios. Different anomaly detection mechanisms could be more suitable for a given network scenario, according to its characteristics. Analyzing the operation of different mechanisms in different environments will lead to important insights.

Deployment on production networks. Our results are based on realistic data sets generated according to real-world observations. However, production networks may present unpredicted behaviors, such as communication problems between NVFOs and other network elements. In this way, analyzing SIM operation in production networks is another important step of this PhD.

4 Acknowledgements

This research was performed partially within the FWO project “Service-oriented management of a virtualised future internet”.

References

1. Chiosi, M., et al.: Network Functions Virtualisation (NFV). White Paper 1, ETSI NFV ISG (2012) available at: https://portal.etsi.org/NFV/NFV_White_Paper.pdf.
2. Bondan, L., Wauters, T., Volckaert, B., Turck, F.D., Granville, L.Z.: Anomaly Detection Framework for SFC Integrity in NFV Environments. In: IEEE Conference on Network Softwarization (NetSoft). (jul 2017 (to appear))
3. Quittek, J., et al.: Network Functions Virtualisation (NFV) - Management and Orchestration. White paper, ETSI NFV ISG (2014)
4. Combe, T., Martin, A., Pietro, R.D.: To Docker or Not to Docker: A Security Perspective. IEEE Cloud Computing **3**(5) (Sept 2016) 54–62
5. Thongthua, A., Ngamsuriyaroj, S.: Assessment of hypervisor vulnerabilities. In: International Conference on Cloud Computing Research and Innovations (ICCCRI). (May 2016) 71–77
6. Wang, Z., Yang, R., Fu, X., Du, X., Luo, B.: A shared memory based cross-vm side channel attacks in iaas cloud. In: IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). (April 2016) 181–186
7. Giotis, K., Androulidakis, G., Maglaris, B.S.: A scalable anomaly detection and mitigation architecture for legacy networks via an openflow middlebox. Security and Communication Networks **9** (Oct 2015) 1958–1970
8. Xilouris, G.K., Kourtis, M.A., Gardikis, G., Koutras, I.: Statistical-based Anomaly Detection for NFV Services. In: IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN). (2016) (To appear)
9. Sauvanaud, C., Lazri, K., Kaâniche, M., Kanoun, K.: Anomaly Detection and Root Cause Localization in Virtual Network Functions. In: IEEE International Symposium on Software Reliability Engineering (ISSRE). (Oct 2016) 196–206
10. Briscoe, B., et al.: Network Functions Virtualisation (NFV) - NFV Security: Problem Statement. White paper, ETSI NFV ISG (2014)
11. Rankothge, W., Le, F., Russo, A., Lobo, J.: Data Modelling for the Evaluation of Virtualized Network Functions Resource Allocation Algorithms. Computing Research Repository (CoRR) **abs/1702.00369** (2017) Available at: http://arxiv.org/abs/1702.00369.
12. Sherry, J., Hasan, S., Scott, C., Krishnamurthy, A., Ratnasamy, S., Sekar, V.: Making Middleboxes Someone else’s Problem: Network Processing As a Cloud Service. In: ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication. (2012) 13–24
13. Anstee, D., Bowen, P., Chui, C., Sockrider, G.: Worldwide infrastructure security report. Technical report, Arbor Networks (2017) Available at: https://www.arbornetworks.com/insight-into-the-global-threat-landscape.

ANNEX D PUBLISHED PAPER – NETSOFT 2017 - SHORT PAPER

In this short paper, the first version of the anomaly detection framework was published, validating using simulated data through the implementation of an entropy-based anomaly detection mechanism.

- **Title:**
  Anomaly Detection Framework for SFC Integrity in NFV Environments
- **Conference:**
  IEEE Conference on Network Softwarization (NetSoft) - Short Paper
- **URL:**
  <http://sites.ieee.org/netsoft/>
- **Date:**
  3-7 July, 2017
- **Held at:**
  Bologna, Italy
- **Digital Object Identifier (DOI):**
  <10.1007/978-3-319-60774-0_18>

Anomaly Detection Framework for SFC Integrity in NFV Environments

Lucas Bondan*†, Tim Wauters†, Bruno Volckaert†, Filip De Turck†, Lisandro Zambenedetti Granville*
*Institute of Informatics – Federal University of Rio Grande do Sul – Brazil
†Department of Information Technology (INTEC) – Ghent University – Belgium
Email: {lbondan, granville}@inf.ufrgs.br, {tim.wauters, bruno.volckaert, filip.deturck}@intec.ugent.be

Abstract—**With the increasing deployments of Network Functions Virtualization (NFV) in both industry and academia, it becomes necessary to design mechanisms for keeping the integrity of Service Function Chains (SFC) responsible for NFV services delivering. Despite the advances in the development of management and orchestration for NFV, solutions to keep SFCs resilient to well-known and zero-day threats are still much needed. In this paper, we introduce a framework for deploying anomaly detection techniques for SFC in NFV environments. Our framework consists of a set of functional blocks with well-defined functions, composing an additional SFC Integrity Module (SIM) for the standard NFV architecture. The proposed SIM enables NFV orchestrators to analyze NFV elements and perform suggested actions with the goal of keeping service integrity in the network. The results obtained through the evaluation of a Proof-of-Concept implementation show that the proposed framework is able to properly detect different types of anomalies using entropy-based detection techniques.**

Index Terms—**network functions virtualization, service function chaining, services integrity, anomaly detection**

I. Introduction

Introduced by the European Telecommunications Standards Institute (ETSI), the concept of Network Functions Virtualization (NFV) is already a reality in computer networks [1]. NFV deals with the virtualization of network functions usually performed by dedicated hardware devices, such as load balancing, Deep Packet Inspection (DPI), and firewalling. In NFV, service provisioning is achieved by chaining Virtual Network Functions (VNF) to compose Service Function Chains (SFC). Both industry and academia are taking advantage of NFV and SFC for boosting innovation and providing flexibility in network service provisioning and management [2].

Many solutions for NFV Management and Orchestration (MANO) emerged recently, mainly focused on service lifecycle management. However, there still are many challenges in NFV MANO not properly addressed [3]. Among them, SFC integrity is extremely important for service delivery [4]. SFCs are vulnerable to many types of exploits, such as unauthorized reconfiguration of VNFs (for denial of service or unauthorized privilege for specific users), flow redirection, and duplication. Despite its importance, there currently is no focus on SFC integrity solutions for network safety and guaranteeing the proper operation of SFCs in NFV environments.

Taking into account the security of NFV deployments, ETSI created a working group focused on NFV security issues, evidencing the importance of protecting NFV environments. The literature in the area lists several vulnerabilities related to different virtualization approaches that can be exploited for malicious purposes [5] [6]. Moreover, undisclosed vulnerabilities (so-called zero days) are under constant investigation by security firms [7]. However, there is a lack of solutions for guaranteeing the integrity of SFC deployments against exploits of potential known and unknown vulnerabilities. Malicious users take advantage of network operators' assumption that following network security best practices will keep their environment protected against malicious behaviors.

In this paper, we propose an anomaly detection framework for SFC deployments in operators' data centers, using SFC models based on ETSI NFV MANO network service catalogs [8]. Our solution interacts with NFV Orchestrators (NFVO) to provide reactive resiliency executing operations like stopping anomalous VNFs, deploying new valid VNFs, stopping anomalous redirected traffic, and detecting re-chained SFCs. Moreover, the proposed solution is based on two views: (*i*) a general SFC view, which results from monitoring the entire SFC operation and the interactions among its elements (*e.g.,* connection points, member VNFs, virtual links); and (*ii*) a VNF view, which is computed from analyzing local information regarding VNFs operation (*e.g.,* connection points, dependencies, localization).

We consider an operator network scenario as presented by the ETSI NFV security group, where the same organization that operates the VNFs deploys and controls the resources. As main contributions of our work, we highlight: (*i*) an SFC anomaly detection solution, (*ii*) the proposal of an *SFC Integrity Module* (SIM) for the NFV MANO architecture, and (*iii*) an information model based on the NFV MANO network service register. Results obtained based on a Proof-of-Concept (PoC) implementation show that the proposed framework is able to properly detect different types of anomalies using entropy-based detection techniques.

The remainder of this paper is organized as follows. In Section II, we provide the background, related work, and motivation for this work. In Section III, we present the proposed SIM framework in detail. In Section IV, we detail the design of the SIM PoC developed to validate the proposed framework. The evaluation performed to validate the SIM framework is presented and discussed in Section V. Finally, our conclusions and perspectives for future work are discussed in Section VI.

II. Background and Related Work

The ETSI NFV Industry Specification Group was established aiming at a consensus for interoperability and management of Virtualized Network Functions (VNFs), creating

978-1-5090-6008-5/17/$31.00 © 2017 IEEE

the concept of NFV [1]. As the NFV concept evolved, it became an enabler for flexible service deployment, delivery, and management [2]. Essentially, NFV decouples network functions from dedicated hardware to run as software in commercial-of-the-shelf (COTS) servers as VNFs.

The NFV architecture presents a Management and Orchestration (MANO) plane designed to handle operations related to services and functions life-cycle management, a well as resource sharing [8]. The central element in NFV MANO is the NFVO, responsible for deploying and monitoring functions and services. In NFV, service delivery is provided by connecting VNFs through SFCs (or VNF Forwarding Graphs – VNFFG – according to ETSI's nomenclature), enabling automated provisioning of network services with different characteristics. Taking into account the importance of SFC for service delivery, the Internet Engineering Task Force (IETF) created a working group focused on defining an architecture for SFC operation [9].

Despite all its benefits, NFV has challenges to be overcome, from small NFV deployments [10] to performance issues [11], but especially MANO-related issues [3]. The CloudNFV project [12] provides an architecture for deploying and managing VNFs in a cloud environment using open standards. TeNOR [13] is an NFVO designed for supporting NFV as a Service (NFVaaS), focusing on automated deployment and configuration of services and resource sharing optimization for VNF hosting. In the same way, Maestro [14] is the first NFVO that considers the internal composition of VNFs for selecting their best deployment setup on wireless networks.

Despite NFV MANO solutions properly address the challenges they aim for, there are still a lack of proposals for dealing with security and integrity of NFV deployments, especially in the context of SFC [4]. Lee and Shen [15] proposed a path self-recovery scheme for SFCs, without taking into account anomalies on the SFC elements. Examples of exploitable elements are container engines [5], hypervisors [6], and Virtual Machines (VM) [16].

In NFV environments, vulnerabilities and exploits can lead to different types of malicious behavior for compromising the integrity of SFC operation. Examples of such malicious behavior are information redirection and duplication, Denial of Service (DoS), and unauthorized privileges for specific users. However, there is still a lack of proposals for guaranteeing SFC integrity in NFV scenarios. In this paper, we propose an anomaly detection framework, detailed in the next section. To the best of our knowledge, the SIM framework proposed in this paper is the first SFC integrity approach for NFV.

III. SFC Anomaly Detection Framework

The framework proposed in this paper is based on the addition of a new module called SIM in the NFV MANO architecture, as depicted in Figure 1. The SIM communicates directly with the NFVO, using standard northbound APIs of the NFVO to request information regarding NFV elements operation and to forward the results of the anomaly detection.

Operations and Business Support Systems (OSS/BSS) are responsible for enforcing access control rules in data centers shared with different network operators. NFVO is the NFV element responsible for bringing intelligence to service provisioning and composition processes, directly interacting with VNF Managers (VNFM) for managing the VNF operation life-cycle. In the same way, the resource sharing orchestration among different virtualized elements is performed by the NFVO through the Virtual Infrastructure Managers (VIM).

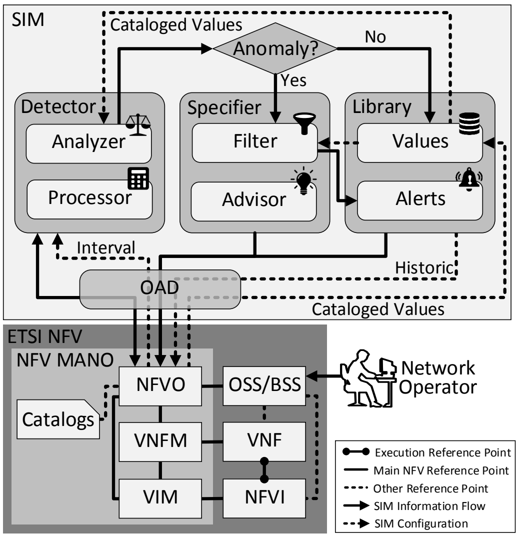


Figure 1. SIM architecture and its internal components

We designed SIM detached from NFVO to make it independent of the NFVO implementation. Therefore, any NFVO should be able to communicate and operate with SIM using standard northbound APIs. Moreover, SIM is a modular framework, providing flexibility for implementing different anomaly detection techniques. SIM can be directly configured by network operators. However, the most suitable approach is controlling and configuring SIM through NFVO, taking advantage of management interfaces already provided by NFVO.

As depicted in Figure 1, network operators configure services through OSS/BSS and NFVO, as well as SFCs and VNFs responsible for delivering network operators services. According to the NFV architecture, the NFVO must deal with all responsibilities regarding services' life-cycle management [8]. To do so, NFVO manages services, SFCs, and VNFs available through a catalog with information regarding their operation. For deploying a new VNF, the network operator should first catalog it. Once deployed, VNFs operation should be monitored and registered by NFVO. SIM is composed of four functional components, described in detail as follows.

Orchestrator Abstraction Driver (OAD): Responsible for handling all communication between NFVO and SIM. Since SIM was designed to operate with any NFVO, SIM should be able to adapt its communication to fit their northbound APIs. OAD hosts the communication functions of the NFVO being used. To change the NFVO or communicate with multiple NFVOs, only the OAD component needs to be modified, avoiding changes and bringing flexibility to SIM operation.

Detector: Requests and receives information regarding SFCs and VNFs operation to/from NFVO, as well as performs the anomaly detection technique implemented. This component can be configured in two different ways: oriented (*i*) by *events*, where SIM requests and analyzes SFCs and VNFs information only when a new event related to these elements is signalized by NFVO; and (*ii*) by *polling*, in which SIM periodically looks for anomalies based on a predefined time interval. The Detector component is composed of two modules. The first one is the *Processor* module, responsible for processing the information acquired from NFVO and formatting it for the anomaly detection techniques. The second module is called *Analyzer* and it uses the processed information to identify potential anomalies based on cataloged values. If an anomaly is detected, results are forwarded to the Specifier component. Otherwise, the Library component stores the results and the NFVO is notified about the absence of anomalies.

Specifier: Identifies the anomalous elements and selects the most appropriate action to be taken. The main reason for separating the anomaly detection from its specification is to save time and computational resources. To do so, a *Filter* module is defined for filtering the anomalies from the list of monitored elements. After identifying the anomalous elements, the *Advisor* module evaluates which is the most appropriated action to be taken to overcome the anomalies and sends an alert message to the NFVO. The suggestion can be based on both predefined sets of actions and learning mechanisms, depending on the implementation of the Advisor module. After selecting the action, the Specifier component sends a notification containing the information regarding the anomalous elements to the Library. The final choice of whether to apply or not the suggested actions and possible impacts of such actions lies with the NFVO.

Library: Stores the anomaly detection technique results and forwards them to the NFVO when queried. The *Alerts* module handles information regarding alerts generated by the Specifier, which can also be used by the network operator to generate reports regarding the historical occurrence of anomalies in the data center. In the same way, the *Values* module handles the results of analyses that did not detect any anomaly. These values can be used as the baseline for further analyses depending on the anomaly detection technique implemented in the Detector component, or they can be re-evaluated when new anomaly detection techniques are implemented, enabling the detection of previously undetected anomalies [17].

IV. Anomaly Detection Mechanism

In this section, we provide the details related to the design of a SIM Proof-of-Concept (PoC), developed to evaluate the proposed framework. In Subsection IV-A, we present the anomaly detection technique implemented to validate SIM operation. Then, in Subsection IV-B, we present and discuss the information regarding NFV elements under analysis.

A. Anomaly Detection Technique

The choice for a specific anomaly detection technique depends on the network scenarios and monitored information. [17]. Techniques that require supervised training or statistical modeling regarding network operation may not be suitable for NFV scenarios due to their dynamic behavior. However, information theory-based techniques do not require training data sets or statistical models to operate, as required by classification and statistic-based techniques. Moreover, information theory-based techniques are less complex than spectral theory-based techniques, which usually demand high processing capabilities to run in acceptable time.

Based on information theory techniques, we implemented a Shannon's Information Entropy anomaly detection technique into the Detector component. The decision for using Shannon's entropy is based on the type of information monitored and the proven effectiveness of using entropy for detecting anomalies on network environments [18]. Disorders in the data set of monitored elements handled by NFVO indicate anomalies in the operation of NFV elements. The computational cost of calculating the entropy is smaller than comparing element by element (diff). Only if the entropy changes, the filtering process will be started to identify where the anomaly occurs. The two-level approach of SIM (detection and filtering) also improves the accuracy and avoids false negatives alarms, two known issues of entropy-based detection mechanism [19].

B. Monitored Information

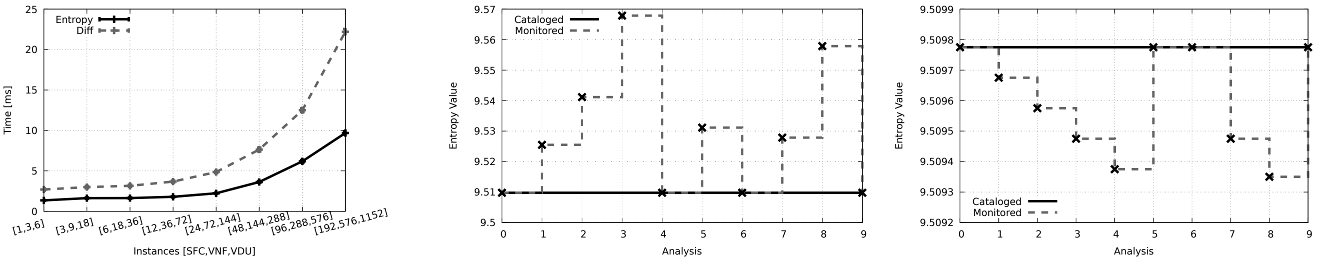
After defining the types of anomalies to be detected and the anomaly detection technique, the final step is selecting which information will be monitored and analyzed. The information monitored determines which types of anomalies and possible threats the system will be able to detect. We based the PoC design on the information model proposed by ETSI NFV MANO [8]. This information model provides a hierarchical structure for NFV elements, composing a tree for cataloging information regarding the operation of SFCs, VNFs, Virtual Deployment Units (VDU – VNFs' execution elements, like virtual machines or containers), among others. For the PoC implementation, we selected the information regarding (*i*) SFC operation (identifier, connection points, virtual links, and member VNFs); and (*ii*) VNF operation (identifier, connection points, virtual links, member VDUs, localization, and VDU dependencies). Although VDU-specific information, such as resource consumption, is not handled in this paper, it can be easily achieved using SIM framework by adding a new third view in the SFC operation to handle VDU information.

V. SIM Framework Validation

In this section, we present the evaluation of SIM based on the definitions presented in the previous section. First, we present and discuss the results obtained through an experimental evaluation of the PoC in Subsection V-A. Then, in Subsection V-B, we analyze the trace of anomalies detected and the actions suggested by SIM PoC.

A. Entropy Result Analysis

Our first evaluation is regarding the time spent by the anomaly detection technique in analyzing the data set. We compared the time expended for calculating the entropy with the time needed for directly comparing cataloged information stored in NFVO catalogs with monitored information obtained from the running NFV elements by NFVO (Diff). We varied the number of instantiated elements, from 1 SFC to 384, each one composed of 3 VNFs and up to 3 VDUs. The results are depicted in Figure 2(a). All experiments have been repeated until we achieved a confidence level of 99%.



(a) Execution time comparison

(b) Anomalous elements detected

(c) Missing elements detected

Figure 2. Evaluation results: Execution time comparison of entropy-based anomaly detection vs extracting differences (Diff) (a); and comparison of entropy value changes when detecting anomalous (b) and missing (c) elements

The entropy-based technique was faster than extracting the difference between monitored and cataloged information for all number of instances (deployed elements) evaluated. As we can observe in Figure 2(a), although both execution times are close to each other for small deployments (*e.g.,* above 6 SFCs, 18 VNFs, 36 VDUS), as the number of instances increases, the entropy presents a less accentuated growth. For this reason, entropy is considered a light-weight anomaly detection technique, suitable for performing the first evaluation of the monitored information for detecting anomalies. When the entropy indicates an anomaly, more complex mechanisms for filtering such anomalies can be applied. This configuration fits in the SIM framework, where the entropy-based technique was implemented in the Detector component, allowing us to implement more sophisticated filters in the Filter module. The advantage of this approach is executing sophisticated heavy-weight algorithms (*e.g.,* Diff) only when an anomaly is detected by the entropy analysis, saving time and computational resources when no anomalies are detected. The drawback, however, is that when anomalies are detected by the entropy analysis, the filtering process should be performed after the entropy calculation, increasing the total execution time.

The second experiment is related to the effectiveness of using entropy for detecting anomalies. For this experiment, we randomized the occurrence of anomalous events in the data set to analyze the changes in the entropy value. The data set is composed of information regarding 150 SFCs, each one composed of 3 VNFs, totaling 450 VNFs. We defined a probability of 60% of occurrence for each anomaly, each one being able to occur up to 5 times in each analysis. Considering that every disorder in the monitored information characterizes an anomaly, the anomalies detected represent 100% of the anomalies inserted in the data set. We measured the overall entropy of the monitored information with randomized anomalies and compared with the unchanged cataloged entropy value. In Figure 2(b), we show the changes in the entropy value when anomalous elements are detected in the data set, while Figure 2(c) shows the entropy when some information of the monitored elements is missing.

As can be observed in Figure 2, the entropy value changes significantly more when anomalous elements are present (Figure 2(b)) than when cataloged elements are missing on the monitored data set (Figure 2(c)). It highlights the difference in the magnitude order of the changes in the entropy value when anomalous and missing elements are detected in the data set and is directly related to the amount of information cataloged in the NFVO. This behavior is especially interesting for our scenario for two main reasons. First, it is impossible for anomalous and missing elements to *cancel* or hide each other in the final entropy value. Second, it is usually worse when an intruder element is found in the network than when there is a missing one. An intruder element (non-cataloged) may indicate a higher threat, such as information leakage, while a missing one usually indicates a DoS. By analyzing these differences, more sophisticated actions could be suggested to NFVO when more dangerous behavior is detected.

B. Anomaly Detection Analysis

For this evaluation, we recorded the anomalies and the actions suggested by SIM to NFVO during the anomaly detection analysis. The recorded values of the tracking can be observed in Tables I and II for uncataloged and missing elements detection, respectively. Tracking these data sets allow us to verify what anomalies were detected, as well as the actions suggested by the SIM PoC Advisor module to NFVO.

When no anomalies occur, SIM sends a standard report to NFVO without any suggested action, as occurs in analyses $0, 4, 6,$ and 9 of Table I, and on $0, 5, 6,$ and 9 of Table II. With regard to the anomalies summarized in Table I, when an anomalous VNF is detected (analyses $1, 2, 3, 5,$ and 8), SIM sends the results of the anomaly detection and suggests NFVO to shutdown the anomalous VNFs. An uncataloged VNF in the middle of an SFC may indicate several threats, such as DoS attacks, flow duplication for obtaining private information, and unauthorized access to services. In the same way, uncataloged virtual links (analyses 5 and 7) and connection points (analyses 7 and 8) may also indicate flow duplication and unauthorized access, as well as unauthorized privilege for users accessing services means of by a side connection.

In the case where missing VNF are detected (all anomalies presented in Table II), the immediate standard action is to re-instantiate the missing VNF to avoid interruption in delivering the services composed of the missing VNF. For missing connection points (analyses $2, 3, 4, 7,$ and 8) and missing virtual links (analysis 4), the suggested action is restarting the connections lost and re-chaining the virtual link, respectively. The most common threat characterized by missing elements is a DoS attack, where one or more SFC elements are turned off or reconfigured for overthrowing service delivery.

Table I
ANOMALOUS ELEMENTS TRACE AND SUGGESTED ACTIONS

Analysis	Anomalies	Possible Threat	Suggested Action
0, 4, 6, 9	None	–	None
1, 2, 3	Uncataloged VNFs (1, 2, 4)	DoS, flow duplication, unauthorized access	Shutdown VNFs
5	Uncataloged virtual link (1) and VNF (1)	DoS, flow duplication, unauthorized access, unauthorized privilege	Remove & trace virtual link, shutdown VNF
7	Uncataloged virtual link (1) and connection point (1)	Flow duplication, unauthorized access, unauthorized privilege	Remove & trace virtual link, remove connection point
8	Uncataloged connection point (1) and VNF (2)	DoS, flow duplication, unauthorized access, unauthorized privilege	Remove connection point, shutdown VNF

Table II
MISSING ELEMENTS TRACE AND SUGGESTED ACTIONS

Analysis	Anomalies	Possible Threat	Suggested Action
0, 5, 6, 9	None	–	None
1	Missing VNF (1)	DoS	Re-instantiate VNF
2, 3, 7, 8	Missing VNFs (1, 2, 2, 2) and connection point (1, 1, 1, 2)	DoS	Re-instantiate VNFs, restart connections
4	Missing VNF (2), connection point (1), and virtual link (1)	DoS	Re-instantiate VNFs, restart connections, re-chain

The Advisor module implemented was configured with a predefined set of standard suggested actions, according to the type of anomaly detected. However, smarter mechanisms may be implemented in the Advisor for suggesting more precise actions to NFVO. For example, machine learning techniques can be implemented to learn over time what is the best action to be executed based on the history of anomalies detected. Complementarily, fine-grained conclusions can be obtained, such as uncovering the origins of a DoS attack by analyzing the missing elements and the last access to these elements.

VI. Final Remarks and Future Work

In this paper, we presented the SIM framework capable of monitoring and maintaining SFC integrity in NFV environments. SIM was designed to be easily adaptable to operate with different NFVOs, and its modular architecture enables the implementation of different anomaly detection and filtering techniques. The choice of such techniques should fulfill network operators' needs for their NFV environment and the information available in such an environment. We implemented an entropy-based anomaly detection technique based on Shannon's entropy to validate SIM operation. The results obtained confirm the entropy-based technique as a suitable solution for detecting anomalies using some elements of the information model proposed by the ETSI NFV MANO group. As future work, we consider extending the SIM views to include VDU information, such as resource consumption. In addition, more advanced anomaly detection techniques can be implemented and evaluated, as well as different filtering mechanisms, according to the network scenario and threats to be detected.

VII. Acknowledgements

This research was performed partially within the FWO project "Service-oriented management of a virtualised future internet".

References

[1] M. Chiosi *et al.*, "Network Functions Virtualisation (NFV)," ETSI NFV ISG, White Paper 1, 2012, available at: https://portal.etsi.org/NFV/NFV_White_Paper.pdf.

[2] 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," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 236–262, Firstquarter 2016.

[3] R. Mijumbi, J. Serrat, J. l. Gorricho, S. Latre, M. Charalambides, and D. Lopez, "Management and orchestration challenges in network functions virtualization," *IEEE Communications Magazine*, vol. 54, no. 1, pp. 98–105, January 2016.

[4] B. Briscoe *et al.*, "Network Functions Virtualisation (NFV) - NFV Security: Problem Statement," ETSI NFV ISG, White Paper, 2014.

[5] T. Combe, A. Martin, and R. D. Pietro, "To Docker or Not to Docker: A Security Perspective," *IEEE Cloud Computing*, vol. 3, no. 5, pp. 54–62, Sept 2016.

[6] A. Thongthua and S. Ngamsuriyaroj, "Assessment of hypervisor vulnerabilities," in *International Conference on Cloud Computing Research and Innovations (ICCCRI)*, May 2016, pp. 71–77.

[7] "Zero-Day Danger: A Survey of Zero-Day Attacks and What They Say About the Traditional Security Model," FireEye, White Paper, 2015.

[8] J. Quittek *et al.*, "Network Functions Virtualisation (NFV) - Management and Orchestration," ETSI NFV ISG, White Paper, 2014.

[9] P. Quinn and U. Elzur, "Network Service Header," Internet Engineering Task Force, Internet-Draft draft-ietf-sfc-nsh-10, Sep. 2016, work in Progress. [Online]. Available: https://tools.ietf.org/html/draft-ietf-sfc-nsh-10

[10] L. Bondan, C. R. P. d. Santos, and L. Z. Granville, "Management requirements for ClickOS-based Network Function Virtualization," in *International Workshop on Management of SDN and NFV Systems (ManSDN/NFV) collocated with the International Conference on Network and Service Management (CNSM)*, Nov 2014, pp. 447–450.

[11] L. Bondan, C. R. P. dos Santos, and L. Z. Granville, "Comparing Virtualization Solutions for NFV Deployment: A Network Management Perspective," in *IEEE Symposium on Computers and Communication (ISCC)*, June 2016, pp. 669–674.

[12] J. Soares, M. Dias, J. Carapinha, B. Parreira, and S. Sargento, "Cloud4NFV: A platform for Virtual Network Functions," in *IEEE International Conference on Cloud Networking (CloudNet)*, Oct 2014, pp. 288–293.

[13] J. F. Riera, J. Batallé, J. Bonnet, M. Días, M. McGrath, G. Petralia, F. Liberati, A. Giuseppi, A. Pietrabissa, A. Ceselli, A. Petrini, M. Trubian, P. Papadimitrou, D. Dietrich, A. Ramos, J. Melián, G. Xilouris, A. Kourtis, T. Kourtis, and E. K. Markakis, "TeNOR: Steps towards an orchestration platform for multi-PoP NFV deployment," in *IEEE NetSoft Conference and Workshops*, June 2016, pp. 243–250.

[14] A. G. Dalla-Costa, L. Bondan, J. A. Wickboldt, C. B. Both, and L. Z. Granville, "Maestro: An NFV Orchestrator for Wireless Environments Aware of VNF Internal Compositions," in *IEEE International Conference on Advanced Information Networking and Applications (AINA)*, Tamkang University, Taipei, Taiwan, Mar. 2017 (to appear).

[15] S. I. Lee and M. K. Shin, "A self-recovery scheme for service function chaining," in *International Conference on Information and Communication Technology Convergence (ICTC)*, Oct 2015, pp. 108–112.

[16] Z. Wang, R. Yang, X. Fu, X. Du, and B. Luo, "A shared memory based cross-vm side channel attacks in iaas cloud," in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, April 2016, pp. 181–186.

[17] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly Detection: A Survey," *ACM Computing Surveys*, vol. 41, no. 3, pp. 15:1–15:58, Jul. 2009.

[18] A. S. da Silva, J. A. Wickboldt, L. Z. Granville, and A. Schaeffer-Filho, "ATLANTIC: A framework for anomaly traffic detection, classification, and mitigation in SDN," in *IEEE/IFIP Network Operations and Management Symposium (NOMS)*, April 2016, pp. 27–35.

[19] P. Berezinski, B. Jasiul, and M. Szpyrka, "An Entropy-Based Network Anomaly Detection Method," *Entropy*, vol. 17, no. 4, pp. 2367–2408, 2015.

ANNEX E PUBLISHED PAPER – ISCC 2016

In this paper, an investigation regarding the effectiveness of emerging visualization solutions was presented, comparing them based on the management aspects of NFV.

- **Title:**
  Comparing Virtualization Solutions for NFV Deployment: a Network Management Perspective
- **Conference:**
  IEEE Symposium on Computers and Communications (ISCC)
- **URL:**
  <http://iscc2016.unime.it/>
- **Date:**
  27-30 June, 2016
- **Held at:**
  Messina, Italy
- **Digital Object Identifier (DOI):**
  <10.1109/ISCC.2016.7543814>

Comparing Virtualization Solutions for NFV Deployment: a Network Management Perspective

Lucas Bondan*, Carlos Raniery Paula dos Santos†, Lisandro Zambenedetti Granville*
*Institute of Informatics – Federal University of Rio Grande do Sul
†Department of Applied Computing – Federal University of Santa Maria
Email: {lbondan, granville}@inf.ufrgs.br, csantos@inf.ufsm.br

Abstract—**Network Functions Virtualization (NFV) is a paradigm designed to promote service agility and able to quickly generate revenue, thus encouraging competition among companies in the computer network industry. Besides the advocated benefits of NFV, management requirements should be properly taken into account. The choice of a particular NFV-based technology must consider its management requirements. However, there is still no evaluation of virtualization solutions providing an in-depth analysis from the management point-of-view. This paper presents a performance analysis of three prominent virtualization solutions: ClickOS, CoreOS, and OS^v. Our results place ClickOS and CoreOS as the best solutions regarding boot time, response time, and memory consumption. Moreover, based on the results obtained for each performance metric, we provide a broad discussion about the effectiveness of each virtualization solution in fulfilling qualitative management requirements.**

Keywords*—*NFV, network management, virtualization solutions, performance analysis, management requirements

I. Introduction

Network Functions Virtualization (NFV) [1] is a networking paradigm where network functions (*e.g.,* firewalls, load balancers, NATs), often requiring dedicated devices, are deployed on virtual servers running on commodity hardware. Introduced by the European Telecommunications Standards Institute (ETSI), NFV complements the more established paradigm of Software-Defined Networking (SDN) [2]. While SDN focuses in decoupling the network control plane, NFV is concerned with moving network functions from dedicated hardware appliances into software running on standard commercial off-the-shelf (COTS) servers [3]. Different than SDN, however, NFV is much less mature, and more concrete developments are just emerging.

As in any novel networking technology, the proper management of its functional aspects is fundamental for its adoption. Unfortunately, the network management discipline is not in the focus of current NFV efforts. We argue, however, that the identification of concrete management functions cannot be neglected. As such, we identified a set of NFV management requirements from the perspective of the network operator [4]. By listing the difficulties faced using a virtualization solution for an NFV deployment, we evidence the importance of the virtualization solution choice for the network management.

Once identified the management requirements, network operators must choose the most appropriate technology to fulfill these requirements. Recently, some efforts from industry and academia led to the development of virtualization solutions aimed to run Virtualized Network Functions (VNFs) [5]–[10]. Although some of these solutions are not directly designed for NFV, they present essential virtualization properties, which turns them promising NFV enablers. The right choice of the virtualization solution is crucial for the network operator, since it has direct implications on the network performance and management support. Besides those efforts and to the best of our knowledge, no investigations were conducted to show how effective virtualization solutions are with regard to management requirements. We argue that the analysis of different virtualization solutions is fundamental to help network operators interested in adopting NFV on their environments.

In this paper, we analyze three different virtualization solutions: ClickOS [5], CoreOS [6], and OS^v [7], which are open source solutions frequently considered NFV enablers. To perform the comparison, we selected the NFV management requirements most appropriate to be analyzed using quantitative metrics, *i.e.,* virtual machines (VMs)/containers instantiation, VNF deployment, VM/container & VNF monitoring, and physical/virtual network functions coexistence. An experimental network setup was deployed using each virtualization solution, supporting their evaluation based on selected performance metrics. The main contributions of this paper are (*i*) a performance evaluation of different virtualization solutions and (*ii*) an in-depth discussion on the effectiveness of each solution regarding the management requirements.

The remaining of this paper is organized as follows. In Section II, we present a background and related work on NFV and emerging solutions. The methodology and the experimental scenario applied to evaluate the performance metrics are presented in Section III. In Section IV, we show the results obtained in the experimental evaluation. In Section V, the results achieved are discussed, relating the performance metrics to the management requirements. Finally, we present the conclusions and perspectives of future work in Section VI.

II. Background and Related Work

NFV is a new networking paradigm where functions (*e.g.,* firewalls, DNS, IDS), traditionally performed by dedicated physical devices, are virtualized and deployed on commodity hardware. Initially, NFV not only enables to reduce both capital and operational expenditures (CAPEX and OPEX) by virtualizing network functions (NFs), but it is also about business and service agility, and the ability to quickly generate revenue, thus encouraging competition among companies in the computer network industry. Moreover, in academia NFV represents a way to develop innovative solutions, by simplifying the design and deployment of network functions [11].

To promote NFV adoption, ETSI released a series of documents detailing NFV concepts. We highlight the NFV Management and Orchestration (MANO) document, which deals with these aspects in the context of NFV [12]. ETSI MANO aims to propose an architecture for NFV management and orchestration, defining some reference points and an information model to manage important data regarding operational NFV elements (*e.g.,* virtual links, VNFs, VMs). However, the reference points defined in ETSI MANO are too vague to be implemented in practice, and the information model seems like a set of abstract requirements for building a model, insufficient for the management of an NFV deployment in practice.

Similarly to ETSI, the Internet Research Task Force (IRTF) is also concerned with promoting NFV adoption. IRTF hosts the NFV Research Group (NFVRG) focused on issues related to NFV environments. NFVRG already produced a set of Internet-Drafts dealing with policy-based management, service verification, resource management, among others research topics[1]. Despite the efforts from both ETSI and IRTF, we identified a lack of practical analysis over virtualization solutions from the management point-of-view. This analysis can certainly help to refine the reference points (*i.e.,* interfaces and functional blocks) and data model proposed in ETSI MANO, highlighting practical necessities of virtualization solutions.

In a previous work, we took a first step in identifying key management requirements of NFV, by deploying a network setup request using a virtualization solution [4]. Now, we advance the research on management requirements, evaluating different virtualization solutions from the management point-of-view. NFV solutions are emerging, as the case ClickOS, CoreOS, OS^v, NetVM (or openNetVM), CirrOS, Alpine Linux, among others [5]–[10]. In this paper, we concentrate our evaluation in three solutions: ClickOS, CoreOS and OS^v.

Our choice was based on three main aspects. The first one is code availability, since that all solutions chosen are open-source and available for download with no cost. Next, these solutions still are under development, so developers are constantly improving and/or fixing issues to guarantee stability for their solutions. Finally, all selected solutions are in accordance with the NFV Virtualization Requirements document published by ETSI [13], which presents low-level requirements for NFV adoption. In our investigations, ClickOS, CoreOS, and OS^v differ from the others by presenting all these aspects together, *i.e.,* in the same solution. We present each one of the selected virtualization solutions in details as follows.

A. ClickOS

ClickOS is a Xen-based software platform optimized for fast network packet processing and designed to support typical network requirements such as high throughput, low latency, and isolation [5]. ClickOS consists of the Click Modular Router running on top of a minimalist Linux [14]. Using ClickOS, network functions are provided by Click libraries, which allows implementation of complex network functions processing configurations by using simple, well-known processing elements. In addition, ClickOS VMs present small sizes and memory (with basic Click libraries, ClickOS VMs are 6 MB in size).

B. CoreOS

CoreOS allows the easy deployment of a wide range of isolated functions using Linux containers (LXC) virtualization [6]. LXC provides similar benefits as complete virtual machines (or full virtualization) but focused on functions instead of entire virtualized hosts. Container-based virtualization allows code to run in isolation from others but safely share the machine resources. Moreover, it is not necessary a dedicated Linux kernel or hypervisor for managing containers, thus presenting almost no performance overhead. In container-based virtualization, the main element is the container engine responsible for containers lifecycle management. In CoreOS, VNFs can be represented by Linux functions (*e.g., iptables* for proxy, firewall, and NAT; *Snort* for IDS Sensor) running on the same server but with isolated memory spaces.

C. OS^v

OS^v is an open source operating system based on the library OS design [7]. Although its standard version is distributed based on the QEMU Kernel Virtual Machine (KVM), OS^v can be managed using other hypervisors (*e.g.,* Xen, VirtualBox) with a minimal amount of architecture-specific code. Moreover, OS^v is flexible, able to run functions designed in different languages, such as C/C++ and Java. In OS^v, each VM runs a single function with its dedicated copy of the library OS. Library OS attempts to address performance and functionality limitations in functions that are caused by traditional operating systems abstractions.

Different works can be found in the literature performing comparisons among virtualization solutions. Estrada *et al.* [15] compared the KVM hypervisor, the Xen para-virtualised hypervisor, and LXC, also performing changes over these solutions in order to improve the runtime to solve sequence alignment problem in bioinformatics. In another work, Felter *et al.* [16] conducted an analysis of the two most common types of virtualization available: VMs and containers based. In their analysis, KVM hypervisor and LXC were compared regarding input/output (IO) operations. Finally, the work of Reddy and Rajamani [17] presents a comparison of different operating systems over the same hypervisor (KVM) in a cloud environment, using performance metrics like CPU usage, memory management, and network communication.

Our focus in this work is neither on the implementation and evaluation of the descriptors proposed by ETSI MANO nor into only comparing different virtualization solutions as the works aforementioned. Our main objective is mapping lower level performance metrics into management requirements, based on the comparison of different virtualization solutions.

III. Methodology

An in-depth discussion about NFV management requirement is fundamental. More specifically, the list previously investigated must be revisited, considering different virtualization solutions [4]. Such discussion is provided in the following.

A. Management Requirements Classification

Management requirements can be observed from two perspectives. The (*i*) **qualitative** analysis requires a **subjective**

[1] https://datatracker.ietf.org/rg/nfvrg/documents/

evaluation, with network operators being interviewed or observed when using a virtualization solution. In the other hand, the (*ii*) **quantitative** analysis is performed by measuring **objective** performance metrics from the related systems. In the context of this paper, we applied the second approach, at the same time we limited the set of requirements for the following:

VMs/containers Instantiation – Once properly configured the VNF servers, network operators have to instantiate the VMs or containers hosting VNFs. Depending on the strategy and technology used, each VNF needs a dedicated VM or container to host it, probably resulting in wide ranges of instances over the NFV Infrastructure (NFVI);

VNF Deployment – Deploying VNFs in the NFVI involves both the configuration and placement of VNFs. The VNF placement is a well-known problem in the NFV literature [18], [19], with solutions focused on optimizing the placement of VNFs to avoid unnecessary migrations over the NFVI;

VM/container & VNF Monitoring – The monitoring of both VMs/containers and VNFs is essential to guarantee the proper service operation. In the context of NFV, the monitoring activity includes the VNFs status acquisition, which may require the instrumentation of VNFs to expose their internal state through management interfaces;

Physical and Virtual NFs Coexistence – The management of both physical and virtual network functions should mostly be transparent for network operators. One exception to this rule is the case where computing resources allocated to VNFs dynamically change. Network operators must be aware of the underlying workload before making any changes to the VNFs.

We performed a quantitative analysis due to the possibility to recreate the evaluated scenario for different virtualization solutions, thus giving us a better perspective on their operation. In order to evaluate the effectiveness of each virtualization solution considering the presented management requirements, we selected three performance metrics: boot time, response time, and memory consumption, which are directly related to each other due to their quantitative nature. In Section V, we provide a detailed discussion regarding the performance metrics and its mapping into the selected management requirements.

B. *Evaluation Scenario*

We first must configure the VNF servers which are responsible for hosting VMs or containers that will run VNFs. In our experiments, each server is an AMD 3.6 GHz with 4 GB of memory, placed according to Figure 1.

Boot time and memory consumption were measured directly from the VNF servers using scripts running on the Server OS. To measure the response time, however, we deployed a VNF acting as network proxy on the respective Server OS for each virtualization solution. This VNF is responsible for forwarding packets from Host A to B and the reply from Host B to A. In Host A, a script is responsible for sending a request, while another one will calculate the elapsed time between Host A send a request and identify the reply sent by Host B. By measuring the elapsed time between a request and its respective reply, it is possible to evaluate the response time inserted by the VNF running in the VNF server. In Algorithm 1 we present the pseudo-code detailing the execution sequence of our proxy.

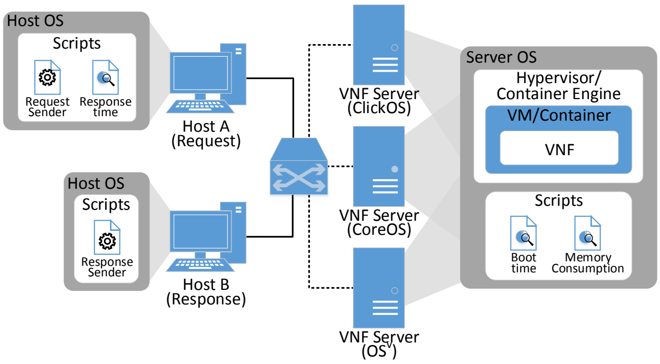


Fig. 1. Evaluation Setup

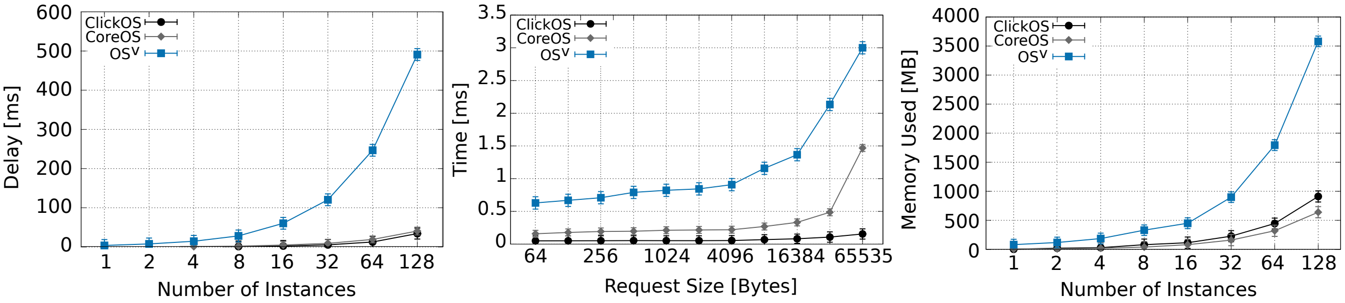
Algorithm 1 Proxy Operation Pseudo-code

```
 1: while $TRUE$ do
 2:     $incoming\_pkt \leftarrow listenInterface(eth0)$
 3:     if $incoming\_pkt \neq NULL$ then
 4:         $outgoing\_pkt \leftarrow incoming\_pkt$
 5:         if $incoming\_pkt.src\_addr == host\_A$ then
 6:             $outgoing\_pkt.dst\_addr \leftarrow host\_B$
 7:         else if $incoming\_pkt.src\_addr == host\_B$ then
 8:             $outgoing\_pkt.dst\_addr \leftarrow host\_A$
 9:         end if
10:     end if
11:     $sendPacket(outgoing\_pkt, eth0)$
12: end while
```

The proxy starts to monitor its network interface ($eth0$) until an incoming packet be received (lines 2 and 3). Then, the incoming packet is copied as an outgoing packet to be forwarded to the correct destination (line 4). Next, the proxy verifies the source of the incoming packet: if the packet is from host A, the destination address is changed to host B (lines 5 and 6). Otherwise, the destination receives host A address (lines 7 and 8). Once configured the new destination, the proxy forward the incoming packet (line 11). Our proxy operates in a limited way purposely, since the objective is to evaluate the performance of the virtualization solutions. Thus, the proxy does not add any significant packet processing time, enabling a clear evaluation over the time spent by each solution.

The next step is the setup of the servers responsible for host VNFs. We first configure the operating system (Server OS), which will provide routines needed by the hypervisor/container engine to access the VNF server functionalities. Next, for each virtualization solution a different kind of VM or container is instantiated to host VNFs, which are implemented according to the libraries provided by the virtualization solution during the creation of a VM/container, *i.e.,* the VNF description/configuration language. For each virtualization solution, a different set of configurations was applied, explained in in the following.

ClickOS: a Xen hypervisor running on top of a Linux-based system is deployed in the VNF server. ClickOS images are responsible for hosting specific Click configurations, *i.e.,* one VNF per ClickOS image. A set of network elements available on Click libraries supports the creation of VNFs. Description files containing network elements for the functions are interpreted and executed by ClickOS. In this way, the proposed



(a) Average boot time

(b) Average response time

(c) VNF server memory consumption

Fig. 2. Evaluation results comparison

proxy function (explained in Algorithm 1) was implemented using some Click elements, making a description file inserted in the ClickOS images. The VNF management in ClickOS is only possible using Cosmos, a tool developed to allow the communication between user and ClickOS domains.

CoreOS: a Linux-based server was used to host containers for each VNF. The main building block of CoreOS is the Docker engine responsible for guarantee VNFs running isolated from each other. In our setup, VNFs in CoreOS are represented by Linux functions installed on the CoreOS server. In our experimental setup, we used a CoreOS container running $iptables$ as the proxy VNF. Two rules were configured on $iptables$, operating like the proxy pseudo-code presented in Algorithm 1: one rule to forward packets coming from host A to B and another one for the opposite direction.

OS^{v}: a KVM hypervisor was configured on a Linux-based server since it is the standard hypervisor for OS^{v} management. Each OS^{v} VM was set to host one particular VNF, which is the recommended behavior by OS^{v} developers. Once designed, VNFs source codes are compiled and transferred to OS^{v} VMs, running immediately after the VMs startup. Each OS^{v} VM can be accessed using a shell, easing the access and operation control of VNFs inside them. An OS^{v} proxy image available at the OS^{v} project repository was used to instantiate the proxy. Algorithm 1 was implemented in the C programming language using Socket RAW libraries and deployed inside OS^{v} images.

For all virtualization solutions, configuration templates were used to specify VMs characteristics (*e.g.,* processing power, storage, memory, and networking). In this work, we configured the minimum recommended amount of resources needed to start a simple VNF to reply network requests: for ClickOS, one virtual CPU with 12 MB of memory; for OS^{v}, one virtual CPU with 64 MB of memory. In the CoreOS configuration, however, the processing and memory available for containers are based on the total amount of these resources on the server due to its container-based nature.

IV. Evaluation Results

In this section, the results obtained in the evaluation of each performance metric are presented in the following order: boot time, response time, and memory consumption. All measurements were repeated until guarantee a minimum confidence interval of 95%.

A. Boot Time

Fast boot time is important in any NFV deployment since that VNFs must be available as soon as possible. For example, when a VM or container that hosts a VNF crashes, a new one must be started-up as fast as possible to avoid losses in packet processing. The boot time evaluation of each virtualization solution was performed using a script to measure the time needed by a VM or container to start its operation and response to a network request (*i.e.,* send an ICMP Echo Reply in response to an ICMP Echo Request packet). It represents the effective time the VM/container needs to be operational in the network. The script used can be configured to measure the boot time for one VM/container or a group of VMs/containers, starting all the instances and calculating the total amount of time to receive the response from all VMs/containers. The results of this evaluation are summarized in Figure 2(a).

OS^{v} boot time increases faster than the others. We credit this poor performance to the wide number of libraries and commands that must be loaded during OS^{v} VMs startup. Moreover, we can conclude that ClickOS has the fastest boot time among the virtualization solutions evaluated – less than 0.25 milliseconds for 1 VM and close to 50 milliseconds with 128 VMs. This good performance, however, has a drawback: the optimizations made turned ClickOS not very user-friendly. For example, there is not a shell to access ClickOS VMs, requiring use the Cosmos tool to have a minimal level of access to ClickOS VMs domain. CoreOS also presented good performance results, being just some milliseconds slower than ClickOS, since that the functions are already installed on the CoreOS server, and the container engine (*i.e.,* Docker) needs only to manage server resources sharing among containers. Thus, CoreOS appears as a good alternative to ClickOS, by presenting a very similar performance, with the benefit of having friendly access to containers domain.

B. Response Time

The main objective here is to analyze how much time each virtualization solution spends to receive, process, and response packets. A simple VNF was implemented for each virtualization solution, acting as a proxy (*i.e.,* receiving packets from a host and forwarding them to another) according to the experimental scenario presented in Section III-B. An important point to highlight is that all virtualization solutions were analyzed without any additional network improvements, providing a fair evaluation of each virtualization solution.

Analyzing Figure 2(b) we can conclude that ClickOS presents the smallest response time for all packets sizes, being around 0.16 milliseconds in the worst case (*i.e.,* packets 64 KB sized). We credit this performance to the improvements made to directly map packet buffers into ClickOS VMs memory space. Very close to ClickOS performance we found CoreOS – under 0.5 milliseconds until packets reach sizes bigger than 16 KB. OS^{v} presented the worst results, showing a response time close to 3 milliseconds with the maximum packets size analyzed, twice as much time as the worst case of CoreOS and around eight times than the worst case of ClickOS.

C. VNF Server Memory Consumption

We measured the available memory before and after VMs/containers instantiation. We varied the number of VMs/containers instantiated from 1 to 128. Inside each host, the same function used in the boot time evaluation was instantiated. Despite the available memory for each VM can be configured using predefined templates, the total amount of memory used for each solution is also influenced by the hypervisor/container engine. The results obtained are presented in Figure 2(c).

Despite all solutions presented the same behavior, the smallest consumption was presented by CoreOS, with less than 700 MB used with 128 containers instantiated. We credit this performance to the container-based virtualization approach used by CoreOS, where the server memory is allocated according to the necessity of the functions running in each container. Very close to CoreOS performance we found ClickOS, with less than 1000 MB (or 1 GB) of memory used in the last case, which is a good performance considering the full virtualization approach used by ClickOS. Finally, OS^{v} presented the worst memory consumption, using more than 3500 MB to instantiate 128 VMs. We credit this behavior to the complexity of OS^{v} VMs, since that each VM is a single function with its own copy of OS^{v} libraries, that should be loaded with each VM. Despite different evaluations, our results for ClickOS and OS^{v} were very close to those achieved by their authors [5], [7].

V. Results Discussion

In this section, we discuss in details the relationship of the performance metrics evaluated in the previous section with the management requirements presented in Section III.

A. VMs/containers Instantiation

The process of instantiating a VM or container must be performed as fast as possible, in particular when the VNF to be hosted on it is part of an entire service chaining (or VNF-FG). Further, in order to instantiate a new VM or container, the network operator (or the placement algorithm) must be aware of how much memory will be required and, consequently, the amount of available memory in the VNF server. If the memory required by the new VM/container is high enough to compromise others VMs/containers or even the VNF server operation, another VNF server with sufficient memory must be selected. For these reasons, we argue that boot time and memory consumption must be considered for the VMs/containers instantiation management requirement.

Based on the results obtained, ClickOS and CoreOS presented best boot time and memory consumption performances, respectively. However, OS^{v} has a differential: a tool called Capstan allows the network operator to access a wide set of OS^{v} VMs images available in the project repository, facilitating the deployment of new VMs. In this way, CoreOS is a good choice regarding performance, due to its fast boot time and low memory footprint. However, if the priority is more flexibility in the overall process of VM instantiation, OS^{v} appears as an alternative, able to ease this process with the drawback of worse memory consumption.

B. VNF Deployment

The VNF deployment is related to two tasks performed over VNFs. The first one is the VNF location, also known as VNF placement problem, widely investigated in NFV literature [18], [19]. The VNF placement problem consists in the distribution of VNFs in a pull of servers according to some criteria. One of the main criteria to be considered in this problem is the resource utilization. The second task involved in the VNF deployment is the configuration/reconfiguration of VNFs. It must occur as fast as possible to avoid information loss or even service outages. Moreover, depending on the strategy used more memory may be required for VNF reconfiguration [20]. For this reason, memory consumption is as important as boot time regarding VNF deployment.

ClickOS and CoreOS are the best virtualization solutions regarding boot time and memory consumption, characterizing them as good choices in terms of VNF configuration/reconfiguration, with a small advantage for ClickOS regarding boot time, and for CoreOS concerning memory consumption. However, despite results showed CoreOS as the best choice regarding memory consumption, there is a drawback in using CoreOS: it is not suitable for migration due to its container-based virtualization. Then, we believe CoreOS is the best choice regarding memory consumption and suitable for use in static scenarios, *i.e.,* network scenarios where VNF migration is not needed. However, in cases where VNF migration is often performed (*e.g.,* for load balancing), we recommend the use of ClickOS, which is more suitable for migration with memory consumption results close to the values presented by CoreOS.

C. VM/container & VNF Monitoring

VNFs must be continuously monitored to keep the network working properly. In this way, the network response time of VMs and containers must be as fast as possible, to quickly obtain information from their operation, such as the current VNF location and its status. When the monitored VNF is a part of a VNF-FG, delays in the VNF response may impact the entire service provided by the VNF-FG. For this reason, the network response time of VMs/containers is an important metric to reflect the VNF monitoring requirement.

In our results, ClickOS appears as the best choice in terms of response time and, consequently, covering the VNF monitoring requirement better than the others solutions. Moreover, ClickOS is very suitable for VNF migration due to the small size of ClickOS VMs (approximately 12 MB), avoiding bottlenecks in the network paths during the migration and its full virtualization approach, turning the memory dump easily.

D. Physical and Virtual NFs Coexistence

Memory consumption is a key metric to reflect how good a virtualization solution is regarding VNF coexistence. If the amount of memory needed to instantiate and manage VMs or containers is too high, probably a few number of VNFs may coexist in the same VNF server. As well known by network operators, the network elements management is easier when they are close to each other. Another important metric is the network response. In cases where VNFs should communicate with each other, forming a VNF-FG, they need to fast reply to others VNFs of the VNF-FG. Moreover, delays in the network response from a VNF belonging to a VNF-FG may affect communication among VNFs composing another VNF-FG.

In our evaluation, CoreOS presented the best results regarding memory consumption while ClickOS is the best virtualization solution regarding network response time. Considering the similarity in the results of both ClickOS and CoreOS concerning memory consumption, we believe that ClickOS is the best choice regarding VNF coexistence, since that the network response of ClickOS is better than CoreOS.

We can conclude that there is no definitive virtualization solution for all evaluated performance metrics and, consequently, for all management requirements considered in this work. Despite the best performance presented by ClickOS, the creation, access, and management of ClickOS VMs is a hard task. CoreOS provides a simple way to create and access containers when compared to ClickOS, but container-based virtualization may impose some difficulties related to VNF migration due to its shared memory approach. Finally, OS^{v} appears as an alternative, by given up performance to achieve easy creation, access, and management of VMs. The choice of one or another virtualization solution depends on the network operator needs and the network scenario.

VI. Conclusion and Future Work

In this paper, we provided a comparison among emerging NFV enablers: ClickOS, CoreOS, and OS^{v}. We refined and classified management requirements from the literature in two classes, selecting quantitative ones to be evaluated based on three different performance metrics: boot time, response time, and memory consumption. Our main objective is to help network operators to choose a virtualization solution to for their NFV deployment.

Our results show that ClickOS and CoreOS are superior to OS^{v} , with advantage for ClickOS in terms of boot time ($18,61\%$ faster than CoreOS in the best case) and response time (approximately 10 times faster than CoreOS for packets 64 KB sized). Nevertheless, CoreOS required $42,34\%$ less memory than ClickOS for 128 instances. However, MANO solutions should also consider the drawbacks using one or another solution in NFV environments. For example, in more dynamic scenarios where VNFs are often updated or reconfigured, solutions like OS^{v} could be an promising alternative, due its diversified images database.

Now, we plan to design a comprehensive management system for NFV, selecting the best virtualization solution to cover as many as possible the management requirements. We also plan to investigate orchestration mechanisms for NFV, working in a synergy among all the good practices proposed by ETSI and the management requirements evaluated in this work, proposing a unified design pattern for NFV MANO.

References

[1] M. Chiosi *et al.*, "Network Functions Virtualisation (NFV)," ETSI NFV ISG, White Paper, 2012, https://portal.etsi.org/nfv/nfv_white_paper2.pdf.

[2] N. Feamster, J. Rexford, and E. Zegura, "The Road to SDN: An Intellectual History of Programmable Networks," *ACM Sigcomm Computer Communication*, vol. 44, no. 2, pp. 87–98, 2014.

[3] F. Risso, A. Manzalini, and M. Nemirovsky, "Some Controversial Opinions on Software-Defined Data Plane Services," *IEEE SDN for Future Networks and Services (SDN4FNS)*, pp. 1–7, Nov. 2013.

[4] L. Bondan, C. R. P. d. Santos, and L. Z. Granville, "Management requirements for ClickOS-based Network Function Virtualization," in *International Workshop on Management of SDN and NFV Systems (ManSDN/NFV) collocated with the International Conference on Network and Service Management (CNSM)*, Nov 2014, pp. 447–450.

[5] J. Martins, M. Ahmed, C. Raiciu, V. Olteanu, and M. Honda, "ClickOS and the Art of Network Function Virtualization," *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2014.

[6] CoreOS: Open Source Projects for Linux Containers. Available at: https://coreos.com/. Accessed December, 2015.

[7] A. Kivity, D. Laor, G. Costa, P. Enberg, N. Har'El, D. Marti, and V. Zolotarov, "OSv—Optimizing the Operating System for Virtual Machines," in *USENIX Annual Technical Conference (USENIX ATC 14)*. Philadelphia, PA: USENIX Association, Jun. 2014, pp. 61–72.

[8] J. Hwang, 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, March 2015.

[9] CirrOS: a Tiny OS that specializes in running on a cloud. Available at: https://launchpad.net/cirros. Accessed December, 2015.

[10] Alpine Linux. Available at: http://www.alpinelinux.org/. Accessed December, 2015.

[11] R. Mijumbi, J. Serrat, J. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Surveys & Tutorials*, 2015, to appear.

[12] J. Quittek *et al.*, "Network Functions Virtualisation (NFV) - Management and Orchestration," ETSI NFV ISG, White Paper, 2014.

[13] M. Chiosi *et al.*, "Network Functions Virtualisation (NFV) - Virtualisation Requirements," ETSI NFV ISG, White Paper, 2013.

[14] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, "The Click Modular Router," *ACM Transactions on Computer Systems*, vol. 18, no. 3, pp. 263–297, Aug. 2000.

[15] Z. J. Estrada, F. Deng, Z. Stephens, C. Pham, Z. Kalbarczyk, and R. Iyer, "Performance comparison and tuning of virtual machines for sequence alignment software," *Scalable Computing*, vol. 16, no. 1, pp. 71–84, 2015.

[16] W. Felter, A. Ferreira, R. Rajamony, and J. Rubio, "An updated performance comparison of virtual machines and linux containers," in *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, March 2015, pp. 171–172.

[17] P. Reddy and L. Rajamani, "Performance comparison of different operating systems in the private cloud with kvm hypervisor using sigar framework," in *International Conference on Communication, Information Computing Technology (ICCICT)*, Jan 2015, pp. 1–6.

[18] H. Moens and F. D. Turck, "VNF-P : A Model for Efficient Placement of Virtualized Network Functions," in *International Conference on Network and Service Management (CNSM)*, Nov 2014, pp. 418–423.

[19] S. Clayman, E. Maini, A. Galis, A. Manzalini, and N. Mazzocca, "The dynamic placement of virtual network functions," in *IEEE Network Operations and Management Symposium (NOMS)*, May 2014, pp. 1–9.

[20] V. A. Olteanu and C. Raiciu, "Efficiently migrating stateful middleboxes," *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 4, p. 93, Sep. 2012. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2377677.2377697

ANNEX F PUBLISHED PAPER – WCNC 2016

This paper is a late result of my master degree, in which we proposed a channels classification solution for opportunistic wireless networks, based on concepts of cognitive radio.

- **Title:**
  ChiMaS: A Spectrum Sensing-based Channels Classification System for Cognitive Radio Networks
- **Conference:**
  IEEE Wireless Communications and Networking Conference (WCNC)
- **URL:**
  <https://wcnc2016.ieee-wcnc.org/>
- **Date:**
  3-6 April, 2016
- **Held at:**
  Doha, Catar
- **Digital Object Identifier (DOI):**
  <10.1109/WCNC.2016.7564911>

ChiMaS: A Spectrum Sensing-based Channels Classification System for Cognitive Radio Networks

Lucas Bondan, Marcelo Antonio Marotta, Leonardo Roveda Faganello,
Juergen Rochol, Lisandro Zambenedetti Granville
Institute of Informatics – Federal University of Rio Grande do Sul
Av. Bento Gonçalves, 9500 – Porto Alegre, Brazil
E-mail: {lbondan, mamarotta, lrfaganello, juergen, granville}@inf.ufrgs.br

***Abstract*—Cognitive radio devices are able to sense the spectrum of frequencies and share access to vacant channels. These devices usually have a candidate channels list that must be sensed to find a vacant channel. In this paper, we propose a novel system called *ChiMaS*, which is able to manage the candidate channels list implementing three tasks: Analysis, Creation, and Sort. Analysis applies reinforcement learning algorithms to evaluate the channels quality based on their historical occupancy and their conditions; Creation is responsible for creating the Candidate Channels List; and Sort ranks the channels to obtain an Ordered Channels List in terms of quality. Results show that *ChiMaS* manages the candidate channels list following the IEEE 802.22 definition, while it finds the best channel in terms of availability and quality faster than Q-Noise+ algorithm, which was implemented for comparison purpose.**

Keywords*—*cognitive radio, channel sensing order, channel list management, reinforcement learning

I. Introduction

Cognitive Radio (CR) devices have been designed to improve the efficiency of spectrum allocation [1]. Such improvement is obtained by enabling these devices to opportunistically access vacant radio frequency channels. However, before accessing a channel, a CR device must decide whether or not that channel is occupied. This decision is performed by the Spectrum Sensing Function (SSF), to find vacant channels and allow multiple devices to share the spectrum without interfering with each other. While selecting a channel, CR devices should also be able to take into account the channel quality. In this sense, the historical occupancy and the channel conditions, which can be measured by the Received Signal Strength Indicator (RSSI) [2], are two of the main characteristics used to determine the channel quality.

Finding the best channel is desirable to provide better transmission conditions to CR devices. However, it is hard to find high-quality vacant channels in short periods of time, since the analysis of channel conditions demands more time than just deciding whether or not the channel is vacant [3]. The amount of time spent to sense the spectrum is also related to the number of channels previously defined to be analyzed. Therefore, the IEEE 802.22 standard [4] defines that CR devices must keep a Candidate Channels List (CCL), in order to limit the duration of the spectrum sensing. Although the IEEE 802.22 standard specified the existence of a list of candidate channels, the classification of the channels in this list was left open to encourage innovation.

In the recent literature, solutions have been presented to sort available channels and quickly select one of them. We organized the solutions into two classes: statistical sorting [5] [6] and reinforcement learning sorting [7] [8] [9]. Both classes aim to dynamically decide in which order channels should be sensed when a device needs to change its operating channel. Towards this aim, the first class is composed of solutions that demand prior knowledge about channel quality metrics, such as the historical occupancy and RSSI of the channel [10]. Based on the defined metrics, statistics are applied to sort the channels according to their quality. The second class, in turn, is composed of solutions that demand no prior knowledge about the channel quality. The knowledge is built over the time by analyzing the transmissions performed by CR devices.

Although broadly applied by current solutions, the approach of analyzing the transmissions performed by CR devices might cause some drawbacks. The first drawback is that the current solutions assume that transmissions occur before CR devices start learning about the channel quality. Since CR devices do not have any prior knowledge about the channel occupancy, this leads to a second drawback related to the probability of a CR device to interfere with other transmissions that may be already occupying the analyzed channel. Finally, another drawback of this current approach is that the chosen channel may not be the best one in terms of quality, especially at the beginning of system operation, since no historical information is available. Therefore, the selected channel may have a high occupancy rate or bad channel conditions, what leads to a low-quality transmission.

To deal with these drawbacks, in this paper we introduce *ChiMaS*, a solution to classify channels in the CCL. The classification of the channels in this list includes three tasks: Analysis, Creation, and Sort, which are defined as follows: *(i)* to analyze the radio frequency channels, *ChiMaS* uses a reinforcement learning based solution, taking into account its historical occupancy and conditions. *(ii)* To create the CCL, only channels considered as vacant are selected. *(iii)* To sort the CCL, scoring and ranking functions are applied to the created list. By evaluating the channels with the spectrum sensing, *ChiMaS* allows CR devices to learn, and consequently find the best available channel, without the need to transmit. The performance of *ChiMaS* is evaluated in a controlled radio environment and the results obtained are compared with the Q-Noise+ algorithm from the literature [8]. Results show that *ChiMaS* is able to find the best available channel faster than Q-Noise+ and to sort CCL in different scenarios.

The remainder of this paper is organized as follows. In Section II, we present related work on solutions to sort vacant channels. *ChiMaS* is described in Section III. Performance evaluations are presented and discussed in Section IV. Finally, we present the conclusions and future work in Section V.

II. Related Work

Solutions to sort channels have been investigated in the scientific community in the recent past. Therefore, we organize the main solutions in two classes: statistical sorting [5] [6] and reinforcement learning sorting [7] [8] [9]. Statistical sorting assumes that conditions and occupancy of the channel are known. Jiang *et al.* [5] analyzed a scenario in which the occupancy probability of each channel is known. The authors considered a CR network with opportunistic transmissions to find an optimal channel order to achieve the maximum gain in terms of transmission rate. Rostami, Arshad, and Moessner [6] proposed an ordered statistic SSF based on a non-parametric method considering the presence of Additive White Gaussian Noise (AWGN) and assuming that the information about the Signal to Interference plus Noise Ratio (SINR) is known. Although allowing the ordering of the channels, the main limitation of statistic sorting class is the need of prior knowledge about channel characteristics.

The second class of sorting solutions applies reinforcement learning algorithms to dynamically define the channel order. This learning dispenses any prior knowledge about the channel occupancy, since knowledge is acquired during transmissions. Mendes *et al.* [7] applied the reinforcement learning algorithm called Q-Learning to obtain information about channel occupancy. Q-Learning algorithm calculates a reward for each transmission in a channel. Based on this reward, the algorithm defines the order in which channels must be analyzed. Faganello *et al.* [8] proposed an improvement of Q-Learning algorithm for cognitive sensor networks, called Q-Noise+, which considers historical analysis of channel occupancy and channel conditions based on SINR. Q-Noise+ also calculates the reward considering transmissions of CR devices. Finally, Zhang *et al.* [9] modeled the sensing order selection as a Q-Learning problem, defining the sensing order based on the results of transmissions and historical sensing performed over the channels.

Both classes described above are based on information obtained during the transmissions performed by CR devices. To obtain this information, a device faces the following drawbacks: *(i)* the learning is performed only in the channel in which the CR device transmits, *(ii)* a transmission can cause interference with other transmitters, and *(iii)* the chosen channel may have low availability and poor conditions in the initial transmissions because there is no prior knowledge about the channel characteristics. The main contribution of *ChiMaS* is to deal with these drawbacks, by managing the channel list in order to find the best vacant channel. Moreover, to the best of our knowledge, *ChiMaS* is the first approach to order the channel list using channels quality considering only the SSF results, *i.e.* CR devices are not required to transmit.

III. *ChiMaS* Components

ChiMaS is divided into three classification tasks, as can be seen in Figure 1. The first one, called Analysis, receives information from the SSF regarding the occupancy status of a Global Channels List (GCL). This list comprises a group of channels previously defined to be analyzed by *ChiMaS*. Channels in the GCL can be defined based on some criteria, like unlicensed channels for IEEE 802.22 operation [11], for example. The GCL is processed by a reinforcement learning algorithm to become aware about both the historical occupancy and conditions of each channel. Based on the results of such analysis, the second task, Creation, is responsible for the generation of the CCL. Finally, the third task, Sort, uses a scoring and a ranking function to obtain an Ordered Channels List (OCL), which is the output of *ChiMaS*. The first element of the OCL may be used as the Operation Channel of a Base Station, as defined by the IEEE 802.22 standard [4], while the second element may be considered a good Backup Channel. The Analysis task of *ChiMaS* is presented in Subsection III-A, while Creation and Sort tasks are described in Subsection III-B.

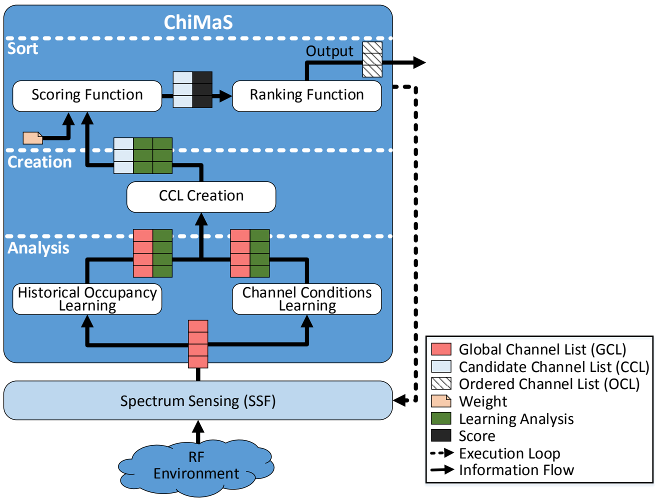


Fig. 1. *ChiMaS* components

A. Analysis Task

The Analysis task receives from the SSF, a GCL containing the channels to be analyzed by *ChiMaS* along with information regarding these channels. At each *ChiMaS* execution, all channels are sensed to prevent sudden changes in channels status do not be perceived. The GCL information is composed of two types of data structures for each channel as defined by the IEEE 802.22 standard and represented in Figure 2. The first data structure is a tuple composed of Signal Vector and Confidence Vector. The Signal Vector contains information regarding the occupancy status of the channel, *i.e*, the result of the SSF. In this case, SSF must indicate if the channel is occupied (0x00), vacant (0xFF) or if it was unable to decide (0x7F). The Confidence Vector carries information about the assurance of the SSF in the current result. The confidence level received by *ChiMaS* varies between 0 (0x00), indicating no confidence and 1 (0xFF), representing full confidence. The second data structure is a vector containing RSSI measurements. This vector ranges from -104dBm (0x00) to +23.5dBm (0xFF). Values outside this range shall be assigned to the closest extreme.

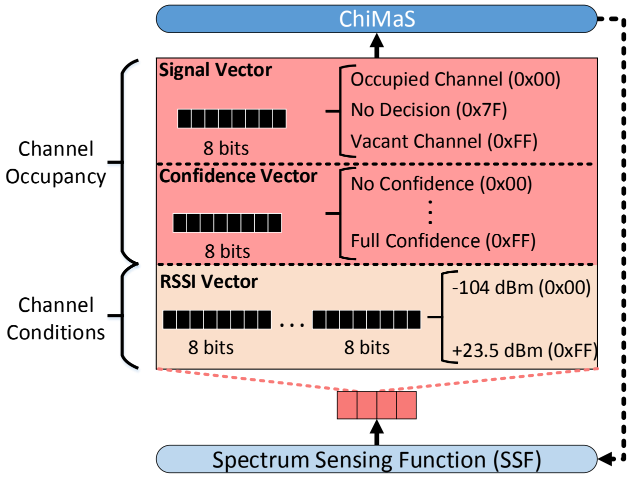


Fig. 2. GCL data structures

GCL contains information about the occupancy status and conditions of each sensed channel. This information is analyzed by two reinforcement learning algorithms to update the historical knowledge about both channels occupancy and conditions. We chose reinforcement learning because it presents a good performance on structured networks solutions, such as the IEEE 802.22 [12]. The results of this analysis allow *ChiMaS* to evaluate the quality of the channels. The reinforcement learning algorithms were implemented adapting the specifications of Q-Noise+, proposed by Faganello *et al.* [8]. The main adaptation we made was to eliminate the need for transmission, which is a drawback of Q-Noise+. Instead, we use information regarding the channel occupancy and conditions collected by SSF to learn about channels quality and compose the OCL. As proposed by Faganello *et al.* [8] we defined weights for each epoch in order to add greater importance to more recent analyzes.

The first learning algorithm of *ChiMaS* is called Historical Occupancy Learning, which is responsible for analyzing the usage profile of the channels. This analysis considers that SSF executions are performed in epochs (t). The goal of this feature is to use SSF information to assess the future occupancy of the channel. Towards this goal, a reward-based approach is applied considering two criteria to calculate Qh, which represents the results of the Historic Occupancy Learning. The criteria used to calculate Qh are: *(i)* the channel occupancy rate in the current epoch (r_t) and *(ii)* the weighted sum of this rate in a defined amount of past epochs (l). The former rate is defined by analyzing every partial analysis conducted by SSF to define whether or not the channel is occupied. This information is obtained in the Confidence Vector, which determines how accurate was the SSF analysis. Let G be the set of channels in the GCL, the Qh of a given channel c for the next epoch is then defined according to Equation 1.

$$\forall c \in G \Rightarrow Qh_t(c) = (1-\alpha)\sum_{i=1}^{l}[w_{t-i}r_{t-i}](c) + \alpha r_t(c) \quad (1)$$

where, $0 \leq \alpha \leq 1$ represents the weight of the reward (r_t) obtained in the last epoch. The higher the α value, the more importance is given to the last epoch and consequently, less importance to past epochs. The number of past epochs to be considered for Qh calculation is defined by l. In this sense, w is the weight of each one of the last l epochs. This value is pondered by the weight of the past epochs, which is $(1-\alpha)$.

Channel Conditions Learning is the second algorithm proposed in *ChiMaS*. This algorithm receives information about the mean RSSI level of a radio frequency channel to obtain knowledge about its conditions and calculate Qn, which represents the results of the algorithm. The criteria used to calculate the Qn are *(i)* the rate of RSSI in the current epoch (η_t) and *(ii)* the weighted sum of this rate in a defined amount of past epochs (l). It is important to highlight that RSSI measurements performed by SSF are considered by *ChiMaS* analysis task only in epochs where the channel is considered vacant, since only noise is present in this case. The Qn for a given channel c is calculated according to Equation 2.

$$\forall c \in G \Rightarrow Qn_t(c) = (1-\beta)\sum_{i=1}^{l}[w_{t-i}\eta_{t-i}](c) + \beta\eta_t(c) \quad (2)$$

where, $0 \leq \beta \leq 1$ is the weight of the current channel conditions, and its complement is the weight of the conditions of past l epochs where the channel was considered vacant. The β value works for the channels condition learning like the α value works for historical occupation learning: higher values implies in more importance to the last epoch and less importance to past epochs. Finally, η is a factor regarding the channel conditions. This factor represents the reward of the Channel Conditions Learning. The better the channel, the higher η is.

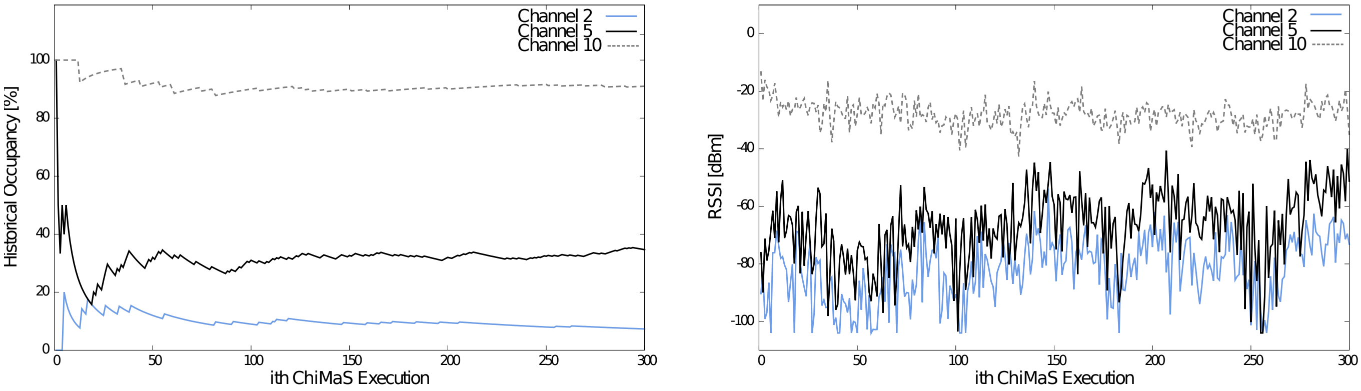
B. Creation and Sort Tasks

In the Creation task, the CCL Creation function receives the GCL from the Historical Occupancy Learning and Channel Conditions Learning and creates the CCL taking into account the occupancy status of every channel in GCL. Only vacant channels are used to create the CCL. The results of the analysis of both historical occupancy and channel conditions of vacant channels are also part of the created list. It is important to emphasize that in the next execution all channels are sensed and analyzed, even those considered occupied by SSF in the current execution.

The Sort task is responsible for sorting the CCL using two functions, called Scoring and Ranking. The former receives the weight of both historical occupancy and channel conditions to calculate a score associated with each channel. The latter sorts the list according to the results of the Scoring function. The obtained score is called Q-Value and indicates how suitable a channel is for opportunistic transmissions, considering its historical occupancy and conditions. In this sense, let C be the set of channels in the CCL. The Q-Value of a given channel $c \in C$ is obtained using Equation 3.

$$\forall c \in C \Rightarrow Q_{-Value} = \gamma * Qh_{t+1}(c) + (1-\gamma) * Qn_{t+1}(c) \quad (3)$$

where γ is the weight of historical occupancy, and $(1-\gamma)$ represents the weight of channel conditions. The score of



(a) Historical Occupancy

(b) Channel Condition

Fig. 3. Radio Frequency Scenario

each channel is then processed by a Ranking function, which is responsible for finishing the sort and creating the OCL. Therefore, the most suitable channel for transmission (channel with the highest Q-Value) will be placed at the beginning of the OCL, while the worst channel (lowest Q-Value) will be at the end of the list.

Once presented and explained each component of *ChiMaS*, in Algorithm 1 we present the pseudo-code detailing the execution sequence of *ChiMaS*, exploring the components presented in this section in order to better understand the entire *ChiMaS* execution. After defining the GCL, the SSF is invoked to obtain information about current channels occupancy and conditions (line 1). Then, during the Analysis Task, Q_h and Q_n values of each channel are calculated for the current epoch t (0, line 2). The Historical Occupancy Learning (Q_h) and the Channel Conditions Learning (Q_n) for each channel are calculated based on the past epochs (lines 8 and 10), resulting in its respective reward value (lines 12 and 13). Next, the Creation task analyses the availability of each channel (line 19), including in the CCL only channels available during the last sensing (C in the algorithm, line 20). Finally, in the Sort task the resulting Q-Value for each channel is calculated by the Scoring function (line 27), attributing the predefined weights for both Q_h and Q_n. The final values are used by the Ranking function to create the OCL (O in the algorithm, line 29), ordered from the best channel (highest Q-Value) to the worst (lowest Q-Value).

Algorithm 1 *ChiMaS* Operation

```
 1: $G \leftarrow SSF(GCL)$
 2: $t = current\_epoch$
 3:
 4: # ANALYSIS:
 5: for each $c \in G$ do
 6:     for $i = 1$ to $l$ do
 7:         # OccupancyLearning:
 8:         $c[Q_h] = c[Q_h] + c[w[t]] * c[r[t-i]] + \alpha * c[r[t]]$
 9:         # ConditionsLearning:
10:         $c[Q_n] = c[Q_n] + c[w[t]] * c[\eta[t-i]] + \beta * c[\eta[t]]$
11:     end for
12:     $c[Q_h] = (1-\alpha) * c[Q_h]$
13:     $c[Q_n] = (1-\beta) * c[Q_n]$
14:     $t$++
15: end for
16:
17: # CREATION:
18: for each $c \in G$ do
19:     if $isFree(c)$ then
20:         $C \leftarrow append(c)$
21:     end if
22: end for
23:
24: # SORT:
25: for each $c \in C$ do
26:     # Scoring:
27:     $c[Q\text{-}Value] = \gamma * c[Qh[t+1]] + (1-\gamma) * c[Qn[t+1]]$
28: end for
29: $O \leftarrow Ranking(C)$
30:
31: return $O$
```

IV. Evaluation and Results

The methodology used for assessing the performance of *ChiMaS*, the experimentation parameters, and their values are presented in Subsection IV-A. Experimental results obtained in a controlled radio frequency environment are presented and discussed in Subsection IV-B.

A. Evaluation Methodology

ChiMaS was evaluated using GNU Radio framework[1] and USRP2 radio front-end[2]. SSF was performed considering eleven channels, using a combination of energy detection and waveform detection algorithms proposed in a previous work [13]. The occupancy rate of each channel is modeled following a Poisson distribution, as proposed by Gosh *et al.* [14]. The mean and variance of this distribution, which give the channel occupancy rate was varied from 0 to 1 in steps of 0.1. To assess the channel conditions, RSSI is measured during SSF. The resulting RSSI is fit into a range described by the η factor to be used as a reward for the Channel Conditions Learning. This factor assumes values according to Table I. RSSI value of a channel may vary between +23.5dBm and -104dBm, as

[1] http://www.gnuradio.org; [2] http://www.ettus.com

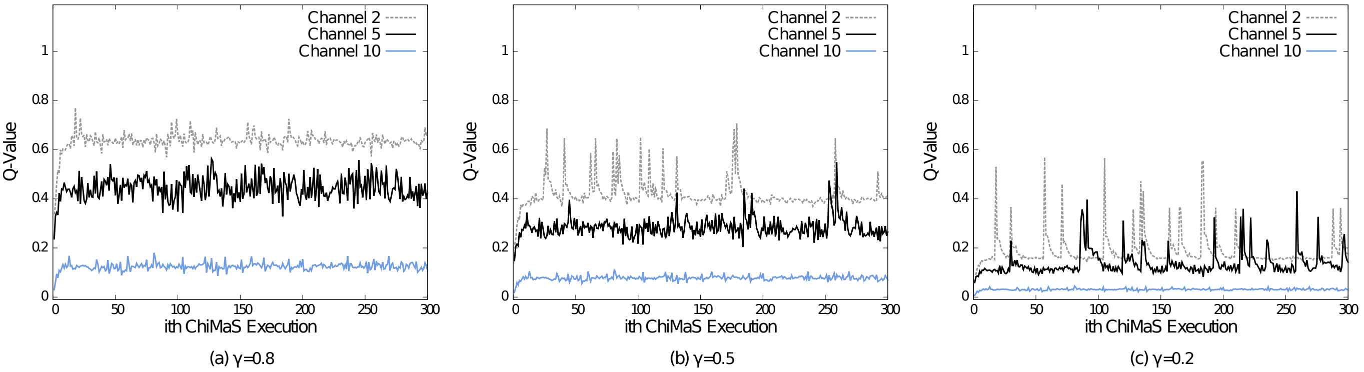


Fig. 4. Q-Value analysis

defined in the IEEE 802.22 standard [4].

TABLE I. RSSI LEVEL CORRESPONDENCE

RSSI value	Corresponding η
+23.5dBm > RSSI ≥ -30dBm	0
-30dBm > RSSI ≥ -60dBm	0.2
-60dBm > RSSI ≥ -80dBm	0.5
-80dBm > RSSI ≥ -90dBm	0.75
-90dBm > RSSI > -104dBm	0.90
RSSI ≤ -104dBm	1

ChiMaS is configured to consider a history of 3 epochs (l) for both Historical Occupancy Learning and Channel Conditions Learning. The current epoch weight for both α and β was set to 0.5 and the past epochs weights (w), defined from the most recent to the oldest one, are 0.45, 0.35, and 0.2, respectively. These values were chosen to give a higher importance to more recent epochs because it is important to consider the current state of the channel in scenarios with high RSSI variability. Moreover, it avoids the selection of overloaded channels or intermittent use, like sensor TV channels or sensor networks. γ was set to 0.8, 0.5, and 0.2, while the Channel Conditions Learning weight is $1 - \gamma$. It implies in three different setups to compute the Q-Value used for sorting the CCL. This values were selected to explore both situations where one weight is higher than the other (*e.g.* $\gamma = 0.2$ or $\gamma = 0.8$) and where the weights are balanced ($\gamma = 0.5$). We performed 300 executions of *ChiMaS* for each evaluation, guaranteeing a confidence interval of 95%.

TABLE II. TABLE OF PARAMETERS

Parameter	Default Value
Number of channels (C)	11
Current epoch weight (α and β)	0.5
Past epochs (l)	3
Past epochs weights (w)	[0.45, 0.35, 0.2]
Historical Occupancy Learning weight (γ)	0.8, 0.5, and 0.2
ChiMaS Executions	300
Confidence interval	95%

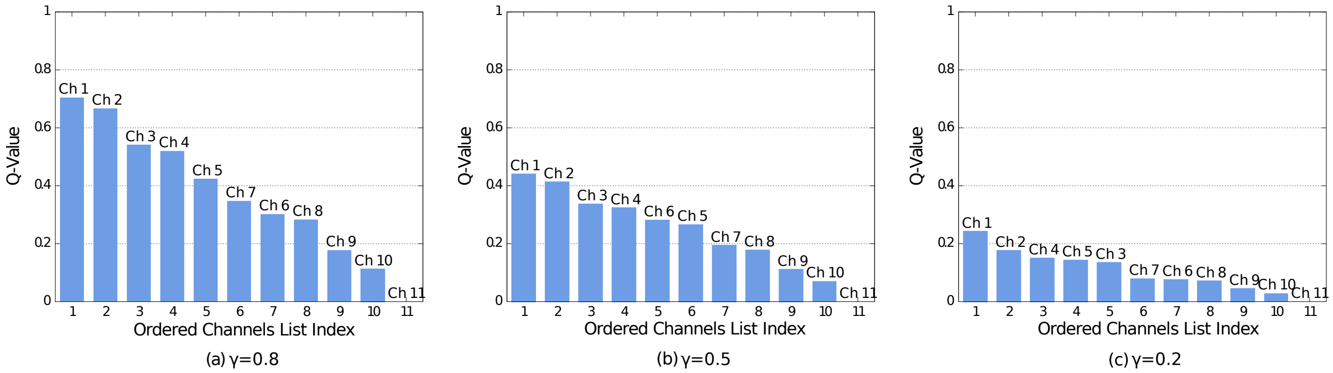
In the first scenario, we measured the average historical occupancy of TV channels, as shown in Figure 3 (a). We also analyzed the channel conditions, based on the RSSI observed during SSF, as can be seen in Figure 3 (b). In this analysis, channel 2 presented a low occupancy rate, of approximately 10%, and good conditions, since its RSSI is about -85dBm. On the other hand, channel 10 had a high occupancy rate, close to 90%, and a higher RSSI (-30dBm), resulting in a worse channel compared to channel 2. Furthermore, channel 5 presented intermediate values for both the occupancy rate and channel conditions, about 30% and -70dBm, respectively. This scenario was used as input to evaluate the process implemented by *ChiMaS* to manage the CCL.

B. Experimental Results

In the first analysis performed, we investigate the impact on Q-Value of three different setups of weights for *ChiMaS* operation, as can be seen in Figure 4. In the first setup, shown in Figure 4 (a), we consider $\gamma = 0.8$, representing a weight of 80% for historical occupancy and 20% for channel conditions. In the second setup, the weights are balanced, as can be seen in Figure 4 (b), while, in the third setup, shown in Figure 4 (c), we consider $\gamma = 0.2$.

Analyzing the behavior of channels in different setups, it is possible to observe that Q-Value decreases as γ decreases. It occurs because the historical occupancy varies less than the channel conditions, as shown in Figure 3. Therefore, as γ reduces, the variations on RSSI tend to result in peaks in the Q-Value of the observed channels. Another important analysis regards the behavior of channel 2 in the same setups. As can be seen in Figure 4 (c), the Q-Value of this channel presents a higher variability when compared to the remaining setups. This variation was caused due to two reasons. The first one is related to the higher weight attributed to the channel conditions. The second one is due to the frequent changes on the RSSI of the analyzed channel, causing variations in the η factor, as defined in Table I. Similar behaviors are observable in other channels as well. However, in the remaining visualized channels, the intensity of Q-Value changes is lower than in channel 2 because the historical occupancy and channel conditions change smoothly, as shown in Figure 3. This evaluation allows us to conclude γ must be defined according to the radio behavior. For example, in a very noisy environment γ may be configured to a small value, aggregating more importance to channels conditions learning. In contrast, with good signal propagation conditions, γ can be set with higher values, making the analysis of channels historical occupancy more effective.

Once analyzed the impact of the Q-Value in the *ChiMaS* operation, we defined a GCL composed by 11 channels to be classified, showing the final result of *ChiMaS* classification over an predefined GCL. The occupation and conditions of the channels present in the GCL where configured to vary from

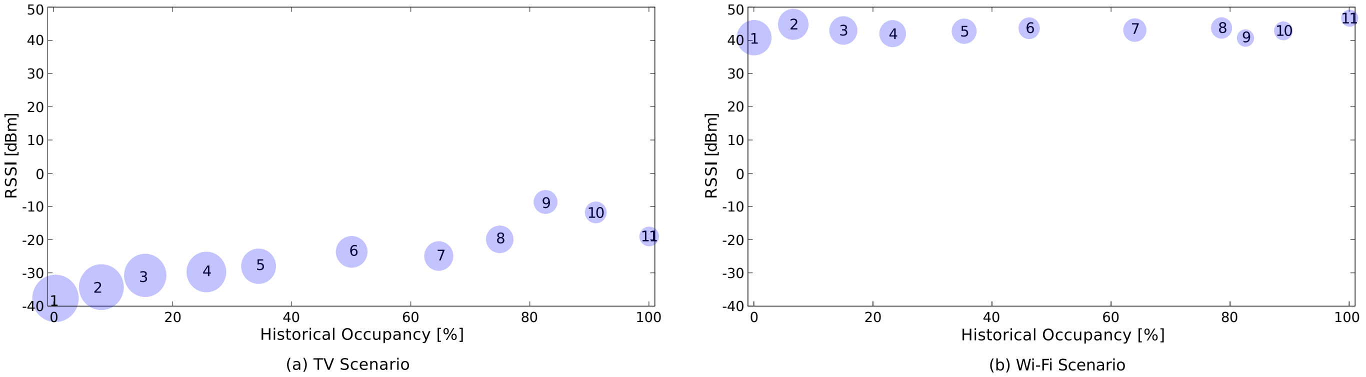


(a) γ=0.8

(b) γ=0.5

(c) γ=0.2

Fig. 5. Channel list analysis



(a) TV Scenario

(b) Wi-Fi Scenario

Fig. 6. OCL analysis

the best channel (channel 1) to the worse (channel 11). As described at the beginning of this section, the occupancy rate of each channel is modeled following a Poisson distribution, as proposed by Gosh *et al.* [14] and the channels conditions where varied between +23.5dBm and -104dBm.

The first evaluation over the 11 channels was made to analyze the impact of γ variation in the composition of the OCL. In Figure 5, we show the results regarding the OCL, where the x-axis represents OCL index and the y-axis represents the Q-Value calculated for each channel. Analyzing the best channel of the first setup, we can observe that the Q-Value is around 0.7. As γ decreases, channel 1 remains as the best channel. However, the Q-Value of all channels is reduced. Another interesting result we can observe in Figure 5 is the behavior of channels in the middle of the OCL, since their positions tend to vary as the setup is changed. For example, channel 5 is one of the most affected by the changes in weight parameters. In setup (a), channel 5 is placed in the 5th position, moving to 6th place when the weights are balanced (b), and finally moving to the 4th position in (c). On the other hand, the channels at the end of the OCL tend to keep a constant position in the ranking, because the learning features of the Analysis task are reward-based. Therefore, as the Q-Value of these channels is low, a small reward is obtained.

Another analysis conducted in this paper is a comparison between two different scenarios: (a) TV channels scenario and (b) Wi-Fi channels scenario. We assume a setup where $\gamma = 0.5$ in both scenarios. The results regarding such comparison are presented in Figure 6, where both the historical channel occupancy (x-axis) and the channel conditions (y-axis) are correlated with the final Q-Value (circles). The larger the size of the circle, the greater the Q-Value of the channel.

In Figure 6 we can also analyze the outputs of the learning algorithms in the Analysis task. As shown in Figure 6 (a), the best channel is the one with higher Q-Value, *i.e.* channel 1. This channel is the one with the lowest historical occupancy rate and the best channel conditions. On the other hand, the worst channel (*i.e.* channel 11) is highly used and presents bad channel conditions, resulting in the lowest Q-Value. The results regarding the second scenario are presented in Figure 6 (b). We can observe that despite the average RSSI is higher in the second scenario, *ChiMaS* remains able to manage the CCL. Another conclusion we can take from the plots is that considering the analyzed scenarios, (a) is more suitable for operation of CR devices. It is justified because the average Q-Value is higher than in scenario (b).

Finally, we present a comparison between *ChiMaS* and Q-Noise+ algorithms. Q-Noise+ parameters are set to the same values used by Faganello *et al.* [8]. In this specific analysis, both algorithms are compared considering a variable amount of channels ranging from 1 to 48. We perform a comparison between the time spent by *ChiMaS* and Q-Noise+ to analyze these channels, considering a sensing time of 2 seconds for both algorithms and a transmission time of 2 seconds for Q-

Noise+. The results obtained are shown in the Figure 7.

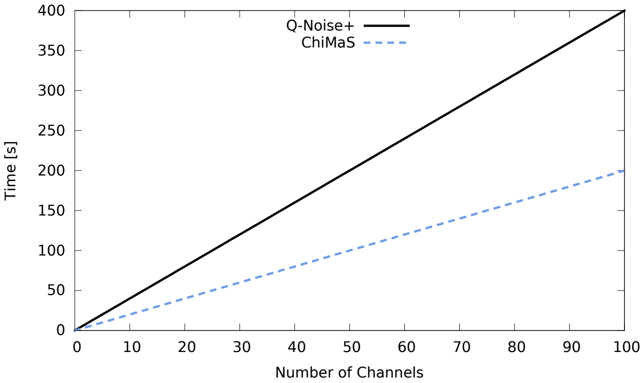


Fig. 7. Time analysis

Figure 7 shows that the time necessary to analyze the channels linearly grows, for both algorithms, as the number of channels to be analyzed increases. However, as the number of channels increase, Q-Noise+ time grows considerably faster than *ChiMaS* time. This behavior occurs because the time demanded by *ChiMaS* depends only on the number of channels to be analyzed and the sensing time, while Q-Noise+ also needs to consider the transmission duration in the composition of the total demanded time.

In Table III, we highlight the main differences between *ChiMaS* and Q-Noise+. Each evaluation performed by *ChiMaS* analyzes a set of channels, defined before starting its execution, while Q-Noise+ is able to analyze only one channel per execution. It could lead to a situation where some channels are never analyzed by Q-Noise+, since when it is analyzing a reasonable channel, it tends to keep transmitting in this channel, without searching for better possibilities. To calculate the reward of the learning process, *ChiMaS* evaluates the Confidence Vector, while Q-Noise+ needs to transmit over the channels to calculate its reward, which may cause interference with transmissions performed by other devices.

TABLE III. COMPARISON OF *ChiMaS* AND Q-NOISE+

Characteristic	***ChiMaS***	**Q-Noise+**
Number of channels analyzed per execution	*N* Channels	One channel
Time spent to find the best channel	Depends on the number of analyzed channels	The best channel may never be found
Reward method	Confidence Vector analysis	Transmission Analysis
Interference with primary user	No interference	May interfere with primary users

V. CONCLUSIONS AND FUTURE WORK

In this paper we presented *ChiMaS*, a novel system able to manage the CCL defined in the IEEE 802.22 standard. Three classification tasks are proposed: Analysis, Creation, and Sort. One of the main contributions of *ChiMaS* is to eliminate the need for transmissions in order to learn about the quality of the channels. Instead of transmitting, our proposed system receives, from SSF, information regarding the channel occupancy, the confidence in its result, and RSSI measurements. Results were obtained using a controlled radio environment and showed that *ChiMaS* is able to find the best available channel and sort the CCL in different scenarios, according to different setups.

As future work, we intend to consider a CR network where a central node performs the channel classification and disseminates the OCL to CR devices through a control channel. Additionally, we plan to apply rules for channel dissemination aiming to provide quality of service for applications in the context of CR networks. One possible approach to apply rules should be managing the channel dissemination by using pricing related techniques.

ACKNOWLEDGEMENTS

The authors would like to thanks to Cristiano Bonato Both and Maicon Kist for their significant contributions at the beginning of this work.

REFERENCES

[1] I. Mitola, J. and J. Maguire, G.Q., “Cognitive radio: making software radios more personal,” *IEEE Personal Communications*, vol. 6, pp. 13–18, 1999.

[2] C. Bas and S. Ergen, “Spatio-Temporal Characteristics of Link Quality in Wireless Sensor Networks,” in *IEEE Wireless Communications and Networking Conference (WCNC)*, April 2012, pp. 1152–1157.

[3] I. Akyildiz, W.-Y. Lee, M. Vuran, and S. Mohanty, “A survey on spectrum management in cognitive radio networks,” *IEEE Communications Magazine*, pp. 40–48, 2008.

[4] IEEE, “IEEE Standard for Information Technology - Telecommunications and information exchange between systems Wireless Regional Area Networks (WRAN) - Specific requirements Part 22:,” *IEEE Std 802.22*, pp. 1–680, 2011.

[5] H. Jiang, L. Lai, R. Fan, and H. Poor, “Optimal selection of channel sensing order in cognitive radio,” *IEEE Transactions on Wireless Communications*, vol. 8, pp. 297–307, 2009.

[6] S. Rostami, K. Arshad, and K. Moessner, “Order-Statistic Based Spectrum Sensing for Cognitive Radio,” *IEEE Communications Letters*, vol. 16, no. 5, pp. 592–595, 2012.

[7] A. Mendes, C. Augusto, M. da Silva, R. Guedes, and J. de Rezende, “Channel Sensing Order for Cognitive Radio Networks Using Reinforcement Learning,” in *IEEE Conference Local Computer Networks (LCN)*, Oct 2011, pp. 546–553.

[8] L. Faganello, R. Kunst, C. Both, L. Granville, and J. Rochol, “Improving Reinforcement Learning Algorithms for Dynamic Spectrum Allocation in Cognitive Sensor Networks,” in *IEEE Wireless Communications and Networking Conference (WCNC)*, April 2013, pp. 35–40.

[9] Y. Zhang, Q. Zhang, B. Cao, and P. Chen, “Model free dynamic sensing order selection for imperfect sensing multichannel cognitive radio networks: A q-learning approach,” in *IEEE International Conference on Communication Systems (ICCS)*, Nov 2014, pp. 364–368.

[10] M. Vallejo, J. Recas Piorno, and J. Ayala Rodrigo, “A Link Quality Estimator for Power-Efficient Communication Over On-Body Channels,” in *IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, Aug 2014, pp. 250–257.

[11] FCC, “In the Matter of Unlicensed Operation in the TV Broadcast Bands: Second Report and Order and Memorandum Opinion and Order,” Federal Communications Commision, Tech. Rep., 2008.

[12] M. Bkassiny, Y. Li, and S. Jayaweera, “A Survey on Machine-Learning Techniques in Cognitive Radios,” *IEEE Communications Surveys Tutorials*, no. 99, pp. 1–24, 2012.

[13] M. Kist, L. Faganello, L. Bondan, M. Marotta, L. Granville, J. Rochol, and C. Both, “Adaptive Threshold Architecture for Spectrum Sensing in Public Safety Radio Channels,” in *IEEE Wireless Communications and Networking Conference (WCNC)*, March 2015, pp. 287–292.

[14] C. Ghosh, S. Pagadarai, D. Agrawal, and A. Wyglinski, “A framework for statistical wireless spectrum occupancy modeling,” *IEEE Transactions on Wireless Communications*, vol. 9, pp. 38–44, 2010.

ANNEX G PUBLISHED PAPER – MANSDN/NFV 2014

This short paper presents the results of our first investigation regarding the management of NFV-based deployments, considering the aspects involving the deployment of a scenario based on the ClickOS, a minimalistic virtualized operating system to run Click-based network functions.

- **Title:**
  Management Requirements for ClickOS-based Network Function Virtualization
- **Conference:**
  International Workshop on Management of SDN and NFV Systems (ManSDN/NFV)
- **URL:**
  <http://www.cnsm-conf.org/2014/sdnnfv2014.html>
- **Date:**
  21 November, 2014
- **Held at:**
  Rio de Janeiro, Brazil
- **Digital Object Identifier (DOI):**
  <10.1109/CNSM.2014.7014210>

Management Requirements for ClickOS-based Network Function Virtualization

Lucas Bondan, Carlos Raniery Paula dos Santos, Lisandro Zambenedetti Granville
Institute of Informatics – Federal University of Rio Grande do Sul
Av. Bento Gonçalves, 9500 – Porto Alegre, Brazil
Email: {lbondan, crpsantos, granville}@inf.ufrgs.br

***Abstract*—Network Functions Virtualization (NFV) is a new approach to design, deploy, and manage network functions. In a recent past, such functions used to be implemented at hardware. This approach, besides effective, presents many disadvantages such as increased operational costs, difficulties to scale up or down the network, and deploy new functions. The rise of virtualization technologies, on the other side, provides new ways to rethink about network functions. Instead of specialized and expensive hardware, multiple network functions can share the same commodity hardware, thus contributing to a better utilization of resources. Besides its advantages, NFV is still on its early stages of employment. Important aspects are not yet being investigated by the research community. For example, to this date, the management requirements of NFV remain unclear. Therefore, the present paper addresses this subject, it presents a realistic network function request, which is used to identify management requirements in the context of a specific NFV enabler platform called ClickOS.**

***Index Terms*—Network Functions Virtualization, Network Management, Function Requests**

I. Introduction

Network Function Virtualization (NFV) [1] is a novel network paradigm that separates data plane software from the underlying hardware. Different than Software Defined Networking (SDN) [2], which deals with control plane through more mature technologies like OpenFlow [3], NFV is still in its infancy, struggling to establish itself as viable way to reduce costs of network deployment and maintenance, *i.e.,* CAPEX and OPEX [4].

Industry, academia, and standardization bodies have shown increased interest in NFV. That can be observed, for example, in important consortiums formed by network vendors, the proliferation of papers and conferences about NFV, and on the NFV-related work under development in the European Telecommunications Standards Institute (ETSI) and on the attempt to create a NFV research group (RG) in the Internet Research Task Force (IRTF).

As in any new networking technology, network management aspects are crucial for the success of NFV. However, despite the increased interest, network management has been neglected in current NFV efforts. Because we believe that network management cannot be an afterthought, in this paper we deal with the issue of identifying NFV management requirements in the context of a specific NFV enabler platform, *i.e.,* ClickOS [5].

Because NFV is still in its infancy, as mentioned before, there is no widely deployed NFV platform. In fact, several platforms seem to be under development, but most of them is unavailable or not based on open source software. ClickOS is a fortunate exception in this landscape. Based on open source software, ClickOS is frequently considered an NFV enabler. Although far from being an NFV materialization as mature and concrete as OpenFlow is for SDN, ClickOS allows us to identify the challenges that network administrators wanting to operate a network with NFV will face.

NFV allows several different networking scenarios, including those with complex relationships between network operators and service providers. Although we recognize that complex scenarios would emerge to support relevant business models, we concentrate our investigation in a simpler scenario where the network administrator is interested in operating its network using NFV by deploying virtualized functions hosted inside the managed network itself. Our methodology to identify management requirements is based on using ClickOS over real virtualized servers and observing the challenges that the operator faces to take advantage of NFV's advertised benefits.

Our investigation starts with a network setup request, which describes the network functions and their relationships in a test network scenario. Our network setup request is introduced in Section II. In Section III, we present our ClickOS environment, describing in details the network infrastructure and the set of operations on top of ClickOS used in our investigation to materialize the network setup request. The list of management requirements is then extracted from the activities the network operator carried out over ClickOS. In Section IV, we present and discuss such a list of management requirements. Finally, we present our conclusions and discuss opportunities of future work in Section V.

II. Network Setup Request

NFV uses virtualization technologies to deploy network functions (NFs) (*e.g.,* firewalls, IDSes, load balancers). In comparison to traditional hardware-based networks, NFV has the advantage of decreasing operational costs, since multiple NFs can share the same commodity hardware. Another benefit of NFV is that it provides a more dynamic environment where NFs can be quickly scaled up or down to address changing demands.

In NFV, virtualized network functions (VNFs) act as building blocks that are connected and orchestrated from a Management System. Through this system, a network operator is able to manage the functions' life-cycle (*e.g.,* instantiation, scaling, termination), as well as to define the chain of VNFs that creates more sophisticated network functionalities. This chain of functions is defined through VNF Forwarding Graphs (VNF-FG) [6].

In this work we use ClickOS as the platform for NFV provisioning. Although other alternatives (*e.g.,* using containers or hypervisors) are available to achieve the same goal, ClickOS is the most prominent of them. ClickOS is a minimalist operating system based on the Click Modular Router [7], focused on supporting typical network requirements such as high throughput, low delay, and isolation. In its current version, ClickOS supports a significant variety of network functions, including traffic shaping, network monitoring, and DDoS prevention.

A. Perimeter Network Design

To identify management requirements in the context of ClickOS-based NFV, we consider the scenario where a network operator needs to deploy a perimeter network – also know as Demilitarized Zone (DMZ) – to provide a protected environment for the organizations' services. A DMZ is used to offer some of an organization's services (*e.g.,* Web server, e-mail server, VoIP server) to an untrusted external network such as the Internet. Hosts placed the DMZ have only limited connectivity to services running inside the internal network. This approach allows the internal network to be protected in the case of an intruder compromising any DMZ's host.

In order to deploy a DMZ, the network operator needs to use multiple and disparate NFs. The most basic NF in such an environment is firewalling, used to filter incoming and outgoing traffic based on specific rules. In simple DMZs, a single firewall is used to protect the internal network from the public one. In this paper, we consider the case where a more secure setup is requested, composed of two firewalls as presented in Fig. 1.

Four NFs are highlighted in the gray boxes of the network setup request of Fig. 1: firewall, load balancer, IDS sensor, and NAT. From top to bottom, the first firewall filters traffic to allow only HTTP/S communications. Any packet that does not match this filtering rule is discarded. All traffic allowed by the firewall is then captured by an IDS sensor, which is used to monitor network traffic looking for malicious activities. Also inside the DMZ, a load balancer distributes user requests among Web servers, according to a probability distribution function, thus ensuring that no Web server becomes overloaded.

A second firewall separates the DMZ from the internal network. This firewall is a second barrier in case of an intrusion in the DMZ, thus preventing important services inside the organization (*i.e.,* LDAP, Application Server, and Database) of becoming compromised. Since Web servers at the DMZ and Application Server (responsible for the business logic) at the internal network communicate using multiple protocols (*e.g.,* SOAP, JMS, RMI), the second firewall must allows traffic of these protocols. A good practice of network security is to place a second IDS sensor inside the internal network to detect attacks coming from insiders (*e.g.,* hosts within the internal network). In this way, the sensor is also less prone to attacks directed to the IDS itself. Finally, a traditional NAT is placed inside the internal network for translating public IP addresses to the corresponding local ones.

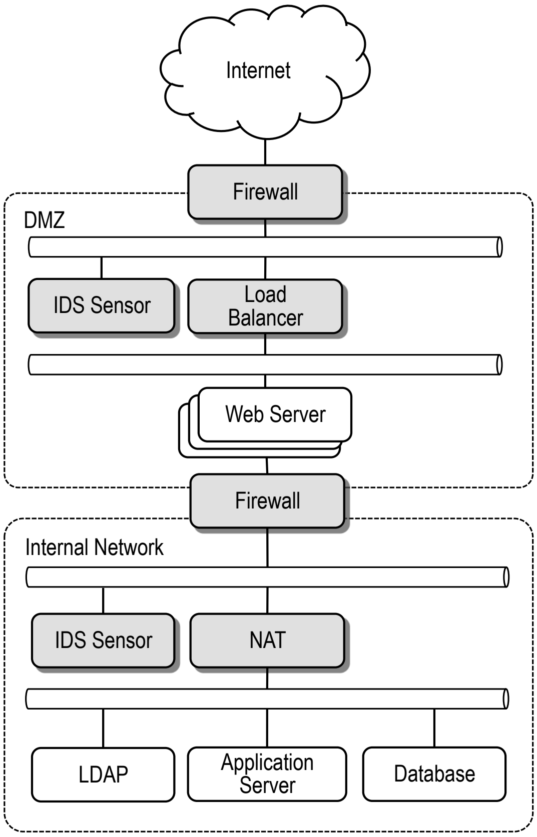


Fig. 1. DMZ Topology Request

In the next section, we describe in details all the steps performed to deploy the network setup request presented in this section, highlighting the main characteristics related to the NFV concepts applied in our experimentation.

III. Network Infrastructure Deployment

In order to deploy the network request presented in the previous section, we employed two VNF servers: one responsible for hosting the DMZ functions, and a second VNF server responsible for hosting the internal network functions. This approach enabled us to isolate both networks, thus supporting an increased security level for the organization's services. This setup is presented in Fig. 2.

Each VNF server consists of a XenServer hypervisor running on top of a Linux-based OS. A generic ClickOS image is compiled for each hosting OS, and is initialized according to a predefined configuration template. In this work, we used

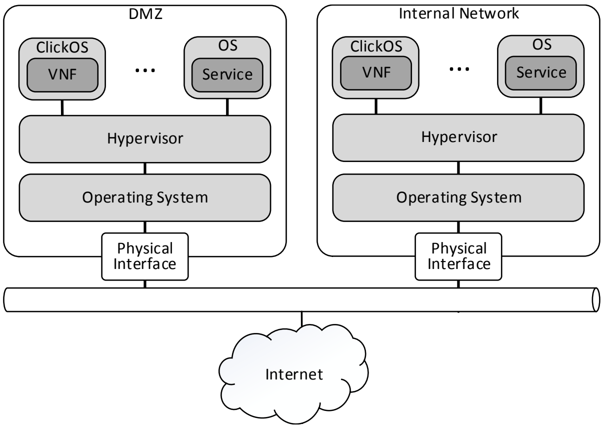


Fig. 2. Topology Request Deployment with ClickOS

a template consisting of one CPU, 12MB of memory, and up to 2 virtual network interfaces depending on the function. For example, firewall functions require two interfaces: one for incoming traffic, and the other for outgoing.

In ClickOS, network functions are defined using the network elements provided by the Click Modular Router. These definitions are represented in description files, which are interpreted and executed by ClickOS. Network operators manages function's lifecycle through the Cosmos tool, which implements the communication interface between the user domain and the ClickOS domain.

Once the VNFs are defined and ready for execution, the network operators needs to specify how to interconnect them (*i.e.,* define the VNF-FG). The connection between two or more NFs can be either physical or virtual (*e.g.,* through bridges). Virtual interfaces (*i.e.,* between the VNF server and ClickOS VMs), can be created using XenServer. The VNF-FG used in this work is presented in Fig. 3, with dark lines representing real connections, and dashed lines representing logical connections.

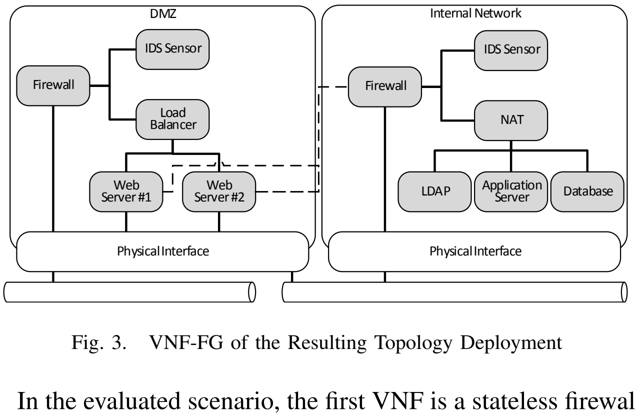


Fig. 3. VNF-FG of the Resulting Topology Deployment

In the evaluated scenario, the first VNF is a stateless firewall located in the DMZ. This firewall is used to filter incoming traffic, allowing only HTTP/S packets. We use a network bridge to connect the physical network interface of the VNF Server #1 to the virtual interface of the ClickOS VM hosting the firewall function. A second network bridge is used to connect the load balancer to the virtualized Web servers. In this work, we deployed two Web servers on the DMZ to process user requests. These servers are connected to the physical interface using a third bridge. We decided to use virtualized Web servers to facilitate the deployment of the network setup request, otherwise it would be necessary to employ additional physical interfaces and network devices. This decision, however, doesn't have any impact on the management requirements presented in the next section.

Inside the internal network, a second firewall is used to allow network traffic related to specific protocols. This firewall is connected with the VNF Server #2 physical interface, and performs the first packet processing in the internal network. A network bridge is used to connect the firewall's output with the virtualized IDS sensor and NAT. Finally, the output interface of the NAT function is also connected using a network bridge with three virtualized services inside the internal network: a LDAP, an Application Server, and a Database.

All the connections (physical or virtual) are part of VNF-FG of the service provided by the network setup scenario. When the service be required, it will be provided using the VNF-FG corresponding to this service, *i.e.,* , the request will be processed following all the network elements that compose the respecting VNF-FG. Thus, the creation of the VNF-FG is a key factor in the deployment of a NFV scenario, allowing a set of NFs be composed to provide a whole service.

In next section, we will discuss all the difficulties faced with the deployment of the proposed network setup on the presented NFV infrastructure. The main objective is to derive the management requirements based on these difficulties, composing a first summary about NFV management requirements.

IV. Management Requirements

The deployment of a DMZ using ClickOS revealed significant difficulties in the adoption of current NFV technologies. Based on such difficulties we identified a list of important management requirements for properly maintaining NFV-based networks. These requirements are discussed in the following. Our objective with this list is to provide starting point for network administrators interested in using NFV.

A. VNF Server Configuration

The current landscape of NFV lacks a common platform for NFs embedding. Network administrators are obligated to use multiple solutions in order to configure and maintain a VNF server. In the case of ClickOS, for example, the VNF server should support specific libraries to build the virtual images as well as an instance of the Citrix XenServer hypervisor to host those images [5]. These solutions, however, are not created with integration in mind, thus leading to an additional effort for the network administrator.

B. VMs Instantiation

While there is a good support for virtualizing traditional OSs (*e.g.,* Linux, Windows, and OSX) using Xen technology, its not the case for ClickOS. Changes on its compilation process to improve network performance prevent existing tool (*e.g.,* XenCenter and XenManager) to be used. The lack of standardized communication methods between these tools and ClickOS images, led the ClickOS maintainers to develop a specific tool, called Cosmos, to provide user interaction. In this context, VMs instantiation should be supported by appropriate management solutions, able to provide feasible and effective methods for the network operators to configure and instantiate ClickOS VMs.

C. Infrastructure Deployment and VNF Location

The provision of NFV-based networks requires an infrastructure of VNF servers properly configured. VNFs connections (*i.e.,* network bridges) are usually manually defined by the network administrator using the XenServer CLI tool. This tool, besides effective, doesn't provide a complete view of the network for the network administrators. Connections between VNF servers are individually defined, using command line instructions for that. Systems focused on the the network administrator needs, like graphical network representation, can be helpful in the infrastructure deployment process, and consequently on its management.

D. Network Functions Design and Deployment

NFs are defined in ClickOS using a configuration language based on the Click Modular Router elements. Each element refers to a basic network operation, such as IP filtering, traffic shaping, and address translation. The way such elements are connected represents the processing flow applied to incoming/outgoing network packets. Although NFs can be manually defined, the usage of a high-level and visual design tool, such as Clicky [8], improves the process of creating new functions. Besides making it easier for network administators to design NFs, Clicky doesn't support VNF deployment. It forces administrators to design a NF in one tool and manually apply them using the Cosmos interface.

E. VNF Monitoring

Network monitoring is one of the main tasks performed by network administrators. In the NFV context, ClickOS VMs can be monitored using the Xen console, which displays information regarding to network traffic and Click operation. Although this approach may be enough for small networks, as the number of VNFs increases, new methods becomes necessary. Moreover, considering the scenario where a NFV

F. VNF Reconfiguration

In order to reach higher performance levels, network administrators are constantly reconfiguring network functions. Such reconfigurations may be performed in ClickOS VM instances by using the Cosmos tool. However, Cosmos only provides access to individual VMs, while some scenarios might benefit from a batch reconfiguration of the network. This approach

orchestrator is responsible for migrating VNFs, it becomes easy for network administrators to lose control of VNFs location [9].

can, for example, handle dynamic provisioning changes and improve the overall management task by saving time from the network administrators.

Once summarized and discussed the list of management requirements derived from our experimentation, in the next section we present some conclusions about this work. Moreover, we present our perspectives for future work based on the results obtained with the experiments presented in this paper.

V. Conclusion and Future Work

In this paper we presented an initial effort to identify NFV management requirements. Our investigation is based on the deployment of a NFV request in a ClickOS-based infrastructure. Besides using a specific technology (*i.e.,* ClickOS), we believe that the identified requirements are still valid for other NFV platforms and more sophisticated network scenarios. As future research, we will design an integrated management system based on the identified requirements. The objective of this system is to promote the adoption of NFV by network operators. For example, we plan to provide the level of functionalities supported by XenCenter, assist the design of VNF-FG, and the development of new network functions through visual interfaces.

Acknowledgements

The authors thanks to João Martins and Filipe Manco from NEC Europe for the support with creation and configuration of ClickOS images.

References

[1] White Paper, "Network Functions Virtualisation (NFV)," *ETSI NFV ISG White Paper*, no. 1, pp. 1–16, 2012.

[2] N. Feamster, J. Rexford, and E. Zegura, "The Road to SDN: An Intellectual History of Programmable Networks," *ACM Sigcomm Computer Communication*, vol. 44, no. 2, pp. 87–98, 2014.

[3] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling Innovation in Campus Networks," *SIGCOMM Computer Communications Review*, vol. 38, no. 2, pp. 69–74, Mar. 2008.

[4] Y. Jarraya, T. Madi, and M. Debbabi, "A Survey and a Layered Taxonomy of Software-Defined Networking," *IEEE Communications Surveys Tutorials*, vol. PP, no. 99, pp. 1–1, 2014.

[5] J. Martins, M. Ahmed, C. Raiciu, V. Olteanu, and M. Honda, "ClickOS and the Art of Network Function Virtualization," *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2014.

[6] White Paper, "Network Functions Virtualisation (NFV) - Use Cases," *ETSI NFV ISG*, vol. 1, pp. 1–50, 2013.

[7] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, "The Click Modular Router," *ACM Trans. Comput. Syst.*, vol. 18, no. 3, pp. 263–297, Aug. 2000.

[8] Clicky. (http://www.read.cs.ucla.edu/click/clicky) Accessed on August, 3.

[9] S. Wright, D. Clarke, P. Runcy, K. Siva, and M. Mularczyk, "Virtual Function State Migration and Interoperability," *NFV ISG Proof of Concept Proposal*, pp. 1–5, 2013.

ANNEX H OTHER COLLABORATIONS

- GARCIA, V. F.; VENÂNCIO, G.; DUARTE, E. P.; MARCUZZO, L.; TAVARES, T.; SANTOS, C.; FRANCO, M.; BONDAN, L.;GRANVILLE, L. Z.;SCHAEFFER-FILHO, A. E.; DE TURCK, F.. On the Design and Development of Emulation Platforms for NFV-based Infrastructures. International Journal of Grid and Utility Computing, 2019 (to appear).
- GARCIA, V. F.; MARCUZZO, L.; VENÂNCIO, G.; BONDAN, L.; NOBRE, J.; SCHAEFFER-FILHO, A. E.; SANTOS, C.; GRANVILLE, L. Z.; DUARTE, E. P.. An NSH-Enabled Architecture for Virtualized Network Function Platforms. International Conference on Advanced Information Networking and Applications (AINA), 27-29 March 2018, Matsue - Japan.
- MAROTTA, M. A., FAGANELLO, L.R.; KIST, M.; BONDAN, L.; WICKBOLDT, J. A.; GRANVILLE, L. Z.; ROCHOL, J.; BOTH, C.. Integrating Dynamic Spectrum Access and Device-to-Device via Cloud Radio Access Networks and Cognitive Radio. International Journal of Communication Systems (IJCS), vol. 31, no. 11, pp. 1–11, Jul 2018.
- VENÂNCIO, G.; GARCIA, V.; MARCUZZO, L.; TAVARES, T.; FRANCO, M.; BONDAN, L.; SCHAEFFER-FILHO, A. E.; SANTOS, C.; GRANVILLE, L. Z.; DUARTE, E. P.. Simplificando o Gerenciamento do Ciclo de Vida de Funções Virtualizadas de Rede. Workshop de Gerência e Operação de Redes e Serviços (WGRS), 7 May 2018, Campos do Jordão - Brazil
- TAVARES, T.; MARCUZZO, L.; GARCIA, V.; SOUZA, G.; FRANCO, M.; BONDAN, L.; DE TURCK, F.; GRANVILLE, L. Z.; DUARTE, E. P.; SANTOS, C.; SCHAEFFER-FILHO, A. E.. NIEP: NFV Infrastructure Emulation Platform. IEEE International Conference on Advanced Information Networking and Applications (AINA), 16-18 May 2018, Krakow - Poland.
- DALLA-COSTA, A. G.; BONDAN, L.; WICKBOLDT, J. A.; BOTH, C. B.; GRANVILLE, L. Z.. Maestro: An NFV Orchestrator for Wireless Environments Aware of VNF Internal Compositions. In IEEE International Conference on Advanced Information Networking and Applications (AINA), 27-29 March 2017, Taipei - Taiwan.
- KIST, M.; FAGANELLO, L.R.; BONDAN, L.; MAROTTA, M. A.; BOTH, C.; GRANVILLE, L. Z.; ROCHOL, J.. Adaptive Threshold Architecture for Spectrum Sensing in Public Safety Radio Channels. In IEEE Wireless Communications and

Networking Conference (WCNC), 9-12 March 2015, New Orleans - USA