Attacks against Network Functions Virtualization and Software-Defined Networking: State-of-the-art

François Reynaud, François-Xavier Aguessy, Olivier Bettan, Mathieu Bouet, Vania Conan

Attacks against Network Functions Virtualization and Software-Defined Networking: State-of-the-art

François Reynaud*, François-Xavier Aguessy*, Olivier Bettan*, Mathieu Bouet* and Vania Conan*
*Thales Services
Campus Polytechnique, 1 avenue Augustin Fresnel, 91767 Palaiseau cedex, France

***Abstract*—Network Functions Virtualization (NFV) and Software-Defined Networking (SDN) are two emerging paradigms for networks. While being independent from each other, they may be deployed together, which is likely to happen more frequently in the future, as they bring many opportunities for simpler, more flexible and energy-efficient networks. However, they also come with weaknesses that evil-minded users could exploit to disrupt such architectures. In this paper, we survey attacks that have been or could be performed against NFV and SDN, and propose practical countermeasures when applicable.**

I. Introduction

Network operators use many different devices running proprietary software to provide specific network functions. As a consequence, when they wish to provide new network services, they have to buy and configure new devices. This raises several issues, such as the increasing equipment costs, the power consumption increasing with each new device, a greater complexity that leads to higher operational expenses and misconfiguration proneness, and low dynamism and scalability.

To address these issues, new network paradigms implementing the virtualization and softwarization of the network are emerging. For example, Network Functions Virtualization is a network architecture concept, standardized by the European Telecommunications Standards Institute (ETSI) [1], that consists in using standard hardware for hosting various, vendor-independent, network software components. Instead of having a myriad of devices providing the Network Functions (NFs) as vendor-specific hardware and software, the NFs are virtualized and consolidated onto standard hardware. The most obvious advantage to this is the reduced equipment costs for network operators since they do not have to buy one device per NF.

Software-Defined Networking is another paradigm, whose progress is led by the ONF [2], that consists in using a centralized programmable controller (control plane) to manage an entire infrastructure (data plane) made of simple forwarding devices. As a consequence, network operators need not to configure every device separately; they only have to program the controller to reconfigure the network. This allows more flexible and less error-prone network configuration.

Although NFV and SDN can be deployed independently, they are complementary to each other, so it is obvious that combining both would augment individual benefits. Moreover, it is very likely that NFV will rely more and more on SDN, especially because of the inherent dynamicity of such infrastructure. In fact, these two paradigms are already starting to be deployed in real systems, either separately or together (*e.g.* Andromeda [3], or UNICA [4]).

However, for NFV and SDN to achieve widespread adoption by the industry, their security needs to be assessed, to be assured that they do not bring new security flaws that cannot be dealt with effectively. In this paper, to measure the risks faced by NFV and SDN, we adopt a practical point of view and survey the attacks (*i.e.* any kind of malicious activity that attempts to collect, disrupt, deny, degrade, or destroy information system resources or the information itself [5]) that could be performed against these two paradigms. For both paradigms, we identify the components that are likely to be targeted by attackers, then attacks against these components.

This paper is organized as follows: Section II surveys the threats and attacks against NFV. Section III surveys the threats and attacks against SDN. Section IV analyzes the root causes of the threats against NFV and SDN, and gives possible countermeasures. Finally, Section V concludes this work.

II. Network Functions Virtualization

A. Overview

The concept of NFV is very recent: it was born in October 2012 from the collaboration of some of the world's leading Telecommunication Service Providers [6]. The ETSI was selected to be the home of the Industry Specification Group for NFV (ETSI ISG NFV). Since then, a fair amount of standardization activities and collaborative NFV projects have been conducted, such as Open Platform for NFV (OPNFV) [7], or OpenMANO [8].

NFV is expected to bring many benefits to network operators [9], like reduced equipment costs, reduced energy consumption, shorter time to market, and increased service agility and possibility to optimize the network configuration and/or topology on the fly.

The ETSI has defined an architectural framework for NFV in [1], whose a simplified version is presented in Figure 1.

In short, it is composed of two main functional blocks: the Network Functions Virtualization Infrastructure (NFVI) on the left side of the picture, and the NFV Management and Orchestration (NFV MANO) on the right side of the picture. The NFVI is the set of hardware and software components which build up the environment in which the Virtualized Network Functions (VNFs) are deployed [1], while the NFV MANO [10] is in charge of managing the lifecycle and

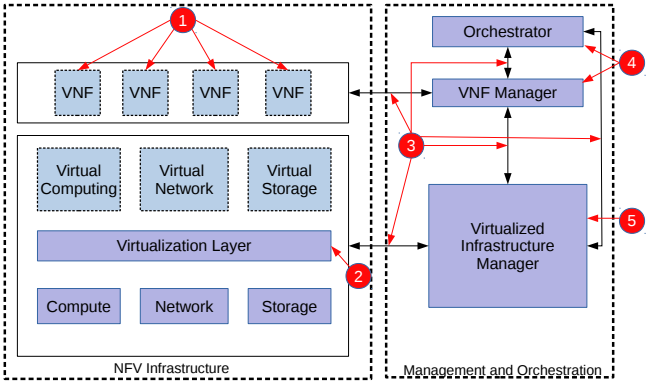

Fig. 1. NFV architecture and its critical assets

chaining of the VNFs in order to provide the needed Network Services (composition of network functions and defined by its functional and behavioural specification [11]).

B. Targetable components

We identify the critical assets in the NFV architecture. They are represented on Figure 1.

1) **Virtualized Network Functions:** VNFs could be either the source or the target of an attack. Indeed, a VNF is a software component provided by a vendor independent of the infrastructure provider. It can thus have software vulnerabilities or even be a malware, designed to perform attacks.

2) **Virtualization layer:** Attackers could take advantage of vulnerabilities present in hypervisors, for example, to escape from the virtual computing, network or storage to the host's physical compute, network or storage resources. This could allow an attacker to undermine the confidentiality, integrity and/or availability of VNFs resources.

3) **Communications with and within NFV MANO:** Attackers may try to eavesdrop or modify the traffic that transits between the NFVI and the NFV MANO, as well as traffic within the NFV MANO.

4) **Orchestrator and/or VNF manager:** Attackers may attempt to exploit these two components to disrupt the life-cycle management of the Network Services (purpose of the Orchestrator) or of individual VNFs (main role of the VNF Manager) [10].

5) **Virtualized Infrastructure Manager (VIM):** The VIM is responsible for the management of the NFVI resources used by the VNFs (compute, network and storage), and attacking it could for example allow Denial of Service (DoS) or data theft, bypassing hypervisor isolation.

C. Attacks

Although some work was done to improve the security of NFV environments (*e.g.* CloudBand [12] or integration of policy enforcement [13]), most of the efforts to develop NFV focused on management and orchestration (*e.g.* OpenMANO [8] or T-NOVA [14]).

The ETSI identifies in [15] the threat surface of NFV as being the union of the threats to generic virtualization and generic networking. NFV being an implementation of Cloud computing for networking, we surveyed attacks that have been performed against Cloud computing systems and hypervisors and analyzed the impact of such attacks on NFV. Potential areas of concerns for NFV are also identified in [15], and some of them are related to the attacks we surveyed.

These attacks can be categorized depending on the components previously listed:

1) **Virtualized Network Functions:** Denial of Service (DoS) attacks are a serious threat to Cloud and NFV environments. There are several examples of DoS attacks on services hosted in the Cloud, like Bitbucket, a web-based hosting service company hosted by Amazon that was victim of massive DDoS (Distributed DoS) attacks, making it unavailable to many developers [16]. The danger of DDoS is even greater in the context of Cloud Computing or NFV because it could also affect untargeted services and tenants that are hosted on the same physical host.

VNFs are software components providing network functions, so they are likely to be vulnerable to software flaws: it could be possible to bypass firewall restrictions or to take advantage of a buffer overflow to execute arbitrary code. CVE-2012-2663 for iptables and CVE-2006-5276 for Snort are examples of such vulnerabilities and, albeit old, they give a first insight on the kind of dangers that may threaten typical NFs that are going to be deployed in NFV infrastructures.

2) **Virtualization layer:** Several types of attacks can be performed on the virtualization layer:

• *Code execution on the physical host*: Wojtczuk [17] presents several attacks against common hypervisors (QEMU-KVM, Virtualbox, Xen) that allow code execution on the host from a compromised or malicious Virtual Machine (VM).

The first one allows the attacker to obtain code execution in Xen's privileged paravirtualization domain by making Xen run a VM with a filesystem corrupted in such a way that it can trigger CVE-2007-5497.

The second one allows code execution on the host by using a use-after-free vulnerability in QEMU-KVM, triggered by requesting a PCI unplug action on the virtual RTC, that was not hotplugged in.

Finally, the third attack uses a buffer overflow related to the emulation of the e1000 router to gain execution of arbitrary code.

Return-oriented-programming-based attacks: Riddle and Chang [18] introduce an attack on the Xen hypervisor that allows the attacker to escalate their VMs to a privileged state by using return-oriented-programming.

In the context of NFV, these attacks could be used to read or modify the memory of, take control of, or deny resources to VNFs co-resident with a malicious VNF (possibly disrupting several Network Services), or even deploy more malicious VNFs.

• *Resource monopolization*: Riddle and Chang [18] present two attacks to steal resources from other VMs:

Monopolization of CPU: If the VMs are running over a Xen hypervisor then it is possible either to use up to 98% of the

physical host's CPU, hence denying the CPU to other VMs, or to determine whether 2 VMs are co-resident (which can be the starting point of another attack), by taking advantage of Xen's credit scheduler.

I/O performance-based attacks: If the attacker knows the scheduling characteristics of the hypervisor, they can use that information to overload I/O resources, resulting in slowing down co-resident VMs (or VNFs).

• *Data theft*: Riddle and Chang [18] explain that if the target VM is co-resident with the attackers' malicious VM and is infected with malware, then the attacker can use memory bus or cache contention to stealthily steal data, *e.g.* cryptographic keys, from the target VM.

• *VM monitoring evasion*: Riddle and Chang [18] present the VM rollback attack: if the hypervisor is already compromised, then the attacker may execute a VM from an older snapshot without the VM owner knowing it, allowing them to bypass security systems. For example, while the attacker is brute forcing a password, when the VM raises a security alert, the compromised hypervisor rolls back to the previous snapshot, and the attacker can continue their attack.

Wang *et al.* [19] present the hypervisor introspection technique: when passively monitoring VMs, the hypervisor needs to suspend the VM to get a consistent view of the hardware state. By determining the frequency at which the hypervisor pauses the VM for inspection, attackers can perform operations between the monitoring checks. This allow them, for example, to stealthily exfiltrate data (*e.g.* network traffic, in the case of NFV) or to maintain a back-door shell on the VM.

4) **Orchestrator and/or VNF manager:** *Using ephemeral storage to steal data (CVE-2013-7130)*: The create_images_and_backing method in libvirt driver in OpenStack Compute (Nova), when using KVM live block migration, does not properly create all expected files, which allows attackers to obtain snapshot root disk contents of other users via ephemeral storage. In an NFV over OpenStack environment this could be used to steal cryptographic keys from other VNFs, thus enabling, for example, eavesdropping, data modification, or impersonation.

5) **Virtualized Infrastructure Manager:** *Privilege escalation (CVE-2014-3790)*: Ruby vSphere Console (RVC) in VMware vCenter Server appliance (centralized management and operation, resource provisioning and performance evaluation of VMs in a distributed virtual data center) allows remote authenticated users to execute arbitrary commands as root by escaping from a chroot jail, thus gaining great control over the infrastructure domain managed by the VIM.

Regarding the areas of concerns about NFV identified by the ETSI in [15], among the attacks we just presented, (D)DoS attacks can be related to *availability of management support infrastructure*, *secure crash*, and *performance isolation*. Resource monopolization attacks and the vulnerability CVE-2014-3790 can be related to *performance isolation*, and data theft can be related to *private keys within cloned images*.

III. Software-Defined Networking

A. Overview

Although SDN has been getting more attention over the past few years, it originates from ideas that appeared and evolved since more than 20 years ago [20], such as active networking in the early 90s, and separating control and data plane in the early 2000s. These ideas did not meet adoption from the industry because they lacked the pragmatism that would allow real-world deployment; the ONF introduced OpenFlow, immediately deployable, as a compromise between fully programmable networks and real-world deployment, allowing the SDN movement to be both pragmatic and bold [20].

According to the ONF, OpenFlow-based SDN is expected to have a certain number of benefits [2], including:

• Centralized control of multi-vendor environments: no need to manage groups of devices from individual vendors anymore.

• Automation: SDN makes it possible to automate many management tasks that are done manually today.

• Higher rate of innovation: possibility to program or reprogram the network in real time.

• Increased network reliability and security: Reduced risk of network failures due to configuration or policy inconsistencies.

• More granular network control: OpenFlow's flow-based control allows to apply policies at a very granular level.

As shown in Figure 2, the architecture of SDN consists of 3 layers:

The application layer: the end-user applications that consume the SDN communication services.

The control layer: the consolidated control functionality that supervises the network forwarding behavior through an open interface.

The infrastructure layer: the network elements and devices that provide packet switching and forwarding.

These layers interact with each other through 2 interfaces: the Northbound Interface (NBI) between the application layer and the control layer, and the Control to Data-Plane Interface (CDPI) between the control layer and the infrastructure layer [21]. The NBI allows SDN applications to express their network behavior and requirements, and the CDPI provides programmatic control of all forwarding operations, capabilities advertisement, statistics reporting and event notification [21].

Although SDN comes with promising benefits, it also raises new security concerns, and Kreutz *et al.* [22] have identified seven threat vectors in SDNs. In this paper however, we chose another point of view and decided to focus on the SDN-specific components that are the most likely to be attacked, illustrated in Figure 3:

1) **SDN switches:** The traffic flows transit through the switches, making them interesting targets for an attacker.

2) **Communications between control plane and data plane:** Switches are totally dependent on the controllers, so communications between control plane and data plane also are interesting targets, maybe even more than the switches themselves.

3) **Controllers:** Controllers bear all the intelligence of the network, making them the most valuable target to attack.

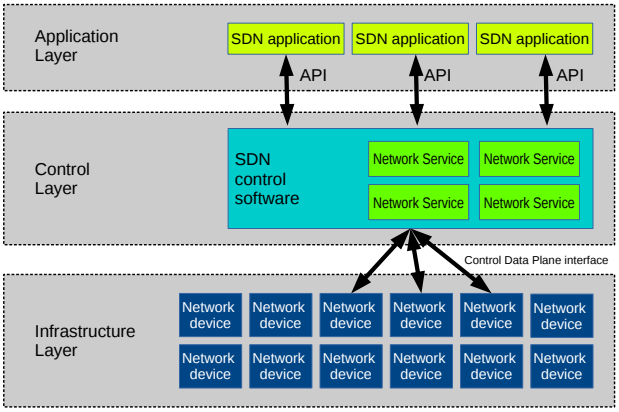

Fig. 2. Software-Defined Networking Architecture

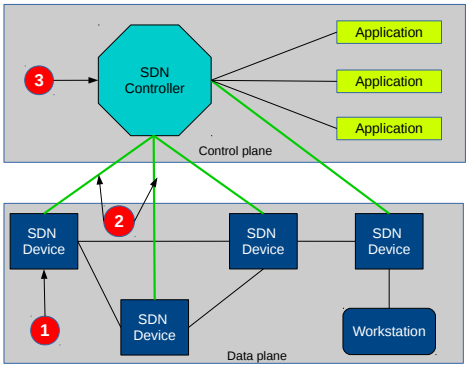

Fig. 3. Targetable SDN components

B. Attacks

Several attacks have been performed against Software-Defined Networks, each targeting one or more components.

1) **Switches:** Shin and Gu [23] show that a Denial of Service (DoS) attack against a remote SDN network could significantly decrease its performance without requiring a large bandwidth or high performance devices.

Romão *et al.* [24] perform a DoS on various switches by inserting permanent flows into them, either inserting the flows directly by the controller or by sending a huge amount of ICMP requests to different IP addresses. The authors also try the debug mode, enabled by default on some switch implementations, *e.g.* OpenWrt/Pantou, that basically provides total control on the switch.

2) **Communications between control plane and data plane:** Romão *et al.* [24] perform 3 different Man-in-the-Middle (MitM) attacks, with different objectives:

- Interrupt traffic between the switch and the controller. This was done by ARP poisoning both the switch and the controller, and caused undesired communication between networks that are supposed to be logically separated.
- Eavesdrop traffic between two hosts. This was done by mirroring the traffic between the hosts toward a third host.
- Stealthily modify the traffic between two hosts. This was done by interrupting traffic between the two target host, so they send ARP requests to which the attacker responds with his own MAC address, instead of simply sending ARP replies.

3) **Controllers:** Hong *et al.* [25] present two topology poisoning attacks unique to SDN that affect major SDN controllers such as Floodlight [26] and OpenDaylight [27].

Host Location Hijacking: This attack exploits the Host Tracking Service, the controller's service that maintains a profile for each host in the network, and updates it as the host migrates to impersonate a specific web server and phish users. To do so, the attacker retrieves the target's identifier used by the controller to identify the host (in the present case it is the MAC address), and can then inject fake packets in the name of the target host. As a result, users trying to access the genuine Web server are directed to the malicious server.

Link Fabrication: This attack consists in fabricating a link in the network either by injecting fake LLDP packets, or in a relay fashion, *i.e.* without modifying the packets. This attack can be a first step for other attacks such as DoS attack, by taking advantage of the Spanning Tree algorithm used by OpenFlow controllers to incapacitate normal switch ports, or MitM attack by using the fact that once it detects that a new link is up, the controller recomputes the shortest route.

Shin *et al.* [28] focus on malicious applications directly attacking the controller. They tested 4 SDN controllers: Floodlight, OpenDaylight, POX and Beacon [29]. Their attacks aim at crashing the controller, or confuse other control layer applications, which they achieved with minimal effort. In the case of Floodlight, for instance, they crashed the controller simply by calling *sys.exit()* function or continuously allocating memory, leading to an out of memory crash. They also tricked a monitoring application into "thinking" there was only one link in the network by deleting a network link in an internal Floodlight data structure (Link Deletion).

There also are several CVE Identifiers related to controllers:

The REST layer on HP SDN VAN Controller devices allows remote attackers to cause a Denial of Service via network traffic to the REST port *(CVE-2015-2122)*.

The Netconf service in OpenDaylight 1.0 allows remote attackers to read arbitrary files via an XML eXternal Entity (XXE) declaration in conjunction with an entity reference in an XML-RPC message, related to an XXE issue *(CVE-2014-5035)*. This could allow attackers to gain information about the configuration of OpenDaylight or the Operating System on which it is running (*e.g.* installed services), as a first step for another attack, for example.

IV. Root cause analysis of attacks on NFV and SDN and possible countermeasures

A. Network Functions Virtualization

As said in section II-C, NFV is an implementation of Cloud Computing for networking, and is exposed to similar threats:

Distributed Denial of Service: The problems enabling this kind of attack are the fact that resources are not unlimited, and the fact it is hard to distinguish normal traffic from attacking traffic. While there is not much that can be done about the resources, solutions have been proposed to defend

against DDoS, *e.g.* Joshi *et al.* [30] proposed Cloud Trace Back, a solution using a back-propagation neural network trained to detect attack traffic. Once attack traffic can be distinguished from normal traffic, it becomes possible to defend against (D)DoS attacks, by using techniques such as selective blackholing [31].

Code execution on the host, privilege escalation, isolation breaking: There are many vulnerabilities in hypervisors whose exploitation allows these. Between 2013 and 2015 (included), more than 50 CVEs concerning VMWare's VSphere, Qemu, KVM, XEN, Hyper-V, LXC and Docker having a CVSS (Common Vulnerability Scoring System) score greater than 7 (out of 10) were published. The fact that so many CVEs with such high scores were published in 3 years is a sign that a lot of work has yet to be done regarding the security of hypervisors, essential for the security of NFV.

Some of these attacks exploit buffer overflows, which can be countered with techniques such as ASLR or canaries.

Side-channel attacks: In side-channel attacks, attackers infer information in an indirect manner, *e.g.* measuring the frequency at which a VM is paused. To defend against this type of attack, the basic idea is either to eliminate or reduce the information released by the side channel (*e.g.* reduce electromagnetic emissions in case of TEMPEST [32] attacks), or to introduce some kind of noise to the channel.

B. Software-Defined Networking

As we have seen, Software-Defined Networks are exposed to 3 main types of attacks: Denial of Service, Man-in-the-Middle, and Network Visibility Poisoning (NVP). For each of these types of attacks, the vulnerabilities exploited may lie in one or more of the targetable components; *e.g.* a (D)DoS attack may be at the forwarding layer level, saturating the storage resources of one or more switches, or it could be at the control layer level, saturating the compute and storage resources of the controller. As there are several points of attack, it seems pretty reasonable to say that all these points need to be considered when securing SDN-enabled networks.

Denial of Service: At the forwarding plane level, each SDN switch stores one ore more flow tables. The problem is that the flow table storage capacity of switches is finite, which makes it possible for switches to be saturated. As switches capacity cannot be increased at will, other mitigation solutions have to be found. At the control plane level, beside the controller's limited resources, the lack of application resource checking and isolation from the controller is another reason why controllers are vulnerable to DoS attacks [28]. The main difficulty when trying to defend against DoS attacks is that it is not easy to make the difference between normal traffic and attacking traffic. Shu *et al.* [33] present several countermeasures that can be used to mitigate the effects of DoS attacks, among others. For example: FlowVisor [34] allows SDN network operators to partition their network in *slices*; it rewrites rules it receives from the controller so that these rules only affect the slice the network is allowed to control. This can be used to prevent a DoS attack to affect the whole network.

Virtual source Address Validation Edge [35] is a countermeasure against DoS attacks caused by IP spoofing.

FloodGuard [36] is a protocol-independent framework that uses proactive flow rule analyzer to detect traffic flows caused by DoS attacks, and packet migration to prevent the controller from consuming too much computing resources.

Man-in-the-Middle: Just like in traditional networks, MitM attacks are possible when there is no proper authentication mechanism. As most vendors did not provide support for TLS in their switches, the configuration of TLS was declared optional in OpenFlow specification in versions after v1.0. But even without SSL/TLS, solutions exists. For example, FortNOX [37] enforces rules inserted by a security application, *i.e.* if a security application inserts a rule and another application tries to insert a new rule, FortNOX will prevent other applications to insert rules that conflict with the rules defined by the security applications. We however think that SSL/TLS should be implemented to help counter MitM attacks.

Network Visibility Poisoning: The feasibility of NVP attacks may come from several security issues such as the lack of authentication like in Host Location Hijacking [25] and Link Fabrication [25] attacks, or the fact that the controller is not sufficiently protected from the Northbound applications like in the Link Deletion attack [28]. To address these issues, Rosemary [28] and TopoGuard [25] have been proposed. Rosemary is a SDN controller that rectifies, among other issues, the lack of access control and authentication for the applications responsible for the Link Deletion attack by employing a sandbox approach (*App Zone*). TopoGuard uses Topology Update Checker to verify the legitimacy of a host migration, the integrity/origin of an LLDP packet and switch port property once detecting a topology update.

C. Combining Network Functions Virtualization and Software-Defined Networking

There is not a unique way to use NFV and SDN conjointly. The ETSI has recently published a report [38] on the usage of SDN in the NFV architectural framework that presents several possibilities for integrating SDN in the NFV architectural framework. There are many combinations, depending on where the SDN controllers, switches and applications are positioned in the NFV architecture. These combinations can be grouped in two main categories:

SDN used in the tenant domain: the SDN controller is a VNF, instructing other VNFs for taking actions on the traffic. In that case the SDN controller, and thus, the VNFs it controls, may be impacted by vulnerabilities of NFV, possibly disrupting an entire Network Service.

SDN used in the infrastructure domain: the SDN controller is part of the NFVI. It supports the infrastructure network, for example by setting up the required connectivity between NFVI Points of Presence. Consequently, the vulnerabilities that affect the SDN controller may impact the entire NFVI as well.

The main use case for NFV and SDN to be used conjointly that we identified is 5G (*i.e.* the convergence of both fixed and mobile accesses relying on programmable networks), where

multiple tenants share services instantiated as VNFs, provided by various vendors and coordinated using SDN. Although no new vulnerabilities arise from the fact of combining NFV and SDN itself, a multi-tenancy context like that of 5G changes the confidentiality, integrity and availability requirements, and the exploitability and impact of already existing vulnerabilities. So, combining NFV and SDN may have a non negligible impact on the security of the 5G infrastructure.

V. Conclusion

In this paper, we surveyed attacks that may impact Network Functions Virtualization and Software-Defined Networking in order to have a concrete vision of the dangers that threaten them. We are not aware of any attack that specifically targeted an NFV infrastructure, so we looked for CVE Identifiers and attacks related to Cloud computing and hypervisors and analyzed the impact of such attacks on NFV. Indeed, the security of both NFV and Cloud computing depend strongly on that of hypervisors.

What we found out is that, albeit promising, NFV and SDN have a non negligible number of weaknesses. We believe that, being the core elements of these two paradigms, the security of SDN controllers, NFV MANO and hypervisors require particular attention. This will be especially important for infrastructures like 5G's, that will leverage NFV and SDN. The security of 5G is a wide topic that still needs to be further studied, which is the objective of our future works.

Acknowledgment

This work is partially funded by the French National Research Agency (ANR), DOCTOR project, <ANR-14- CE28-000>, started in 01/12/2014 and supported by the French Systematic cluster.

References

[1] ETSI-ISG-NFV, "Network Functions Virtualization; Architectural Framework," *https://www.etsi.org/deliver/etsi_gs/NFV/001_099/002/01.01.01_60/gs_NFV002v010101p.pdf*, Dec. 2014.

[2] ONF, "Software-Defined Networking: The New Norm for Networks - ONF White Paper," Apr. 2012.

[3] "Google Cloud Platform's latest networking stack," *http://googlecloudplatform.blogspot.fr/2014/04/enter-andromeda-zone-google-cloud-platforms-latest-networking-stack.html*, 2014.

[4] "Telefónica Selects HP OpenNFV Platform to Build its UNICA Infrastructure," *http://www8.hp.com/us/en/hp-news/press-release.html?id=1923363#.VrxXjmN2024*, 2015.

[5] C. on National Security Systems (CNSS), "National Information Assurance Glossary," *http://www.ncsc.gov/nittf/docs/CNSSI-4009_National_Information_Assurance.pdf*.

[6] R. Mijumbi et. al, "Network Function Virtualization: State-of-the-art and Research Challenges," *IEEE COMST*, 2015.

[7] "Open Platform for NFV (OPNFV)," *https://www.opnfv.org/about*, 2015.

[8] D. R. Lopezj, "OpenMANO - the Dataplane-Ready Open Source NFV MANO Stack," *IETF Meeting Proceedings*, Mar. 2015.

[9] ETSI-ISG-NFV, "Network Functions Virtualisation: An Introduction, Benefits, Enablers, Challenges and Call for Action," *http://portal.etsi.org/NFV/NFV_White_Paper.pdf*, Oct. 2014.

[10] ——, "Network Function Virtualization; Management and Orchestration," *https://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_NFV-MAN001v010101p.pdf*, Dec. 2014.

[11] ——, "Network Function Virtualization; Terminology for Main Concepts in NFV," *https://www.etsi.org/deliver/etsi_gs/NFV/001_099/003/01.02.01_60/gs_NFV003v010201p.pdf*, 2014.

[12] "Providing Security in NFV: Challenges and Opportunities - Strategic White Paper," *http://www.tmcnet.com/redir/?u=1011422*, 2014.

[13] C. Basile, C. Pitscheider, F. Valenza, and M. Vallini, "A novel approach for integrating security policy enforcement with dynamic network virtualization," *NetSoft 2015*, 2015.

[14] "T-NOVA: Network Functions-as-a-Service (NFaaS) over Virtualized Infrastructures," *http://www.t-nova.eu*, 2015.

[15] ETSI-ISG-NFV, "Network Functions Virtualisation (NFV); NFV Security; Problem Statement," *http://www.etsi.org/deliver/etsi_gs/NFV-SEC/001_099/001/01.01.01_60/gs_NFV-SEC001v010101p.pdf*, 2014.

[16] T.-S. Chou, "Security Threats on Cloud Computing Vulnerabilities," *IJCSIT*, Jun. 2013.

[17] R. Wojtczuk, "Poacher turned gamekeeper: Lessons learned from eight years of breaking hypervisors," in *Black Hat USA 2014*, Jul. 2014.

[18] A. R. Riddle and S. M. Chung, "A Survey on the Security of Hypervisors in Cloud Computing," *2015 IEEE 35th International Conference on Distributed Computing Systems Workshops*, 2015.

[19] G. Wang, Z. J. Estrada, C. Pham, Z. Kalbarczyk, and R. K. Iyer, "Hypervisor Introspection: A Technique for Evading Passive Virtual Machine Monitoring," *WOOT'15*, 2015.

[20] N. Feamster, J. Rexford, and E. Zegura, "The Road to SDN: An Intellectual History of Programmable Networks," *Queue*, Dec. 2013.

[21] ONF, "SDN Architecture Overview," *https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/SDN-architecture-overview-1.0.pdf*, 2013.

[22] D. Kreutz, F. M. V. Ramos, and P. Verissimo, "Towards Secure and Dependable Software-Defined Networks," *HotSDN' 13*, Aug. 2013.

[23] S. Shin and G. Gu, "Attacking software-defined networks: A first feasibility study," Aug. 2013.

[24] D. Romão, N. van Dijkhuizen, S. Konstantaras, and G. Thessalonikefs, "Practical Security Analysis of OpenFlow," 2013.

[25] S. Hong, L. Xu, H. Wang, and G. Gu, "Poisoning Network Visibility in Software-Defined Networks: New Attacks and Countermeasures," *NDSS'15*, Feb. 2015.

[26] "Floodlight OpenFlow Controller," *http://www.projectfloodlight.org/floodlight/*.

[27] "The OpenDaylight Platform," *https://www.opendaylight.org/*.

[28] S. Shin *et al.* , "Rosemary: A Robust, Secure, and High-performance Network Operating System," *CCS'14*, Nov. 2014.

[29] D. Erickson, "The Beacon OpenFlow Controller," in *HotSDN'13*, 2013.

[30] B. Joshi, A. Santhana Vijayan, and B. Kumar Joshi, "Securing Cloud Computing Environment Against DDoS Attacks," *ICCCI-2012*, 2012.

[31] T. Jayawardena and L. Morales, "Distributed denial-of-service attack mitigation by selective black-holing in ip networks," Oct. 2008.

[32] NSA, "TEMPEST Certification Program," *https://www.nsa.gov/applications/ia/tempest*.

[33] Z. Shu *et al.* , "Security in Software-Defined Networking: Threats and Countermeasures," *Mobile Networks and Applications*, Jan. 2016.

[34] R. Sherwood, G. Gibb, K.-K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar, "FlowVisor: A Network Virtualization Layer," *OpenFlow Switch Consortium, Tech. Rep*, Oct. 2009.

[35] G. Yao, J. Bi, and P. Xiao, "Source address validation solution with OpenFlow/NOX architecture," in *19th IEEE International Conference on Network Protocols*, Oct 2011.

[36] H. Wang, L. Xu, and G. Gu, "FloodGuard: A DoS Attack Prevention Extension in Software-Defined Networks," *45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, 2015.

[37] P. Porras *et al.* , "A Security Enforcement Kernel for OpenFlow Networks," *HotSDN'12*, Aug. 2012.

[38] ETSI-ISG-NFV, "Network Function Virtualisation; Ecosystem; Report on SDN Usage in NFV Architectural Framework," *http://www.etsi.org/deliver/etsi_gs/NFV-EVE/001_099/005/01.01.01_60/gs_NFV-EVE005v010101p.pdf*, Dec. 2015.