

A SDN and NFV use-case: NDN implementation and security monitoring

Théo Combe, Wissam Mallouli, Thibault Cholez, Guillaume Doyen, Bertrand Mathieu, Edgardo Montes de Oca

▶ **To cite this version:**

Théo Combe, Wissam Mallouli, Thibault Cholez, Guillaume Doyen, Bertrand Mathieu, et al.. A SDN and NFV use-case: NDN implementation and security monitoring. Guide to Security in SDN and NFV, Springer, 2017, Computer Communications and Networks book series (CCN). hal-01652639

HAL Id: hal-01652639
https://inria.hal.science/hal-01652639

Submitted on 30 Nov 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A SDN and NFV use-case: NDN implementation and security monitoring

Théo Combe, Wissam Mallouli, Thibault Cholez, Guillaume Doyen, Bertrand Mathieu, and Edgardo Montes de Oca[1]

Abstract: Combining NFV fast service deployment and SDN fine grained control of data flows allows comprehensive network security monitoring. The DOCTOR architecture[2] allows detecting, assessing and remediating attacks. DOCTOR is an ANR funded project designing a NFV platform enabling to securely deploy virtual network functions. The project relies on open-source technologies providing a platform on top of which a Named Data Networking architecture (NDN [2]) is implemented. NDN is an example of application made possible by SDN and NFV coexistence, since hardware implementation would be too expansive. We show how NDN routers can be implemented and managed as VNFs.

Security monitoring of the DOCTOR architecture is performed at two levels. First, host-level monitoring, provided by CyberCAPTOR, uses an attack graph approach based on network topology knowledge. It then suggests remediations to cut attack paths. We show how our monitoring tool integrates SDN and NFV specificities and how SDN and NFV make security monitoring more efficient. Then, application-level monitoring relies on the MMT probe. It monitors NDN-specific metrics from inside the VNFs and a central component can detect attack patterns corre-

---

[1] Théo Combe • e-mail: theo.combe@thalesgroup.com
Wissam Mallouli • e-mail: wissam.mallouli@montimage.com
Guillaume Doyen • e-mail: guillaume.doyen@utt.fr
Bertrand Mathieu • e-mail: bertrand2.mathieu@orange.com
Thibault Cholez • e-mail: thibault.cholez@inria.fr
Edgardo Montes de Oca • e-mail: edgardo.montesdeoca@montimage.com

[2] The DOCTOR project (http://doctor-project.org) is a collaborative research project partially financed by the French National Research Agency (ANR) under grant <ANR-14-CE28-0001>.

sponding to known flaws of the NDN protocol. These attacks are fed to the CyberCAPTOR module to integrate NDN attacks in attack graphs.

Introduction

The development of Software Defined Networking (SDN) in the past few years and the more recent introduction of Network Functions Virtualization (NFV) promise to simplify network management, with a significantly increased flexibility and real-time reconfiguration. The first application has been to apply this new SDN paradigm to existing networks, in order to simplify them. However one can go further in network softwarization, leveraging virtualization in both the control plane and the data plane, to build a fully virtualized network stack. Due to softwarization and standardization of hardware, development costs and times are shortened, and the development of innovative network stacks from scratch is made possible. In this chapter, we present the DOCTOR architecture, which makes use of SDN and NFV to implement NDN, a networking paradigm in which routing is based on content names rather than host addresses. The goal of this paper is double: following a presentation of the NDN paradigm, we show how SDN and NFV can be used to provide an infrastructure layer on top of which the NDN stack can be deployed. In the context of DOCTOR, we aim at running this stack in a production network involving real users. We will detail the innovative aspects of the envisioned virtualized infrastructure, from the design of the architecture, to its monitoring and interconnection with the IP-world. We then focus on how to address security in this infrastructure. We perform a survey of the vulnerabilities introduced by NFV, SDN and NDN, and sort them in categories depending on the targeted components. For each attack we identify the target (SDN, NFV or NDN), review possible remediations and assess their feasibility. We finally propose a practical monitoring solution depending on NFV orchestration to collect information on network topology, and on SDN to perform real-time remediation actions. This monitoring is performed by the CyberCAPTOR tool and Montimage Monitoring Tool (MMT) [1].

1. A VIRTUALIZED ARCHITECTURE for the deployment of emerging NETWORK FUNCTIONS

Current networks generally consist of heterogeneous and vendor-locked hardware and software components, with little or no support for interoperability. This leads to complex network management. This vertical segmentation prevents telecom operators from rapidly deploying new services. Moreover, innovation cycles are

often long, meaning that network operators are reluctant to introduce new paradigms or technology. New networking solutions require being fully designed (often including cumbersome standardization procedures), evaluated, monitored and secured to ensure that they do not disturb existing services and can provide rapid return on investments. Faced with these limitations, telecom operators are adopting new approaches for building networks stemming from the wide adoption of virtualization techniques in data centers. Virtualization provides greater flexibility in sharing hardware resources, which result in cost reductions and faster service deployment. We are thus seeing the emergence of network softwarization consisting in building Network Functions Virtualization components, which are treated as virtualized software instances deployed in Virtual Machines (VMs). In turn these Virtual Network Functions (VNFs) can be chained and managed via Software-Defined Networking controllers to create end-to-end communication services.

Our main objective in the DOCTOR project is to design a flexible and secure service-aware network architecture. The DOCTOR virtualized network architecture is designed with the NFV concept in mind to efficiently host network functions and services which can be performed at high throughput. Based on the SDN principles, the network control is separated from the data plane and is delegated to a controller. This controller allows configuring data routing, managing and orchestrating network services. These services include network monitoring that makes it possible to secure the overall virtualized architecture for the detection of network anomalies and attacks.

1.1 Architecture overview

Fig. 1 shows an overview of the DOCTOR virtualized network infrastructure, including the functional blocks and their interactions. Note that the interactions or interfaces are numbered in the figure with different two colors (green and purple) to separate the SDN control plane for virtual network configuration from the NFV management plane, which concerns the virtualized functions.

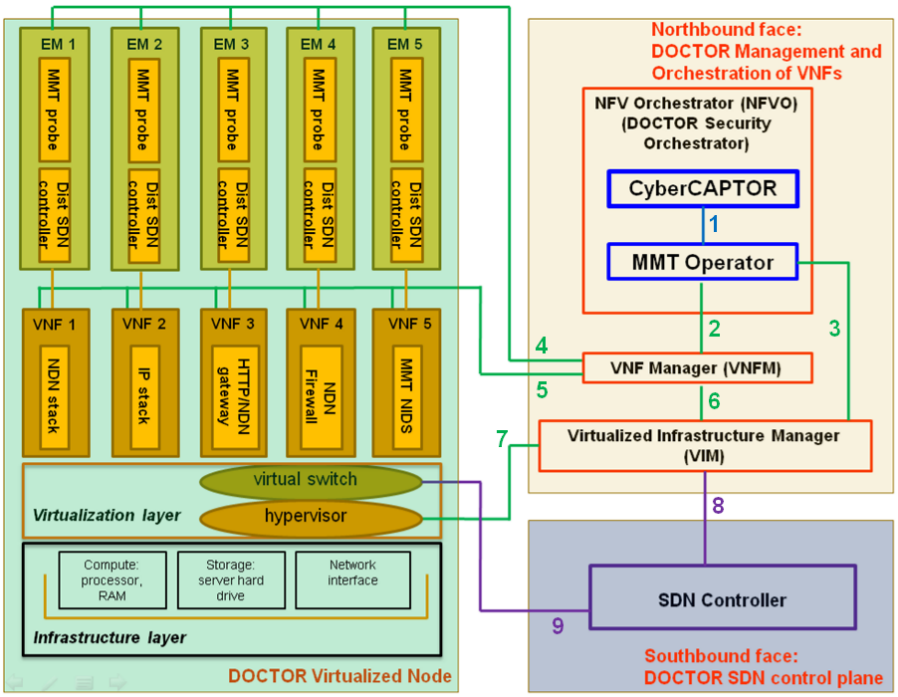


Fig. 1 Overview of the DOCTOR virtualized network infrastructure

We first designed a virtualized node to be able to deploy multiple network services as software instances or Virtualized Network Functions over a single physical host. Each deployed VNF thus run on one or several Virtual Machines, depending on the design. As such, the DOCTOR virtualized node can be structured into three layers. The application layer contains the VNFs, deployed as virtual machines over a virtualization layer which provides an abstraction for the underlying hardware resources offered by the physical hosts. A virtual network based on programmable virtual switches is then implemented to ensure end-to-end network connectivity between the virtualized machines, but also to enable network automation at the control plane.

1.2 Deploying ICN-based VNF: the DOCTOR use-case

The flexibility of the DOCTOR virtualized infrastructure makes it possible to host existing or new network services. To demonstrate this, the deployment of both IP and NDN protocol stacks was undertaken. NDN [2] is a recent networking paradigm that proposes moving away from host-based communication networks toward content-based ones. The goal is to find a solution that is better suited for the

massive diffusion of content in today's major Internet use-cases, such as video delivery or social networks applications.

1.2.1 NAMED DATA NETWORKING background

The novelty of NDN [2] relies on the key concept of naming content objects instead of naming hosts with IP addresses. NDN uses a hierarchical naming scheme for content objects, such as the Uniform Resource Identifier (URI). Communication in NDN is achieved using two types of packets: (1) Interest packets, and (2) Data packets. A user issues a request for some content by sending an Interest packet. In return, a Data packet containing the requested content is sent back to the user. In NDN, a router implements many interfaces that represent a generalization of those provided by IP networks and include three main components that enable the forwarding process. First, the Content Store (CS) that is a local cache intended for improving content delivery by storing recently requested or popular content. Secondly, the Forwarding Information Base (FIB) that contains routing information related to the name of Interest packets. Finally, the Pending Interest Table (PIT) that contains the state of emitted Interests with the purpose to route back Data packets and to aggregate requests. More precisely, for each forwarded Interest, the incoming interface is added to the corresponding PIT entry if not already present, so that the corresponding Data can be sent back to the user. For each Data received, the corresponding PIT entry is removed. Consequently, NDN defines a stateful data-plane which enables efficient routing of Interest and Data packets.

1.2.2 On the necessity of coupling IP and NDN

Even though NDN is considered as a clean-slate approach eventually aiming to replace the current IP-based data plane, it appears that such a deployment will reasonably not occur in one shot. To address this, several studies [21, 22, 23, 24, 25] show to what extent IP and ICN (Information-Centric Networks) can co-exist by leveraging SDN. However, if each of these solutions brings a proof of feasibility, they also induce some limits (e.g., inability to carry standard IP traffic, need for an extension of the Openflow protocol) due to the antagonist nature of these two networking paradigms. Consequently, a progressive deployment approach, standing for a serial combination of these protocol stacks, seems more realistic. This relies on the deployment of ICN islands inserted in the global IP network. Here, dedicated ICN/IP gateways are required to enable data transit through a boundary between heterogeneous domains. This solution can be of great value where ICN presents proven advantages when deployed on a particular topological location; and, NFV appears as a promising means to enable such a deployment strategy.

The DOCTOR project advocates this type of deployment strategy that allows NDN to operate and to be assessed in real contexts without entailing high risks and costs. A typical use case could be the provision of a service (e.g., HTTP web traffic) consumed by real users generating real traffic patterns. From the perspective of users and the Internet, the deployment of NDN must be transparent and the services must continue uninterrupted. To achieve this aim, dedicated gateways that convert the HTTP requests and responses respectively into Interest and Data packets have to be implemented and deployed. Fig. 2 illustrates the operation of an NDN/HTTP gateway. Basically, an HTTP client sends an HTTP request (red arrow) to the Ingress Gateway which transforms it into Interest packets by mapping the initial URL to a name prefix. These are sent through the NDN network via standard NDN routing to the Egress Gateway, thus benefiting from NDN mechanisms such as caching. The Egress Gateway collects the unresolved Interest packets, reconstructs the HTTP request and sends them to the corresponding web server. The server then sends the data in response to the Egress Gateway (green arrow) in the form of HTTP messages which, in a similar way, creates Data packets and sends them through the NDN network to reach the HTTP client via the Ingress Gateway.

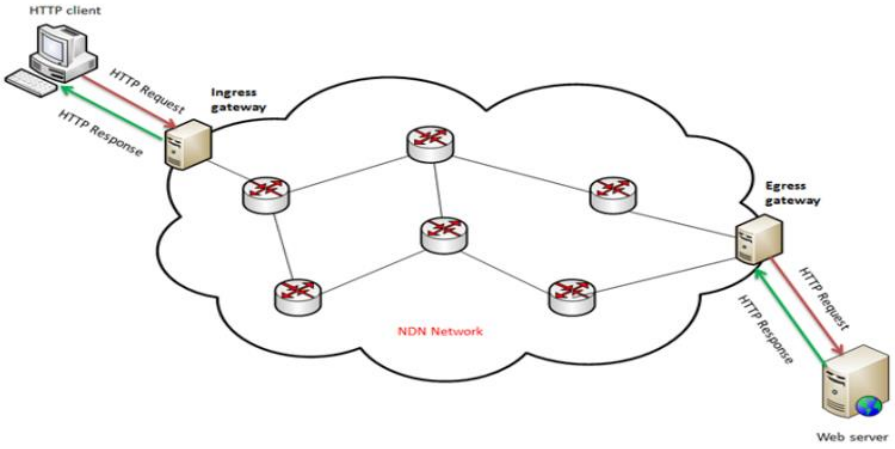


Fig. 2 NDN/HTTP gateway deployment

1.3 Managing the DOCTOR ARCHITECTURE

Monitoring and managing the network is a critical task for network operators. It is necessary for guaranteeing the security and the performance of the network. It also provides valuable knowledge (e.g., network load, type of traffic, peak hours) use-

ful for deploying and assessing new network services, such as the NDN protocol, or scaling existing services. To this end, specific virtualized functions were implemented for traffic monitoring and analysis, in particular for the detection and mitigation of network attacks specific to NDN. In this respect, each virtualized network service deployed in the application layer of the virtualized node is linked with an Element Manager (EM), which integrates a network monitoring function (provided by MMT (Montimage Monitoring Tool)), along with a distributed SDN Controller (dSDNC). These virtualized MMT probes and distributed SDN controller pairs allow to distribute the complexity of traffic monitoring over different virtualized network functions in order to consolidate, inter-correlate and aggregate monitored data (pre-processing) before sending them to the MMT Operator for deeper analysis in the context of unveiling network anomalies or attacks.

The DOCTOR virtualized network infrastructure also includes a framework providing dynamic configuration and management, as well as real-time security enforcement in the virtualized network. The proposed control and management plane (as represented on the right side in Fig. 1) consists in two function blocks:

- Infrastructure management and orchestration on the Northbound interface.
- Virtual network control on the Southbound interface.

The **Northbound interface** consists in functions for management and orchestration of VNFs, which are:

1. The Virtualized Infrastructure Manager (VIM), responsible for **provisioning hardware resources to VMs** (computing, storage, networking, including VM (re)configuration or migration, etc.) when necessary, based on the MMT Operator decisions (interface 3 in Fig. 1). To this end, the VIM controls the hypervisors of the DOCTOR virtualized node by using the interface 7.
2. **Monitoring and securing the VNFs**, to secure the whole virtualized networking infrastructure. This is implemented by the DOCTOR Security Orchestrator. The MMT Operator is responsible for coordinating traffic monitoring provided by the MMT probes distributed in each virtualized network service deployed in the project (interface 4). The MMT Operator interacts with the CyberCAPTOR manager (interface 1) for network security analysis (attack path detection and remediation).
3. **Management and configuration of the network functions** implemented with the VNFs. The MMT Operator obtains information from the CyberCAPTOR manager related to network security policies. It is thus able

to apply remediations or corrections on the virtualized network functions in response to network misuses (interface 2), through the VNF Manager using the interface 5. If needed, the VNF Manager can ask the VIM, via interface 6, to orchestrate (or allocate new) hardware resources for the VNFs.

The **Southbound interface** of the DOCTOR Control and Management Plane implements the **DOCTOR SDN control plane** which consists of a SDN controller interacting with virtual networks for dynamic configuration (interface 9). Following the SDN principles, the DOCTOR controller is mainly designed to acquire a global view of the network and enable centralized, intelligence-based network control. It actually interfaces with the DOCTOR Security Orchestrator (via the VIM using interface 8) to be notified of attacks or anomalies detected with the assistance of CyberCAPTOR, so as to correctly configure virtual networks to mitigate attacks. Its role includes, e.g., setting up the HTTP/NDN gateway to deliver traffic between heterogeneous network domains (i.e., IP and NDN), traffic load balancing, deploying rules in a firewall or an Intrusion Detection System / Intrusion Prevention System (IDS/IPS) service, adding/removing routes in NDN or IP router's forwarding tables, etc.

It is worth noting that the DOCTOR virtualized network infrastructure is designed respecting the recommendations from the ETSI NFV group, while leveraging the SDN principles for decoupling the control functions from the data plane. Thus, the application layer of the DOCTOR virtualized node consists of different VNFs which provide the suite of network services needed to deploy NDN; the virtualization and infrastructure layers of the node represent the NFV Infrastructure (NFVI); and, the Northbound interface of the Control and Management plane in the DOCTOR virtualized infrastructure implements the NFV Management and Orchestration. The DOCTOR controller in the southbound interface is intended for making the virtualized network services programmable, allowing them to be managed and controlled by a central element. The SDN principles are thus implemented by the controller, enabling: a clear separation between the control and forwarding planes; and, the centralization of network control to dynamically configure the network functions through well-defined interfaces.

2. SECURITY RISKS of SDN and NFV: DOCTOR use-case

SDN and NFV facilitate security management but also introduce new threats. The flexibility they provide to network infrastructures allows their in-depth monitor-

ing, with a central point gathering all needed information (the SDN controller and NFV orchestrator). However, these new features come with many new software elements and protocols, which increase the attack surface of the infrastructure. Moreover, some attacks specific to SDN and NFV have emerged. Consequently SDN and NFV have a bidirectional relationship with security: they both are security enablers and introduce vulnerabilities.

2.1 SECURITY ISSUES introduced by SDN and NFV

Network softwarization, in both control and data planes, generate a new attack surface that can be expressed as a set of vulnerabilities. These vulnerabilities target SDN control and data planes, NFV control plane, the virtualization layer, the accounting system, etc. In this section we propose a classification of the vulnerabilities related to SDN and NFV that need to be taken into account in a virtualized infrastructure.

In order to measure the risks faced by NFV and SDN, we adopt a practical point of view and survey the attacks. By attacks we mean any kind of malicious activity trying to collect, disrupt, deny, degrade, or destroy information or resources [3]] particularly targeting NVF and SDN. For this we identify the components that are likely targets, the possible attacks against them, and propose ways to detect and mitigate attack occurrences. Although SDN and NFV are distinct technologies, they are complementary to form the infrastructure layer on top of which services are built. Therefore, threats on them can be assessed following the same taxonomy, i.e., the separation between control and data planes, which leads to a similar separation of threats.

2.1.1 Network Function Virtualization ATTACKS

In a study [5], ETSI identifies the threat surface of NFV as the union of the threats to generic virtualization and networking. NFV being an implementation of Cloud computing technologies for networking, we surveyed attacks that have been performed against Cloud computing systems and hypervisors and analyzed the impact of such attacks on NFV.

Attacks on VIRTUAL NETWORK FUNCTIONS

VNFs are software components providing network functions, so they are likely to be vulnerable to classic software flaws, such as: Denial of Service DoS), bypass of isolation, and arbitrary code execution using, e.g., buffer overflows. Denial of service is not a new threat, but in a virtualized environment, its scope changes since DoS attacks can have side effects and affect other services collocated with the tar-

get. Arbitrary code execution allows an attacker to take over a VM or a VNF component, potentially compromising the whole VNF and providing a machine to continue or launch attacks. The principle is the same as for classical software, and such vulnerabilities are widely described in the CVE (Common Vulnerabilities and Exposures) database.

Against these attacks, the proposed solutions consist in leveraging Virtual Machine Introspection (VMI) which allows a monitor running on the host to check the integrity of the VMs. If a VM is taken over by an attacker (meaning that all detection mechanisms inside the VM are hence disabled or bypassed), the VMI can still detect and report the attack. An overview of VMIs is available in [6]. It provides a classification of VMIs that only report attacks, and other ones that can take action against them. A detailed formalism is proposed in [7].

Attacks on VIRTUALIZATION LAYER

Several types of attacks can be performed on the virtualization layer, such as:

- *Code execution on the physical host*: Wojtczuk [6] presents several attacks against common hypervisors (QEMU-KVM, Virtualbox, Xen) that allow code execution on host from a compromised or malicious Virtual Machine. These attacks allow a malicious VM to escape isolation and execute code on the host. For instance, concerning the lightweight virtualization, older versions (e.g. < 1.6.2) of Docker (used for implementing VNFs in DOCTOR) contain a vulnerability. It is identified by CVE-2014-9357 and allows uncompressing a Docker image to traverse the file system back to the root, permitting to override system binaries and leading to delayed arbitrary code execution. Another example, identified by CVE-2015-3630, shows how containers can modify shared resources to change host kernel parameters. This is possible when the isolation between hosts and containers is being assured by a blacklist of resources that cannot be accessed by containers, but the list is missing some elements that let containers access critical data on the host. For instance, some subdirectories like /proc and /sys are container-specific and others are system-wide.
- *Resource monopolization*: These attacks aim at overriding the hypervisor's resource limitations. Riddle and Chang [7] present attacks that steal resources. One, monopolization of CPU, concerns VMs running over a Xen hypervisor that can use up to 98% of the physical host's CPU, hence denying the CPU to other VMs. This will provoke a DoS or abusive charging fees in a pay-per-cycle model. Another, determining whether 2 VMs are co-resident, can be the starting point of another attack such as a side-channel attack to steal data, tak-

ing advantage of Xen's credit scheduler.
Yet another, I/O performance-based attacks, is based on knowing the scheduling of the hypervisor. This information can be used to overload I/O resources, resulting in slowing down co-resident VMs (or VNFs).

- *Data theft*: Data theft on the hypervisor can be performed by directly reading another VM's memory or disk by exploiting a vulnerability in the hypervisor or using a side-channel attack against cryptographic keys. Riddle and Chang [7] 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., keys, from the target VM. [8] proposes a method to infer execution path in a co-resident VM from cache timing attacks. Containers are even more vulnerable than VMs. The vulnerability CVE-2015-3630 in Docker allows a malicious container to directly access information related to other containers. This is possible due to a shared file in the `/proc` directory. The kernel vulnerability CVE-2015-2925 allows to escape mount namespace (double-chroot-like) and can give a malicious container at least read access to another container's disk image.
- *VM monitoring evasion:* These attacks aim at evading VM monitoring. Riddle and Chang [7] present the VM rollback attack that is possible when the hypervisor is already compromised. The attacker may execute a VM from an older snapshot without the VM owner knowing of it, allowing to bypass security mechanisms. For example, if the attacker is bruteforcing a password, causing the VM to raise a security alert, the compromised hypervisor can roll back to the previous snapshot so that the attacker can continue the attack. This allows avoiding internal VM monitoring.

Against all these attacks targeting the virtualization layer, a dedicated protection layer can be added by hardening the host to prevent the hypervisor process from accessing anything but the resources its associated VM can access (suited in the case of hypervisors running one or more processes per VM, as in the case of KVM or Docker). If an attacker manages to escape from a VM it will have access only to the resources related to this VM on the host. Such hardening systems exist in Linux environments (e.g., SELinux and Apparmor). Moreover, attempts to access forbidden resources can be logged and reported (e.g., using `auditd`).

More generally, an Intrusion Detection System on the host (HIDS) can be used to detect VM evasion attacks (code execution and data theft on VM disks). For instance, in [9] the authors propose an architecture to automatically build Apparmor profiles that match the Docker containers needs and trace their execution with an HIDS.

ORCHESTRATOR and/or VNF Manager

The orchestrator is in charge of placing the VMs on the nodes, triggering automatic scaling, chaining in the case of service chaining, live reconfiguring the VNFs (e.g., changing firewall rules in the case of a firewall VNF), etc. Thus, orchestrators are critical elements that centralize all configuration information. Attackers can target them either to disrupt services (DoS), to gain information on the infrastructure, or even to take control of the data path of the VNFs. For instance, the `create_images_and_backing` method in `libvirt` driver in OpenStack Compute (Nova), using KVM live block migration, does not properly create all the expected files. This 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, enabling further *eavesdropping*, data modification, or impersonation.

The orchestrator is subject to classical software vulnerabilities, so detection methods include hardening the machine on which it runs, logging all events and syscalls, or running the orchestrator inside a VM to benefit from Virtual Machine Introspection. Since it is a single point of failure, redundancy is required to avoid DoS attacks.

Other threats and attacks

Apart from the above mentioned elements that are core components of a NFV architecture, security must also be ensured for the following miscellaneous elements:

Communications with and within NFV MANO (Management and Orchestration): Communications between the VNFs and NFV MANO are subject to classical network eavesdropping and tampering though Man-in-the-Middle (MitM) attacks. However, Authentication, encryption (TLS) of the communication and the use of a dedicated control network can prevent this type of attacks.

- *Virtualized Infrastructure Manager:* The VIM is in charge of managing virtual resources and to directly control the hypervisor including VM images, snapshots, compute, RAM, storage, located at the infrastructure operator domain. An attacker may breach the VIM to launch his own VNFs, modify VM images to add some code, exfiltrate data, etc. As we see here, the VMI is subject to classical software vulnerabilities, so detection methods are similar to those described in the previous subsection.

2.1.1 *SOFTWARE-DEFINED NETWORKING attacks*

Decoupling the data-plane from the control-plane, SDN also suffers from threats on both of these two planes.

Packet flooding

On the data-plane, a known Distributed DoS (DDoS) attack against SDN consists in flooding a switch by sending crafted packets with many different source addresses/ports. Each different source address leads to a flow miss and the packet is forwarded to the controller. This results in the saturation of the link between the controller and the switch, and of the controller's computing capacities.

Regarding the control-plane, the following two topology poisoning attacks are unique to SDN and affect major SDN controllers such as Floodlight [10] and OpenDaylight [11]. They aim at deceiving the controller regarding the topology.

Host Location Hijacking

This attack exploits the Host Tracking Service of the controller 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 first retrieves the target's identifier used by the controller to identify the host (here: the MAC address), then injects fake packets in the name of the target host. As a result, users trying to access the genuine server are redirected to the malicious server.

The *Host-Location Hijacking attack* could be tackled by adding an authentication mechanism on the packets, making sure received packets are issued by the legitimate host. However, this would require signature verification for each packet (with large overhead) and an additional Public Key Infrastructure (PKI) for the hosts. Another proposed defence is to monitor pre-conditions and post-conditions surrounding a host migration. For instance, the pre-condition for a legitimate host migration is that the former location of the host and the corresponding switch port are not used anymore, and that the controller has received a Port_Down message. Similarly, a post-condition is that the host is unreachable at the previous location after migration. As a detection mechanism, the controller / switch could check these conditions when a migration occurs, and any migration that violates them (spoofed message from an attacker) could be detected and ignored in the Host Profile.

Link Fabrication

This attack consists in creating a fake link in the network either by injecting fake Link Layer Discovery Protocol (LLDP), a protocol used by the switches to automatically discover neighbors) packets, or via a relay fashion, i.e., without modifying the packets. This attack can be a first step for other attacks, such as a DoS attack, by taking advantage of the Spanning Tree algorithm used by OpenFlow controllers to incapacitate normal switch ports; or a MitM attack, by using the fact that once it detects that a new link is up, the controller re-computes the shortest route (Shin *et al* [12 change]) and could redirect packets to a host controlled by the attacker.

The *Link Fabrication attack* could be detected by authenticating LLDP packets, introducing the same large overhead and PKI issues as for Host Location Hijacking. Another detection mechanism proposed is in the hypothesis that the attacker is not on an SDN switch but on a host linked to the network. In this case, SDN switches could tag all their ports as HOST or SWITCH, depending on whether they are connected to a host or another switch. Such identification is possible by detecting host-specific traffic (e.g. DNS, ARP) on the links. Since LLDP packets are only exchanged with other switches and the controller, any LLDP packet coming from a HOST-tagged port would be detected as an attack and dropped. To evade this detection an attacker would have to stop all host-specific traffic on his machine. While this is possible on the attacker's own machine, it would disrupt normal service on a compromised host, leading to detection.

2.2 Security threats in IP vs. NDN

NDN is designed to intrinsically prevent some types of threats that IP needs to solve using external mechanisms. In IP networks, an attacker can send altered data to end-users, thus causing damage when content is delivered. To avoid this, IPsec or Transport Layer Security (TLS) needs to be used to prevent any data alteration and avoid other security issues. On the other hand, NDN signatures are intrinsically computed and included in each NDN Data packet. The user receiving the Data packet can use the information to verify the signature, hence ensuring the authenticity of the content and avoid tampered data.

The caching technique also helps reduce the impact of Denial of Service attacks. In this type of attack, a targeted machine is flooded with superfluous requests in an attempt to overload systems and prevent legitimate requests from being ful-

filled, but the caching mechanism intrinsically protects content servers from flooding attacks.

However, each NDN router keeps the incoming Interest packets in its PIT after forwarding them to enable the routing of related Data packets and also avoid the duplication of Interests. This exposes the PIT to an attack that consists of sending a large amount of Interest packets of non-existing content in a short period of time. The consequence of this stateful routing mechanism is that the PIT can be overloaded and thus cannot process Interest packets from legitimate users. This type of attack is called Interest Flooding Attack (IFA) and performing such attack is simple because NDN enables requesting content by name which can be easily crafted by attackers. It has been extensively studied [14, 15, 16, 17, 18, 19], as in [20] that relies on a custom simulator component, provides guidelines for its design and implementation. A recent release of the NDN reference implementation (NDN Forwarding Daemon – NFD) partially solves this issue by implementing a NACK packet which enables the rapid removal of Interests for non-existing content from the PIT. Nevertheless, there are still some attack patterns that are possible as indicated in [26].

For instance, if we consider the serial combination of IP and NDN networking domains, deployed into a virtualized infrastructure, one can easily understand that the stateful nature of NDN combined with in-network caching will exhibit different security properties as compared to the stateless nature of IP. To further understand the impact of this coupling on the overall security, we consider the IFA use-case previously described but now implemented in a scenario in which NDN and IP are coupled to forward web traffic. In this case, an attacker, located in an IP domain who wants to reproduce an IFA in an intermediate NDN island by leveraging HTTP traffic, may try to flood the network with HTTP requests for non-existing web content. However, as illustrated in Fig. 2, users are not directly connected to the NDN network but to the Ingress Gateway, thus moving the problem to this entry point that should be able to detect flooding attacks with regular DoS mitigation strategies for IP networks.

In order to successfully perform the IFA in a combination of NDN and IP domains, an attack must go beyond the basic IFA mechanism. A possible attack scenario consists in stretching the responding delay of any HTTP answers with the help of a malicious website (Fig. 3 IFA setup in an IP/NDN environment). The consequence of this scenario is that IP and NDN do not protect themselves, as before, but rather make the phenomenon harder to mitigate. From an IP perspective, the symmetric nature of the traffic, as well as its rate-limited nature, makes it an ideal candidate for the definition of detection rules that an Intrusion Detection

System can implement. In IP domains, the attack traffic cannot be separated from the legitimate one. By contrast, in the NDN domain, the delay spent by Interest to get Data packets unavoidably fills the PIT and prevents the NACK from removing these illegitimate entries. Decupling this pattern by endorsing a sufficient amount of partner websites can easily lead to PIT collapses in the NDN nodes.

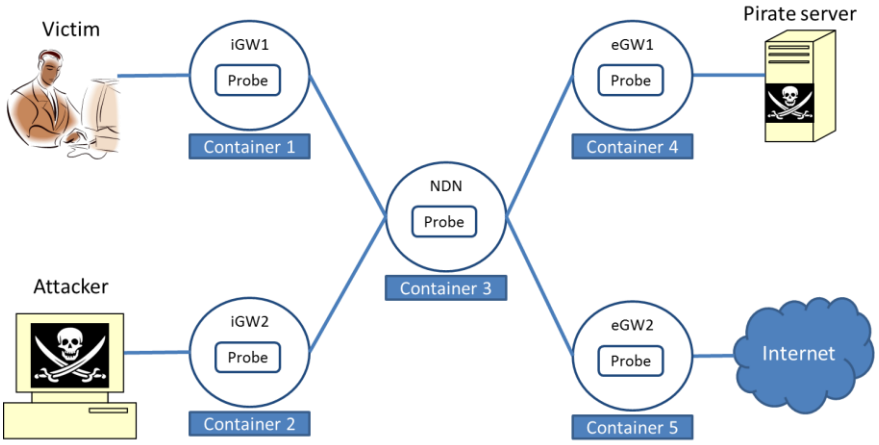


Fig. 3 IFA setup in an IP/NDN environment

To conclude, we have shown how the combination of networking domains can be easily deployed in a virtualized infrastructure. We have also shown that in the case of Denial of Service attacks, for instance, novel security mitigations are possible but new threats also exist and need to be addressed. The normal behavior in one domain may be considered as an abnormal in another due to the different protocols and network functions running. Furthermore, the security mechanisms are divided and network operators in charge of a particular domain lack a global view of the threats that would allow them to better understand what is occurring in the network to be able to detect and mitigate attacks and malfunctions. The next section presents how practical tools can be used to defend against the aforementioned security threats.

3 CYBERCAPTOR and MMT: a set of tools for a SECURE DEPLOYMENT of NDN as VIRTUAL NETWORK FUNCTIONS

A major asset of SDN and NFV is to provide a high level of programmability to networks. This can be used to enforce complex security policies, detailed monitoring and fast reaction on threat detection. In the DOCTOR project, collaboration between Thales and Montimage resulted in a cyber-monitoring and reaction toolset that leverages SDN and NFV concepts, and is adapted to the particular context of NDN. The Montimage Monitoring Tool provides network information on topology, metrics and alerts of the NDN and NFV/SDN network to Thales' CyberCAPTOR tool, which relies on an analysis of attack graphs to assess possible attack paths and their level of risk. We describe these two components and their functionalities in this section.

3.1 The MONTIMAGE MONITORING TOOL

MMT is a monitoring solution that combines a set of functionalities that include:

- Data capture, filtering, and storage;
- Events extraction and statistics collection; and,
- Traffic analysis and reporting for providing, network, application, flow and user level visibility.

MMT is composed of a set of complementary and independent modules as shown in Fig. 44 and described below.

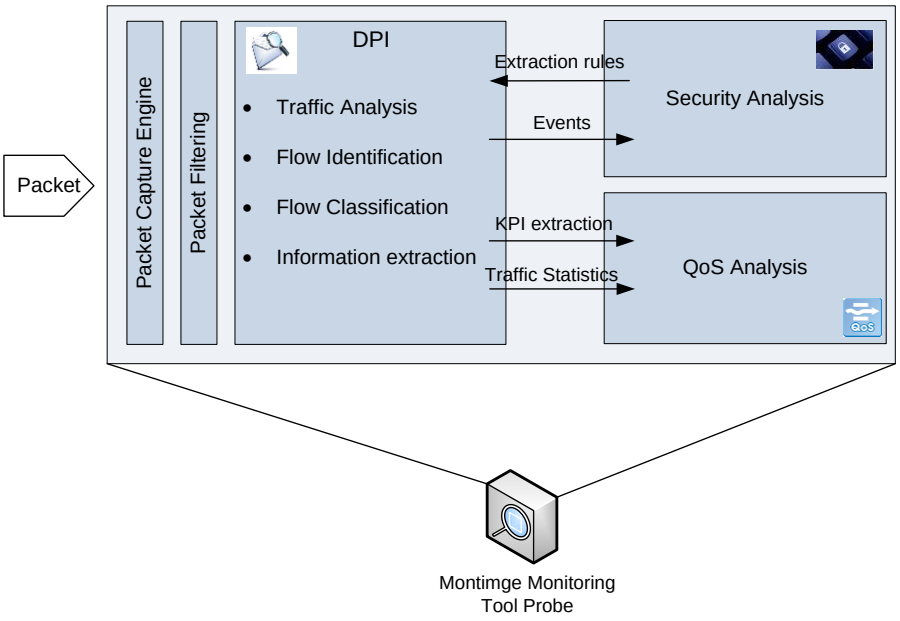


Fig. 4 MMT Global Architecture

- **MMT-Capture:** allows the capture of network packets using the `libpcap` or other packet capture libraries including DPDK.
- **MMT-Filter:** is a basic filtering tool that permits focusing on only some specific types of traffic depending on the usage of the network probe.
- **MMT-DPI** is the core packet processing module. It is a C library that analyses network traffic using Deep Packet and Flow Inspection (DPI/DFI) techniques in order to extract network and application based events, measure network and per-application QoS/QoE parameters and Key Performance Indicators (KPIs). In the context of DOCTOR, a new plugin to monitor the NDN protocol stack has been developed to extract different NDN protocol field values and perform basis statistics. This extracted metadata is important for performing security analysis of the communications between different NDN nodes and detecting potential security flaws specific or not specific to NDN.
- **MMT-Security** is a rule engine that analyses and correlates network and application events to detect performance, operational and security incidents. The rules are written in XML and permit to aggregate detected events using logical (AND, OR, NOT) and temporal (BEFORE, AFTER) operators. It has self-learning capabilities to obtain network intelligence,

perform dynamic threshold based analysis, and identify possible Denial of Service attacks.

- **MMT-QoS** allows providing visibility on the quality of the network in terms of different KPI, such as delays, jitter, response times, etc, that can also be used to help detect DoS attacks.
- **MMT-Operator** is a JavaScript web application that allows visualizing reports and alarms generated by the probes.

3.1.1 MMT as NIDS

MMT can be deployed as a Network based Intrusion Detection System (NIDS**)** in a separate virtual machine. This NIDS can be placed at strategic points within the network to monitor traffic to and from the different Virtualized Network Functions (e.g., NDN nodes, HTTP/NDN Gateway, Firewall). The chaining of the virtual machine is configured by the virtualization layer component (e.g., Open vSwitch) to place the MMT NIDS just after the HTTP/NDN for intercepting the NDN based network traffic. In this way, MMT can passively analyze traffic on the entire subnet, and match the traffic passed on the subnets to the library of known attacks. Once an attack or abnormal behavior is identified, an alert will be sent to the administrator via the MMT-Operator.

Deploying MMT as a NIDS allows monitoring the NDN network traffic to obtain a global view of the network comprised of metrics related to QoS (e.g., response times) and detections of attacks targeting different NDN nodes. However, NIDSs are used to monitor NDN network traffic and alert on suspicious activity that violates network security policy. Typically, one network node is tapped from which the NIDS then gains its input. What network node should actually be tapped for the NIDS depends on the network structure in use. However, IDS systems in general function best in environments with limited amounts of noise. In very noisy environments the systems typically produce large amounts of alerts including a number of false positives. Thus, NIDS need to be placed at strategic points to monitor traffic to and from the different devices and virtual machines and network policy will be enforced by the security rules defined and activated.

3.1.2 MMT deployed inside each VNF

A lightweight version of MMT probes can be co-located with each VNF. This allows the analysis of metrics and security indicators related to the VNF. In this scenario, only parts of the parsing plugins in MMT-DPI are needed to fulfill the list of protocols used by the VNF. Besides, the security analysis and intrusion detection needs only to target the risks and vulnerabilities identified for the given VNF application and differentiate abnormal activity from allowed activity. The

security analysis methodology and properties of an NDN node are indeed different from the ones for a firewall or a HTTP/NDN gateway.

The performance impact of the monitoring probe can be reduced when it focuses only on part of the network traffic. Besides, the monitoring tool can analyze specific VNF security issues and apply advanced algorithms to detect pre-identified risks and attacks targeting the single VNF. It can be adapted to the specific requirements of NDN nodes to analyze NDN activity and detect any abnormal behavior. However, the monitoring tool installed in each VNF consumes part of the memory and CPU allocated for the VNF. This can have an impact on the network operation and can add delays in communications. Furthermore, the monitoring tool will have only local visibility of the VNF traffic which compromises the detection of collaborative attacks or attacks involving different network paths. This last limitation is addressed by the sharing of data between MMT probes (P2P cooperation) and by performing centralized analysis (done by the MMT Operator) in order to improve intrusion detection capability.

3.1.3 COLLABORATIVE MONITORING

The deployment of MMT probes inside VNFs or as NIDS and the collaboration between distributed probes, directly using P2P communications or through the centralized application, allow to dynamically build the network topology and even to detect at runtime any change that may occur during the network operation (e.g., adding or removing network nodes and functions).

This information, as well as the detection of network incidents including functional or non-functional incidents, allows providing valuable input to the CyberCAPTOR tool to assess the risk of such adaptive virtual network and propose relevant remediation to mitigate the impact of a vulnerability or stop an ongoing attack (e.g., malicious data exfiltration or scans). The remediation action to be taken needs to be selected at runtime (preferably in an automated way) and then orchestrated by the VNF manager and/or SDN controller to ensure the security of the NFV/SDN-based environment.

3.2 CYBERCAPTOR

CyberCAPTOR is a security monitoring tool based on an attack graph model. Initially developed for physical networks, it was later adapted to virtualized networks and eventually NDN in the particular context of the DOCTOR project. It is composed of four main modules forming a data pipeline, and a graphical visualization interface. These modules are attack graph generation, attack paths extraction, attack path scoring and remediation. The first three modules are automatically

chained (with parameters given by the operator), while the remediation module requires manual validation to commit a remediation proposal.

CyberCAPTOR's inputs are the network topology, vulnerability scans of the machines, fixed and variable costs for applying elementary remediation, operational costs for the infection of a given machine or denial of service and an up-to-date vulnerability database (the NVD database [13]). Its outputs are the complete attack graph, all the extracted attack paths, their scores and a list of remedies (i.e., list of actions to perform) for a given attack path.

3.2.1 Attack graph generation

The attack graph approach allows a defender to enumerate all possible attack paths for an attacker, given a network topology (i.e., network and software configuration, VMs placement and domain dependencies). It relies on an up-to-date vulnerability database and a global knowledge of the network. CyberCAPTOR depends on the MulVAL attack graph engine [4]. It is an engine that uses generic rules and vulnerability information from the system to produce attack graphs. A few dozen rules are enough to model most attack steps. System topology and vulnerability information are used as parameters for the generic rules, thus forming attack steps. These attack steps have several inputs, called preconditions, and an output, called postcondition. MulVAL then produces an AND-OR graph, composed of 3 types of nodes: AND nodes, OR nodes and LEAF nodes.

An attack step needs all its preconditions to be true to satisfy its postcondition. For this, "AND" logical nodes are used. On the other hand, "OR" nodes represent different ways for an attacker to gain some level of privileges on the network (e.g., different attack steps that lead to the same postcondition). LEAF nodes are nodes without preconditions. They correspond to elementary preconditions, or "facts", i.e. information given as input. These facts are the conditions that can further be remediated. In the example shown by 5, there are 4 leaves, 2 AND nodes and 2 OR nodes.

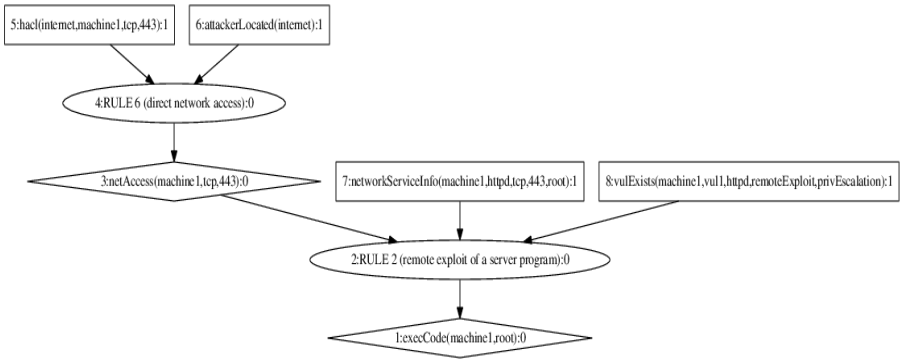


Fig. 5 Simple attack graph

3.2.2 Attack path extraction

The complete attack graph for a company network is very large (potentially millions of edges for a few hundred machines), so that it is not relevant to present it to an operator. Due to the complexity of many information systems, focusing interest on particular subgraphs of the attack graph is necessary. A noticeable subgraph category is attack paths.

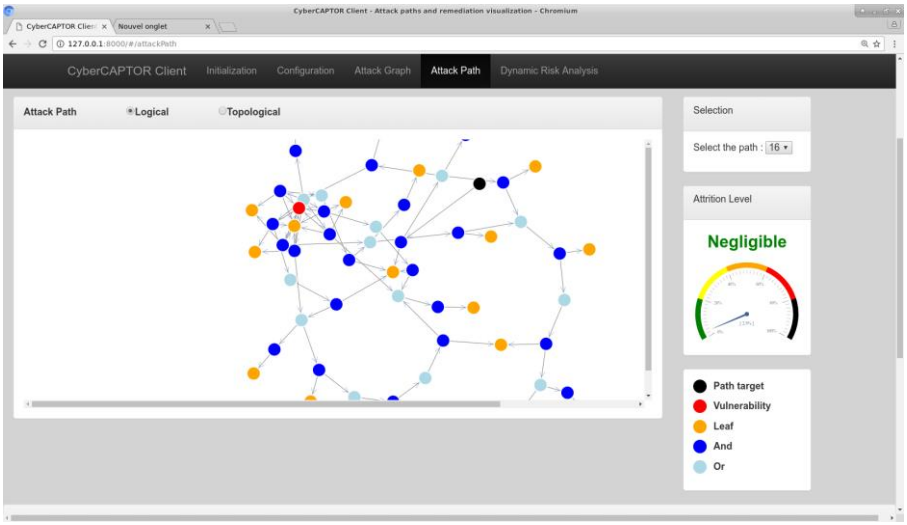


Fig. 6 Attack path extraction with CyberCAPTOR

An attack path is a subgraph of an attack graph corresponding to all graph nodes an attacker can cross to reach a certain objective (generally execute code on a given machine). It is a Directed Acyclic Graph (DAG) rooted on the target machine.

Its LEAF nodes are all facts of the topology that can be used to attack a particular target. Attack paths consequently show the subset of facts that can be changed in order to thwart the attack. (**Erreur ! Source du renvoi introuvable.**).

3.2.3 Scoring

Attack paths are scored according to various metrics, in order to automatically present the most relevant paths to an operator. This is done by assessing the criticality of each attack path or the likelihood of their occurrence. Attack path scores have 2 components: impact score and risk score.

The impact score is defined as the sum of local impacts for all vertices of the attack graph. The local impact for each vertex is defined by the user, often motivated by operational aspects. By default, each rule (e.g., vulnerability exploitation, network access) has a constant local impact.

Risk scores model the likelihood of the realization of an attack path. It is computed from the LEAF nodes of the attack path to its root: each LEAF represents a fact, with a default risk (depending on the fact), and each AND and OR nodes has a risk depending on the corresponding fact or rule and the number of ingoing and outgoing vertices of the node.

Each attack path is given a score, which are then normalized between 0 and 1, and sorted.

3.2.4 Remediation

CyberCAPTOR provides information on possible remediation actions to prevent the exploitation of identified attack paths. This corresponds to a list of actions that need to be carried out on the network topology that will disable the attack path. A remediation action is an elementary change in the topology. Each remediation action roughly corresponds to a different precondition. For instance, a patch remediates a vulnerability, a firewall rule remediates a network access, and moving a VM protects it from security incidents on a particular host.

Since multiple action combinations can be applied, all combinations are proposed so that the operator can choose the best one according to functional / business needs. Once a remediation has been chosen, the attack paths are recomputed to take into account the topology changes.

3.2.5 Interactions with SDN and NFV

Although CyberCAPTOR does not depend on specific methods to gather the necessary knowledge (e.g., network scans, static configuration file analysis, vulnerability scans), SDN and NFV offer ways to obtain the required information. For instance, the monitoring tool can retrieve the network topology from the SDN controller, and the orchestration relations and VMs placement from the NFV orchestrator.

CyberCAPTOR does not directly depend on SDN or NFV, but it can improve its efficiency through the combination of both technologies. More specifically, the control plane centralization allows obtaining information on the network's configuration from a single point: the SDN controller. The controller keeps track of all the allowed flows in real time, while in a classical network one would need to periodically gather information on the configuration of firewalls and routers. Similarly, the NFV orchestrator can provide information concerning software versions and configuration of the VMs without launching scans. Furthermore, the remediation recommendations provided by CyberCAPTOR can be, after being validated by an operator, directly sent to the SDN controller and/or NFV orchestrator to be applied. This enables much faster and less error-prone information collection and remediation enforcement than can be achieved manually.

MMT and CyberCAPTOR are therefore fully complementary: the first can provide from its deep monitoring the detailed states of the virtualized architecture to CyberCAPTOR, which can in turn give back the critical attack paths to be monitored and the remediations to perform, leading to a very efficient architecture to secure the deployment of NDN as virtual network functions.

Conclusion

SDN and NFV promise a greater flexibility in networks, by the means of a separation between the control and data planes, a centralization of management via controllers and orchestrators and the massive use of virtualization for the data plane, at the expanse of an increasing complexity of the infrastructure. We showed through the architecture of the DOCTOR project how these emerging technologies allow deploying novel network stacks such as NDN that can co-exist with IP thanks to network slicing while bringing new services like optimizing content distribution at the network level.

Moreover SDN and NFV allow improved security monitoring, permitting faster and more accurate knowledge of the network through a centralized control plane.

However the added complexity, both in SDN/NFV and in the NDN stack, brings a large attack surface, which we tried to assess, in order to thwart the most likely attacks. For each technology, we presented the main known attacks and ways to detect and mitigate them.

Our prototype is monitored and secured thanks to a pro-active approach with CyberCAPTOR and a reactive approach thanks to Montimage Monitoring Tool, both tools being complementary and needed to secure such a complex and innovative architecture.

The natural following of this research work is to assess the whole infrastructure while processing real user traffic while facing attacks in the same time. Therefore, we plan to involve soon real users thanks to the HTTP-NDN gateway.

References

[1] Montimage website. Available: http://www.montimage.com/products.html
[2] NDN. Available: https://named-data.net/
[3] CNSS, "National Information Assurance Glossary,". Available: http://www.ncsc.gov/nittf/docs/CNSSI-4009_National_Information_Assurance.pdf
[4] MulVAL Project at Kansas University. Available: http://people.cs.ksu.edu/~xou/mulval/
[5] ETSI-ISG-NFV, "Network Functions Virtualisation (NFV); NFV Security; Problem Statement," 2014. Available: http://www.etsi.org/deliver/etsi_gs/NFVSEC/001_099/001/01.01.01_60/gs_NFVSEC001v010101p.pdf.
[6] R. Wojtczuk (2014) "Poacher turned gamekeeper: Lessons learned from eight years of breaking hypervisors,". In: *Black Hat USA*, 2014.
[7] A. R. C. a. S. M. Riddle (2015) "A Survey on the Security of Hypervisors in Cloud Computing". In: *IEEE 35th International Conference on Distributed Computing Systems Workshops*, 2015.
[8] G. Wang, Z. J. Estrada, C. Pham, Z. Kalbarczyk and a. R. K. Iyer (2015) "Hypervisor Introspection: A Technique for Evading Passive Virtual Machine Monitoring,". In: *WOOT*, 2015.
[9] D. Kreutz, F. M. V. Ramos and a. P. Verissimo (2013) "Towards Secure and Dependable Software-Defined Networks,". In: *HotSDN*, 2013.
[10] "Floodlight OpenFlow Controller,". Available: http://www.projectfloodlight.org/floodlight/.
[11] "The OpenDaylight Platform,". Available: https://www.opendaylight.org/.

[12] S. Shin (2014) “Rosemary: A Robust, Secure, and High-performance Network Operating System,”. In: *CCS*, 2014.
[13] National Vulnerability Database. Available: https://nvd.nist.gov/download.cfm
[14] P. Gasti et al. (2013) “DoS and DDoS in Named Data Networking”. In: Computer Communications and Networks (ICCCN). In: Conference on. IEEE, 2013, pp. 1–7.
[15] H. Dai et al. (2013) “Mitigate DDoD attacks in NDN by Interest traceback”. In: Proc. of IEEE INFOCOM NOMEN Workshop, 2013.
[16] A. Compagno et al. (2013) “Poseidon: Mitigating Interest flooding DDoS attacks in Named Data Networking”. In: Local Computer Networks (LCN), Intl’ Conference on. IEEE, 2013, pp. 630–638.
[17] A. Afanasyev et al. (2013) “Interest flooding attack and countermeasures in Named Data Networking”. In: IFIP Networking Conference. IEEE. 2013, pp. 1–9.
[18] Nguyen T., R. Cogranne, and G. Doyen. (2015) “An optimal statistical test for robust detection against Interest flooding attacks in CCN”. In: Integrated Network Management (IM), IFIP/IEEE Intl’ Symposium on. 2015, pp. 252–260.
[19] T.N. Nguyen et al. (2015) “Detection of Interest flooding attacks in Named Data Networking using hypothesis testing”. In: Information Forensics and Security (WIFS), IEEE Intl’ Workshop on. 2015, pp. 1–6.
[20] M. Virgilio, G. Marchetto, and R. Sisto. (2013) “PIT overload analysis in content centric networks”. In: Proc. of 3rd ACM SIGCOMM workshop on Information-centric networking. ACM. 2013, pp. 67–72.
[21] M. Vahlenkamp, F. Schneider, D. Kutscher and J. Seedorf (2013) "Enabling Information Centric Networking in IP Networks Using SDN,". In: *Future Networks and Services (SDN4FNS), 2013 IEEE SDN for*, Trento, pp. 1-6.
[22] S. Salsano, N. Blefari-Melazzi, A. Detti, G. Morabito, L. Veltri (2013) In: "Information centric networking over SDN and OpenFlow: Architectural aspects and experiments on the OFELIA testbed", Computer Networks, Volume 57, Issue 16, 13 November 2013, Pages 3207-3221, ISSN 1389-1286
[23] N. L. M. van Adrichem and F. A. Kuipers (2015) "NDNFlow: Software-defined Named Data Networking,". In: *Network Softwarization (NetSoft), 2015 1st IEEE Conference on*, London, 2015, pp. 1-5.
[24] X. N. Nguyen, D. Saucez and T. Turletti, (2013) "Efficient caching in content-centric networks using OpenFlow," *INFOCOM, Proceedings IEEE*, Turin, 2013, pp. 1-2.
[25] Peyman TalebiFard, Ravishankar Ravindran, Asit Chakraborti, Jianli Pan, Anu Mercian, Guoqiang Wang, Victor C.M. Leung (2015) "An Information Cen-

tric Networking approach towards contextualized edge service". In: *12th Annual IEEE Consumer Communications and Networking Conference (CCNC)*, Las Vegas, NV, 2015, pp. 250-255.

[26] Hoang Long Mai, Ngoc Tan Nguyen, Guillaume Doyen, Alain Ploix, Remi Cogranne (2016) On the Readiness of NDN for a Secure Deployment: The Case of Pending Interest Table. In: proceedings of the 10th IFIP WG 6.6 International Conference on Autonomous Infrastructure, Management, and Security, AIMS 2016. pp 98-110. Lecture Notes in Computer Science 9701. Springer International Publishing, 2016.

Exercises

- What is NDN? Is it secure? How to detect a NDN attack?
- Which network use-cases are addressed by NDN?
- What are some practical applications of SDN and NFV?
- How do SDN and NFV make NDN implementation possible? Are they necessary?
- What risks and vulnerabilities are brought by SDN and NFV?
- How to use SDN and NFV as levers to secure an information system?

Author biographies

Thibault Cholez is an Associate Professor at the University of Lorraine. He teaches at TELECOM Nancy, an engineering school in Computer Science, and undertakes his research activities within the laboratory LORIA / INRIA Nancy-Grand Est. He previously got a PhD degree in Computer Science from Henri-Poincare University. His research interests concern Data Network Monitoring and Analytics, with a particular focus on content diffusion protocols and their security.

Guillaume Doyen is an associate professor in Troyes University of Technology (UTT), France, since 2006. He is affiliated to the Charles Delaunay institute (ICD - UMR CNRS 6281) where he is the co-chair of the Cyber-Security transversal research project. His current research interest focuses on the design of autonomous management and control solutions applied to the performance and security of content distribution and cloud computing. He has published more than 40 papers in the network and service management community. As an active member, he is a TPC member of high-venue conferences (e.g. IEEE/IFIP CNSM, IEEE/IFIP IM and NOMS, IFIP AIMS) and a regular reviewer for the top-related journals (e.g. IEEE CommMag, IEEE TNSM, Springer JNSM, Wiley IJNM) and has been a co-chair of several events (e.g. ManSDN/NFV, IFIP AIMS). He has been involved in several research projects (e.g. ANR-Doctor, IA-Request, ANR BBNet).

Dr Bertrand Mathieu joined France Telecom, Orange Labs in 1994. He received a Diploma of Engineering in Toulon, the MsC degree from the University of Marseille and the PhD degree from the University Pierre et Marie Curie in Paris. Until 1999, he worked on network management including interfaces, protocols and platforms. Since 1999, he is working on distributed computing, programmable networks and he is currently focusing his research activity on dynamic overlay networks, P2P networks and Information-Centric Networking. He contributed to several national and European projects (Corsica, Safari, FAIN, Ambient Networks, OneLab, P2Pim@ges, Envision, eCousin, Doctor). He published more than 50 papers in international conferences, journals or books. He is member of several conferences Technical Program Committee and an IEEE and SEE senior member.

Dr Wissam Mallouli is currently a research & development project manager at Montimage, France. He received his PhD in computer science from Telecom and Management SudParis (France) in 2008. His topics of interest cover formal testing and monitoring of functional, performance and security aspects of networks and cloud based systems. He is working on several European and French research pro-

jects. He also participates to the program/organizing committees of numerous national and international conferences. He published more than 30 papers in conference proceedings, books and journals.

Théo Combe is a research engineer at Thales Services. He graduated from the Ecole polytechnique, France and followed a double degree at Télécom ParisTech, where he studied networks and cyber-security. In 2014, he had a 4-month internship at the National Cybersecurity Agency (ANSSI, Paris, France) on side-channel attacks against asymmetric cryptography. In 2015, he worked at Thales Communications & Security in the SDN-NFV research group, as a part-time project along with his studies.

Edgardo Montes de Oca graduated as engineer in 1985 from Paris XI University, Orsay and DEA from Paris VI, Paris. He has worked as research engineer in the Alcatel Corporate Research center in Marcoussis, France and in Ericsson's Research center in Massy, France. In 2004 he founded Montimage, and is currently its CEO. Montimage specializes in the development of network and application monitoring tools for performance and security analysis. His main interests are designing state-of-the-art tools to test and monitor applications and telecommunication protocol exchanges, and the development of software solutions with strong performance and security requirements. He has participated in many research collaboration and product development projects for Alcatel, Ericsson and Montimage (e.g. Diamonds-Itea2, SIGMONA-CelticPlus, SENDATE-CelticPlus, SISSDEN-H2020, DOCTOR-ANR). He is member of NetWorld2020 and has published many papers and book chapters on SDN/SVN, testing, network monitoring, network security and performance.