Misuse Patterns for NFV based on Privilege Escalation

Abdulrahman K. Alnaim, Florida Atlantic University
Ahmed M. Alwakeel, Florida Atlantic University
Eduardo B. Fernandez, Florida Atlantic University

ABSTRACT

Network Function Virtualization (NFV) leverages cloud computing and virtualization technology to deliver on-demand network functions as software services, which are hosted by virtual machines (VMs). These functions are created and managed by the NFV hypervisor, which is a collection of software modules that provide virtualization of hardware resources and thus enable several VMs to be run on a single physical server. The hypervisor has to be secured to ensure service continuity, and protect the data of the VMs. In this paper, we use misuse patterns to study some attacks that may jeopardize the hypervisor. Misuse patterns describe how an attack is performed from the point view of the attacker. We are building a catalog of misuse patterns for NFV virtual machine environments (VME), and we present here two of them, which are unauthorized access to hardware resources and stop victims' VNFs causing Denial of Service (DoS), both of which are based on privilege escalation threat.

Keywords: network function virtualization (NFV), cloud computing, virtualization, virtual machine environment (VME), hypervisor, misuse patterns.

1. INTRODUCTION

The telecommunication industry is having a new and advanced shift in its infrastructure and network service delivery. Traditionally, telecommunication service providers (TSP) are required to deploy proprietary network hardware to establish and deliver network functions such as firewalls, routers, switches, domain name server (DNS), etc. However, with the increasing need of network services, TSPs are required to deploy additional hardware to meet the consumers' demands, which makes managing the network infrastructure a cumbersome process, not to mention the operational cost (OpEx) that would increase with the expansion of network infrastructure.

However, TSPs are now able to deliver a better network service using network function virtualization. NFV is a network architecture that takes advantage of cloud technology to virtualize network functions that may be chained together to build a virtualized communication service. NFV is a new paradigm of cloud computing virtualization that ensures the provision of a shared, scalable, and securable network environment. Here, we consider the TSPs as NFV providers.

As shown in Fig 1, NFV consists of three main components [Ets14]. First, the virtualized network functions (VNF) that are software implementations of network functions. Second, NFV management and orchestration (MANO), that covers the lifecycle management and orchestration of NFV resources, and consists of three parts; the *Orchestrator* is responsible of managing the lifecycle of network services; the *VNF Manager* is responsible for VNFs lifecycle management; and the *Virtualized Infrastructure Manager* (VIM), which is responsible for managing and controlling the interaction of the VNFs with the NFVI resources. Third, network function virtualization infrastructure (NFVI), which comprises all the hardware and software components to support the execution of the virtualized network functions. The NFVI also contains the hypervisor, which resides within the virtual machine environment. The hypervisor is a collection of software modules that provides virtualization of hardware resources and enables several virtual machines (VMs) to be run on a single physical server [Ets15]. It is responsible for creating and managing the VMs and mediating the access between hardware and these VMs, ensuring isolation among VMs, process scheduling of VMs, and managing VMs lifecycle [Cha18].

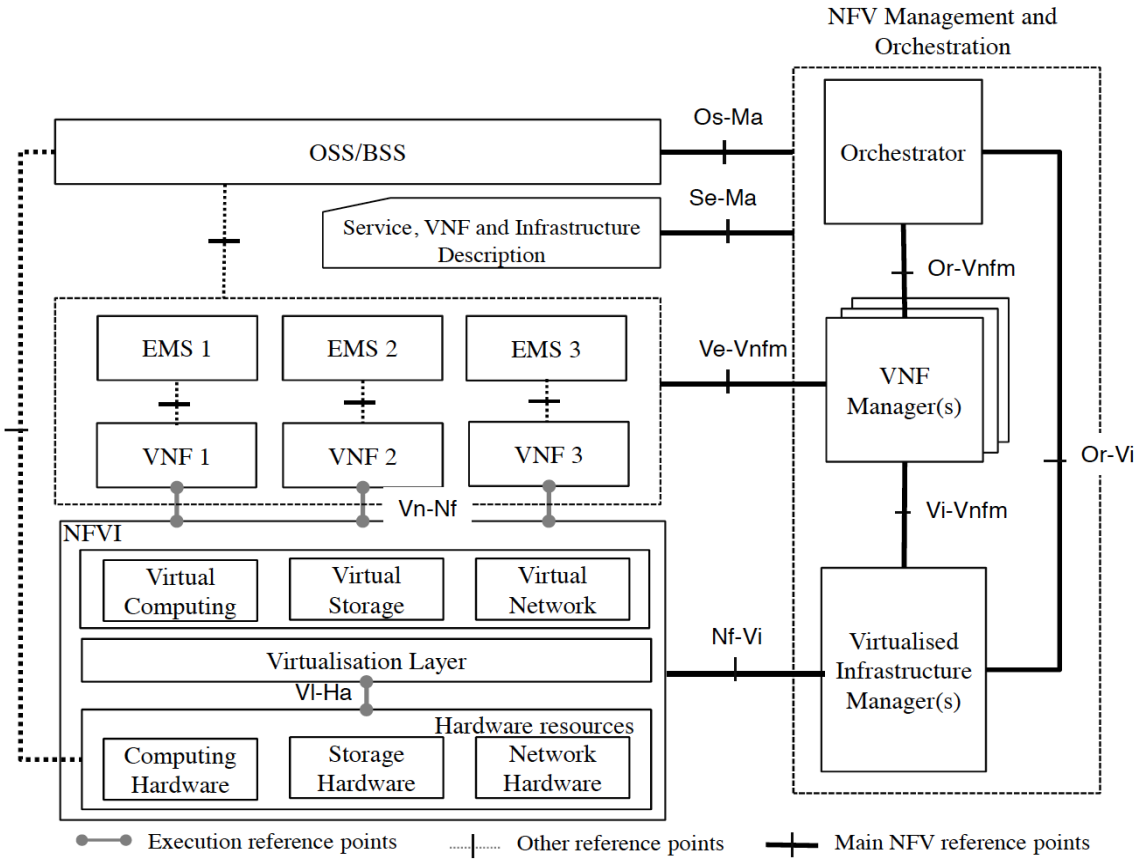

Fig. 1. NFV Reference Architecture Framework [Ets14]

In spite of the benefits that NFV promises, the NFV providers have to undertake a substantial effort to secure their services. Thus, in order to design a secured NFV system, we need to understand its possible threats. In [Alw18] we have surveyed the main security threats of NFV and the possible countermeasures to mitigate these threats. Our approach was to categorize the threats based on the vulnerabilities in NFV. In this work, we describe one of these threats, the privilege escalation threat. As shown in fig. 2, each threat may lead to several misuses of information. A misuse pattern describes how an attack is performed from the point of view of the attacker [Fer13]. It also defines the environment where the attack is performed, what security mechanisms are needed as countermeasures to stop it, and how to find forensic information to trace the attack once it happens.

In this paper, we present two misuse patterns, unauthorized access to hardware and stop victim VM causing denial of network service. Both of these misuses are based on the privilege escalation threat of VMs that allows the attacker to perform hypervisor privileged operations. The patterns are part of an ongoing catalog that can be used by system designers to consider security aspects when building an NFV system.

Section 2 presents misuse patterns for unauthorized access to hardware resources and stop victims' VNFs causing DoS using privilege escalation in NFV as a common threat. We end with conclusion and future work in section 3. We add an appendix at the end of the paper that describes the POSA template used for misuse patterns. We consider the POSA template as it is more suitable for describing misuse patterns [Bus96].

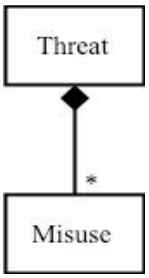

Fig. 2. Relationship between threats and misuses

2. MISUSE PATTERNS FOR UNAUTHORIZED ACCESS TO HARDWARE AND STOP VICTIMS' VNFs CAUSING DOS USING PRIVILEGE ESCALATION IN NFV

2.1 Intent

VMs are created and managed by the hypervisor, which has rights to fulfill their hypercall requests and ensures isolation among them. An attacker may escalate the privilege of his VM allowing him to perform hypervisor privileged operations that may lead to several misuses.

2.2 Context

NFV providers offer network services requested by their consumers. An attacker can be also a regular NFV consumer as long as he has a valid account, and able to run malicious applications in his VNF that sends a malicious hypercall to the hypervisor. Hypercalls are system calls used by domain VMs to request privilege operations from the hypervisor.

2.3 Problem

In order to perform the misuses, the attacker runs a malicious application in his VNF that sends a malicious hypercall to the hypervisor. The misuse can be done by exploiting the following vulnerabilities:

1. VMs can send any type of hypercalls, whether they are legitimate or malicious, to the hypervisor.
2. Hypercalls are low-level requests for basic processing and resource access, and it is difficult to differentiate between legitimate and malicious hypercalls.
3. The network service is hosted on a sharable environment, if a VNF is compromised, that may affect the other co-resident VNFs.
4. Emergence of new attacks such as return-oriented programming (ROP) that enable attackers to modify the data in the hypervisor that controls the VM privilege level.

2.4 Solution

When an attacker has a valid account, he would be able to request network services. Then, he could run malicious codes in an application running on top of his VNF that sends process and resource requests to the hypervisor as hypercalls. These hypercalls can be malicious and be able to control the hypervisor. One way attackers can escalate their VM privilege level is by exploiting the CVE-2011-1583 vulnerability that leads to control the hypervisor and escalate their VMs level [Nist]. Another way is using a malicious hypercall as the one developed by [Din12] that is applied to Xen hypervisor as we show further.

The attacker needs to know that each VM has a domain structure stored in the hypervisor; this structure contains basic information about a particular VM such as *domain_id*, which indicates an identification number for the VM, *is_privilege* that indicates whether the VM is privileged or not, and *next_in_list* which is used to link these domain structures together in ascending order by their *domain_id*. For example, the Xen hypervisor creates a parent VM (domain 0) that manages child VMs (domain 1, domain 2, etc.). So, by traversing dom0, we can know the domain

structure of dom1 by knowing the *domain_id* of dom1. The misuse starts when a malicious hypercall that has an overflow in its stack buffer is sent to the hypervisor enabling the malicious application to access the hypervisor address space, and then launches a return-oriented programming (ROP) attack. A ROP is a type of attack that modifies codes already existing in the hypervisor memory space instead of injecting new codes. The result of the ROP attack is escalating the privilege of attacker's VM by changing the *is_privilege* value from 0 to 1. If the attack has successfully been performed, the attacker can then perform hypervisor operations such as stop VMs causing denial of service, and unauthorized access to hardware resources that may lead to jeopardize the whole virtualization environment. This attack scenario is applied to cloud computing, and potentially possible in NFV in a way, for instance, to stop a VNF co-resident with a malicious VNF, or even create more malicious VNFs [Rey16].

Structure

Fig. 3 shows a class diagram for network service in NFV. The **NFV Consumer** connects to the **NFV** service through the **Portal**. The VIM is a MANO software unit that manages the NFV infrastructure including the hypervisor. The **Hypervisor** controls the **VMI Repository** that stores VM images (VMIs), creates **VMs,** assigns them to consumers' **Accounts**, and manages the **Hardware** resources (**compute**, **storage**, **network**) necessary to provide network services, as well as mediates access to them. Each VM is created using a **VMI**. Once a VM is launched, the **Hypervisor** assigns hardware resources to it. Each VM can contain many **VNFs** that are assigned to each **Consumer**. **VMs** send system calls called hypercalls to the **Hypervisor** that executes these calls in the **Hardware**.

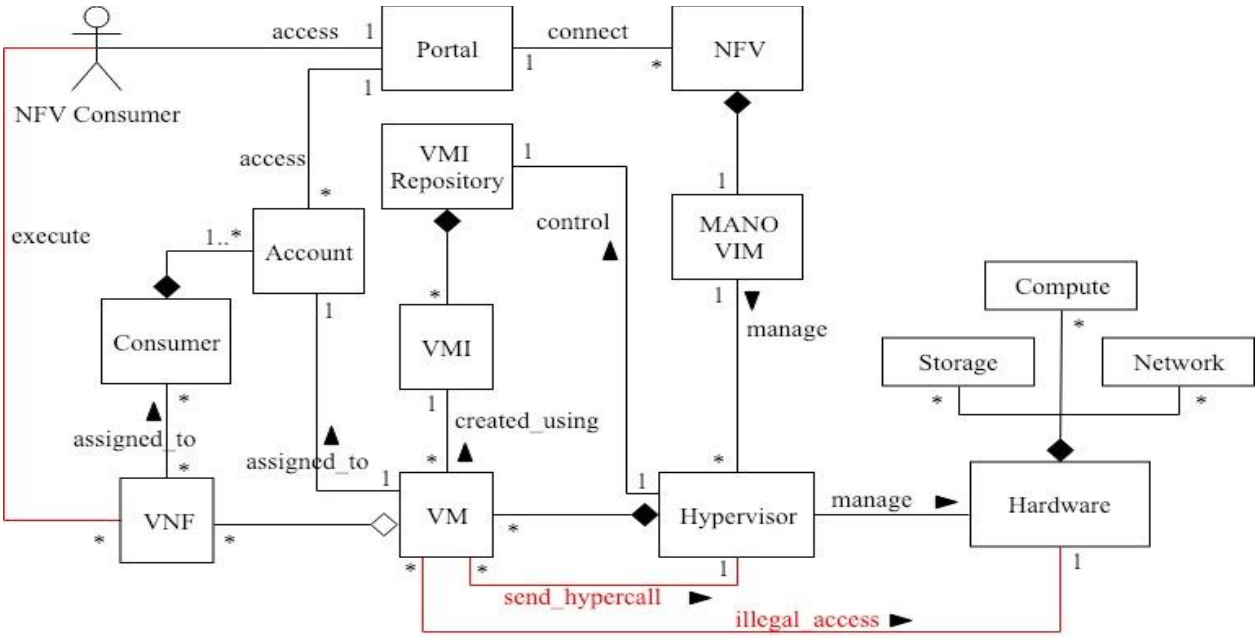

Fig. 3. Class diagram for network service in NFV.

Dynamics

UC1 (a misuse): Unauthorized access to hardware based on privilege escalation of an attacker's VM

Summary: The attacker, who is an NFV consumer, runs a malicious application in his VNF to send a malicious hypercall to the hypervisor. The application accesses the hypervisor address space and launches a ROP attack that modifies the *is_privilege* value of the attacker's VM from 0 to 1 (unprivileged to privileged). The success of this misuse allows the attacker to access hardware resources illegally and retrieve confidential data.

Actor: NFV consumer (Attacker)

Precondition: The attacker has a valid account and active network services.

Description:

1. The attacker first runs a malicious application in his VNF.
2. Using the malicious application, a malicious resource request is sent as a hypercall.
3. The malicious hypercall is forwarded to the hypervisor through the VM.
4. The hypervisor receives the malicious hypercall request and fulfills it. In this case, the malicious application accesses the hypervisor address space and launches a ROP attack. The result of the ROP is escalating the privilege of the attacker's VM by changing the *is_privilege* value from 0 to 1.
5. The hypervisor escalates the VM privilege level.
6. The attacker is notified that the VM has successfully been escalated.
7. The attacker is now able to access to hardware resources illegally.

Postcondition: The attacker controls the hypervisor, and has a direct access to the hardware resources, which he can perform malicious operations.

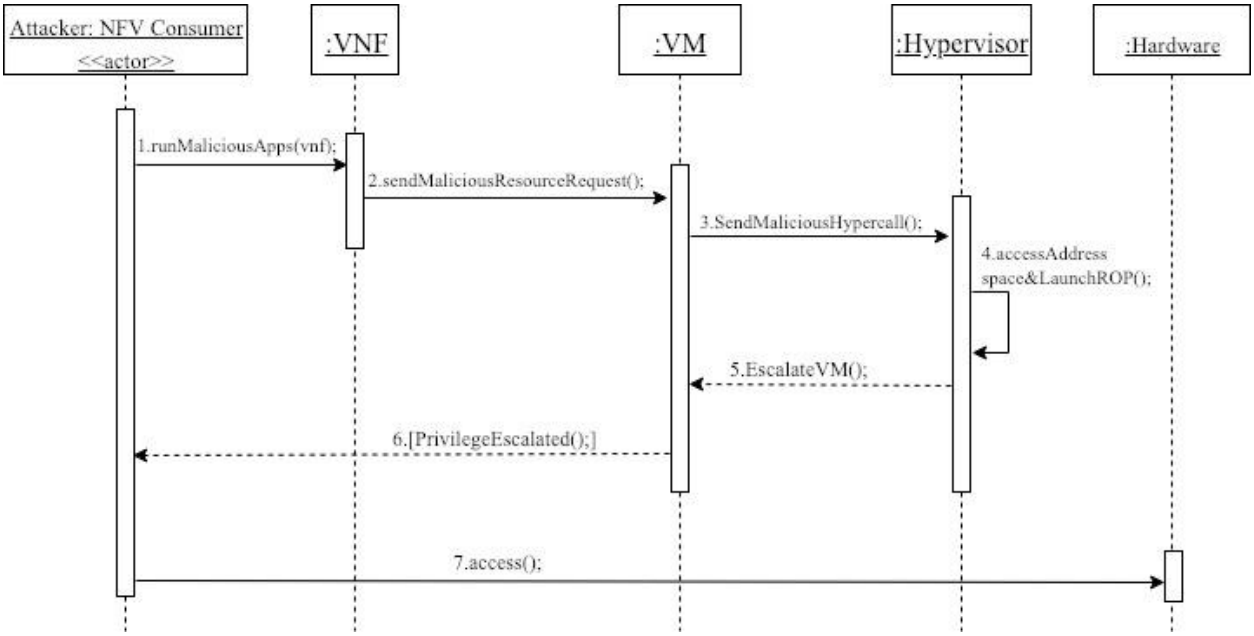

Fig. 4. Sequence diagram for unauthorized access to hardware resources.

UC2 (a misuse): Stop Victims VNFs (Denial-of-Service)

Summary: The attacker, who is an NFV consumer, stops VNFs related to other NFV consumers sharing the same resources by stopping their VMs.

Actor: NFV consumer (Attacker)

Precondition: The attacker has a valid account and active network services.

Description:

1. Do from step 1 to step 4 in UC1.
2. The result of UC1 is escalating the privilege level of the attacker's VM successfully.
3. The attacker controls the hypervisor.
4. The controlled hypervisor maliciously stops victims' VNFs by stopping their VMs and causing denial of the network service.

Postcondition: The network service has been denied to other NFV consumers.

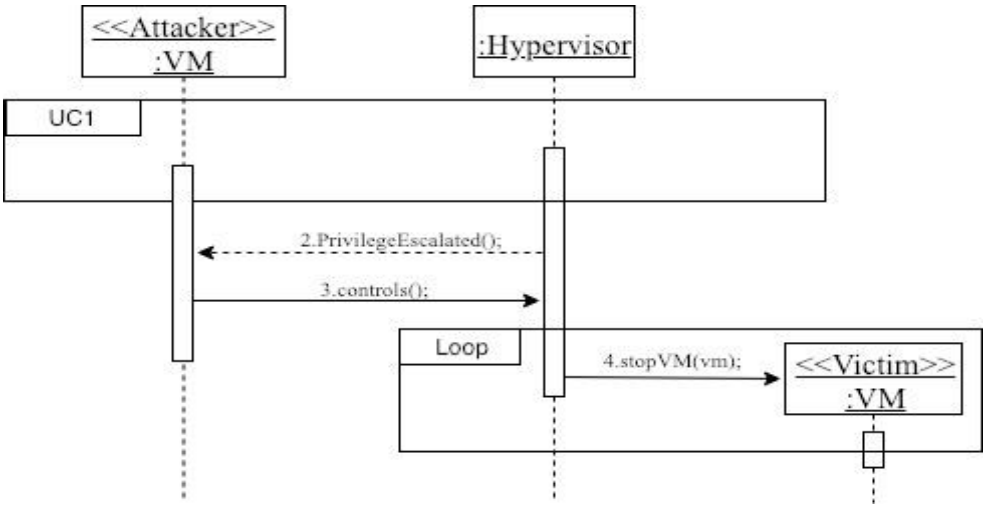

Fig. 5. Sequence diagram for stop victims' VNFs.

2.5 Consequences

A successful attack leads to the following consequences:

1. The attacker is able to compromise the system and its data as the the would be able to access hardware resources illegally.
2. The attacker can disrupt the network services completely (DoS) preventing NFV consumers from using the service.
3. Escalating the privilege of attacker's VM enables him to perform hypervisor operations such as accessing hardware resources directly and jeopardizing the system servers, creating, starting, stopping, migrating, and terminating victims VMs.
4. The attacker may be a competitor in the network service market and aims to damage to the reputation of the NFV provider as their service has been disrupted and will appear to have security issues.

Possible sources of attack failure include:

1. In virtualization environment, different versions of Xen are being used; the attacker has to know which Xen version the NFV provider is using to ensure attack succession.
2. The fields of the VM domain structure may also vary based on the configuration of the Xen hypervisor.
3. The method used in Xen to find the structure domain of attacker's VM may not work with some hypervisors, like VMware ESXi, as they don't initiate a parent privileged VM.
4. ROP attack is implemented in open source hypervisors. For closed source hypervisors it is difficult to implement such an attack as the data layout is not known.
5. Some countermeasures that can mitigate this attack are described in the following section.

2.6 Countermeasures

VM Privilege Escalation can be mitigated using the following countermeasures:

1. System bugs are patched by hypervisors' vendors, therefore, the attack can be mitigated using vendor's patch in [Xen] if the attacker instead exploits this vulnerability [Nist].
2. Using some security tools that aims to mitigate ROP attacks such as G-Free [Ona10], HyperCrop [Jia11], HyperVerify [Din13], ROPecker [Che14], as well as the hardware virtualization mechanism proposed in [Shu12].

2.7 Forensics

We can find evidence of this attack using the following actions:

1. NFV providers can keep logs of VMs hypercalls for all NFV consumers.
2. NFV providers can also keep logs of all the activities of the privileged VMs (Dom0 VMs).

2.8 Known Uses

This attack scenario is applied to cloud computing as [Din12] presented an approach to attack Xen hypervisor using ROP attack. In the context of NFV, the attack is potentially possible in a way to stop VNFs (DoS) co-resident to a malicious VNF, or even create malicious VNFs by escalating the privilege level of attacker's VM.

2.9 Related Patterns

1. NFV Virtual Machine Environment [Aln19]: describes the environment where VMs are created and managed for the purpose of NFV.
2. Pattern for Network Function Virtualization [Fer15]: presents the NFV architecture that shows how to create network services using cloud Software-as-a-Service (SaaS).
3. Reference architecture for NFV [Aln18]: shows a generic architecture for NFV and the NFVI.
4. Virtual machine operating system architecture (VMOS) [Fer13]: shows how VMs can be used to execute different operating systems with strong isolation among them.
5. Cloud ecosystem [Fer16]: shows how the NFV pattern interacts with the different parts of the ecosystem patterns.
6. A pattern for virtual machine environment [Sye16]: provides an environment in which VMs can be created and managed according to user requests.

3. CONCLUSION

NFV is a network concept that takes advantage of virtualization technology to provide reliable, scalable, isolated, and secured network services. The hypervisor plays a vital role a in this technology, as it is responsible for creating and managing the VMs used to deliver network service. Hence, it is important to ensure that the hypervisor is secure throughout its lifecycle.

There are several threats that can jeopardize the hypervisor and slow down the adoption of NFV [Alw18]. We have presented one of them as a form of misuse pattern, which is privilege escalation. We show how an attacker can exploit a vulnerability in the hypervisor to escalate the level of his VM privilege using malicious hypercalls. We demonstrated how this attack allows the attacker to perform two misuses, which are unauthorized access to hardware resources and stop the VMs of other NFV consumers sharing the same resources causing denial of network service.

We will continue to develop misuse patterns as we intend to build a catalog of misuse patterns for the virtual machine environment (VME) of NFV that can be used by system designers to build a secure and reliable NFV system. We will also use this catalog to develop security patterns that can add defenses for NFV.

AKNOWLEDGMENT

We thank our shepherd Jiwon Kim for his useful comments that helped improve this paper.

REFERENCES

[Aln18] A. K. Alnaim, A. M. Alwakeel, and E. B. Fernandez, “Towards a reference architecture for NFV,” 2018 (to be submitted).

[Aln19] A. K. Alnaim, A. M. Alwakeel, and E. B. Fernandez, “A pattern for an NFV Virtual Machine Environment,” 2019 (accepted in the 13th annual IEEE international systems conference 2019).

[Alw18] A. M. Alwakeel, A. K. Alnaim, and E. B. Fernandez, “A Survey of Network Function Virtualization Security,” in Proceedings of the IEEE SoutheastCon 2018, 2018, pp. 1–8.

[Bus96] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, “Pattern-Oriented Software Architecture”, Wiley.

[Cha18] R. Chandramouli, “Security Recommendations for Hypervisor Deployment on Servers”, National Institute of Standards and Technology (NIST), NIST Special Publication 800-125A Rev. 1, 2018.

[Che14] Y. Cheng, Z. Zhou, M. Yu, X. Ding, and R. H. Deng, “ROPecker: A Generic and Practical Approach For Defending Against ROP Attacks,” in Proceedings of the 2014 Network and Distributed System Security Symposium, 2014.

[Din12] B. Ding, Y. Wu, Y. He, S. Tian, G. Guan, G. Wu, “Return-Oriented Programming Attack on the Xen Hypervisor,” in Proceedings of the 2012 Seventh International Conference on Availability, Reliability and Security, 2012, pp. 479–484.

[Din13] B. Ding, Y. He, Y. Wu, and Y. Lin, “HyperVerify: A VM-assisted Architecture for Monitoring Hypervisor Non-control Data,” in Proceedings of the 2013 IEEE Seventh International Conference on Software Security and Reliability Companion, 2013, pp. 26–34.

[Ets14] ETSI, “GS NFV 002 - V1.2.1 - Network Functions Virtualisation (NFV); Architectural Framework,” 2014, https://www.etsi.org/deliver/etsi_gs/NFV/001_099/002/01.02.01_60/gs_NFV002v010201p.pdf

[Ets15] ETSI, “GS NFV-INF 004 - V1.1.1 - Network Functions Virtualisation (NFV); Infrastructure; Hypervisor Domain,” 2015, https://www.etsi.org/deliver/etsi_gs/NFV-INF/001_099/004/01.01.01_60/gs_nfv-inf004v010101p.pdf

[Fer13] E. B. Fernandez, “Security patterns in practice: Building secure architectures using software patterns”, Wiley Series on Software Design Patterns, 2013.

[Fer15] E. B. Fernandez and B. Hamid, “A pattern for network functions virtualization,” in Proceedings of the 20th European Conference on Pattern Languages of Programs, 2015, p. 47.

[Fer16] E. B. Fernandez, N. Yoshioka, H. Washizaki, and M. H. Syed, “Modeling and security in cloud ecosystems,” Future Internet, 2016, 8(2), 13; doi:10.3390/fi8020013.

[Jia11] J. Jiang, X. Jia, D. Feng, S. Zhang, and P. Liu, “HyperCrop: A Hypervisor-Based Countermeasure for Return Oriented Programming,” Springer, Berlin, Heidelberg, 2011, pp. 360–373.

[Nist] NIST-National Vulnerability Database. CVE-201101583 Multiple Integer Overflows. http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2011-1583

[Ona10] K. Onarlioglu, L. Bilge, A. Lanzi, D. Balzarotti, and E. Kirda, “G-Free,” in Proceedings of the 26th Annual Computer Security Applications Conference on - ACSAC ’10, 2010, p. 49.

[Rey16] F. Reynaud, F.-X. Aguessy, O. Bettan, M. Bouet, and V. Conan, “Attacks against network functions virtualization and software-defined networking: state-of-the-art,” in Proceedings of the NetSoft Conference and Workshops (NetSoft). IEEE, 2016, pp. 471–476.

[Shu12] T. Shuo, H. Yeping, and D. Baozeng, “Prevent Kernel Return-Oriented Programming Attacks Using Hardware Virtualization,” Springer, Berlin, Heidelberg, 2012, pp. 289–300.

[Sye16] M. H. Syed and E. B. Fernandez, “A Pattern for a Virtual Machine Environment Virtual Machine Environment (VME),” in Proceedings of the 23rd European Conference on Pattern Languages of Programs. The Hillside Group, 2016.

[Xen] Xen Security Advisory. CVE-2011-1583. http://old-list-archives.xenproject.org/archives/html/xen-devel/2011-05/msg00483.html

Appendix

A Template for Misuse Patterns

In this section, we show the template used in this paper to describe the misuse pattern.

Name

The name of the misuse pattern should correspond to the generic name given to the specific type of threat in standard attack repositories.

Intent

A short description of the intended purpose of the pattern (what problem it solves for an attacker).

Context

It describes the generic environment including the conditions under which the attack may occur. This may include minimal defenses present in the system as well as standard vulnerabilities of the system.

Problem

From an attacker's perspective, the problem is how to find a way to attack the system. The forces indicate what factors may be required in order to accomplish the attack and in what way.

Solution

This section describes the solution of the attacker's problem, i.e., how the attack can reach its objectives and the expected results of the attack. UML class diagrams show the system units involved in the attack. Sequence or collaboration diagrams show the exchange of messages needed to accomplish the attack.

Structure (where to look for evidence, targets)

The pattern should indicate in the UML class diagram the role of all components that are involved in the attack. From a forensic viewpoint, this section describes what information can be obtained at each stage tracing back the attack and what can be deduced from this data.

Dynamics

The pattern should include sequence diagrams to show the exchange of messages needed to accomplish the attack.

Consequences for the attacker

Discusses the benefits and drawbacks of a threat pattern from the attacker's viewpoint. The enumeration includes good and bad aspects and should match the forces.

Countermeasures

Describes the security measures necessary in order to stop, mitigate, or trace this type of attack. This implies an enumeration of which security patterns or other practical measures are effective against this attack.

Forensics

It describes what information can be obtained at each stage tracing back the attack. It also may indicates what additional information should be collected at the involved units to improve forensic analysis

Known uses

List of the security incidents where the attack has already occurred.

Related Patterns

Discusses other misuse patterns with different objectives but performed in a similar way or with similar objectives but performed in a different way.