Securing Virtual Network Function (VNF) in Telco Cloud

Bharathkumar Ravichandran

Electronics and Communication Engineering, National Institute of Technology, Puducherry, India
E-mail: rbkv2.0@gmail.com

Received 07 December 2019; Accepted 31 December 2019;
Publication 18 July 2020

Abstract

In the fifth generation mobile communication architecture *(5G)*, network functions which traditionally existed as discrete hardware entities based on custom architectures, are replaced with dynamic, scalable Virtual Network Functions (VNF) that run on general purpose (x86) cloud computing platforms, under the paradigm Network Function Virtualization (NFV). The shift towards a virtualized infrastructure poses its own set of security challenges that need to be addressed. One such challenge that we seek to address in this paper is providing integrity, authenticity and confidentiality protection for VNFs.

Keywords: VNF, orchestrator, virtualization, integrity, authenticity, confidentiality, virtualized infrastructure manager.

1 Introduction

With the advent of private cloud infrastructures and virtualized cloud environments, telecommunication vendors and operators alike are taking to the

Journal of ICT, Vol. 8_3, 235–246. *River Publishers*
doi: 10.13052/jicts2245-800X.834

idea of cloud computing and virtualization. There is a marked shift towards deploying network services on standard x86 servers rather than relying on proprietary hardware architectures. Network Functions that used to exist as dedicated hardware devices such as firewalls and load balancers are deployed as virtualized network functions on x86 servers.

In the context of security, cloud computing presents its own set of shortcomings in addition to those inherent to general computing systems.

Public clouds do not evoke the level of trust in customers as on-premise compute resources do, in the sense that the machines are not physically under their control - the customers are forced to implicitly trust the cloud service provider.

The other approach to leverage a scalable virtualization platform is to establish a private cloud infrastructure. The key security requirements that TSPs (Telecom Service Providers) look for in a private cloud infrastructure that will be hosting mission critical network elements are Confidentiality, Integrity and Availability - also known as the *CIA* triad of Information Security.

Owing to its scalability, NFV (Network Function Virtualization) is being leveraged by TSPs to deploy 5G in order to combat poor resource utilization, tight coupling with proprietary hardware and lack of flexible control interfaces. By implementing a virtualized architecture as the backbone for their network, TSPs can reduce their capital and operational expenditure without having to increase subscription costs as a result of the massive scalability and flexibility provided by the architecture.

In a private cloud environment that deploys, scales and manages several Virtualized Network Functions (VNFs), the integrity, authenticity and confidentiality of the VNFs have to be ensured.

This paper focuses on the handling of VNF packages in a private cloud environment and the assurance of their integrity, authenticity and confidentiality at rest in the VNF Manager.

The rest of this paper is structured as follows. Section 1.2 discusses relevant work in the field of VNF security. In Section 1.3, we discuss the elements of the infrastructure and the implementation of the test-bed setup. In Section 1.4, we explain the process of providing integrity, authenticity and confidentiality protections to VNFs. In Section 1.5, we summarize the findings derived from our experimental setup.

2 Related Work

In this section we take a look at the existing work that sets the direction for this paper.

An experimental setup described in [4] provides assurance of VNF (Virtual Network Function) integrity in a private cloud infrastructure utilizing OpenStack as the VIM (Virtualized Infrastructure Manager). It comprises a dedicated Security Orchestrator (Sec-O) to verify the integrity of the package by signing the VNFs. This study forms the basis for our experimental setup, wherein we try to incorporate the integrity and confidentiality verification mechanisms in OSM (OpenSource MANO) Several studies [3] [7] [1] [2] have discussed the inherent weaknesses of OpenStack and emphasized that the security of private cloud infrastructures is rather dependent on the implementation. In the context of virtualized environments based on NFV, ensuring security [5] [6] entails a deeper understanding of the underlying infrastructure.

The attack vector we are focusing in this paper pertains to malicious tampering and injection of backdoors in the VNF packages.

This paper aims to establish a reference private cloud environment with OpenStack as the Virtualized Infrastructure Manager, while OSM represents the Orchestrator and VNFM to provide integrity, authenticity and confidentiality protection to VNFs.

3 Test-Bed Setup

In this section, we discuss the components of the infrastructure and explain the implementation of the test-bed setup.

3.1 Components

3.1.1 Virtualized infrastructure manager

OpenStack aims to provide a flexible solution for both public and private clouds of any size, and in this regard two basic requirements are considered: clouds must be simple to implement and massively scalable. To meet these principles, OpenStack is divided into different components that work together. This integration is achieved through application programming interfaces (APIs) offered and consumed by each service.

In our virtualized cloud test-bed setup, OpenStack acts as the Virtualized Infrastructure Manager serving as the backbone of the architecture.

3.1.2 Orchestrator and VNFM

Our test-bed leverages OSM as both the Orchestrator and VNFM (VNF Manager). OSM consists of several modules, of which NBI (unified northbound interface) and LCM (lifecycle manager) are of special interest. The NBI is reponsible for the onboarding of the VNF to the VIM, while the LCM handles the instantiation of the VNF in the virtualized cloud environment through the VIM.

3.1.3 Secure Storage

We use a secure storage backend to store the keys used for signing and verification, and encryption and decryption. The keys are retrieved from the secure storage backend by the Orchestrator during onboarding and instantiation as required via a REST API.

3.2 Implementation

Our test-bed is basically a three-node deployment of OpenStack comprising one controller node and two compute nodes. The controller node and one of the compute nodes are based on the Intel Core i5 4570 processor having 8 GB of RAM each. The other compute node is a Dell server with Dual Intel Xeon E5 2697 processors and 32GB of RAM. We used the OpenStack Rocky release for this test-bed. Dedicated internal and external networks were created in Neutron, OpenStack's networking service, to realize this proof-of-concept environment. OSM is run on a dedicated node based on an 8-core Intel Xeon processor and 16 GB of RAM. Hashicorp vault is run as a virtual machine (VM) in the same node running OSM.

The complete topology of the setup is in Figure 1

4 Integrity, Authenticity and Confidentiality Protection for VNF

In this section, we elaborate on the process of providing integrity, authenticity and confidentiality protection to VNF.

4.1 Onboarding and Instantiation of VNF

The following are the procedures to onboard a VNF in a virtualized cloud environment as described in Figure 2:

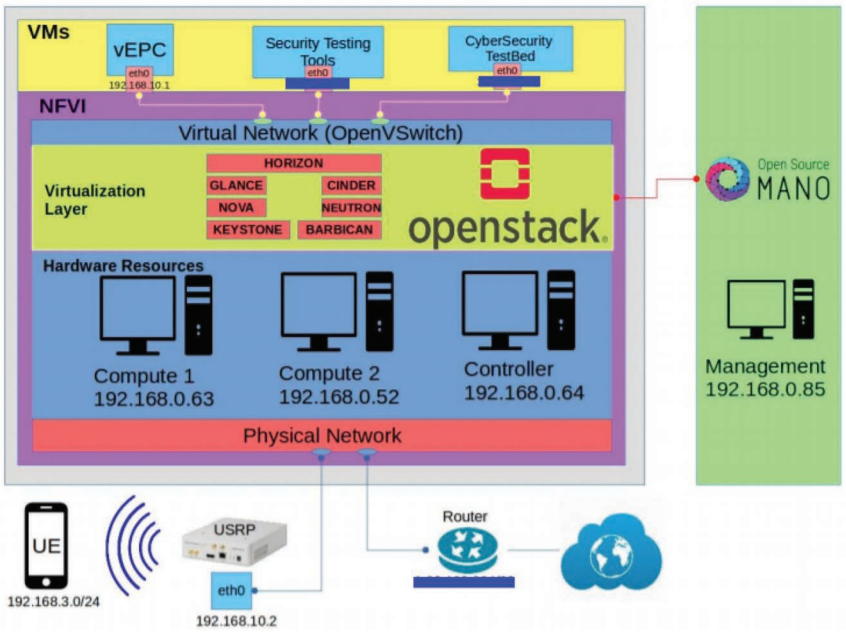

Figure 1 NFV Setup.

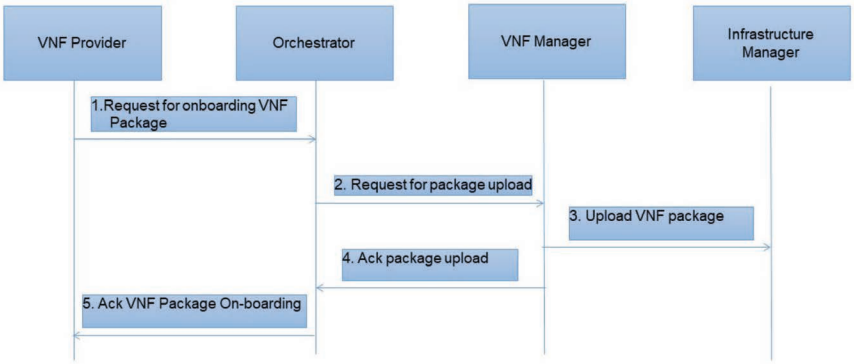

Figure 2 VNF Onboarding.

1. The VNF Provider initiates the onboarding of the VNF in the virtualized cloud environment by sending the VNF package to the Orchestrator
2. The Orchestrator checks the VNF package for presence of necessary configuration files and requests the VNF Manager to upload the VNF package

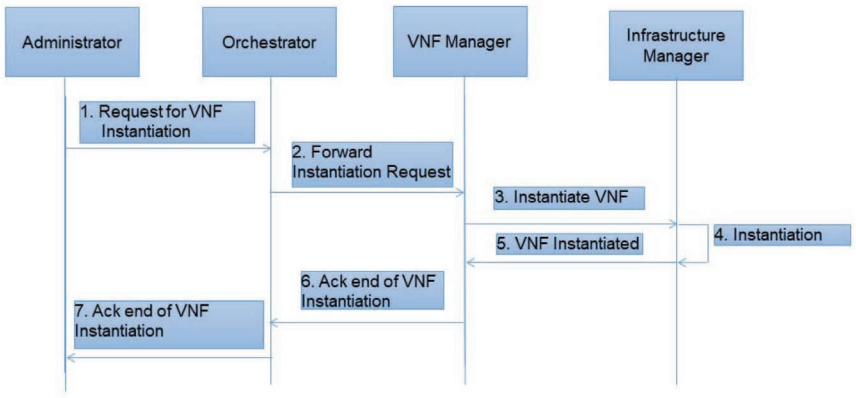

Figure 3 VNF Instantiation.

3. The VNF Manager then uploads the VNF package to the container repository in the VIM.
4. The VIM acknowledges the successful uploading of the package.
5. The VNF Manager acknowledges the upload of the VNF package to the Orchestrator which in turn intimates the successful onboarding to the VNF provider.

The following are the procedures to instantiate a VNF in a virtualized cloud environment as described in Figure 3:

1. The administrator requests the Orchestrator for VNF instantiation.
2. The Orchestrator forwards the request to the VNF Manager.
3. The VNF Manager instantiates the VNF through the Infrastructure manager
4. Once the VNF is instantiated the Infrastructure manager intimates the VNF Manager, which in turn notifies the Administrator via the Orchestrator

4.2 Signing and Verification of VNF

The process of onboarding a VNF with integrity and authenticity protection in a virtualized cloud environment is as described in Figure 4:

1. The VNF Provider initates the onboarding of the VNF in the virtualized cloud environment by sending the VNF package to the Orchestrator.

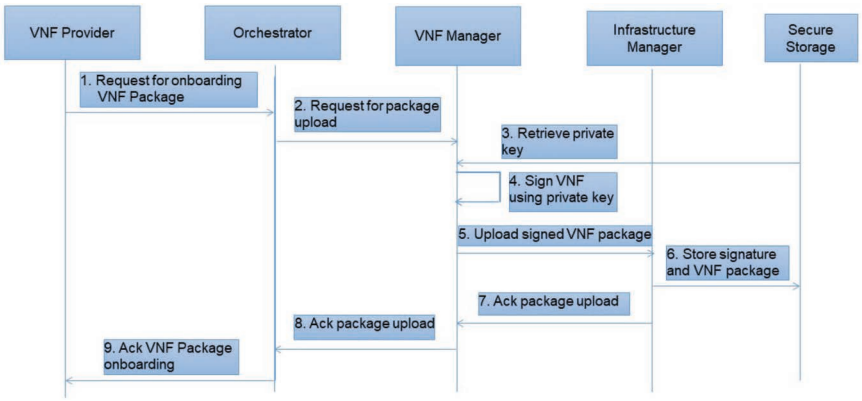

Figure 4 VNF onboarding with signing.

2. The Orchestrator checks the VNF package for presence of necessary configuration files and requests the VNF Manager to upload the VNF package.
3. The VNF Manager retrieves the private key from the secure storage and signs the VNF package.
4. The VNF Manager then uploads the signature and the VNF package to the VIM.
5. The VIM then stores the VNF package and the signature in the secure storage and sends acknowledgement to the VNF Manager.
6. The VNF Manager acknowledges upload of the VNF package to the Orchestrator which in turn intimates successful onboarding of the VNF package to the VNF provider.

The process of instantiating a VNF with integrity and authentication checks in a virtualized cloud environment is as described in Figure 5:

1. The administrator requests the Orchestrator for VNF instantiation.
2. The Orchestrator forwards the request to the VNF Manager.
3. The VNF Manager retrieves the public key from the secure storage and verifies the signature of the VNF package.
4. If the signature verification succeeds, the VNF Manager sends the instantiation request to the VIM.
5. The VIM instantiates the VNF and sends an acknowledgement to the VNF Manager.
6. The VNF Manager acknowledges the successful instantiation to the administrator via the Orchestrator.

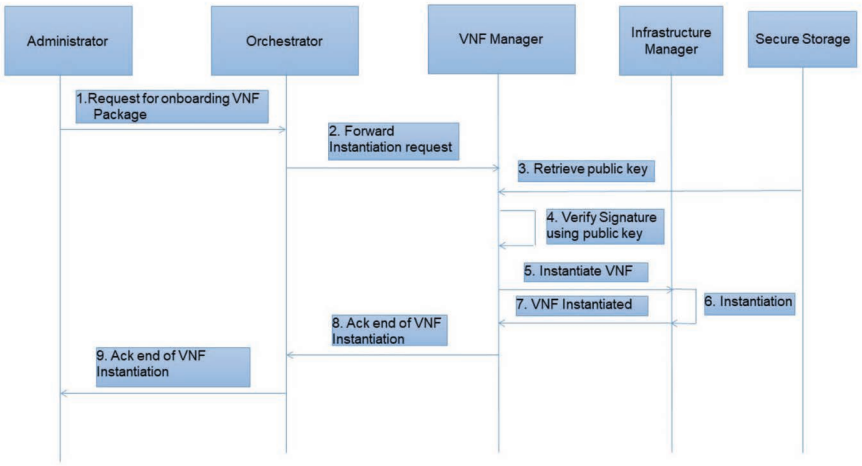

Figure 5 VNF instantiation with signature verification.

4.3 Encryption and Decryption of VNF

The process of onboarding a VNF with confidentiality protection in a virtualized cloud environment is as described in Figure 6:

1. The VNF provider initiates the onboarding of the VNF in the virtualized cloud environment by sending the VNF package to the Orchestrator.
2. The Orchestrator checks the VNF package for presence of necessary configuration files and requests the VNF Manager to upload the VNF package.
3. The VNF Manager retrieves the public key from the secure storage and encrypts the VNF package
4. The VNF Manager then uploads the encrypted VNF package to the VIM which then stores it in the secure storage and sends acknowledgement back to the VNF Manager.
5. The VNF Manager acknowledges the upload of the VNF package to the VNF Provider via the Orchestrator

The process of instantiating a VNF with confidentiality protection in a virtualized cloud environment is as described in Figure 7:

1. The administrator requests the Orchestrator for VNF instantiation.
2. The Orchestrator forwards the request to the VNF Manager.
3. The VNF Manager retrieves the private key from the secure storage and decrypts the VNF package.

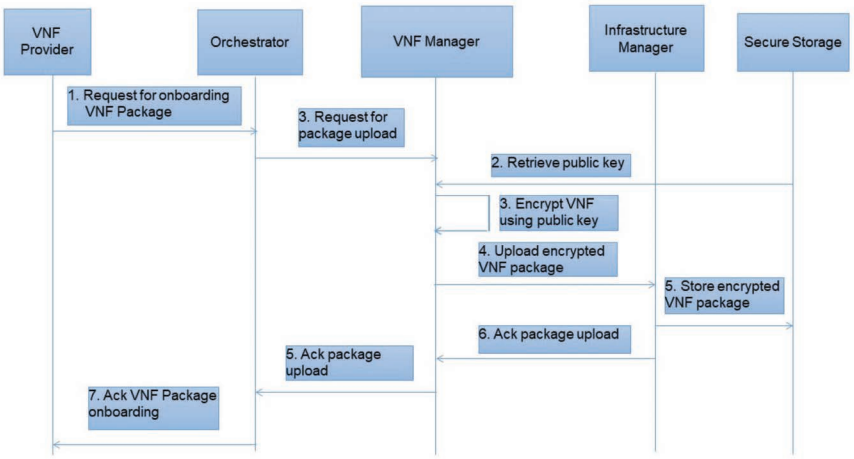

Figure 6 VNF onboarding with encryption.

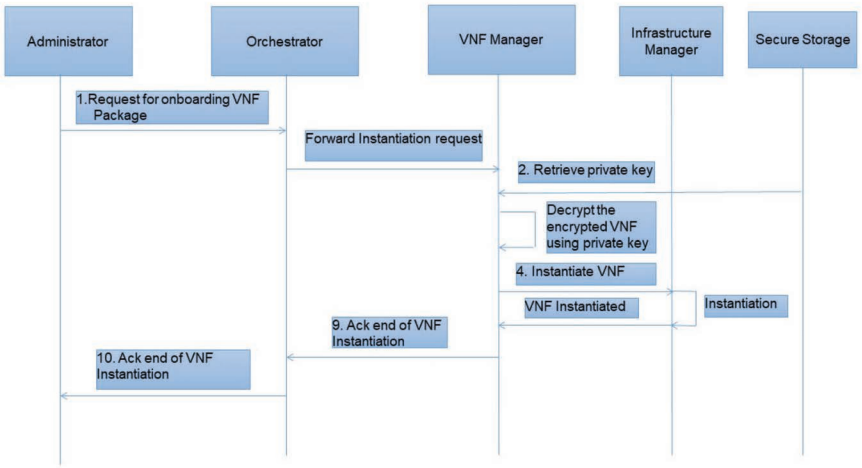

Figure 7 VNF instantiation with decryption.

4. The decrypted package is now sent to the VIM to be instantiated.
5. The VIM instantiates the VNF and sends an acknowledgement to the VNF Manger.
6. The VNF Manager acknowledges the successful instantiation to the administrator via the Orchestrator.

Having defined the process, we proceeded to implement those protections in OSM by modifying the onboarding and instantiation process to ensure the integrity, authenticity and confidentiality of the VNF packages at rest.

With integrity and authenticity protection, when a new VNF package is uploaded, the NBI module now calculates a SHA256 hash of the uploaded package and signs it with a 2048-bit RSA key retrieved from the secure storage backend. The package along with the digital signature is now stored in the catalog. During instantiation, a new hash of the stored package is calculated and verified against the hash in the signature after decryption with the public key retrieved from the secure storage backend. The verification of the signature ensures that the integrity and authenticity of the VNF is maintained at rest from onboarding to instantiation.

With confidentiality protection, when a new VNF package is uploaded, the NBI module retrieves the public key derived from a 2048-bit RSA key from the secure storage backend, encrypts the VNF package and stores the encrypted package in the catalog. During instantiation, the LCM module retrieves the private key from the secure storage backend and attempts to decrypt the encrypted VNF package. On successful decryption, the VNF is then instantiated by the VIM.

5 Conclusions

In this paper, we discussed a potential attack vector in virtualized cloud environments and the need for providing integrity, authenticity and confidentiality protection to VNFs. In this vein, we devised an experimental virtualized cloud test-bed consisting of OpenStack and OSM and demostrated digital signature verification checks as well as encryption and decryption for VNFs. The storage of the signing keys is handled by a secure storage backend and the keys are retrieved by the VNF Manager on demand using tokens for authorized users.

Integrity and Authenticity protections are provided by signing the VNFs during onboarding and verifying them before instantiation. This prevents malicious tampering of those packages. Confidentiality protection is provided by encrypting the VNFs during onboarding and decrypting them before instantiation. This prevents unauthorized access to the VNF package stored in the catalog of the VNF Manager.

References

[1] Hala Albaroodi, Selvakumar Manickam, and Parminder Singh. Critical review of openstack security: Issues and weaknesses. *Journal of Computer Science*, 2013.

[2] Marco Anisetti, Claudio A. Ardagna, Ernesto Damiani, and Filippo Gaudenzi. A Security Benchmark for OpenStack. In *IEEE International Conference on Cloud Computing, CLOUD*, 2017.

[3] Ivano Alessandro Elia, Nuno Antunes, Nuno Laranjeiro, and Marco Vieira. An Analysis of OpenStack Vulnerabilities. In *Proceedings – 2017 13th European Dependable Computing Conference, EDCC 2017*, 2017.

[4] Shankar Lal, Aapo Kalliola, Ian Oliver, Kimmo Ahola, and Tarik Taleb. Securing VNF communication in NFVI. In *2017 IEEE Conference on Standards for Communications and Networking, CSCN 2017*, 2017.

[5] Shankar Lal, Sowmya Ravidas, Ian Oliver, and Tarik Taleb. Assuring virtual network function image integrity and host sealing in Telco cloue. In *IEEE International Conference on Communications*, 2017.

[6] Sowmya Ravidas, Shankar Lal, Ian Oliver, and Leo Hippelainen. Incorporating trust in NFV: Addressing the challenges. In *Proceedings of the 2017 20th Conference on Innovations in Clouds, Internet and Networks, ICIN 2017*, 2017.

[7] Sasko Ristov, Marjan Gusev, and Aleksandar Donevski. Security vulnerability assessment of OpenStack cloud. In *Proceedings – 6th International Conference on Computational Intelligence, Communication Systems and Networks, CICSyN 2014*, 2014.

Biography

Bharathkumar Ravichandran received B.Tech in Electronics and Communication Engineering from National Institute of Technology, Puducherry in 2019. He was an intern working on NFV Security at NEC India for 1.5 years. His areas of interest include NFV, 5G and container security.