Attestation of Trusted and Reliable Service Function Chains in the ETSI-NFV Framework

Antonio Suriano[1,3], Domenico Striccoli[1,3], Giuseppe Piro[1,3], Raffele Bolla[2,3], Gennaro Boggia[1,3]
[1]*Department of Electrical and Information Engineering (DEI)*, Politecnico di Bari, Bari, Italy
Email: {antonio.suriano, domenico.striccoli, giuseppe.piro, gennaro.boggia}@poliba.it
[2]*Department of Naval, Electrical, Electronical and Telecommunications Engineering (DITEN)*,
Università di Genova, Genova, Italy
Email: {raffaele.bolla@unige.it}
[3]*Consorzio Nazionale Interuniversitario per le Telecomunicazioni (CNIT)*

***Abstract*—The new generation of digital services are natively conceived as an ordered set of Virtual Network Functions, deployed across boundaries and organizations. In this context, security threats, variable network conditions, computational and memory capabilities and software vulnerabilities may significantly weaken the whole service chain, thus making very difficult to combat the newest kinds of attacks. It is thus extremely important to conceive a flexible (and standard-compliant) framework able to attest the trustworthiness and the reliability of each single function of a Service Function Chain. At the time of this writing, and to the best of authors knowledge, the scientific literature addressed all of these problems almost separately. To bridge this gap, this paper proposes a novel methodology, properly tailored within the ETSI-NFV framework. From one side, Software-Defined Controllers continuously monitor the properties and the performance indicators taken from networking domains of each single Virtual Network Function available in the architecture. From another side, a high-level orchestrator combines, on demand, the suitable Virtual Network Functions into a Service Function Chain, based on the user requests, targeted security requirements, and measured reliability levels. The paper concludes by further explaining the functionalities of the proposed architecture through a use case.**

***Index Terms*—Network Function Virtualization, Service Function Chain, Remote Attestation, Trustworthiness**

I. Introduction

In the last years, there has been an exponential growth in the users demand of new network services, favoured by the explosion of new network technologies and infrastructures that have increased the degree of pervasiveness and connectivity among the plethora of heterogeneous devices. Unfortunately, hardware infrastructures cannot easily adapt to this rapidly evolving scenario: hardware resources are limited and suffer from scalability problems, rigidity in configuration changes, and consistent deployment and maintenance costs.

An effective solution to solve these issues is virtualization, that decouples software applications from the underlying hardware, and gives rise to the Network Function Virtualization (NFV) paradigm where different Virtual Network Functions (VNFs) are defined, interconnected and executed, allocating resources to all of them to compose a Service Function Chain [1]. Each NFV is designed, deployed and managed via software, so that it can be seen as an instance of the software decoupled by the underlying hardware and running on it, thus increasing the service reliability and flexibility, reducing costs, and optimizing the lifecycle management of a service.

The main advantages of a Service Function Chain of VNFs are counterbalanced by security concerns. In virtualized environments, security threats and software vulnerabilities can easily increase and blur the attack surface, making more difficult to combat the newest kinds of attacks. In addition, the service chain should guarantee reliability requirements while providing a service to end users. To tackle these issues, a flexible and standard-compliant framework would be of great help in assessing and certifying the trustworthiness of Service Function Chains composed by VNFs and extending across multiple and heterogeneous domains, while satisfying reliability guarantees. Unfortunately, this topic has not been properly addressed yet in the current literature.

The goal of this paper is exactly to fill this gap, proposing a methodology, tailored within the ETSI-NFV framework, to verify the trustworthiness and reliability of Service Function Chains of VNFs through remote attestation, an approach that is still unexplored in the recent literature to the best of the authors knowledge. The proposed methodology relies on the continuous monitoring of the VNF properties and performance indicators by Software-Defined Controllers, that report all this state information to a high-level orchestrator. In turn, the orchestrator selects and combines the VNFs on the basis of both the trustworthiness of the VNFs bootcode and the VNFs suitability to meet service requirements in terms of networking, computational, and memory capabilities. This approach is very general and customizable according to the targeted use case: each single aspect of the designed solution could be further extended, improved, and customized before being implemented in real use cases.

The contribution of this paper can be synthesized into the following points:

- a strategy to build a Service Function Chain (SFC) of VNF based on attestation and reliability criteria;
- the proposal of a novel architecture;
- a flexible and customizable method.

The reminder of this paper is organized as follows. Section II reviews the most representative literature on Remote Attestation, Service Function Chains and NFV. Section III describes

the methodology for the remote attestation of VNF-based Service Function Chains. Section IV provides an example of the application of the proposed methodology to a real use case. Finally, Section V concludes this work.

II. Background and State of the art

In this section a review of the most recent literature on remote attestation of services, service function chains and NFV standardization efforts is carried out. In each of the three subsections that follow, the discussion of the state-of-the-art literature is preceded by the most relevant background notions on the topic the subsection refers to.

A. Reference technologies for the remote attestation

A service is said to be trusted when it is in the same operating state as it was when released by the developer. In the current scientific literature the evaluation of the trustworthiness of an application is generally based on the bootcode of the application. Specifically, a hash function (h) is applied to this piece of code; then the hash value is compared with a reference value provided by the producer. If a match occurs, the trustworthiness is verified. Several works in literature deal with remote attestation techniques. Methodologies to perform attestation for distributed systems are proposed in [2], [3]. A fine-grained attestation method is proposed in [2], while the work [3] presents a property-based remote attestation method oriented to cloud computing. Another fine-grained remote attestation architecture for containerized system (like guest Operating Systems in Virtual Machine) is presented in [4]. Domain-specific integrity reports are built to ease system and sub-system verification, and provide desirable properties such as measurement log stability and constrained disclosure for multi-domain systems. Privacy-preserving blockchains are adopted in [5], to implement remote attestation in the Internet of Vehicles (IoV) context. Different attestation schemes are analyzed in [6], that satisfy scalability requirements for their application in large networks of embedded devices. The schemes are analyzed with respect to malware and run-time attacks detection capabilities. Scalability is also the focus of the remote attestation mechanism presented in [7]. It is based on Machine Learning algorithms embedded into functions of the public cloud, which are available on "pay-per-use" basis and have more sophisticated and efficient hardware, while lightweight CPU tasks and privacy preserving verifications are carried out in the private cloud. The goal of the scheme is to preserve privacy of the applications on the efficient and less expensive hardware of the private clouds. Hardware-based attestation and isolation architectures are also presented in [8], comparing them all in terms of security properties and architectural features they offer. Another study on remote attestation techniques for embedded devices is carried out in [9], where devices are resource-constrained and have limited connectivity. A real use case (hydroelectric power plants) is also considered, evaluating the system components and their privileges related to their influence on the control task, and integrating a trusting computer architecture.

The paper [10] proposes a three-phase remote attestation protocol for distributed system. It binds the system measurements with server-signed certificates, to authenticate clients to a server; policies are enforced to make the environment more suitable according to network needs. A worm propagation detection scheme for sensor networks is proposed in [11]. It exploits rooted software remote attestation to detect infected nodes. The software-based remote attestation technique proposed in [12] guarantees the execution of trusted and correct code on a remote host, so that the control domain becomes the root of trust and provides services such as code measurement, trusted execution, and remote attestation.

Some works deal with remote attestation techniques in Internet of Things (IoT) scenarios [13]–[17]. In [13] the remote attestation technique aims at detecting hardware and software modifications of the original configuration of a device in the network through static and dynamic attestation phases. The scheme in [14] is a many-to-one scheme, i.e. many verifiers attest a single untrusted node (the "prover"), eliminating the single point of failure (the single verifier). The scheme, suitable for IoT swarms, aims at quickly reporting compromised nodes, to reduce the run-time of attestation. A remote collective and scalable attestation scheme for IoT networks is presented in [15]. The scheme is based on a mechanism of key exchange and Proofs-of-non-Absence, that verifies that nodes do not disconnect from the network. The protocol proposed in [16], called RADIS, is based on a control-flow attestation technique that detects the IoT services that perform an unexpected operation because of their interactions with a malicious remote service. A definition of the required security properties for distributed IoT services is also presented, together with an adversary model whose goal is to compromise the correct execution of distributed IoT services. A study of collective Remote Attestation schemes for large networks of embedded devices (like IoT networks) is carried out in [17]. A formal model of collective remote attestation for a use case is built, that encompasses desirable efficiency, soundness, and security notions. Starting from this point, a collective remote attestation protocol adhering to the model requirements is built and implemented.

B. Provisioning of security services for Service Function Chains

To provide a service requested by a user, there is often the need to offer a series of functions that have to be executed sequentially in a predetermined order. This sequence is defined as a Service Function Chain. Few works can be found in the recent literature that deal with Service Function Chains. In [18] a set of requirements is proposed to solve the problem of isolating and encrypting Service Function Chains. In [19], the management of a Service Function Chain is integrated in the Software Defined Network controller, in order to handle large-scale data center networks.

Works [20]–[23] study some problems related to this topic. A design method is proposed in [20] to increase the reliability of a service chain while reducing the resource consumption.

The survey [21] focuses on the motivations and efforts in implementing Service Function Chains. In [22], an anomaly detection method is proposed for a service chain of VNFs, in order to ensure their correct placement. Finally, a study is performed in [23] on the Service Function Chain composition and mapping, focusing on resource optimization issues. In [24], a formal trust chain model is proposed, which includes important rules of the trust chain. Also the work [25] presents a diagrammatic approach to model rules of trust using an extended version of concept diagrams.

C. Proposals related to ETSI-NFV specifications

The NFV framework decouples a network function from the underlying dedicated hardware. It has been standardized by the NFV ETSI Industry Specification Group into specifications that describe the main functionalities of the NFV framework architecture, the design principles of the related infrastructure, and the VNFs that are the virtualized functions making part of the framework [26]. As such, the NFV framework is composed by a number of VNFs that are properly managed and orchestrated.

Some works can be found in literature that analyze security aspects of VNFs. A survey aiming to analyze NFV from a security perspective is found in [27]. An analysis of security threats is carried out, conducting comparative studies on security mechanisms applied in traditional scenarios. Some NFV use cases are analyzed in [28], pointing out their threats and misuse activities. Starting from this analysis, some security policies are proposed to improve system security. The problem of placing VNFs on the NFV infrastructure and establishing the paths between them is tackled in [29]. This work proposes to solve this problem through Linear Programming techniques with trust, placement, flow and capacity constraints. Trustworthiness in NFVs is the main focus of [30], [31]. In [30] the challenges in incorporating trust in NFV are discussed, also evaluating some requirements for establishing the trust. The design of a centralized monitoring and reporting solution is presented in [31] to assess the trustworthiness of a NFV infrastructure, tailored for the Security-as-a-Service paradigm. In [32], different methods are discussed to protect the context of the headers of packets exchanged among service function, and an experimental study is conducted to evaluate the performance of the solutions. Papers [33], [34] try to solve the problem of composition and reconfiguration of a Service Function Chain in NFV architectures.

D. Final Considerations

The analysis of the current state of the art clearly demonstrates that security of virtualized services and service chains is of paramount importance. Nevertheless, as reported in Table I, there is still the lack of a general framework that jointly addresses security threats, variable network conditions and computational capabilities, and software vulnerability, through a standard-compliant methodology.

III. The proposed architecture

This Section presents the proposed architecture for the attestation of trusted and reliable Service Function Chains, based on the ETSI-NFV framework.

Since the proposed procedure is compliant with the ETSI-NFV standard, among the involved entities there are some components typical of the NFV framework [26]. Overall, the entities involved in the process are:

- **VNF**: it is the entity dedicated to provide a certain function;
- **Orchestrator**: it is the logical entity in charge of managing the overall architecture. It handles the VNF placement and deployment, the composition of the service chain and the attestation procedure. For the attestation process it integrates the Remote Verifier and the Trust Assessor;
- **Attestation Agent**: it is installed on the target platform, listens and receives the attestation requests;
- **Remote Verifier**: it is in charge to validate the integrity measurements of the target platform;
- **Trust Assessor**: it is the component that exploits the result of the integrity validation to determine further actions;
- **Software Defined Network controller**: it is the logical entity that monitors and configures the network. It receives the state information and passes it to the orchestrator;
- **Trusted Platform Module**: it is the cryptographic coprocessor (that acts as root of trust for measurements). It contains the Platform Configuration Registers storing the hash (or digest) of the bootcode of the function [31], [35]. Hashes are used during the attestation procedure to decide on the trustworthiness of the service chain; as known, hash functions are mathematically built so that the probability that two different messages generate the same digest is about zero. Since the bootcode is characteristic of the VNF and does not depend on data it generates and manages, the hash of a VNF guarantees its originality, and that it has not been altered during its lifecycle operation in the service chain;
- **Database**: it is the physical place where the reference information of the VNFs is stored. It is supposed to be secured and tamper-resistant.

The reference scenario envisaged in this work is depicted in Figure 1.

There are different servers, each one managing a cloud environment that runs some VNFs. The same VNF can be hosted by one or more servers, also belonging to different domains or organizations, for redundancy and load balancing. Indeed, if a VNF is not accessible, another one performing the same function can be chosen in replacement of the first one. On each server, there is a Trusted Platform Module installed, to perform the cryptographic operations needed to create the integrity measurement report. Additionally, on each cloud environment several Attestation Agents are installed, one for each VNF. Finally, on the orchestrator, the Trust Assessor

TABLE I
AN OVERVIEW OF RELATED WORKS

PAPER	Definition of Attestation	Attestation Mehodology	Definition of Secure Service	NFV Attestation	Service Function Chain	IoT	Distributed Systems	Formalization, Discussion of Concepts
[3]	✓	✓					✓	
[2]	✓	✓					✓	
[4]	✓	✓					✓	✓
[5]	✓	✓				✓		
[6]	✓	✓					✓	✓
[7]	✓	✓					✓	
[8]	✓	✓	✓					
[9]	✓	✓				✓		
[10]	✓	✓					✓	
[11]	✓	✓				✓		
[12]	✓	✓	✓				✓	
[13]	✓	✓				✓		
[14]	✓	✓				✓	✓	
[15]	✓	✓				✓		
[16]	✓	✓	✓			✓	✓	
[17]	✓	✓				✓		✓
[18]			✓		✓			✓
[19]					✓			
[20]					✓			
[21]					✓			
[22]					✓			
[23]					✓			
[24]	✓							✓
[25]	✓	✓					✓	✓
[26]	✓	✓	✓	✓				
[27]	✓	✓		✓				✓
[28]				✓				✓
[29]			✓		✓			
[30]	✓	✓		✓			✓	✓
[31]	✓	✓		✓			✓	
[32]					✓			
[33]					✓			
[34]					✓			
Our Work	✓	✓	✓	✓	✓	✓	✓	✓

and the Remote Verifier are installed, to manage the requests and handle the attestation procedures.

In order to perform the service attestation, three different procedures are required: *Installation*, *Reporting* and *Attestation*.

A. The Installation procedure

The Installation procedure is performed each time a new VNF is installed on the server. During this phase, after the VNF installation, all the reference parameters of its state $S^{(init)}$ are distributed and stored in the database. The following set of parameters is defined, to evaluate the trustworthiness and the reliability of a VNF:

- $h^{(init)}$: it represents the hash of the bootcode;
- $b^{(init)}$: it represents the bandwidth consumption;
- $c^{(init)}$: it represents the CPU usage;
- $m^{(init)}$: it represents the memory footprint.

The initial state $S^{(init)}$ of a VNF is thus defined as:

$$S^{(init)} = \{h^{(init)}, b^{(init)}, c^{(init)}, m^{(init)}\} \quad (1)$$

In the Installation procedure the orchestrator first sends an installation message to the Attestation Agent of the VNF. In response, the Attestation Agent sends its initial state $S^{(init)}$ as in (1). In the last step, the orchestrator stores this information in the database. The sequence diagram of this procedure is reported in Fig. 2.

B. The Reporting procedure

In the Reporting procedure, the Attestation Agent of each VNF updates the state information to the Software Defined Network controller and the orchestrator. This procedure should be performed periodically, so that the Attestation Agents can update the state information to notify changes in the reliability requirements of the VNFs they refer to. The servers the VNFs run onto allocate dynamically resources to them, depending on both the total workload they have and their hardware characteristic. So, the impact that each VNF has on a server will be different from the impact the same VNF can have on another one, and can also vary in time. In addition, in this study it is supposed that the periodicity of the updates is long enough, with respect to the total duration of the service, to reduce the overhead due to a possible frequent changes of the elected VNF in the chain. All this given, the reference variables for each VNF are the parameters of the current state $S^{(cur)}$ of the VNF, identified as:

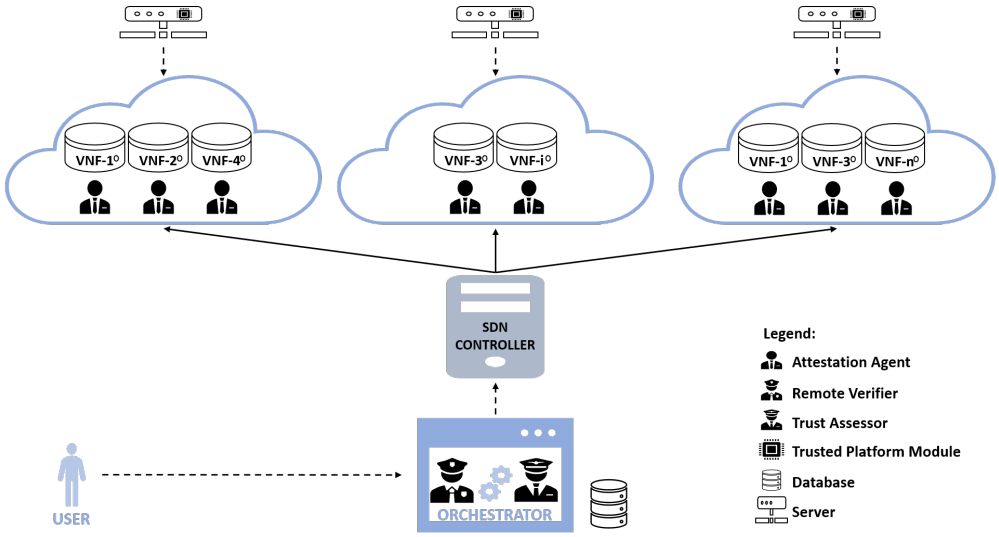

Fig. 1. Reference Scenario

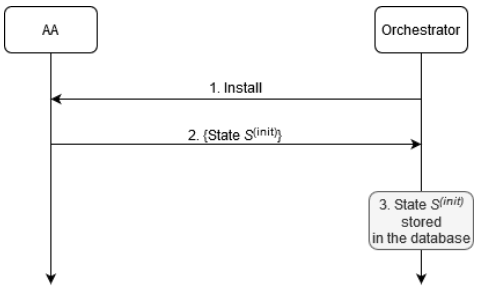

Fig. 2. Sequence Diagram of the VNF Installation procedure

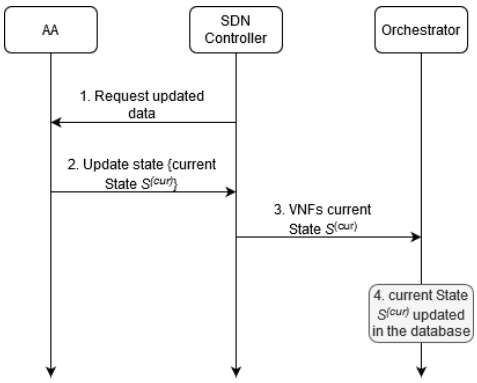

Fig. 3. Sequence Diagram of the VNF Reporting procedure

$$S^{(cur)} = \{h^{(cur)}, b^{(cur)}, c^{(cur)}, m^{(cur)}\} \quad (2)$$

In the Reporting procedure the Software Defined Network controller requests the updated data to the Attestation Agent, that sends back the updated state information $S^{(cur)}$ as in (2). The Software Defined Network controller contacts the orchestrator and sends it the state information. The orchestrator then stores these data in the database. The sequence diagram of this procedure is reported in Figure 3.

C. The Attestation procedure

The Attestation procedure is in charge of choosing the most suitable VNFs that form a service chain, according to both trustworthiness and reliability requirements. This procedure is called each time a user wants to access/use a service composed by a sequence of VNFs. The orchestrator maps the service request into a service chain, identifying the best set of VNFs, based on specific reliability levels (i.e., bandwidth conditions, CPU and memory usage), and setting up the related NFV graph that describes the various interconnections among VNFs. The targeted service chain $\mathcal{C} = \{VNF_1, VNF_2, ..., VNF_N\}$, is derived as follows.

Let $\mathcal{L}$ be the set of all the available VNFs, $N = |\mathcal{C}|$ the number of services of the chain, and l_i the number of instances of VNF performing the i-th service of the chain. Let also $VNF_{i,j}$ be the j-th instance of the VNF performing the i-th service ($1 \leq i \leq N$, $1 \leq j \leq l_i$). The orchestrator periodically receives the current state $S_{i,j}^{(cur)} = (h_{i,j}^{(cur)}, b_{i,j}^{(cur)}, c_{i,j}^{(cur)}, m_{i,j}^{(cur)})$ of $VNF_{i,j}$, and compares it with the initial trusted state $S_{i,j}^{(init)} = (h_{i,j}^{(init)}, b_{i,j}^{(init)}, c_{i,j}^{(init)}, m_{i,j}^{(init)})$ stored in the database. All the VNFs not verifying the hash check (i.e., $h_{i,j}^{(init)} \neq h_{i,j}^{(cur)}$) are considered as untrusted and discarded from the list, because this means that the VNF bootcode has been altered from its original (trusted) version. All the other VNFs passing the check are stored in a set $\mathcal{F}_i$ of trusted VNF of the i-th service, of size $|\mathcal{F}_i| \leq l_i$. For all the VNFs belonging to $\mathcal{F}_i$, the orchestrator computes the *reliability function* $\Delta F_{i,j}$, defined as:

$$\Delta F_{i,j} = w_1(b_{i,j}^{(cur)} - b_{i,j}^{(init)}) + w_2(c_{i,j}^{(init)} - c_{i,j}^{(cur)}) + \\ + w_3(m_{i,j}^{(init)} - m_{i,j}^{(cur)}) \quad (3)$$

where the weights w_1, w_2 and w_3 satisfy the constraint $w_1 + w_2 + w_3 = 1$ and can be arbitrarily chosen, depending on the impact that the specific reliability metric (bandwidth, CPU usage and memory footprint) has on the service chain. Obviously, the more reliable the $VNF_{i,j}$, the higher $\Delta F_{i,j}$; so, the best VNF for the i-th service is chosen as the one corresponding to the maximum value of the reliability function related to the same service:

$$VNF_i = VNF_{i,j_{opt}}, \textit{ where} \\ \{j_{opt} | \Delta F_{i,j_{opt}} = \max_{1 \le j \le |\mathcal{F}_i|} \Delta F_{i,j}\} \quad (4)$$

It is worthy to note that $\mathcal{F}_i$ could also be empty. In this case, the i-th service is declared as untrusted, and so the whole service chain. This attestation procedure is summarized in Algorithm 1.

Algorithm 1: Attestation procedure

Input : Number of services: N
Set of available VNFs: $\mathcal{L}$
Output: Targeted Service Chain: $\mathcal{C}$

for $i = 1 : N$ **do**
  $\mathcal{F}_i \leftarrow \emptyset$
  $l_i \leftarrow$ Number of VNFs selected for the i-th service
  for $j = 1 : l_i$ **do**
    $S_{i,j}^{(cur)} \leftarrow$ Get the current state of $VNF_{i,j} : (h_{i,j}^{(cur)}, b_{i,j}^{(cur)}, c_{i,j}^{(cur)}, m_{i,j}^{(cur)})$;
    if $h_{i,j}^{(cur)} = h_{i,j}^{(init)}$ **then**
      $\mathcal{F}_i \leftarrow \mathcal{F}_i \oplus \{VNF_{i,j}\}$
      $\Delta F_{i,j} = w_1 \Delta b_{i,j} + w_2 \Delta c_{i,j} + w_3 \Delta m_{i,j}$;
    end
  end
  if $\mathcal{F}_i = \emptyset$ **then**
    return Untrusted Service Chain
  else
    select $\{1 \le j_{opt} \le |\mathcal{F}_i| \; | \Delta F_{i,j_{opt}} = \max_{1 \le j \le |\mathcal{F}_i|} \Delta F_{i,j}\}$;
    $VNF_i = VNF_{i,j_{opt}}$
  end
end
$\mathcal{C} = \{VNF_1, VNF_2, \ldots VNF_N\}$

The targeted service chain $\mathcal{C}$ is sent to the Remote Verifier, which in turn sends the attestation requests to the Attestation Agents of the VNFs. The Attestation Agents retrieve the integrity measurements from the Trusted Platform Module and generate the integrity report, that is sent back to the Remote Verifier.

The Remote Verifier verifies the integrity report, computes the integrity result and sends it to the Trust Assessor that reads such result and determines the trustworthiness of the service. The sequence diagram of the Attestation procedure is reported in Figure 4.

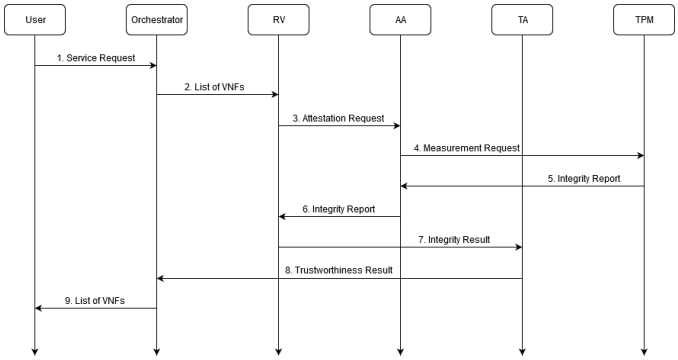

Fig. 4. Sequence Diagram of the Attestation procedure of the service chain

IV. A PRACTICAL USE CASE

This section shows the operative mode of the proposed architecture through a practical use case. Let us consider a video transmission scenario. The service chain is made up of five distinct functions: a firewall, an Intrusion Detection System (IDS), a parental control filter, a video optimizer, and a Network Address Translation (NAT). Each of the above functions is implemented as a VNF. The five resulting VNFs must be executed sequentially to guarantee the overall service to the final user. For example, without loss of generality, the firewall is implemented as VNF_1, the IDS as VNF_2, the NAT as VNF_3, the parental control as VNF_4 and the video optimizer as VNF_5. The VNFs are located on different servers, say, three servers labeled as S1, S2 and S3, that are distributed on a multi-cloud environment. To take into account redundancy, multiple copies of the same VNF are installed on more servers. Let the VNFs be located on the servers as follows. On S1 there are a firewall, an IDS, a video optimizer, and a NAT; on S2 there are an IDS, a parental control, and a NAT; and on S3 there are a firewall, a parental control, and a video optimizer. This example is depicted in Fig. 5

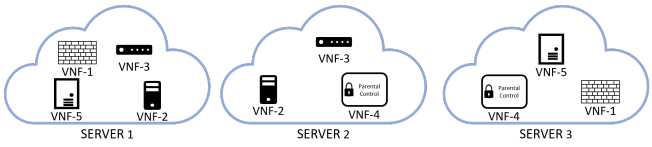

Fig. 5. Use Case Scenario

The first step to be executed is the installation procedure. The orchestrator sends a message to the Attestation Agent of each VNF that triggers its installation. The Attestation Agents send back the initial state S^{init} of the VNFs they refer to. The orchestrator stores all this information in its secure database. Let the initial parameters be as in Table II.

During the reporting procedure, the Attestation Agent of each VNF communicates with the Software Defined Network

TABLE II
VNFs' INITIAL STATE

	Server 1	Server 2	Server 3
VNF_1	{DFCD, 15 Mbps, 0.8%, 430 MB}		{52ED, 8.4 Mbps, 0.78%, 410 MB}
VNF_2	{8A0B, 14.6 Mbps, 1.1%, 360 MB}	{6EB6, 13 Mbps, 1.08%, 340 MB}	
VNF_3		{52E4, 12.7 Mbps, 0.94%, 270 MB}	{4BA3, 11.2 Mbps, 1.14%, 290 MB}
VNF_4	{6C9A, 12 Mbps, 0.83%, 300 MB}		{70F7, 12.8 Mbps, 0.95%, 320 MB}
VNF_5	{B36E, 18 Mbps, 0.7%, 170 MB}	{0B91, 15.8 Mbps, 0.81%, 160 MB}	

controller and sends the state information $S^{(cur)}$ to it. The Software Defined Network controller monitors the parameters of the involved entities and sends all the states to the orchestrator, which stores the information in a database. As previously said, this procedure is performed periodically, to update the reliability requirements of the service.

As an example, let the current parameters be as in Table III.

TABLE III
VNFs' CURRENT STATE

	Server 1	Server 2	Server 3
VNF_1	{DFCD 14.4 Mbps 0.7% 400 MB}		{52ED 8 Mbps 0.75% 370 MB}
VNF_2	{8A8B 13.9 Mbps 0.93% 350 MB}	{6E6B 10.9 Mbps 0.91% 310 MB}	
VNF_3		{52E4 11.9 Mbps 0.91% 250 MB}	{4BA3 10.7 Mbps 1.1% 260 MB}
VNF_4	{6C9A 11.1 Mbps 0.87% 280 MB}		{70F7 12.5 Mbps 0.9% 315 MB}
VNF_5	{B36E 16.2 Mbps 0.59% 180 MB}	{0B91 16.3 Mbps 0.85% 150 MB}	

When a user makes a request for a service, e.g. the fruition of a video content, the attestation procedure begins and the orchestrator starts Algorithm 1 to select the best VNF sequence that compose the service chain. Let us suppose that the reliability of the service relies more on bandwidth consumption than CPU and memory usage. So, the weights of the reliability function (3) are set as $w_1 = 0.4;\ w_2 = 0.35;\ w_3 = 0.25$.

For each VNF, the reliability is then evaluated according to Table IV, and the VNF list is computed according to (4) and chosen as $\mathcal{C} = \{VNF_{1,3}, VNF_{2,2}, VNF_{3,3}, VNF_{4,1}, VNF_{5,2}\}$.

Let us suppose now that some VNFs are attacked; as a result, their hash values change, so that in the subsequent update, the current state of the VNFs becomes as in Table V, where the hash of the attacked VNFs is reported in bold.

The VNFs with the modified hash are no more considered as trusted, and the list of VNFs that compose the service changes accordingly. Referring to Table IV and applying again

TABLE IV
RELIABILITY FUNCTION FOR EACH VNF

	Server 1	Server 2	Server 3
VNF_1	7.775		10.1705
VNF_2	2.8395	8.368	
VNF_3		4.3305	7.714
VNF_4	5.346		1.3875
VNF_5	-1.7415	2.286	

TABLE V
CURRENT STATE AT THE SECOND UPDATE

	Server 1	Server 2	Server 3
VNF_1	{DFCD 14.4 Mbps 0.7% 400 MB}		{52ED 8 Mbps 0.75% 370 MB}
VNF_2	{**8A0C** 13.9 Mbps 0.93% 350 MB}	{6E6B 10.9 Mbps 0.91% 310 MB}	
VNF_3		{52E4 11.9 Mbps 0.91% 250 MB}	{4BA3 10.7 Mbps 1.1% 260 MB}
VNF_4	{**6E82** 11.1 Mbps 0.87% 280 MB}		{70F7 12.5 Mbps 0.9% 315 MB}
VNF_5	{B36E 16.2 Mbps 0.59% 180 MB}	{**0A64** 16.3 Mbps 0.85% 150 MB}	

Algorithm 1 the VNF list that is sent to the final user becomes $\mathcal{C} = \{VNF_{1,3}, VNF_{2,2}, VNF_{3,3}, VNF_{4,3}, VNF_{5,1}\}$.

V. CONCLUSIONS AND FUTURE WORK

Network Function Virtualization (NFV) is a network paradigm that allows to decouple Virtualized Network Functions (VNFs) from the underlying hardware, thus enabling faster deploying of services. The interconnection of such VNFs composes the Service Function Chain. With the emergence of these paradigms, security problems arise that need to be properly takled. In this context, remote attestation is surely of great help in evaluating the trustworthiness and reliability of VNFs. In this paper, a procedure is explained in detail that performs the attestation of NFV-based service function chains, by properly taking into account the targeted security requirements and the measured reliability levels of the VNFs of the chain. The architecture is described in detail, pointing out the main parameters and procedures needed to evaluate the trustworthiness and the reliability guarantees of the service

chain. Finally, a practical use case is described, to show the operation mode of the proposed procedure. An accurate definition of the most representative performance metrics and the evaluation of the effectiveness of the architecture proposed in this work require further investigation, that will be carried out in a future research work.

ACKNOWLEDGMENTS

This work was framed in the context of the GUARD project, which receives funding from the European Union's Horizon 2020 research and innovation programme under grant agreement 833456. It was also supported by Apulia Region (Italy) Research project INTENTO (36A49H6), and by the PRIN project no. 2017NS9FEY entitled "Realtime Control of 5G Wireless Networks: Taming the Complexity of Future Transmission and Computation Challenges" funded by the Italian MIUR; it has been also partially supported by the Italian MIUR PON projects Pico&Pro (ARS01_01061), AGREED (ARS01_00254), FURTHER (ARS01_01283) and RAFAEL (ARS01_00305).

REFERENCES

[1] J. Halpern and C. Pignataro, "Service Function Chaining (SFC) Architecture," Internet Requests for Comments, RFC Editor, RFC 7665, Oct. 2015. [Online]. Available: https://www.rfc-editor.org/info/rfc7665

[2] E. Shi, A. Perrig, and L. Van Doorn, "Bind: A fine-grained attestation service for secure distributed systems," in *2005 IEEE Symposium on Security and Privacy (S&P'05)*. IEEE, 2005, pp. 154–168.

[3] S. Xin, Y. Zhao, and Y. Li, "Property-based remote attestation oriented to cloud computing," in *2011 Seventh International Conference on Computational Intelligence and Security*. IEEE, 2011, pp. 1028–1032.

[4] H. Lauer, A. Sakzad, C. Rudolph, and S. Nepal, "A logic for secure stratified systems and its application to containerized systems," in *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. IEEE, 2019, pp. 562–569.

[5] C. Xu, H. Liu, P. Li, and P. Wang, "A remote attestation security model based on privacy-preserving blockchain for v2x," *IEEE Access*, vol. 6, pp. 67 809–67 818, 2018.

[6] A. Ibrahim, "Collective attestation: for a stronger security in embedded networks," in *2018 IEEE 37th Symposium on Reliable Distributed Systems (SRDS)*. IEEE, 2018, pp. 267–268.

[7] T. A. Syed, S. Jan, S. Musa, and J. Ali, "Providing efficient, scalable and privacy preserved verification mechanism in remote attestation," in *2016 International Conference on Information and Communication Technology (ICICTM)*. IEEE, 2016, pp. 236–245.

[8] P. Maene, J. Götzfried, R. De Clercq, T. Müller, F. Freiling, and I. Verbauwhede, "Hardware-based trusted computing architectures for isolation and attestation," *IEEE Transactions on Computers*, vol. 67, no. 3, pp. 361–374, 2017.

[9] T. Rauter, A. Höller, J. Iber, M. Krisper, and C. Kreiner, "Integration of integrity enforcing technologies into embedded control devices: experiences and evaluation," in *2017 IEEE 22nd Pacific Rim International Symposium on Dependable Computing (PRDC)*. IEEE, 2017, pp. 155–164.

[10] M. Santra, S. K. Peddoju, A. Bhattacharjee, and A. Khan, "Design and analysis of a modified remote attestation protocol," in *2017 IEEE Trustcom/BigDataSE/ICESS*. IEEE, 2017, pp. 578–585.

[11] J.-W. Ho and M. Wright, "Distributed detection of sensor worms using sequential analysis and remote software attestations," *IEEE Access*, vol. 5, pp. 680–695, 2017.

[12] N. Agarwal and K. Paul, "Xebra: Xen based remote attestation," in *2016 IEEE Region 10 Conference (TENCON)*. IEEE, 2016, pp. 2383–2386.

[13] N. Ahmed, M. A. Talib, and Q. Nasir, "Program-flow attestation of iot systems software," in *2018 15th Learning and Technology Conference (L&T)*. IEEE, 2018, pp. 67–73.

[14] B. Kuang, A. Fu, S. Yu, G. Yang, M. Su, and Y. Zhang, "Esdra: An efficient and secure distributed remote attestation scheme for iot swarms," *IEEE Internet of Things Journal*, 2019.

[15] A. Ibrahim, A.-R. Sadeghi, and G. Tsudik, "Us-aid: Unattended scalable attestation of iot devices," in *2018 IEEE 37th Symposium on Reliable Distributed Systems (SRDS)*. IEEE, 2018, pp. 21–30.

[16] M. Conti, E. Dushku, and L. V. Mancini, "Radis: Remote attestation of distributed iot services," in *2019 Sixth International Conference on Software Defined Systems (SDS)*. IEEE, 2019, pp. 25–32.

[17] I. D. O. Nunes, G. Dessouky, A. Ibrahim, N. Rattanavipanon, A.-R. Sadeghi, and G. Tsudik, "Towards systematic design of collective remote attestation protocols," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2019, pp. 1188–1198.

[18] H. Gunleifsen and T. Kemmerich, "Security requirements for service function chaining isolation and encryption," in *2017 IEEE 17th International Conference on Communication Technology (ICCT)*. IEEE, 2017, pp. 1360–1365.

[19] B. Lakshmi and J. Lakshmi, "Integrating service function chain management into software defined network controller," in *2019 IEEE World Congress on Services (SERVICES)*, vol. 2642. IEEE, 2019, pp. 160–165.

[20] O. Aiko, M. Nakajima, Y. Soejima, and M. Tahara, "Reliable design method for service function chaining," in *2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*. IEEE, 2019, pp. 1–4.

[21] G. Mirjalily and L. Zhiquan, "Optimal network function virtualization and service function chaining: A survey," *Chinese Journal of Electronics*, vol. 27, no. 4, pp. 704–717, 2018.

[22] A. Blaise, S. Wong, and A. H. Aghvami, "Virtual network function service chaining anomaly detection," in *2018 25th International Conference on Telecommunications (ICT)*. IEEE, 2018, pp. 411–415.

[23] M. Wang, B. Cheng, B. Li, and J. Chen, "Service function chain composition and mapping in nfv-enabled networks," in *2019 IEEE World Congress on Services (SERVICES)*, vol. 2642. IEEE, 2019, pp. 331–334.

[24] L. Tian and W. Jiang, "A multi trust chain scheme in trusted cross-domain interaction," in *2012 International Conference on Industrial Control and Electronics Engineering*. IEEE, 2012, pp. 550–553.

[25] I. Oliver, J. Howse, G. Stapleton, Z. Shams, and M. Jamnik, "Exploring and conceptualising attestation," in *International Conference on Conceptual Structures*. Springer, 2019, pp. 131–145.

[26] ETSI GS NFV 002, "Network functions virtualization (NFV); architectural framework v1.1.1," ETSI, Tech. Rep., October 2013.

[27] M. Pattaranantakul, R. He, Q. Song, Z. Zhang, and A. Meddahi, "Nfv security survey: From use case driven threat analysis to state-of-the-art countermeasures," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3330–3368, 2018.

[28] A. M. Alwakeel, A. K. Alnaim, and E. B. Fernandez, "Analysis of threats and countermeasures in nfv use cases," in *2019 IEEE International Systems Conference (SysCon)*. IEEE, 2019, pp. 1–6.

[29] N. Torkzaban, C. Papagianni, and J. S. Baras, "Trust-aware service chain embedding," in *2019 Sixth International Conference on Software Defined Systems (SDS)*. IEEE, 2019, pp. 242–247.

[30] S. Ravidas, S. Lal, I. Oliver, and L. Hippelainen, "Incorporating trust in nfv: Addressing the challenges," in *2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN)*. IEEE, 2017, pp. 87–91.

[31] M. De Benedictis and A. Lioy, "A proposal for trust monitoring in a network functions virtualisation infrastructure," in *2019 IEEE Conference on Network Softwarization (NetSoft)*. IEEE, 2019, pp. 1–9.

[32] V.-C. Nguyen, A.-V. Vu, K. Sun, and Y. Kim, "An experimental study of security for service function chaining," in *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE, 2017, pp. 797–799.

[33] Y. Liu, H. Lu, X. Li, and D. Zhao, "An approach for service function chain reconfiguration in network function virtualization architectures," *IEEE Access*, vol. 7, pp. 147 224–147 237, 2019.

[34] Y. Liu, Y. Lu, W. Qiao, and X. Chen, "A dynamic composition mechanism of security service chaining oriented to sdn/nfv-enabled networks," *IEEE Access*, vol. 6, pp. 53 918–53 929, 2018.

[35] T. C. Group, "Tcg specification architecture overview, specification revision 1.4," Tech. Rep., 2007.