Designing security-aware service requests for NFV-enabled networks

Houda Jmila, Gregory Blanc

Designing Security-aware Service Requests for NFV-enabled Networks

Houda Jmila and Gregory Blanc
SAMOVAR, CNRS UMR 5157
Institut Mines-Télécom, Télécom SudParis
Institut Polytechnique de Paris
91011 Evry Cedex, France
{houda.jmila,gregory.blanc}@telecom-sudparis.eu

***Abstract*—Network Function Virtualization (NFV) is a new concept where virtualization is used to shift "network functions" (e.g., routers, switches, load-balancers, proxies) from specialized hardware appliances to software images running on high volume servers. The resource allocation problem in the NFV environment has received considerable attention in the past years. However, little attention was paid to the security aspects of the problem in spite of the increasing number of vulnerabilities faced by cloud-based applications. Securing the services is an urgent need to completely benefit from the advantages offered by NFV. In this paper, we show how a network service request, composed of a set of service function chains (SFC) should be modified and enriched to take into consideration the security requirements of the supported service. We examine the well-known security best practices and propose a two-step algorithm that extends the initial SFC requests to a more complex chaining model that includes the security requirements of the service.**

***Index Terms*—Network Function Virtualization (NFV), Service Function chaining (SFC), resource allocation, Security.**

I. Introduction

Network Function Virtualization [1] has drawn significant attention from both industry and academia as a new concept allowing greater network flexibility and time/cost reduction to introduce new services. Network functions, such as load balancers, WAN optimizers, or Intrusion Detection Systems (IDSs), were traditionally provided by dedicated and special-purpose hardware for each network function. However, this calls for specialized maintenance and limits flexibility. The NFV takes advantage of recent advances in virtualisation technologies to decouple network functions from dedicated hardware and run them as software in virtual machines or containers in a cloud computing infrastructure. In this way, the *Virtualized Network Function*s (VNFs) can be relocated and instantiated at different locations without requiring the purchase and installation of new hardware. Moreover, precise and dynamic routing of the traffic between the VNFs can be achieved using the Software-Defined Networks (SDN) [2] technology.

To fully enjoy the benefits of NFV, security, privacy and resilience should be guaranteed to the services running over the VNFs chain. In fact, NFV faces several security challenges (e.g., multi-tenancy and live migration) that make it vulnerable to some cybersecurity attacks (e.g., side-channel attacks and shared resource misuse attacks). Although the resource allocation problem was extensively investigated in the literature [3]–[5], little attention was paid to the security aspects. In this paper, we investigate the problem of designing a service network request, that takes into account the security requirements. Starting from initial SFCs containing basic VNFs of the supported service, we enrich and extend the SFCs to more complex chaining model that include security needs. We achieve this in two steps. First, we take profit of the flexibility allowed by the NFV and SDN paradigms to include additional customized virtual network *security* functions (VNSFs). VNSFs can be firewalls, intrusion detection systems, parental control systems, among others, and will provide the required security services depending on the service definition. Second, we add new security constraints reflecting the resource sharing and mapping restrictions to guarantee their compliance with the well-known security best practices.

Note that this is a study of a fundamental nature that tries take care of critical details of NFV , and it is a necessary formalization step in order to finally design algorithms that compute the mapping of the VNF request.

The remainder of this paper is structured as follows: Section II gives an introduction to the NFV resource allocation problem, section III details the designed model, section IV exposes a use-case study, section V describes the related work and section VI concludes the paper.

II. NFV resource allocation problem

The application of NFV introduces the problem of efficient resource management. In fact, the physical resources used to host the VNFs have a finite amount of compute, memory and storage capacity. Physical links connecting these resources have also limited amount of bandwidth. Therefore, these physical resources should be managed conveniently to gain the economical benefits promised by NFV.

The resource management problem in NFV is often referred to as the Service Function Chaining problem and is extensively investigated in the literature [4], [6] as well as by different working groups and research projects. Particularly, ETSI NFV ISG [7] describes a network service request by means of three graphs, depicted in Figure 1:

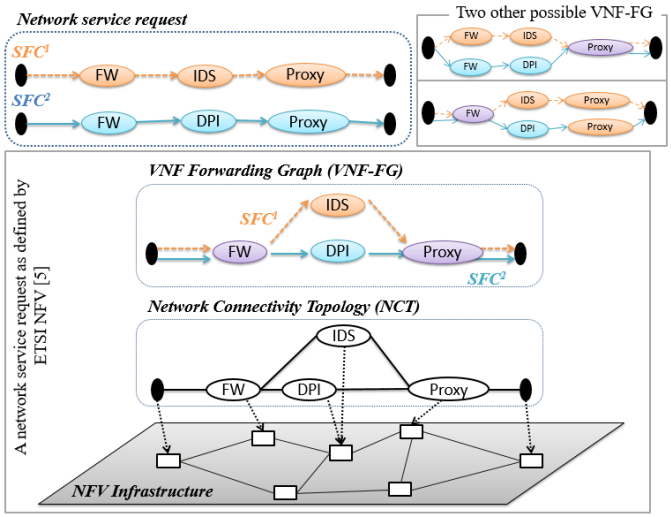


Fig. 1. Network Function Virtualization graphs

- **The Virtual Network Function Service Graph (VNF-SG)** that represents VNF forwarding graphs (VNF-FGs) also called service function chains (SFCs) on top of the network connectivity topology.
- **The Network Connectivity Topology (NCT)** defining the underlying network necessary to support the forwarding graph flows.
- **The NFV infrastructure (NFVI)**, or substrate network that will host the requested resources. It is composed of substrate nodes and links having limited available resources.

A service request, for example a slice request in a 5G environment, can be described by a set of service function chains interacting together to provide the requested service. The top of Figure 1 shows an example of a service request composed of two service function chains SFC^1 and SFC^2. Each SFC is a collection of VNFs and virtual links connecting them in a specific order. For instance, SFC^1 is composed of a Firewall, an Intrusion Detection System and a Proxy. Each SFC requires an amount of bandwidth and each VNF of the chain has a type (firewall, IDS, etc.) and demands an amount of computing resources (CPU, memory, storage) to process the traversing traffic.

The resource provisioning problem in a NFV environment can be achieved in two stages [4]:

1) **VNF service graph composition**
2) **VNF service graph embedding**.

During the first stage, we associate to the service request a VNF service graph that concatenates the different VNFs efficiently in order to compose the supported service in the most adequate way.

Figure 2 shows an illustration of three VNF-SGs that can be associated to the service request *S*. In fact, the two SFCs composing the service require VNFs of the same type (firewall, proxy). Thus, for efficient resource usage, the service provider can decide to mutualize some or all these VNFs between the two chains. Obviously, enough resources should be allocated to the shared VNFs to process the flows of the different chains. We remark that for a single service request, diverse VNF-SGs can be derived depending on how much VNFs are shared between the services, on their order, etc. Actually, optimizing the composition of the VNF-SG to achieve some pre-defined goals (such as resource usage efficiency), while respecting the services needs, is a challenging problem that requires more attention from the research community as evoked in [4].

Once the VNF-SG is selected, the VNF-SG embedding consists in mapping the VNF-SG components (VNFs and links) into the provider's infrastructure. This involves two sub-problems: i) VNFs mapping and ii) virtual links mapping, also called chaining. *VNF mapping* consists in finding the best service nodes in the NFVI having enough resources to host the VNFs and *chaining* is about creating optimal paths to chain the VNFs and steer traffic between them. The mapping cost is the amount of physical resources required to host the service.

Securing the hosted services is essential to face the increasing and various threats introduced by the VNF environment. Conceiving a security-aware service request, *from the beginning*, leads to better resource allocation solutions as it gives richer and more complete information about the service resource needs. Otherwise, ignoring the service security constraints can lead to service disruption if network configurations are required to meet the security needs later. For these reasons, we introduce a **preliminary step** that must be performed before the resource provisioning stages described above, and consists in enriching the initial service request to take into consideration its security needs. Note that the resource provisioning stages are out of the scope of this paper and will be addressed in future work. We focus on the preliminary step described above, more detail are given in the following.

III. Design of a Security-aware Service Request

The problem we tackle is how to enrich an initial service request to take into account the security needs. To do so, we will first describe a *basic service request model* and then detail the different steps leading to a *security-aware service request model*. Next, the impact of these steps over the resource provisioning stages is examined. But before all this, we will define the VNF Infrastructure model.

A. The VNFI model

Without loss of generality, we model the VNFI by a weighted undirected graph $G_s = (N_s, L_s)$, where N_s is the set of *substrate nodes* n_s and L_s is the set of *substrate links* l_s. Let $c(n_s)$ denote the available capacity of node n_s (typically CPU, storage and memory) and $bw(l_s)$ the available bandwidth on link l_s. Variable φ represents a substrate path (a single or a sequence of substrate links) between two substrate nodes. Variable P_φ is the set of loop-free substrate paths. The available bandwidth $bw(\varphi)$ associated to a substrate path φ can be evaluated as the smallest available bandwidth on the links along the substrate path.

B. Basic Service Request Model

A network service request r can be composed by a set of service function chains $Req_r = \{SFC_r^1, SFC_r^2, ...SFC_r^m\}$. Each service function SFC_r^j requires an amount of bandwidth $bw(SFC_r^j)$ and describes the ordered VNF sequence the traffic must traverse, between two endpoints (e.g., a users computer and a remote server).

$$SFC_r^j = \left\{f_1^j, f_2^j, ..., f_y^j\right\} \quad (1)$$

Each VNF f requires an amount of resources $c(f)$ to process the entering traffic. Moreover, as specified by the SFC IETF working group [8], each VNF f is associated to a type $t(f)$, e.g., firewall, DPI, NAT, etc. To model the endpoints of the traffic, we add fictive VNFs f_0^j and f_∞^j at the start and the end of each service function chain SFC_r^j. These VNFs are only defined by their their hosting substrate nodes in the VNF infrastructure that are selected with respect to the placement of the endpoints of the SFC. More formally:

$$SFC_r^j = \left\{f_0^j, f_1^j, f_2^j, ..., f_y^j, f_\infty^j\right\} \quad (2)$$

C. Securing the Service Request

The SFCs will carry user traffic and user specific information. This data will contain sensitive information about the user and the environment in which he is situated. This will require proper considerations in the design and implementation of the SFCs to preserve the privacy of the user and the integrity of the provided data. To do so, we design *a security-aware network service request model* that integrates most of the security measures to be taken into account during the creation and implementation of the request, as described below.

The universal way to secure a service is to route the traffic to appropriate security functions to process security operations. This could be useful to detect intrusion, to classify the traffic, to monitor the flows, etc. However, this solution can not satisfy some other security requirements. For example, deploying the VNFs in untrusted domains is not permitted for services with high sensitivity. Thus, security mapping constraints should also be considered. Moreover, resource sharing policy should be respected during the SFC composition phase. Hence, we construct the security-aware network request model in two steps. First *we add the required Virtual network security functions* to the initial SFCs and second *we define constraints related to the resource sharing policies and the security mapping restrictions*. These two steps are described below.

*1) **Step 1: adding virtual network security functions**:* In order to secure the services, some additional VNSFs should be included to the service function chains. The traffic should traverse these VNSFs where security and monitoring operations will be performed (filtering, inspection, classification, etc.). Note that deciding which VNSFs to instantiate and where they should be placed in the service function chain is specific to the service and it is generally resolved by security experts after studying the possible threats and vulnerabilities. Inspired by the literature and standards [7], [9], [10], we define three ways

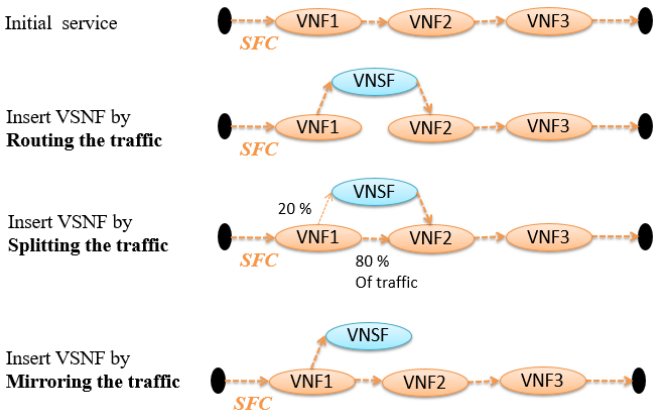


Fig. 2. Three fashions of inserting a VNSF in the SFC

of adding the VNSFs to the SFC, depending on whether the whole traffic is routed, split or mirrored (cf. Figure 2):

- **traffic routing**: this is the case where all traffic is routed to pass through a VNSF before reaching the next VNF.
- **traffic splitting** : this is used if the traffic should be split into different types of flows, each has to traverse different security chains or bypass some of them. For example, some unencrypted traffic needs to be inspected by an IDS.
- **traffic mirroring** : mirroring can be required to deploy VNSF or security monitoring functions in an out-of-band fashion, where the traffic does not need to be routed to a VNSF but simply mirrored. For example, an IDS needs to analyze a copy of the traffic without acting on it.

Obviously, for each VNSF added to the SFC, the amount of required resources and the amount of bandwidth necessary to route the flow to and from the function are specified.

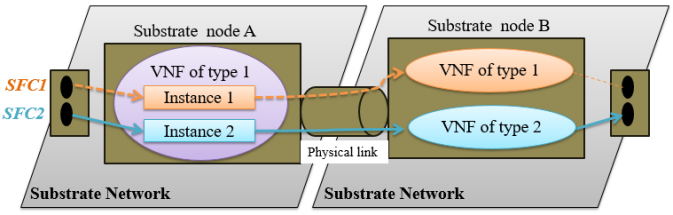
After inserting the VNSFs to a SFC, the result is not a simple linear chain of VNFs anymore, but a more complex graph, as shown in Figure 2. Let $G_{SFC_r^j} = \left\{N_{SFC_r^j}, L_{SFC_r^j}\right\}$ be a weighted oriented graph describing the resulting graph. $N_{SFC_r^j}$ represents the set of virtual network functions. $L_{SFC_r^j}$ is the set of links supporting the various flows passing through the VNFs. Note that $N_{SFC_r^j}$ includes the set of VNFs initially present in SFC_r^j in addition to the VNSFs that have been inserted for security purpose. Formally:

$$N_{SFC_r^j} = \left\{f_k^j | f_k^j \in SFC_r^j\right\} \oplus \{sf\} \quad (3)$$

$\{sf\}$ is the set of added VNSFs and $\oplus$ is an operator inserting them in the existing sequence of VNFs in the requested SFC, $|$ means "except". Note that each virtual link $l = (f, f')$ between two VNFs f and f' is associated to a required bandwidth $bw(l)$. Each VNF f is associated to a type $t(f)$ and required resources $c(f)$. The service network request is composed of the set of resulting graphs:

$$Req_r = \{G_{SFC_r^1}, G_{SFC_r^2}, ...G_{SFC_r^m}\} \quad (4)$$

*2) **Step 2: defining security deployment constraints**:* Along with inserting VNSFs, some security constraints need to



a. *Sharing multiple VNF instances of the same VNF type between different services*

b. *Sharing one physical node (or link) between VNFs of different types (or virtual links of different services).*

Fig. 3. Two scenarios for resource sharing policy

be considered during the *VNF-SG composition* and the *VNF-SG embedding* stages. In fact, taking into account only the resource requirements and types during the request creation is not adequate. Actually, several VNF-SG composition and VNF-SG embedding options may be valid from an optimization point of view but invalid from security best practices and recommendations point of view as they can break tenants' security needs. In the rest of the paper, we will use the word *VNF* to design both VNFs and VNSFs.

In the following, we describe some constraints that can model common security requirements and best practices. Mostly, we define two types of constraints: i) those specifying the resource sharing policy, and ii) constraints related to geographical restrictions when deploying the requested components in the network infrastructure. Details and motivation are described below.

- Resources sharing constraints: the resource sharing policy should be defined for two generic scenarios (cf Fig 3). First, when multiple instances of the same VNF type can be used to process traffic flows of different SFCs. Second, when the same physical host (node or link) can be shared between different VNFs or virtual links.
  - Sharing instances
    Motivation: for a single VNF, one or more subsequent VNF instances can be created. For resource usage efficiency, the service provider can allocate only one VNF and create two instances in the same host to proceed two traffic flows of two different services requiring the same VNF type, as depicted in Figure 3.a. For example, an anti-virus function can be configured once and used for every chain that needs this function. In this case, the workload should be separated in an appropriate way between the different instances to achieve service isolation. Because of potential security breaches, service tenants may refuse sharing instances of the same VNF.
    Design: To formulate this constraint, we specify for each VNF f the set of VNFs of the other chains that should not share instances with it, called $DontShareInstances(f)$.
  - Sharing physical resources
    Motivation: this deals with allowing or not the co-location of VNFs (*resp.* virtual links) in the same physical node (*resp.* physical link) as depicted in Figure 3.b. The co-location can be required to improve the security service performance or to avoid information leakage. For example, a firewall and an IDS are placed in the same node to collaborate and detect malicious activities within a short time. In another scenario, two security functions of the same service need to be placed in the same node to avoid security breaches. However, the co-location of VNFs and links of different services may be forbidden to guarantee isolation between the services.
    Design: As explained above, the co-location of virtual components can be required or forbidden, elsewhere it is simply allowed. To capture these features, we define for each VNF f, two sets $ShouldColoc(f)$ (*resp.* $ForbidColoc(f)$) defining the list of VNFs that should (*resp.* should not) be allocated in the same host as f. We extend this constraint to the virtual links connecting the VNFs by specifying for each virtual link $l = (f, f')$ between two VNFs f and f' if it is allowed to share the physical path hosting another virtual link. $ShouldColoc(l)$ (*resp.* $ForbidColoc(l)$) is the list of virtual links that should (*resp.* should not) share the same physical path as l.
    Obviously, a VNF that should be colocated with f should not be in the set $ForbidColoc(f)$ and vice versa, the same condition is valid for virtual links, formally:

$$ShouldColoc(f) \cap ForbidColoc(f) = \emptyset \quad (5)$$

$$ShouldColoc(l) \cap ForbidColoc(l) = \emptyset \quad (6)$$

    Note that the set of virtual nodes that are neither in $ShouldColoc(f)$ nor $ForbidColoc(f)$ are simply allowed to be co-located with f, the same property is observed for virtual links.
- Geographical/domain related constraints:
  Motivation: Different arguments can motivate the need for taking the geographical and domain constraints into consideration during the VNF placement. First, the Network Function Virtualization concept is designed to support the multi-provider scenario and this can introduce new security vulnerabilities. Particularly, VNFI will be partitioned into *domains of different security levels* depending on the trust towards each provider. In fact, if a VNF is hosted in an insecure physical node, it could be attacked and this can damage the quality of the offered service. Similarly, if the traffic is routed over an insecure physical path, replay and man-in-the-middle attacks could be performed. Thus, the service network providers can limit the subset of physical resources that may host their service to avoid vulnerable and untrusted hosts/domains. Second, the geographical and domain constraints can be used to improve the performance of the virtual security functions by placing them at a certain distance from a

TABLE I
SECURING A SERVICE REQUEST: STEPS AND NOTATIONS

Input: Basic service request & VNF Infrastructure description	
Notation	**Description**
$Req_r = \{SFC_r^1, SFC_r^2, ...SFC_r^m\}$	Basic service request
$SFC_r^j = \left\{f_0^j, f_1^j, f_2^j, ..., f_y^j, f_\infty^j\right\}$	Service function chain j of request r
f_i^j	Virtual network function i of SFC_r^j
$bw(SFC_r^j)$	Bandwidth required by SFC_r^j
$t(f)$	Type of VNF f
$c(f)$	Resources required by VNF f
f_0^j and f_∞^j	Endpoints of SFC SFC_r^j
$G_s = (N_s, L_s)$	VNF infrastructure
N_s	Set of substrate nodes
L_s	Set of substrate links
Step 1: For each SFC of the request, insert the required VNSFs by *routing*, *splitting or mirroring* the traffic	
Notation	**Description**
VNSF	A virtual network security function
$G_{SFC_r^j} = \left\{N_{SFC_r^j}, L_{SFC_r^j}\right\}$	Graph resulting from the SFC_r^j after inserting the required VNSFs
$N_{SFC_r^j}$	virtual network functions of $G_{SFC_r^j}$
$L_{SFC_r^j}$	Virtual links of $G_{SFC_r^j}$
$l = (f, f')$	Virtual link connecting f and f'
$bw(l)$	Bandwidth required by link l
Step 2: For each virtual component, specify the following security deployment constraints:	
Notation	**Description**
$DontShareInstances(f)$	VNFs that should not share instances with VNF f
$ShouldColoc(f)$ resp. $ShouldColoc(l)$	VNFs *resp. virtual link* that should be allocated in the same substrate node *resp. path* as f resp. as l
$ForbidColoc(f)$ resp. $ForbidColoc(l)$	VNFs *resp. v. link* that should NOT be allocated in the same substrate node *resp. path* as f resp. as l
$MayHost(f)$ resp. $MayHost(l)$	Physical nodes *resp. physical links* authorized to host the VNF f *resp v. link* l
Output: Security-aware service request	
Notation	**Description**
$Req_r = \left\{G_{SFC_r^1}, G_{SFC_r^2}, ...G_{SFC_r^m}\right\}$	Security-aware service request

reference point where detection, network analysis and responding would be more efficient. For example, detecting DDoS attack would be easier if performed close to the sources. Similarly, placing firewalls and IDS/IPS close to the border of the Telecommunication Service Provider domain is better to block unwanted traffic before it enters the network. Moreover traffic encryption should be performed before traversing the untrusted network for end-to-end security.

Design: To formulate these constraints, we define for each virtual resource, the set of physical resources allowed to host it. Then $MayHost(f) \subset N_s$ is the set of physical nodes authorized to host the VNF f, and $MayHost(l) \subset L_s$ is the set of physical links where the virtual link $l = (f, f')$ can be mapped. $MayHost(f)$ and $MayHost(l)$ should be constructed in a way that the insecure domains are avoided, and the VNFs performance is increased. In a finer-grained description, let $d(n_s, m_s)$ denote the distance between two substrate nodes n_s and m_s, e.g., the number of hops or the geographical distance. Let $p \in N_s$ be a substrate node defining a reference point. If the VNF f should be placed at a certain distance δ from the reference place p, then $MayHost(f)$ may be deduced as follows:

$$MayHost(f) = \{m_s \in N_s | d(n_s, m_s) < \delta\} \quad (7)$$

D. Security-aware Service Request

In conclusion, a security-aware service request Req_r can be described by a set of secure SFCs $Req_r = \left\{G_{SFC_r^1}, G_{SFC_r^2}, ...G_{SFC_r^m}\right\}$. For each $G_{SFC_r^j} \in Req_r$, each VNF $f_k^j \in N_{SFC_r^j}$ (except f_0^j and f_∞^j) is associated to i) a type $t(f_k^j)$, ii) a requested capacity $c(f_k^j)$ and a set of security deployment constraints, namely, iii) $MayHost(f_k^j)$, the list of physical nodes where f_k^j can be allocated, iv) $ShouldColoc(f_k^j)$ *resp.* v) $ForbidColoc(f_k^j)$, the set of other VNFs that should *resp.* should not be co-located with f_k^j. vi) $DontShareInstances(f_k^j)$, the set of VNFs that should not share instances with it.

For each virtual link $l = (f_k^j, f_{k'}^j)$ connecting two VNFs f_k^j and $f_{k'}^j$, we associate i) a requested bandwidth $bw(l)$ ii) $MayHost(l)$, the list of physical links allowed to host l, iii) $ShouldColoc(l)$ *resp.* iv) $ForbidColoc(l)$, the lists of virtual links that should *resp.* should not be co-located with l.

The different steps and notation of the model are summarized in Table I. In the following, we will examine the impact of the proposed model over the two resource provisioning stages described in Section II; the service graph composition and embedding.

E. Service Graph Composition

From the slice request, and during the service graph composition step (cf. Section II), an optimal VNF service graph is

derived with respect to the resource sharing constraints. This graph contains the routing and forwarding description of the set of SFCs and shows how the functions can be interconnected at a virtual level.

Let $SG_r = (N_r, L_r)$ be a *directed weighted* graph representing the VNF service graph associated to the service request Req_r. L_r is the set of links supporting the various flows passing through the VNFs. The set of nodes $n_r \in N_r$ represents the set of virtual network functions that will be used to support the different SFCs.

During the service graph composition step, two or more VNFs (*resp. virtual links* connecting two VNFs) can be consolidated into one virtual node (*resp. one virtual link*) of the VNF service graph to optimize the resource usage and simplify the service graph. We call this, virtual nodes *resp. virtual links* composition. The different security constraints associated to the service request should be respected and kept to preserve the service integrity. In the following, we show how the security parameters associated to the virtual elements (nodes and links) of the service graph should be calculated during the composition step.

1) Virtual nodes composition: Let $n_r \in N_r$ be a virtual node of the service graph SG_r and let $Share(n_r)$ be the set of virtual network functions merged into n_r.

Two VNFs f and f' can be in the set $Share(n_r)$ **if and only if** :

- f and f' have the same type.
- f and f' accept to share instances:

$$\begin{aligned} f \notin DontShareInstances(f') \ \& \\ f' \notin DontShareInstances(f) \end{aligned} \tag{8}$$

- f and f' accept to share the same substrate host:

$$f \notin ForbidColoc(f') \ \& \ f' \notin ForbidColoc(f) \tag{9}$$

- f and f' do not have conflicting constraints:
  - A VNF that should be co-located with f, should not be banned from the co-location with f' and vice versa:

$$\begin{aligned} ForbidColoc(f') \cap ShouldColoc(f) = \emptyset \ \& \\ ForbidColoc(f) \cap ShouldColoc(f') = \emptyset \end{aligned} \tag{10}$$

  - At least one substrate node is allowed to host both f and f':

$$MayHost(f) \cap MayHost(f') \neq \emptyset \tag{11}$$

Consequently, the constraints associated to the resulting virtual node n_r can be deduced as follows:

- n_r has the type of the grouped VNFs, thus:

$$t(n_r) = t(f), \forall f \in Share(n_r) \tag{12}$$

- The requested capacity is the sum of all the required resources by the grouped VNFs:

$$c(n_r) = \sum_{f \in Share(n_r)} c(f) \tag{13}$$

- Only substrate nodes allowed to host all the grouped VNFs may host n_r, thus:

$$MayHost(n_r) = \bigcap_{f \in Share(n_r)} MayHost(f) \tag{14}$$

- Consider all the VNFs that should be co-colocated:

$$ShouldColoc(n_r) = \bigcup_{f \in Share(n_r)} ShouldColoc(f) \tag{15}$$

- Consider all the VNFs that are prohibited from being co-located:

$$ForbidColoc(n_r) = \bigcup_{f \in Share(n_r)} ForbidColoc(f) \tag{16}$$

In this way, $ShouldColoc(n_r)$ and $ForbidColoc(n_r)$ verify the properties (5) and (6). Finally, $DontShareInstances$ is obvsiouly not defined for n_r, as it is not required during the embedding step.

2) Virtual links composition: To simplify the service graph and the embedding step, two or more virtual links l and l' can be consolidated into one virtual link l_r of the service graph. The conditions that should be respected when merging virtual links and the security parameters associated to the resulting virtual link are calculated in the same fashion as for virtual nodes composition. Details are not given due to lack of space.

F. Service Graph Embedding

Once the service graph is deduced, resources should be allocated and instantiated to support the service. The resource allocation problem in the NFV environment is well investigated [4] but little attention was paid to the security constraints. Below we give a simplified formulation of the embedding problem while considering the security constraints:

1) Objective: Given a substrate network G_s and a service graph request Req_r. The aim is to find a mapping of all the virtual nodes and links of Req_r, to the substrate nodes and links of G_s so that the required resources are provided and the physical resources spent to map Req_r, known as embedding cost, are minimized, under the following constraints:

2) Constraints:

- **Basic constraints:**
  - *Resources constraints:* ensure that the resources required by a virtual element do not exceed the available resources of the substrate element to which it is mapped
  - *Flow conservation constraint*: during link mapping, ensure that outbound flow is equal to inbound flow for each intermediate node in the hosting path.
- **Security related constraints:** For each virtual element, the physical resource sharing and geographical/domain constraints, defined by the sets $MayHost(.)$, $ShouldColoc(.)$ and $ForbidColoc(.)$ should be respected.

IV. Use-case study

In this section, we show through an example, how to build a security-aware service request, starting from a basic request. For clarity's sake, we will use the same presentation format as Section III.

A. The VNFI model

As substrate network, we use the Abilene topology from SNDlib [11], shown in Figure 4. Alibene is a reference for mid-scale network containing 12 nodes and 15 edges.

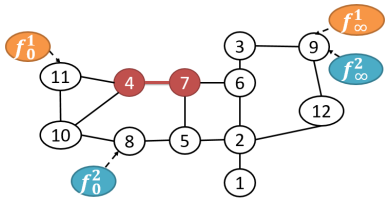


Fig. 4. VNF Infrastructure topology

B. Basic Service Request

Based on the *Service Function Chaining Use Cases in Mobile Networks* proposed by the IETF standardization group [12], we consider the scenario of a service request in a 3GPP-based mobile networks environment and composed of two service function chains presented in Figure 5.

Figure 5 shows a simplified view of a 3GGP-based service networks. It includes user equipments (UE) like smartphones or tablets, the 3GPP mobile network and the packet gateways (P-GW), followed by Service Function Chains where the upstream and downstream user flows will pass. The SFCs will be provided by the NFV Infrastructure, connected to internet or/and an internal application platform like IMS, through a router. IETF described different categories of service functions that can be used in the 3GPP mobile networks scenario. Based on this description, we consider the SFCs depicted in the Figure 5. SFC^1 includes a Video Optimizer (Video Opt.) to improve the users Quality of experience and a firewall (FW) to protect the carrier network from the outside. SCF^2 includes a parental control function (Par. Ctrl), to restrict content, and a firewall. The endpoints of the traffic traversing the SFCs are the packet gateways that will be mapped to substrate nodes 11 and 8, and the routers that will be mapped to the same substrate node 9, as illustrated in Figure 4.

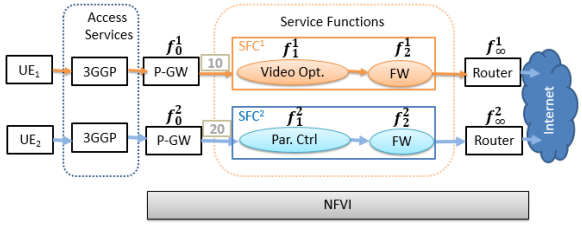


Fig. 5. SFCs scenario in 3GPP-based mobile networks

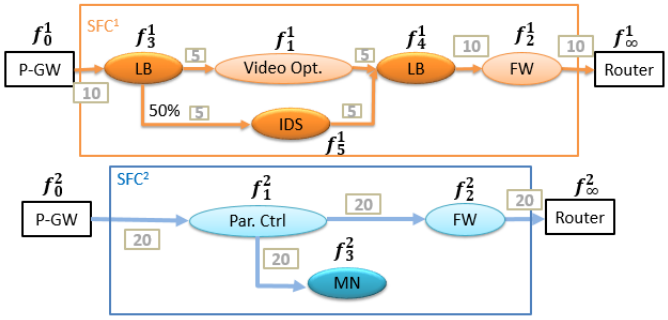


Fig. 6. SFCs after adding VSNFs

Using the basic service request model proposed in Section III-B, the previous service network request description is detailed in Table II, where the bandwidth and virtual node functions resource requirements are specified.

TABLE II
Basic service request

$Req = \{SFC^1, SFC^2\}$			
$bw(SFC^1) = 10$		$bw(SFC^2)$=20	
$SFC^1 = \{f_0^1, f_1^1, f_2^1, f_\infty^1\}$		$SFC^2 = \{f_0^2, f_1^2, f_2^2, f_\infty^2\}$	
$t(f_1^1)$=Video.Opt	$t(f_2^1)$=FW	$t(f_1^2)$=Par.Ctrl.	$t(f_2^2)$=FW
$c(f_1^1) = 25$	$c(f_2^1) = 10$	$c(f_1^2) = 20$	$c(f_2^2) = 10$
f_0^1= P-GW, mapped to node 11		f_0^2= P-GW, mapped to node 8	
f_∞^1= Router, mapped to node 9		f_∞^2= Router, mapped to node 9	

C. Securing the Service Request

As described in Section III-C, we construct the security-aware network service request model in two steps, detailed below.

*1) **Step 1: adding virtual network security functions**:* First we add the required Virtual network security functions to the initial SFCs. In our example, imagine that an Intrusion Detection System needs to be added to SFC^1 in order to detect malicious activity or policy violations. As this VNSF is computationally demanding, it may cause a noticeable performance degradation to latency-sensitive applications like video streaming. So let us insert two load balancers to split video stream away from the other traffic and act on each type separately. In conclusion, we add three VNSF to SFC^1, by **traffic routing** (the two load balancers) and by **traffic splitting** (*IDS*). We also add a monitoring VNSF (*MN*) to SFC^2 by **traffic mirroring**.

The resulting graphs are represented in Figure 6, where the numbers in a rectangle, next to the arrows are the requested bandwidth. Other details describing the service network request, namely the required resources of the newly added VNFs, are detailed in the upper part of Table III, following the service request model proposed in Section III-C1.

TABLE III
SECURITY-AWARE SERVICE REQUEST

$Req = \{G_{SFC^1}, G_{SFC^2}\}$							
$N_{SFC^1} = \{f_0^1, f_1^1, f_2^1, f_3^1, f_4^1, f_5^1, f_\infty^1\}$					$N_{SFC^2} = \{f_0^2, f_1^2, f_2^2, f_3^2, f_\infty^2\}$		
$L_{SFC^1} = \{(f_0^1, f_1^1), (f_1^1, f_2^1), (f_2^1, f_3^1), (f_3^1, f_4^1), (f_4^1, f_\infty^1)(f_1^1, f_5^1), (f_5^1, f_2^1)\}$					$L_{SFC^1} = \{(f_0^1, f_1^1), (f_1^1, f_2^1), (f_2^1, f_3^1), (f_3^1, f_4^1)\}$		
$bw((f_0^1, f_3^1)) = bw((f_4^1, f_2^1)) = bw((f_2^1, f_\infty^1)) = 10$ $bw((f_3^1, f_1^1)) = bw((f_1^1, f_4^1)) = bw((f_3^1, f_5^1)) = bw((f_2^5, f_4^1)) = 5$					$bw((f_0^1, f_1^1)) = bw((f_3^1, f_4^1)) = bw((f_3^1, f_4^1)) = bw((f_4^1, f_\infty^1)) = 20$		
$t(f_3^1)$=LB	$t(f_1^1)$=Video Opt.	$t(f_4^1)$=LB	$t(f_2^1)$=FW	$t(f_5^1)$=IDS	$t(f_1^2)$=Par.Ctrl.	$t(f_2^2)$=FW	$t(f_3^2)$=MN
$c(f_3^1) = 25$	$c(f_1^1) = 25$	$c(f_4^1) = 10$	$c(f_2^1) = 10$	$c(f_5^1) = 20$	$c(f_1^2) = 20$	$c(f_2^2) = 10$	$c(f_3^2) = 20$
f_0^1= P-GW, mapped to node 11					f_0^2= P-GW, mapped to node 8		
f_∞^1= Router, mapped to node 11					f_∞^2= Router, mapped to node 8		
$DontShareInstances(f_3^1) = \{f_4^1\}$, $DontShareInstances(f_4^1) = \{f_3^1\}$, $DontShareInstances(f_1^1) = DontShareInstances(f_2^1) = DontShareInstances(f_5^1) = \{\emptyset\}$					$DontShareInstances(f_1^2) = DontShareInstances(f_2^2) = DontShareInstances(f_3^2) = \{\emptyset\}$		
$ForbidColoc(f_3^1) = \{f_4^1\}$, $ForbidColoc(f_4^1) = \{f_3^1\}$, $ForbidColoc(f_1^1) = ForbidColoc(f_2^1) = ForbidColoc(f_5^1) = \{\emptyset\}$					$ForbidColoc(f_1^2) = ForbidColoc(f_2^2) = ForbidColoc(f_3^2) = \{\emptyset\}$		
$ShouldColoc(f_2^1) = \{f_5^1\}$, $ShouldColoc(f_5^1) = \{f_2^1\}$, $ShouldColoc(f_3^1) = ShouldColoc(f_1^1) = ShouldColoc(f_4^1) = \{\emptyset\}$					$ShouldColoc(f_1^2) = ShouldColoc(f_2^2) = ShouldColoc(f_3^2) = \{\emptyset\}$		
$MayHost(f_1^1) = MayHost(f_2^1) = MayHost(f_3^1) = MayHost(f_4^1) = MayHost(f_5^1) = N_s \setminus \{4, 7\}$					$MayHost(f_3^2) = \{1\}$, $MayHost(f_1^2) = MayHost(f_2^2) = N_s \setminus \{4, 7\}$		
$MayHost((f_0^1, f_3^1)) = MayHost((f_3^1, f_1^1)) = MayHost((f_1^1, f_4^1)) = MayHost((f_4^1, f_2^1)) = MayHost((f_2^1, f_\infty^1)) = MayHost((f_3^1, f_5^1)) = MayHost((f_5^1, f_4^1)) = L_s \setminus \{(4, 7)\}$					$MayHost((f_0^2, f_1^2)) = MayHost((f_1^2, f_1^2)) = MayHost((f_2^2, f_\infty^2)) = MayHost((f_1^2, f_3^2)) = L_s \setminus \{(4, 7)\}$		

*2) **Step 2: defining security deployment constraints**:* We specify the set of constraints related to the resource sharing policies and the security mapping restrictions for each virtual component of the request. An illustration is given in the lower part of Table III. For instance, the firewall and an IDS of SFC^1 should be placed in the same substrate node to collaborate and detect malicious activities rapidly, thus, $ShouldColoc(f_5^1) = \{f_2^1\}$ and $ShouldColoc(f_2^1) = \{f_5^1\}$. Moreover, let us imagine that only the substrate node 1 can provide the monitoring function, so MN of SFC^2 should be mapped to substrate node 1, thus $MayHost(f_3^2) = \{1\}$. Next, let the two load balancers be de-localized and prohibited from sharing instances. Consequently, $DontShareInstances(f_3^1) = \{f_4^1\}$, $DontShareInstances(f_4^1) = \{f_3^1\}$, and $ForbidColoc(f_3^1) = \{f_4^1\}$ and $ForbidColoc(f_4^1) = \{f_3^1\}$. Finally, as depicted in figure 4, the substrate nodes 4 and 7 of the VNFI, and the substrate link connecting them are considered to be a dangerous area with security breaches, thus they should be avoided during the mapping. Therefore, they are excluded from the $MayHost(.)$ set of all the requested virtual components.

D. Security-aware Service Request

Table III illustrates the security-aware request resulting from applying the previous steps on the basic request shown in Table II. The comparison between the Tables III and II shows that the security-aware request is richer and more complex.

E. Service Graph Composition and Embedding

Conceiving algorithms to resolve the Service graph composition and embedding problems described in sections III-E and III-F is out of the scope of this paper. Existing algorithms in the NFV resource allocation literature should be adapted and enhanced to resolve these problems because of the complexity added when considering the security constraints. This complexity can lead to slower computation times, lower query acceptance rates, and higher embedding costs. Figure 7 shows a possible solution of the service graph composition where the firewalls of the two SFCs are merged to utilize two instances of the same image for efficient resource usage. Compared to the initial service request shown in Figure 5, composed of simple linear chains, we can see that the resulting graph topology is more complex.

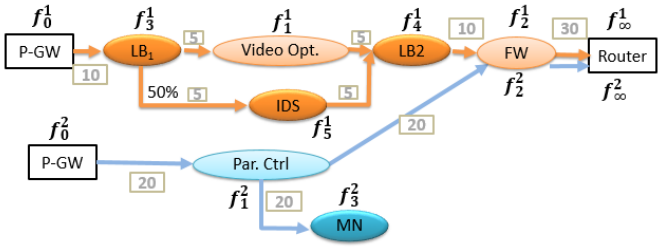


Fig. 7. A possible result of service graph composition

V. RELATED WORK

The resource allocation problem in NFV has already attracted the attention of the research community. But most of the existing work focused on the service function chaining problem, referred to as *the VNF service graph embedding* in this paper. Only a few papers investigated the security aspects of the problem.

Namely, the authors of [13] tackled the problem of efficient *Security Service Chain (SSC)* deployment. A SSC is an ordered set of security functions composing a logical security service. The authors proposed heuristic algorithms to select hosting nodes and establish routing paths. The proposal avoids VNFI resource fragmentation and security service latency. The authors of [14], [15] proposed an Integer Linear Programming (ILP) formulation for placing VNSFs with respect to both *the quality of service requirements* and *the security constraints*. They argue that omitting the QoS requirements by forcing all the traffic to traverse the whole VNSFs chain can cause performance degradation to latency-sensitive applications, especially when traversing computationally-demanding security functions such as IDS. The authors of [16] proposed a three-stage model to solve the VNSFs mapping problem: first, the requested security policies are translated into chained VNSFs; second, the security SFC mapping is formulated as a mixed ILP problem; inally, the MILP problem is solved. The authors of [17] formulated the problem of optimal placement of ordered sequences of security middle-boxes as a traveling purchaser problem. In [18], the same authors designed a scalable algorithm to solve the mapping problem. The solution is based on a multistage approach to cope with the complexity of the problem. Later, in [19], the authors enhanced their work by defining network security defense patterns (NSDP) that capture best security practices and efficiently select deployment options for security functions. The authors of [20] used a modified version of the Best-Fit Decreasing (BFD) algorithm to solve the problem of allocating VNSFs in cloud data centers. In [21], the authors focused on searching optimal placement for VNSFs. The best hosts are the most capable to control the traffic. This is measured by the node centrality that represents the degree of connectivity between nodes.

Existing work tackled the problem of securing a network service request as a resource provisioning problem where the proposals attempt to find an optimal mapping of VNSFs chains. Unlike them, our proposal is a preliminary step where the initial service request is extended to express its security needs before its embedding.

VI. Conclusion

This paper focuses on the security aspects of service request in an NFV environment. It tackles the problem of conceiving a security-aware service request. We propose a two-step model that enriches a basic query description to take into consideration the eventual security needs of the request. The model incorporates well thought-out security parameters to meet most known security needs and recommended best practices. The resulting security-aware request is richer and more complex. Future works include the design of adequate service graph composition and embedding algorithms, which are required to resolve a problem with such complexity.

Acknowledgements

This work is supported by the Celtic-Plus SENDATE-TANDEM project (2016-2019).

References

[1] B. Yi, X. Wang, K. Li, M. Huang *et al.*, "A comprehensive survey of network function virtualization," *Computer Networks*, vol. 133, pp. 212–262, 2018.

[2] D. Kreutz, F. M. Ramos, P. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[3] D. Bhamare, R. Jain, M. Samaka, and A. Erbad, "A survey on service function chaining," *Journal of Network and Computer Applications*, vol. 75, pp. 138–155, 2016.

[4] J. G. Herrera and J. F. Botero, "Resource allocation in nfv: A comprehensive survey," vol. 13, no. 3, 2016, pp. 518–532.

[5] N. F. S. de Sousa, D. A. L. Perez, R. V. Rosa, M. A. S. Santos, and C. E. Rothenberg, "Network service orchestration: A survey," *CoRR*, vol. abs/1803.06596, 2018.

[6] Y. Xie, Z. Liu, S. Wang, and Y. Wang, "Service function chaining resource allocation: A survey," vol. abs/1608.00095, 2016.

[7] "Network Functions Virtualisation (NFV); Management and Orchestration," European Telecommunications Standards Institute, Standard ETSI GS NFV-MAN 001, Dec. 2014, version 1.1.1.

[8] ETSI GS NFV 003, "Network functions virtualisation (nfv); terminology for main concepts in nfv."

[9] P. C. Lin, C. F. Wu, and P. H. Shih, "Optimal placement of network security monitoring functions in nfv-enabled data centers," in *2017 IEEE 7th International Symposium on Cloud and Service Computing (SC2)*, Nov 2017, pp. 9–16.

[10] A. Shameli-Sendi, Y. Jarraya, M. Pourzandi, and M. Cheriet, "Efficient provisioning of security service function chaining using network security defense patterns," *IEEE Transactions on Services Computing*, 2017.

[11] S. Orlowski, M. Pióro, A. Tomaszewski, and R. Wessäly, "SNDlib 1.0–Survivable Network Design Library," in *Proceedings of the 3rd International Network Optimization Conference (INOC 2007), Spa, Belgium*, April 2007, http://sndlib.zib.de, extended version accepted in Networks, 2009. [Online]. Available: http://www.zib.de/orlowski/Paper/OrlowskiPioroTomaszewskiWessaely2007-SNDlib-INOC.pdf.gz

[12] IETF, "Service function chaining use cases in mobile networks," 2014.

[13] Y. Liu, H. Zhang, J. Liu, and Y. Yang, "A new approach for delivering customized security everywhere: Security service chain," in *Security and Communication Networks*, 2017, pp. 1–17.

[14] R. Doriguzzi-Corin, S. Scott-Hayward, D. Siracusa, and E. Salvadori, "Application-centric provisioning of virtual security network functions," in *2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, Nov 2017, pp. 276–279.

[15] R. Doriguzzi-Corin, S. Scott-Hayward, D. Siracusa, M. Savi, and E. Salvadori, "Dynamic and application-aware provisioning of chained virtual security network functions," *CoRR*, vol. abs/1901.01704, 2019. [Online]. Available: http://arxiv.org/abs/1901.01704

[16] C. Basile, C. Pitscheider, F. Risso, F. Valenza, and M. Vallini, "Towards the dynamic provision of virtualized security services," in *Cyber Security and Privacy*, F. Cleary and M. Felici, Eds. Cham: Springer International Publishing, 2015, pp. 65–76.

[17] A. Shameli-Sendi, Y. Jarraya, M. Fekih-Ahmed, M. Pourzandi, C. Talhi, and M. Cheriet, "Optimal placement of sequentially ordered virtual security appliances in the cloud," in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, May 2015, pp. 818–821.

[18] Y. Jarraya, A. Shameli-Sendi, M. Pourzandi, and M. Cheriet, "Multistage ocdo: Scalable security provisioning optimization in sdn-based cloud," in *2015 IEEE 8th International Conference on Cloud Computing*, June 2015, pp. 572–579.

[19] A. Shameli-Sendi, Y. Jarraya, M. Pourzandi, and M. Cheriet, "Efficient provisioning of security service function chaining using network security defense patterns," *IEEE Transactions on Services Computing*, 2016.

[20] A. Ali, C. Anagnostopoulos, and D. P. Pezaros, "Resource-aware placement of softwarised security services in cloud data centers," in *2017 13th International Conference on Network and Service Management (CNSM)*, Nov 2017, pp. 1–5.

[21] J. Guan, Z. Wei, and I. You, "Grbc-based network security functions placement scheme in sds for 5g security," *Journal of Network and Computer Applications*, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1084804518300948