I2NSF Working Group
Internet-Draft
Intended status: Informational
Expires: November 17, 2019

J. Jeong
Sungkyunkwan University
S. Hyun
Chosun University
T. Ahn
Korea Telecom
S. Hares
Huawei
D. Lopez
Telefonica I+D
May 16, 2019

Applicability of Interfaces to Network Security Functions to Network-Based Security Services
draft-ietf-i2nsf-applicability-11

Abstract

This document describes the applicability of Interface to Network Security Functions (I2NSF) to network-based security services in Network Functions Virtualization (NFV) environments, such as firewall, deep packet inspection, or attack mitigation engines.

Table of Contents

1. Introduction

Interface to Network Security Functions (I2NSF) defines a framework and interfaces for interacting with Network Security Functions (NSFs). Note that an NSF is defined as software that provides a set of security-related services, such as (i) detecting unwanted activity, (ii) blocking or mitigating the effect of such unwanted activity in order to fulfil service requirements, and (iii) supporting communication stream integrity and confidentiality [i2nsf-terminology].

The I2NSF framework allows heterogeneous NSFs developed by different security solution vendors to be used in the Network Functions Virtualization (NFV) environment [ETSI-NFV] by utilizing the capabilities of such NSFs through I2NSF interfaces such as Customer-Facing Interface [consumer-facing-inf-dm] and NSF-Facing Interface [nsf-facing-inf-dm]. In the I2NSF framework, each NSF initially

registers the profile of its own capabilities into the Security Controller (i.e., network operator management system [RFC8329]) in the I2NSF system via Registration Interface [registration-inf-dm] so that each NSF can be selected and used to enforce a given security policy from I2NSF User (i.e., network security administrator). Note that Developer's Management System (DMS) is management software that provides a vendor's security service software as a Virtual Network Function (VNF) in an NFV environment (or middlebox in the legacy network) as an NSF, and registers the capabilities of an NSF into Security Controller via Registration Interface for a security service [RFC8329].

Security Controller is defined as a management component that contains control plane functions to manage NSFs and facilitate information sharing among other components (e.g., NSFs and I2NSF User) in an I2NSF system [i2nsf-terminology]. Security Controller maintains the mapping between a capability and an NSF, so it can perform to translate a high-level security policy received from I2NSF User to a low-level security policy configured and enforced in an NSF [policy-translation]. Security Controller can monitor the states and security attacks in NSFs through NSF monitoring [nsf-monitoring-dm].

This document illustrates the applicability of the I2NSF framework with four different scenarios:

1. The enforcement of time-dependent web access control.

2. The application of I2NSF to a Service Function Chaining (SFC) environment [RFC7665].

3. The integration of the I2NSF framework with Software-Defined Networking (SDN) [RFC7149] to provide different security functionality such as firewalls [opsawg-firewalls], Deep Packet Inspection (DPI), and Distributed Denial of Service (DDoS) attack mitigation.

4. The use of Network Functions Virtualization (NFV) [ETSI-NFV] as a supporting technology.

The implementation of I2NSF in these scenarios has allowed us to verify the applicability and effectiveness of the I2NSF framework for a variety of use cases.

2. Terminology

This document uses the terminology described in [RFC7665], [RFC7149], [ITU-T.Y.3300], [ONF-SDN-Architecture], [ITU-T.X.800],

[NFV-Terminology], [RFC8329], and [i2nsf-terminology]. In addition, the following terms are defined below:

- o Software-Defined Networking (SDN): A set of techniques that enables to directly program, orchestrate, control, and manage network resources, which facilitates the design, delivery and operation of network services in a dynamic and scalable manner [ITU-T.Y.3300].

- o Network Function: A functional block within a network infrastructure that has well-defined external interfaces and well-defined functional behavior [NFV-Terminology].

- o Network Security Function (NSF): Software that provides a set of security-related services. Examples include detecting unwanted activity and blocking or mitigating the effect of such unwanted activity in order to fulfil service requirements. The NSF can also help in supporting communication stream integrity and confidentiality [i2nsf-terminology].

- o Network Functions Virtualization (NFV): A principle of separating network functions (or network security functions) from the hardware they run on by using virtual hardware abstraction [NFV-Terminology].

- o Service Function Chaining (SFC): The execution of an ordered set of abstract service functions (i.e., network functions) according to ordering constraints that must be applied to packets, frames, and flows selected as a result of classification. The implied order may not be a linear progression as the architecture allows for SFCs that copy to more than one branch, and also allows for cases where there is flexibility in the order in which service functions need to be applied [RFC7665].

- o Firewall: A service function at the junction of two network segments that inspects some suspicious packets that attempt to cross the boundary. It also rejects any packet that does not satisfy certain criteria for, for example, disallowed port numbers or IP addresses.

- o Centralized Firewall System: A centralized firewall that can establish and distribute policy rules into network resources for efficient firewall management.

- o Centralized VoIP Security System: A centralized security system that handles the security functions required for VoIP and VoLTE services.

- o Centralized DDoS-attack Mitigation System: A centralized mitigator that can establish and distribute access control policy rules into network resources for efficient DDoS-attack mitigation.

```
   +------------+
   | I2NSF User |
   +------------+
          ^
          | Consumer-Facing Interface
          v
+--------------------+     Registration     +-----------------------+
|Security Controller|<-------------------->|Developer's Mgmt System|
+--------------------+      Interface       +-----------------------+
          ^
          | NSF-Facing Interface
          v
   +----------------+ +---------------+   +-----------------------+
   |      NSF-1     |-|     NSF-2     |...|         NSF-n         |
   |   (Firewall)   | | (Web Filter)  |   |(DDoS-Attack Mitigator)|
   +----------------+ +---------------+   +-----------------------+
```

Figure 1: I2NSF Framework

3. I2NSF Framework

This section summarizes the I2NSF framework as defined in [RFC8329]. As shown in Figure 1, an I2NSF User can use security functions by delivering high-level security policies, which specify security requirements that the I2NSF user wants to enforce, to the Security Controller via the Consumer-Facing Interface [consumer-facing-inf-dm].

The Security Controller receives and analyzes the high-level security policies from an I2NSF User, and identifies what types of security capabilities are required to meet these high-level security policies. The Security Controller then identifies NSFs that have the required security capabilities, and generates low-level security policies for each of the NSFs so that the high-level security policies are eventually enforced by those NSFs [policy-translation]. Finally, the Security Controller sends the generated low-level security policies to the NSFs via the NSF-Facing Interface [nsf-facing-inf-dm].

As shown in Figure 1, with a Developer's Management System (called DMS), developers (or vendors) inform the Security Controller of the capabilities of the NSFs through the Registration Interface [registration-inf-dm] for registering (or deregistering) the corresponding NSFs. Note that an inside attacker at the DMS can seriously weaken the I2NSF system's security. That is, DMS can be

compromised to attack the Security Controller by providing the Security Controller with malicious NSFs, and controlling those NSFs in real time. To deal with this type of threat, the role of the DMS should be restricted to providing an I2NSF system with the software package/image for NSF execution, and the DMS should never be able to access NSFs in online/activated status for the I2NSF system's security. On the other hand, an access to active NSFs should be allowed only to the Security Controller, not the DMS during the provisioning time of those NSFs to the I2NSF system. However, note that an inside attacker can access the active NSFs, which are being executed as either VNFs or middleboxes in the I2NSF system, through a back door (i.e., an IP address and a port number that are known to the DMS to control an NSF). However, the Security Controller can detect and prevent inside attacks by monitoring the activities of all the DMSs as well as the NSFs through the I2NSF NSF monitoring functionality [nsf-monitoring-dm]. Through the NSF monitoring, the Security Controller can monitor the activities and states of NSFs, and then can make a diagnosis to see whether the NSFs are working in normal conditions or in abnormal conditions including the insider threat. Note that the monitoring of the DMSs is out of scope for I2NSF.

The Consumer-Facing Interface can be implemented as an XML file based on the Consumer-Facing Interface data model [consumer-facing-inf-dm] along with RESTCONF [RFC8040], which befits a web-based user interface for an I2NSF User to send a Security Controller a high-level security policy. Data models specified by YANG [RFC6020] describe high-level security policies to be specified by an I2NSF User. The data model defined in [consumer-facing-inf-dm] can be used for the I2NSF Consumer-Facing Interface. Note that an inside attacker at the I2NSF User can misuse the I2NSF system so that the network system under the I2NSF system is vulnerable to security attacks. To handle this type of threat, the Security Controller needs to monitor the activities of all the I2NSF Users as well as the NSFs through the I2NSF NSF monitoring functionality [nsf-monitoring-dm]. Note that the monitoring of the I2NSF Users is out of scope for I2NSF.

The NSF-Facing Interface can be implemented as an XML file based on the NSF-Facing Interface YANG data model [nsf-facing-inf-dm] along with NETCONF [RFC6241], which befits a command-line-based remote-procedure call for a Security Controller to configure an NSF with a low-level security policy. Data models specified by YANG [RFC6020] describe low-level security policies for the sake of NSFs, which are translated from the high-level security policies by the Security Controller. The data model defined in [nsf-facing-inf-dm] can be used for the I2NSF NSF-Facing Interface.

The Registration Interface can be implemented as an XML file based on the Registration Interface YANG data model [registration-inf-dm] along with NETCONF [RFC6241], which befits a command-line-based remote-procedure call for a DMS to send a Security Controller an NSF's capability information. Data models specified by YANG [RFC6020] describe the registration of an NSF's capabilities to enforce security services at the NSF. The data model defined in [registration-inf-dm] can be used for the I2NSF Registration Interface.

Also, the I2NSF framework can enforce multiple chained NSFs for the low-level security policies by means of SFC techniques for the I2NSF architecture [RFC7665].

The following sections describe different security service scenarios illustrating the applicability of the I2NSF framework.

4. Time-dependent Web Access Control Service

This service scenario assumes that an enterprise network administrator wants to control the staff members' access to a particular Internet service (e.g., Example.com) during business hours. The following is an example high-level security policy rule for a web filter that the administrator requests: Block the staff members' access to Example.com from 9 AM (i.e., 09:00) to 6 PM (i.e., 18:00) by dropping their packets. Figure 2 is an example XML code for this web filter that is sent from the I2NSF User to the Security Controller via the Consumer-Facing Interface [consumer-facing-inf-dm]:

```
<?xml version="1.0" encoding="UTF-8" ?>
<ietf-i2nsf-cfi-policy:policy>
  <policy-name>block_website</policy-name>
  <rule>
    <rule-name>block_website_during_working_hours</rule-name>
    <event>
      <time-information>
        <begin-time>09:00</begin-time>
        <end-time>18:00</end-time>
      </time-information>
    </event>
    <condition>
      <firewall-condition>
        <source-target>
          <src-target>Staff_Member's_PC</src-target>
        </source-target>
      </firewall-condition>
      <custom-condition>
        <destination-target>
          <dest-target>Example.com</dest-target>
        </destination-target>
      </custom-condition>
    </condition>
    <action>
      <primary-action>drop</primary-action>
    </action>
  </rule>
</ietf-i2nsf-cfi-policy:policy>
```

Figure 2: An XML Example for Time-based Web-filter

The security policy name is "block_website" with the tag "policy-name", and the security policy rule name is "block_website_during_working_hours" with the tag "rule-name". The filtering event has the time span where the filtering begin time is the time "09:00" (i.e., 9:00AM) with the tag "begin-time", and the filtering end time is the time "18:00" (i.e., 6:00PM) with the tag "end-time". The filtering condition has the source target of "Staff_Member's_PC" with the tag "src-target", the destination target of a website "Example.com" with the tag "dest-target". The action is to "drop" the packets satisfying the above event and condition with the tag "primary-action".

After receiving the high-level security policy, the Security Controller identifies required security capabilities, e.g., IP address and port number inspection capabilities and URL inspection capability. In this scenario, it is assumed that the IP address and port number inspection capabilities are required to check whether a

received packet is an HTTP packet from a staff member. The URL inspection capability is required to check whether the target URL of a received packet is in the Example.com domain or not.

The Security Controller maintains the security capabilities of each NSF running in the I2NSF system, which have been reported by the Developer's Management System via the Registration interface. Based on this information, the Security Controller identifies NSFs that can perform the IP address and port number inspection and URL inspection [policy-translation]. In this scenario, it is assumed that a firewall NSF has the IP address and port number inspection capabilities and a web filter NSF has URL inspection capability.

The Security Controller generates low-level security rules for the NSFs to perform IP address and port number inspection, URL inspection, and time checking. Specifically, the Security Controller may interoperate with an access control server in the enterprise network in order to retrieve the information (e.g., IP address in use, company identifier (ID), and role) of each employee that is currently using the network. Based on the retrieved information, the Security Controller generates low-level security rules to check whether the source IP address of a received packet matches any one being used by a staff member. In addition, the low-level security rules should be able to determine that a received packet is of HTTP protocol. The low-level security rules for web filter check that the target URL field of a received packet is equal to Example.com. Finally, the Security Controller sends the low-level security rules of the IP address and port number inspection to the firewall NSF and the low-level rules for URL inspection to the web filter NSF.

The following describes how the time-dependent web access control service is enforced by the NSFs of firewall and web filter.

1. A staff member tries to access Example.com during business hours, e.g., 10 AM.

2. The packet is forwarded from the staff member's device to the firewall, and the firewall checks the source IP address and port number. Now the firewall identifies the received packet is an HTTP packet from the staff member.

3. The firewall triggers the web filter to further inspect the packet, and the packet is forwarded from the firewall to the web filter. SFC technology can be utilized to support such packet forwarding in the I2NSF framework [RFC7665].

4. The web filter checks the target URL field of the received packet, and realizes the packet is toward Example.com. The web

filter then checks that the current time is in business hours. If so, the web filter drops the packet, and consequently the staff member's access to Example.com during business hours is blocked.

```
      +------------+
      | I2NSF User |
      +------------+
             ^
             | Consumer-Facing Interface
             v
   +-------------------+     Registration     +-----------------------+
   |Security Controller|<-------------------->|Developer's Mgmt System|
   +-------------------+      Interface       +-----------------------+
         ^       ^
         |       | NSF-Facing Interface
         |       |-------------------------
         |                                |
         | NSF-Facing Interface           |
   +-----v-----------+             +------v-------+
   |  +-----------+  |      ------>|     NSF-1    |
   |  |Classifier |  |      |      |  (Firewall)  |
   |  +-----------+  |      |      +--------------+
   |     +-----+     |<-----|      +--------------+
   |     | SFF |     |      |----->|     NSF-2    |
   |     +-----+     |      |      |     (DPI)    |
   +-----------------+      |      +--------------+
                            |             .
                            |             .
                            |             .
                            |      +-----------------------+
                            ------>|        NSF-n          |
                                   |(DDoS-Attack Mitigator)|
                                   +-----------------------+
```

Figure 3: An I2NSF Framework with SFC

5. I2NSF Framework with SFC

In the I2NSF architecture, an NSF can trigger an advanced security action (e.g., DPI or DDoS attack mitigation) on a packet based on the result of its own security inspection of the packet. For example, a firewall triggers further inspection of a suspicious packet with DPI. For this advanced security action to be fulfilled, the suspicious packet should be forwarded from the current NSF to the successor NSF. SFC [RFC7665] is a technology that enables this advanced security action by steering a packet with multiple service functions (e.g.,

NSFs), and this technology can be utilized by the I2NSF architecture to support the advanced security action.

Figure 3 shows an I2NSF framework with the support of SFC. As shown in the figure, SFC generally requires classifiers and service function forwarders (SFFs); classifiers are responsible for determining which service function path (SFP) (i.e., an ordered sequence of service functions) a given packet should pass through, according to pre-configured classification rules, and SFFs perform forwarding the given packet to the next service function (e.g., NSF) on the SFP of the packet by referring to their forwarding tables. In the I2NSF architecture with SFC, the Security Controller can take responsibilities of generating classification rules for classifiers and forwarding tables for SFFs. By analyzing high-level security policies from I2NSF users, the Security Controller can construct SFPs that are required to meet the high-level security policies, generates classification rules of the SFPs, and then configures classifiers with the classification rules over NSF-Facing Interface so that relevant traffic packets can follow the SFPs. Also, based on the global view of NSF instances available in the system, the Security Controller constructs forwarding tables, which are required for SFFs to forward a given packet to the next NSF over the SFP, and configures SFFs with those forwarding tables over NSF-Facing Interface.

To trigger an advanced security action in the I2NSF architecture, the current NSF appends metadata describing the security capability required for the advanced action to the suspicious packet to the network service header (NSH) of the packet [RFC8300]. It then sends the packet to the classifier. Based on the metadata information, the classifier searches an SFP which includes an NSF with the required security capability, changes the SFP-related information (e.g., service path identifier and service index [RFC8300]) of the packet with the new SFP that has been found, and then forwards the packet to the SFF. When receiving the packet, the SFF checks the SFP-related information such as the service path identifier and service index contained in the packet and forwards the packet to the next NSF on the SFP of the packet, according to its forwarding table.

```
      +------------+
      | I2NSF User |
      +------------+
             ^
             | Consumer-Facing Interface
             v
   +-------------------+     Registration     +-----------------------+
   |Security Controller|<-------------------->|Developer's Mgmt System|
   +-------------------+      Interface       +-----------------------+
      ^     ^
      |     | NSF-Facing Interface
      |     v
      | +----------------+ +---------------+   +-----------------------+
      | |      NSF-1     |-|     NSF-2     |...|         NSF-n         |
      | |   (Firewall)   | |     (DPI)     |   |(DDoS-Attack Mitigator)|
      | +----------------+ +---------------+   +-----------------------+
      |
      |
      |                                                  SDN Network
   +--|----------------------------------------------------------------+
   |  V NSF-Facing Interface                                           |
   |  +----------------+                                               |
   |  | SDN Controller |                                               |
   |  +----------------+                                               |
   |           ^                                                       |
   |           | SDN Southbound Interface                              |
   |           v                                                       |
   |      +--------+ +------------+ +--------+       +--------+        |
   |      |Switch-1|-|  Switch-2  |-|Switch-3|.......|Switch-m|        |
   |      |        | |(Classifier)| | (SFF)  |       |        |        |
   |      +--------+ +------------+ +--------+       +--------+        |
   +-------------------------------------------------------------------+
```

Figure 4: An I2NSF Framework with SDN Network

6. I2NSF Framework with SDN

This section describes an I2NSF framework with SDN for I2NSF applicability and use cases, such as firewall, deep packet inspection, and DDoS-attack mitigation functions. SDN enables some packet filtering rules to be enforced in network forwarding elements (e.g., switch) by controlling their packet forwarding rules. By taking advantage of this capability of SDN, it is possible to optimize the process of security service enforcement in the I2NSF system. For example, for efficient firewall services, simple packet filtering can be performed by SDN forwarding elements (e.g., switches), and complicated packet filtering based on packet payloads can be performed by a firewall NSF. This optimized firewall using

both SDN forwarding elements and a firewall NSF is more efficient than a firewall where SDN forwarding elements forward all the packets to a firewall NSF for packet filtering. This is because packets to be filtered out can be early dropped by SDN forwarding elements without consuming further network bandwidth due to the forwarding of the packets to the firewall NSF.

Figure 4 shows an I2NSF framework [RFC8329] with SDN networks to support network-based security services. In this system, the enforcement of security policy rules is divided into the SDN forwarding elements (e.g., switch running as either a hardware middle box or a software virtual switch) and NSFs (e.g., firewall running in a form of a virtual network function (VNF) [ETSI-NFV]). Note that NSFs are created or removed by the NFV Management and Orchestration (MANO) [ETSI-NFV-MANO], performing the life-cycle management of NSFs as VNFs. Refer to Section 7 for the detailed discussion of the NSF life-cycle management in the NFV MANO for I2NSF. SDN forwarding elements enforce simple packet filtering rules that can be translated into their packet forwarding rules, whereas NSFs enforce complicated NSF-related security rules requiring the security capabilities of the NSFs. Note that SDN packet forwarding rules are for packet forwarding or filtering by flow table entries at SDN forwarding elements, and NSF rules are for security enforcement at NSFs (e.g., firewall). Thus, simple firewall rules can be enforced by SDN packet forwarding rules at SDN forwarding elements (e.g., switches). For the tasks for security enforcement (e.g., packet filtering), the Security Controller instructs the SDN Controller via NSF-Facing Interface so that SDN forwarding elements can perform the required security services with flow tables under the supervision of the SDN Controller.

As an example, let us consider two different types of security rules: Rule A is a simple packet filtering rule that checks only the IP address and port number of a given packet, whereas rule B is a time-consuming packet inspection rule for analyzing whether an attached file being transmitted over a flow of packets contains malware. Rule A can be translated into packet forwarding rules of SDN forwarding elements and thus be enforced by these elements. In contrast, rule B cannot be enforced by forwarding elements, but it has to be enforced by NSFs with anti-malware capability. Specifically, a flow of packets is forwarded to and reassembled by an NSF to reconstruct the attached file stored in the flow of packets. The NSF then analyzes the file to check the existence of malware. If the file contains malware, the NSF drops the packets.

In an I2NSF framework with SDN, the Security Controller can analyze given security policy rules and automatically determine which of the given security policy rules should be enforced by SDN forwarding

elements and which should be enforced by NSFs. If some of the given rules requires security capabilities that can be provided by SDN forwarding elements, then the Security Controller instructs the SDN Controller via NSF-Facing Interface so that SDN forwarding elements can enforce those security policy rules with flow tables under the supervision of the SDN Controller. Or if some rules require security capabilities that cannot be provided by SDN forwarding elements but by NSFs, then the Security Controller instructs relevant NSFs to enforce those rules.

The distinction between software-based SDN forwarding elements and NSFs, which can both run as virtual network functions (VNFs), may be necessary for some management purposes in this system. Note that an SDN forwarding element (i.e., switch) is a specific type of VNF rather than an NSF because an NSF is for security services rather than for packet forwarding. For this distinction, we can take advantage of the NFV MANO where there is a subsystem that maintains the descriptions of the capabilities each VNF can offer [ETSI-NFV-MANO]. This subsystem can determine whether a given software element (VNF instance) is an NSF or a virtualized SDN switch. For example, if a VNF instance has anti-malware capability according to the description of the VNF, it could be considered as an NSF. A VNF onboarding system [VNF-ONBOARDING] can be used as such a subsystem that maintains the descriptions of each VNF to tell whether a VNF instance is for an NSF or for a virtualized SDN switch.

For the support of SFC in the I2NSF framework with SDN, as shown in Figure 4, network forwarding elements (e.g., switch) can play the role of either SFC Classifier or SFF, which are explained in Section 5. Classifier and SFF have an NSF-Facing Interface with Security Controller. This interface is used to update security service function chaining information for traffic flows. For example, when it needs to update an SFP for a traffic flow in an SDN network, as shown in Figure 4, SFF (denoted as Switch-3) asks Security Controller to update the SFP for the traffic flow (needing another security service as an NSF) via NSF-Facing Interface. This update lets Security Controller ask Classifier (denoted as Switch-2) to update the mapping between the traffic flow and SFP in Classifier via NSF-Facing Interface.

The following subsections introduce three use cases from [RFC8192] for cloud-based security services: (i) firewall system, (ii) deep packet inspection system, and (iii) attack mitigation system.

6.1. Firewall: Centralized Firewall System

A centralized network firewall can manage each network resource and apply common rules to individual network elements (e.g., switch). The centralized network firewall controls each forwarding element, and firewall rules can be added or deleted dynamically.

A time-based firewall can be enforced with packet filtering rules and a time span (e.g., work hours). With this time-based firewall, a time-based security policy can be enforced, as explained in Section 4. For example, employees at a company are allowed to access social networking service websites during lunch time or after work hours.

6.2. Deep Packet Inspection: Centralized VoIP/VoLTE Security System

A centralized VoIP/VoLTE security system can monitor each VoIP/VoLTE flow and manage VoIP/VoLTE security rules, according to the configuration of a VoIP/VoLTE security service called VoIP Intrusion Prevention System (IPS). This centralized VoIP/VoLTE security system controls each switch for the VoIP/VoLTE call flow management by manipulating the rules that can be added, deleted or modified dynamically.

The centralized VoIP/VoLTE security system can cooperate with a network firewall to realize VoIP/VoLTE security service. Specifically, a network firewall performs the basic security check of an unknown flow's packet observed by a switch. If the network firewall detects that the packet is an unknown VoIP call flow's packet that exhibits some suspicious patterns, then it triggers the VoIP/VoLTE security system for more specialized security analysis of the suspicious VoIP call packet.

6.3. Attack Mitigation: Centralized DDoS-attack Mitigation System

A centralized DDoS-attack mitigation can manage each network resource and configure rules to each switch for DDoS-attack mitigation (called DDoS-attack Mitigator) on a common server. The centralized DDoS-attack mitigation system defends servers against DDoS attacks outside the private network, that is, from public networks.

Servers are categorized into stateless servers (e.g., DNS servers) and stateful servers (e.g., web servers). For DDoS-attack mitigation, the forwarding of traffic flows in switches can be dynamically configured such that malicious traffic flows are handled by the paths separated from normal traffic flows in order to minimize the impact of those malicious traffic on the servers. This flow path separation can be done by a flow forwarding path management scheme

based on [AVANT-GUARD]. This management should consider the load balance among the switches for the defense against DDoS attacks.

So far this section has described the three use cases for network-based security services using the I2NSF framework with SDN networks. To support these use cases in the proposed data-driven security service framework, YANG data models described in [consumer-facing-inf-dm], [nsf-facing-inf-dm], and [registration-inf-dm] can be used as Consumer-Facing Interface, NSF-Facing Interface, and Registration Interface, respectively, along with RESTCONF [RFC8040] and NETCONF [RFC6241].

```
                                                  +--------------------+
   +-------------------------------------------+  | ----------------   |
   |            I2NSF User (OSS/BSS)           |  | | NFV          |   |
   +------+------------------------------------+  | | Orchestrator +-+ |
          |  Consumer-Facing Interface            | -----+---------- | |
   +------|------------------------------------+  |      |           | |
   | -----+----------  (a)  -----------------  |  |  ----+-----      | |
   | |  Security    +-------+  Developer's  |  |  |  |        |      | |
   | |Controller(EM)|       |Mgmt System(EM)|  +-(b)-+ VNFM(s)|      | |
   | -----+----------       -----------------  |  |  |        |      | |
   |      |  NSF-Facing Interface              |  |  ----+-----      | |
   |  ----+-----    ----+-----    ----+-----   |  |      |           | |
   |  |NSF(VNF)|    |NSF(VNF)|    |NSF(VNF)|   |  |      |           | |
   |  ----+-----    ----+-----    ----+-----   |  |      |           | |
   |      |             |             |        |  |      |           | |
   +------|-------------|-------------|--------+  |      |           | |
          |             |             |           |      |           | |
   +------+-------------+-------------+--------+  |      |           | |
   |         NFV Infrastructure (NFVI)         |  |      |           | |
   | -----------    -----------    ----------- |  |      |           | |
   | | Virtual |    | Virtual |    | Virtual | |  |      |           | |
   | | Compute |    | Storage |    | Network | |  |      |           | |
   | -----------    -----------    ----------- |  |  ----+-----      | |
   | +-------------------------------------+ |  |  |        |      | |
   | |         Virtualization Layer          | +-----+ VIM(s) +------+ |
   | +-------------------------------------+ |  |  |        |        |
   | +-------------------------------------+ |  |  ----------        |
   | | -----------  -----------  ----------- | |  |                    |
   | | | Compute |  | Storage |  | Network | | |  |                    |
   | | | Hardware|  | Hardware|  | Hardware| | |  |                    |
   | | -----------  -----------  ----------- | |  |                    |
   | |           Hardware Resources          | |  |   NFV Management   |
   | +-------------------------------------+ |  | and Orchestration  |
   |                                           |  |       (MANO)       |
   +-------------------------------------------+  +--------------------+
   (a) = Registration Interface
   (b) = Ve-Vnfm Interface
```

Figure 5: I2NSF Framework Implementation with respect to the NFV Reference Architectural Framework

7. I2NSF Framework with NFV

This section discusses the implementation of the I2NSF framework using Network Functions Virtualization (NFV).

NFV is a promising technology for improving the elasticity and efficiency of network resource utilization. In NFV environments,

NSFs can be deployed in the forms of software-based virtual instances rather than physical appliances. Virtualizing NSFs makes it possible to rapidly and flexibly respond to the amount of service requests by dynamically increasing or decreasing the number of NSF instances. Moreover, NFV technology facilitates flexibly including or excluding NSFs from multiple security solution vendors according to the changes on security requirements. In order to take advantages of the NFV technology, the I2NSF framework can be implemented on top of an NFV infrastructure as show in Figure 5.

Figure 5 shows an I2NSF framework implementation based on the NFV reference architecture that the European Telecommunications Standards Institute (ETSI) defines [ETSI-NFV]. The NSFs are deployed as virtual network functions (VNFs) in Figure 5. The Developer's Management System (DMS) in the I2NSF framework is responsible for registering capability information of NSFs into the Security Controller. However, those NSFs are created or removed by a virtual network functions manager (VNFM) in the NFV MANO that performs the life-cycle management of VNFs. Note that the life-cycle management of VNFs are out of scope for I2NSF. The Security Controller controls and monitors the configurations (e.g., function parameters and security policy rules) of VNFs via NSF-Facing Interface along with NSF monitoring capability [nsf-facing-inf-dm][nsf-monitoring-dm]. Both the DMS and Security Controller can be implemented as the Element Managements (EMs) in the NFV architecture. Finally, the I2NSF User can be implemented as OSS/BSS (Operational Support Systems/Business Support Systems) in the NFV architecture that provides interfaces for users in the NFV system.

The operation procedure in the I2NSF framework based on the NFV architecture is as follows:

1. The VNFM has a set of virtual machine (VM) images of NSFs, and each VM image can be used to create an NSF instance that provides a set of security capabilities. The DMS initially registers a mapping table of the ID of each VM image and the set of capabilities that can be provided by an NSF instance created from the VM image into the Security Controller.

2. If the Security Controller does not have any instantiated NSF that has the set of capabilities required to meet the security requirements from users, it searches the mapping table (registered by the DMS) for the VM image ID corresponding to the required set of capabilities.

3. The Security Controller requests the DMS to instantiate an NSF with the VM image ID via VNFM.

4. When receiving the instantiation request, the VNFM first asks the NFV orchestrator for the permission required to create the NSF instance, requests the VIM to allocate resources for the NSF instance, and finally creates the NSF instance based on the allocated resources.

5. Once the NSF instance has been created by the VNFM, the DMS performs the initial configurations of the NSF instance and then notifies the Security Controller of the NSF instance.

6. After being notified of the created NSF instance, the Security Controller delivers low-level security policy rules to the NSF instance for policy enforcement.

We can conclude that the I2NSF framework can be implemented based on the NFV architecture framework. Note that the registration of the capabilities of NSFs is performed through the Registration Interface and the lifecycle management for NSFs (VNFs) is performed through the Ve-Vnfm interface between the DMS and VNFM, as shown in Figure 5.

8. Security Considerations

The same security considerations for the I2NSF framework [RFC8329] are applicable to this document.

This document shares all the security issues of SDN that are specified in the "Security Considerations" section of [ITU-T.Y.3300].

9. Acknowledgments

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (No.R-20160222-002755, Cloud based Security Intelligence Technology Development for the Customized Security Service Provisioning).

This work has been partially supported by the European Commission under Horizon 2020 grant agreement no. 700199 "Securing against intruders and other threats through a NFV-enabled environment (SHIELD)". This support does not imply endorsement.

10. Contributors

I2NSF is a group effort. I2NSF has had a number of contributing authors. The following are considered co-authors:

o Hyoungshick Kim (Sungkyunkwan University)

- Jinyong Tim Kim (Sungkyunkwan University)
- Hyunsik Yang (Soongsil University)
- Younghan Kim (Soongsil University)
- Jung-Soo Park (ETRI)
- Se-Hui Lee (Korea Telecom)
- Mohamed Boucadair (Orange)

11. References

11.1. Normative References

[ETSI-NFV]
"Network Functions Virtualisation (NFV); Architectural Framework", Available: https://www.etsi.org/deliver/etsi_gs/nfv/001_099/002/01.01.01_60/gs_nfv002v010101p.pdf, October 2013.

[ITU-T.Y.3300]
"Framework of Software-Defined Networking", Available: https://www.itu.int/rec/T-REC-Y.3300-201406-I, June 2014.

[NFV-Terminology]
"Network Functions Virtualisation (NFV); Terminology for Main Concepts in NFV", Available: https://www.etsi.org/deliver/etsi_gs/NFV/001_099/003/01.02.01_60/gs_nfv003v010201p.pdf, December 2014.

[ONF-SDN-Architecture]
"SDN Architecture (Issue 1.1)", Available: https://www.opennetworking.org/wp-content/uploads/2014/10/TR-521_SDN_Architecture_issue_1.1.pdf, June 2016.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC7149] Boucadair, M. and C. Jacquenet, "Software-Defined Networking: A Perspective from within a Service Provider Environment", RFC 7149, March 2014.

[RFC7665] Halpern, J. and C. Pignataro, "Service Function Chaining (SFC) Architecture", RFC 7665, October 2015.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, January 2017.

[RFC8192] Hares, S., Lopez, D., Zarny, M., Jacquenet, C., Kumar, R., and J. Jeong, "Interface to Network Security Functions (I2NSF): Problem Statement and Use Cases", RFC 8192, July 2017.

[RFC8300] Quinn, P., Elzur, U., and C. Pignataro, "Network Service Header (NSH)", RFC 8300, January 2018.

[RFC8329] Lopez, D., Lopez, E., Dunbar, L., Strassner, J., and R. Kumar, "Framework for Interface to Network Security Functions", RFC 8329, February 2018.

11.2. Informative References

[AVANT-GUARD]
Shin, S., Yegneswaran, V., Porras, P., and G. Gu, "AVANT-GUARD: Scalable and Vigilant Switch Flow Management in Software-Defined Networks", ACM CCS, November 2013.

[consumer-facing-inf-dm]
Jeong, J., Kim, E., Ahn, T., Kumar, R., and S. Hares, "I2NSF Consumer-Facing Interface YANG Data Model", draft-ietf-i2nsf-consumer-facing-interface-dm-04 (work in progress), April 2019.

[ETSI-NFV-MANO]
"Network Functions Virtualisation (NFV); Management and Orchestration", Available: https://www.etsi.org/deliver/etsi_gs/nfv-man/001_099/001/01.01.01_60/gs_nfv-man001v010101p.pdf, December 2014.

[i2nsf-terminology]
Hares, S., Strassner, J., Lopez, D., Xia, L., and H. Birkholz, "Interface to Network Security Functions (I2NSF) Terminology", draft-ietf-i2nsf-terminology-07 (work in progress), January 2019.

[ITU-T.X.800]
"Security Architecture for Open Systems Interconnection for CCITT Applications", March 1991.

[nsf-facing-inf-dm]
Kim, J., Jeong, J., Park, J., Hares, S., and Q. Lin, "I2NSF Network Security Function-Facing Interface YANG Data Model", draft-ietf-i2nsf-nsf-facing-interface-dm-05 (work in progress), March 2019.

[nsf-monitoring-dm]
Jeong, J., Chung, C., Hares, S., Xia, L., and H. Birkholz, "I2NSF NSF Monitoring YANG Data Model", draft-ietf-i2nsf-nsf-monitoring-data-model-00 (work in progress), March 2019.

[opsawg-firewalls]
Baker, F. and P. Hoffman, "On Firewalls in Internet Security", draft-ietf-opsawg-firewalls-01 (work in progress), October 2012.

[policy-translation]
Yang, J., Jeong, J., and J. Kim, "Security Policy Translation in Interface to Network Security Functions", draft-yang-i2nsf-security-policy-translation-03 (work in progress), March 2019.

[registration-inf-dm]
Hyun, S., Jeong, J., Roh, T., Wi, S., and J. Park, "I2NSF Registration Interface YANG Data Model", draft-ietf-i2nsf-registration-interface-dm-03 (work in progress), March 2019.

[VNF-ONBOARDING]
"VNF Onboarding", Available: https://wiki.opnfv.org/display/mano/VNF+Onboarding, November 2016.

Appendix A. Changes from draft-ietf-i2nsf-applicability-10

The following changes have been made from draft-ietf-i2nsf-applicability-10:

- o In Section 1, "Network Security Function (NSF)" is replaced with "an NSF" because the abbreviation of "Network Security Function" is defined as "NSF" in the previous sentence.

- o In Section 2, a typo in "funcional block" is corrected as "functional block".

Authors' Addresses

Jaehoon Paul Jeong
Department of Software
Sungkyunkwan University
2066 Seobu-Ro, Jangan-Gu
Suwon, Gyeonggi-Do 16419
Republic of Korea

Phone: +82 31 299 4957
Fax: +82 31 290 7996
EMail: pauljeong@skku.edu
URI: http://iotlab.skku.edu/people-jaehoon-jeong.php

Sangwon Hyun
Department of Computer Engineering
Chosun University
309 Pilmun-daero, Dong-Gu
Gwangju 61452
Republic of Korea

Phone: +82 62 230 7473
EMail: shyun@chosun.ac.kr

Tae-Jin Ahn
Korea Telecom
70 Yuseong-Ro, Yuseong-Gu
Daejeon 305-811
Republic of Korea

Phone: +82 42 870 8409
EMail: taejin.ahn@kt.com

Susan Hares
Huawei
7453 Hickory Hill
Saline, MI 48176
USA

Phone: +1-734-604-0332
EMail: shares@ndzh.com

Diego R. Lopez
Telefonica I+D
Jose Manuel Lara, 9
Seville 41013
Spain

Phone: +34 682 051 091
EMail: diego.r.lopez@telefonica.com