Workgroup: I2NSF Working Group
Internet-Draft:
draft-ietf-i2nsf-registration-interface-dm-22
Published: 8 November 2022
Intended Status: Standards Track
Expires: 12 May 2023
Authors: S. Hyun, Ed.
Myongji University
J. Jeong, Ed.
Sungkyunkwan University
T. Roh
Sungkyunkwan University
S. Wi
Sungkyunkwan University
J. Park
ETRI

I2NSF Registration Interface YANG Data Model for NSF Capability Registration

Abstract

This document defines an information model and a YANG data model for the Registration Interface between Security Controller and Developer's Management System (DMS) in the Interface to Network Security Functions (I2NSF) framework to register Network Security Functions (NSF) of the DMS with the Security Controller. The objective of these information and data models is to support NSF capability registration and query via I2NSF Registration Interface.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 12 May 2023.

Copyright Notice

Table of Contents

1. Introduction
2. Terminology
3. Objectives
4. Information Model
  4.1. NSF Capability Registration
    4.1.1. NSF Capability Information
    4.1.2. NSF Access Information
  4.2. NSF Capability Query
5. Data Model
  5.1. YANG Tree Diagram
    5.1.1. Definitions of Symbols in Tree Diagrams
    5.1.2. YANG Tree of I2NSF Registration Interface
  5.2. YANG Data Module
6. IANA Considerations
7. Security Considerations
8. References
  8.1. Normative References
  8.2. Informative References
Appendix A. XML Examples of an NSF Registration with I2NSF Registration Interface Data Model
Appendix B. XML Examples of an NSF Query with I2NSF Registration Interface Data Model
Appendix C. NSF Lifecycle Management in NFV Environments
Appendix D. Acknowledgments
Appendix E. Contributors
Appendix F. Changes from draft-ietf-i2nsf-registration-interface-dm-21
Authors' Addresses

1. Introduction

A number of Network Security Functions (NSF) may exist in the Interface to Network Security Functions (I2NSF) framework [RFC8329]. Since each of these NSFs likely has different security capabilities from each other, it is important to register the security capabilities of the NSFs to the Security Controller (i.e., Network Management Operator System). In addition, it is required to search NSFs of some required security capabilities on demand. As an example, if additional security capabilities are required to serve

some security service request(s) from an I2NSF User, the security controller SHOULD be able to request the DMS for NSFs that have the required security capabilities.

As the main focus of the YANG module defined in [I-D.ietf-i2nsf-capability-data-model] is to define the security capabilities of an NSF, it lacks in some information (e.g., network access information to an NSF) needed by the Security Controller. This information can be provided by the DMS as it is the vendor system that provides and deploys the NSFs. Hence, this document provides extended information for the I2NSF Registration Interface.

This document describes an information model (see Section 4) and an augmented YANG [RFC7950] data model from I2NSF Capability YANG data model [I-D.ietf-i2nsf-capability-data-model] (see Section 5) for the I2NSF Registration Interface [RFC8329] between the Security Controller and the developer's management system (DMS) to support NSF capability registration and query via the registration interface. It also describes the operations which SHOULD be performed by the Security Controller and the DMS via the Registration Interface using the defined model. Note that in either NETCONF [RFC6241] or RESTCONF [RFC8040] parlance through the I2NSF Registration Interface, the Security Controller is the server, and the DMS is the client because the Security Controller and DMS run the server and client for either NETCONF or RESTCONF, respectively.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119][RFC8174] when, and only when, they appear in all capitals, as shown here.

This document uses the following terms defined in [RFC3444], [RFC8329] and [I-D.ietf-i2nsf-capability-data-model].

* Network Security Function (NSF): A function that is responsible for a specific treatment of received packets. A Network Security Function can act at various layers of a protocol stack (e.g., at the network layer or other OSI layers). Sample Network Security Service Functions are as follows: Firewall, Intrusion Prevention/ Detection System (IPS/IDS), Deep Packet Inspection (DPI), Application Visibility and Control (AVC), network virus and malware scanning, sandbox, Data Loss Prevention (DLP), Distributed Denial of Service (DDoS) mitigation and TLS proxy.

* Data Model: Data Models define managed objects at a lower level of abstraction, which include implementation- and protocol-

specific details, e.g., rules that explain how to map managed objects onto lower-level protocol constructs [RFC3444].

*Information Model: Information Models are primarily useful for designers to describe the managed environment, for operators to understand the modeled objects, and for implementers as a guide to the functionality that must be described and coded in the Data Models [RFC3444].

*YANG: This document follows the guidelines of [RFC8407], uses the common YANG types defined in [RFC6991], and adopts the Network Management Datastore Architecture (NMDA) [RFC8342]. The meaning of the symbols in tree diagrams is defined in [RFC8340].

3. Objectives

*Registering NSFs with the I2NSF framework: Developer's Management System (DMS) in I2NSF framework is typically run by an NSF vendor, and uses Registration Interface to provide NSFs information (i.e., capability, specification, and access information) developed by the NSF vendor to Security Controller. Since there may be multiple vendors that provide NSFs for a target network, the I2NSF Registration Interface can be used as a standard interface for the DMSs to provide NSFs capability information to the Security Controller. For the registered NSFs, Security Controller maintains a catalog of the capabilities of those NSFs to select appropriate NSFs for the requested security services.

*Updating the capabilities of registered NSFs: After an NSF is registered with Security Controller, some modifications on the capability of the NSF MAY be required later. In this case, DMS uses Registration Interface to deliver the update of the capability of the NSF to the Security Controller, and this update MUST be reflected on the catalog of NSFs existing in the Security Controller. That is, whenever there are updates on the NSFs, the DMS sends the updated, whole NSF capability information to the Security Controller. The Security Controller updates its catalog of NSFs with the updated NSF capability information.

*Asking DMS about some required capabilities: In cases that some security capabilities are required to serve the security service request from an I2NSF User, the Security Controller searches through the registered NSFs to find ones that can provide the required capabilities. But Security Controller might fail to find any NSFs having the required capabilities among the registered NSFs. In this case, Security Controller needs to request DMS for additional NSF(s) information that can provide the required security capabilities via Registration Interface.

4. Information Model

The I2NSF registration interface is used by Security Controller and Developer's Management System (DMS) in I2NSF framework. Figure 1 shows the information model of the I2NSF registration interface, which consists of two submodels: NSF capability registration and NSF capability query. Each submodel is used for the operations listed above. The remainder of this section will provide in-depth explanation of each submodel.

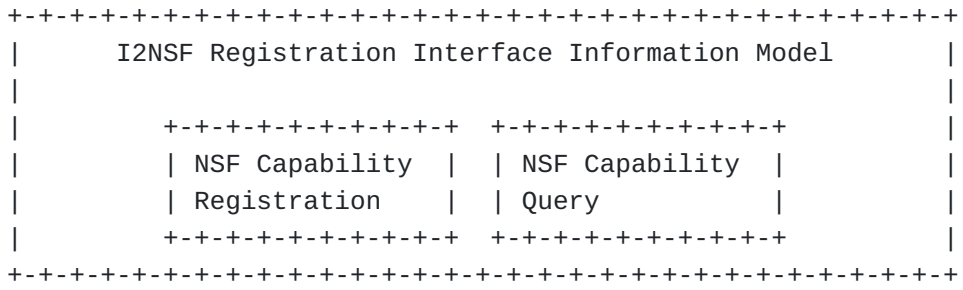

Figure 1: I2NSF Registration Interface Information Model

4.1. NSF Capability Registration

This submodel is used by the DMS to register the capabilities of NSFs with the Security Controller. Figure 2 shows how this submodel is constructed. The most important part in Figure 2 is the NSF capability, and this specifies the set of capabilities that the NSF to be registered can offer. The NSF Name contains a unique name of this NSF with the specified set of capabilities. The NSF name MUST be unique within the registered NSFs in the Security Controller to identify the NSF with the capability. The name can be an arbitrary string including Fully Qualified Domain Name (FQDN). To make sure each vendor does not provide a duplicated name, the name should include the vendor's name (e.g., firewall-cisco, firewall-huawei). When registering the NSF, DMS additionally includes the network access information of the NSF which is required to enable network communications with the NSF.

The following will further explain the NSF capability information and the NSF access information in more detail.

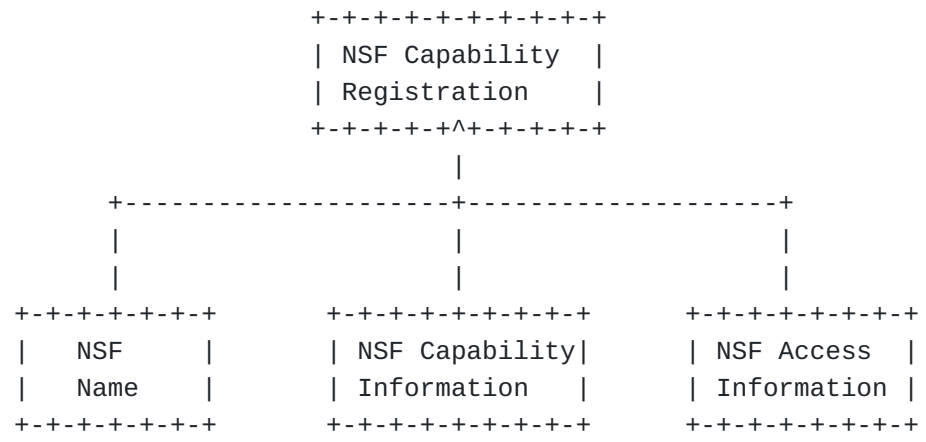

Figure 2: NSF Capability Registration Sub-Model

4.1.1. NSF Capability Information

NSF Capability Information basically describes the security capabilities of an NSF. In Figure 3, we show capability objects of an NSF. Following the information model of NSF capabilities defined in [I-D.ietf-i2nsf-capability-data-model], we share the same I2NSF security capabilities: Directional Capabilities, Event Capabilities, Condition Capabilities, Action Capabilities, Resolution Strategy Capabilities, Default Action Capabilities. Also, NSF Capability Information additionally contains the specification of an NSF as shown in Figure 3.

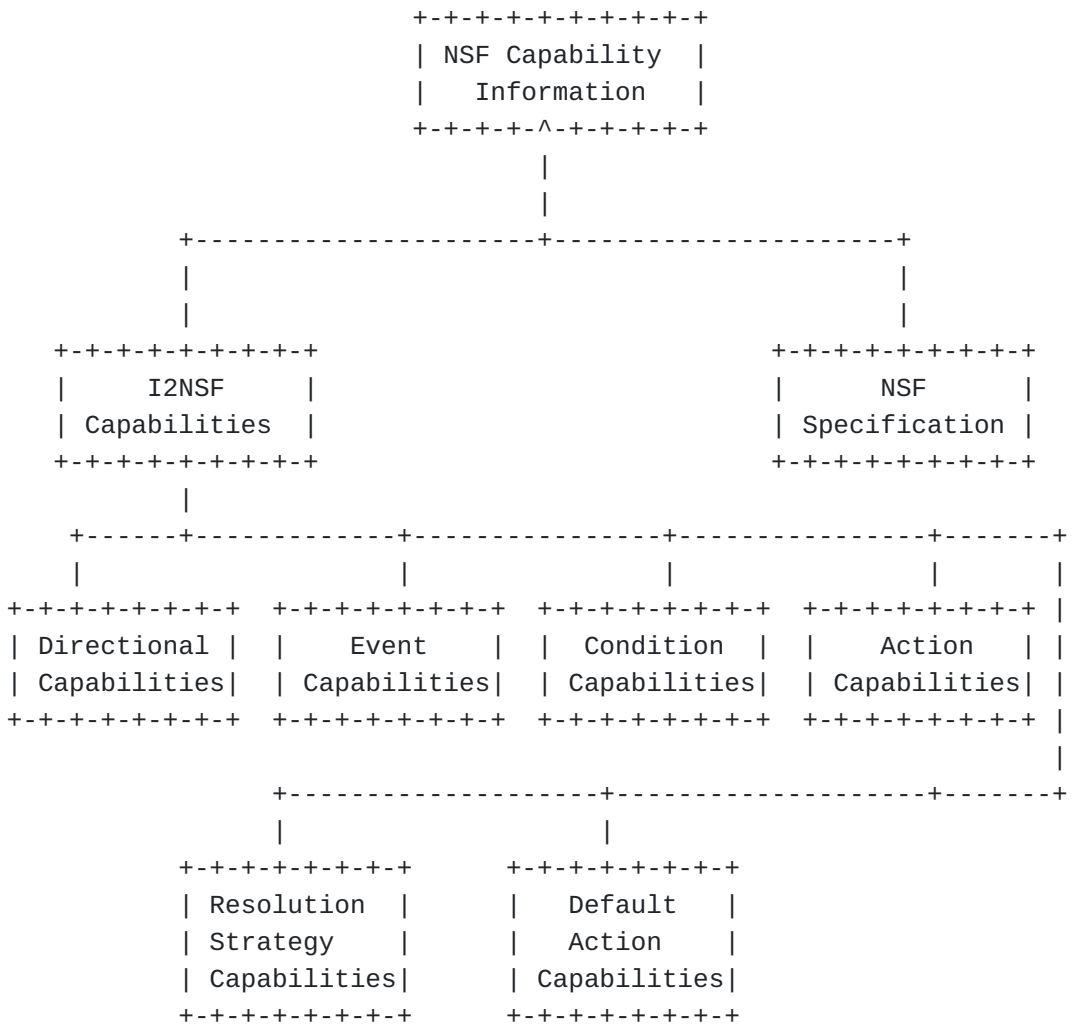

Figure 3: NSF Capability Information

4.1.1.1. NSF Specification

This information represents the specification information (e.g., CPU, memory, disk, and bandwidth) of an NSF. As illustrated in Figure 4, this information consists of CPU, memory, disk, and bandwidth. The CPU information describes the Central Processing Unit (CPU) used by the NSF. The information consists of model name, cores, clock speed, and threads.

The memory information describes the hardware that stores information temporarily, i.e., Random Access Memory (RAM). The information consists of RAM maximum capacity and RAM speed. The disk information describes the storage information, i.e., Hard Disk and Solid-State Drive. The information given is the maximum capacity of the storage available in the NSF.

Bandwidth describes the information about available network amount in two cases, such as outbound and inbound. Assuming that the

current throughput status of each NSF is being collected through NSF monitoring [I-D.ietf-i2nsf-nsf-monitoring-data-model], this capability information of the NSF can be used to determine whether the NSF is in congestion or not by comparing it with the current throughput of the NSF.

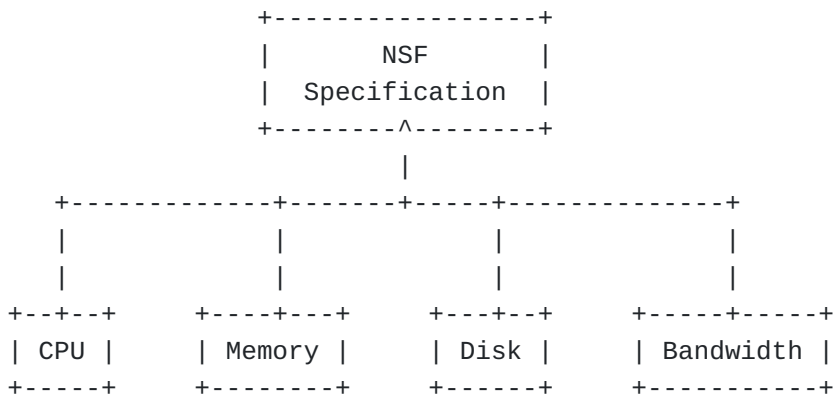

Figure 4: NSF Specification Overview

4.1.2. NSF Access Information

NSF Access Information contains the following that are required to communicate with an NSF through NETCONF [RFC6241] or RESTCONF [RFC8040]: an IP address (i.e., IPv4 or IPv6 address) and a port number. Note that TCP is used as a transport layer protocol due to either NETCONF or RESTCONF. In this document, NSF Access Information is used to identify a specific NSF instance. That is, NSF Access Information is the signature (i.e., unique identifier) of an NSF instance in the overall I2NSF system.

4.2. NSF Capability Query

Security Controller MAY require some additional capabilities to serve the security service request from an I2NSF User, but none of the registered NSFs has the required capabilities. In this case, Security Controller makes a description of the required capabilities by using the NSF capability information submodel in Section 4.1.1, and sends DMS a query about which NSF(s) can provide these capabilities.

5. Data Model

5.1. YANG Tree Diagram

This section provides the YANG Tree diagram of the I2NSF registration interface.

5.1.1. Definitions of Symbols in Tree Diagrams

A simplified graphical representation of the data model is used in this section. The meaning of the symbols used in the following diagrams [RFC8431] is as follows:

Brackets "[" and "]" enclose list keys.

Abbreviations before data node names: "rw" means configuration (read-write) and "ro" state data (read-only).

Symbols after data node names: "?" means an optional node and "*" denotes a "list" and "leaf-list".

Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").

Ellipsis ("...") stands for contents of subtrees that are not shown.

5.1.2. YANG Tree of I2NSF Registration Interface

The I2NSF Registration Interface is used by the Developer's Management System (DMS) to register NSFs and their capabilities with the Security Controller. In case that the Security Controller fails to find any NSF among the registered NSFs which can provide some required capabilities, Security Controller uses the registration interface to query DMS about NSF(s) having the required capabilities. The following sections describe the YANG data models to support these operations.

5.1.2.1. NSF Capability Registration

This section describes the YANG tree for the NSF capability registration and capability update.

```
NSF Capability Registration
augment /i2nsfcap:nsf:
  +--rw nsf-specification
  |  +--rw cpu
  |  |  +--rw model?         string
  |  |  +--rw clock-speed?   uint16
  |  |  +--rw cores?         uint8
  |  |  +--rw threads?       uint16
  |  +--rw memory
  |  |  +--rw capacity?   uint32
  |  |  +--rw speed?      uint32
  |  +--rw disk
  |  |  +--rw capacity?   uint32
  |  +--rw bandwidth
  |     +--rw outbound?   uint64
  |     +--rw inbound?    uint64
  +--rw nsf-access-info
     +--rw ip?                    union
     +--rw port?                  inet:port-number
     +--rw management-protocol?   enumeration
     +--rw name?                  string
     +--rw password?              ianach:crypt-hash
```

Figure 5: YANG Tree of NSF Capability Registration Module

When registering an NSF with Security Controller, DMS uses the augmented I2NSF Capability YANG Data Model [I-D.ietf-i2nsf-capability-data-model] to describe what capabilities the NSF can offer. DMS includes the access information of the NSF which is required to make a network connection with the NSF as well as the specification of the NSFs. The NSF access information consists of ip, port, and management-protocol. The field of ip can have either an IPv4 address or an IPv6 address. The port field is used to get the transport protocol port number. As I2NSF uses a YANG data model, the management protocol can be either NETCONF or RESTCONF. The credentials (i.e., username and password)

The DMS can also include the resource information in terms of CPU, memory, disk, and network bandwidth of the NSF. Detailed overview of NSF specification can be seen in Section 4.1.1.1.

This YANG data model is also used to update the registered NSFs. The update operation can be done by changing the configuration information of the same key (i.e., nsf-name) and utilize the "replace" operation for NETCONF defined in Section 7.2 of [RFC6241] or the "PUT" method for RESTCONF defined in Section 4.5 of [RFC8040].

5.1.2.2. NSF Capability Query

This section describe the YANG tree for the NSF capability query.

```
I2NSF Capability Query
rpcs:
  +---x nsf-capability-query
     +---w input
     |  +---w query-nsf-capability
     |     +---w directional-capabilities*           identityref
     |     +---w event-capabilities
     |     |  +---w system-event-capability*   identityref
     |     |  +---w system-alarm-capability*   identityref
     |     +---w condition-capabilities
     |     |  +---w generic-nsf-capabilities
     |     |  |  +---w ethernet-capability*   identityref
     |     |  |  +---w ipv4-capability*       identityref
     |     |  |  +---w ipv6-capability*       identityref
     |     |  |  +---w icmpv4-capability*     identityref
     |     |  |  +---w icmpv6-capability*     identityref
     |     |  |  +---w tcp-capability*        identityref
     |     |  |  +---w udp-capability*        identityref
     |     |  |  +---w sctp-capability*       identityref
     |     |  |  +---w dccp-capability*       identityref
     |     |  +---w advanced-nsf-capabilities
     |     |  |  +---w anti-ddos-capability*             identityref
     |     |  |  +---w ips-capability*                   identityref
     |     |  |  +---w anti-virus-capability*            identityref
     |     |  |  +---w url-filtering-capability*         identityref
     |     |  |  +---w voip-vocn-filtering-capability*   identityref
     |     |  +---w context-capabilities
     |     |     +---w time-capabilities*                 identityref
     |     |     +---w application-filter-capabilities*   identityref
     |     |     +---w device-type-capabilities*          identityref
     |     |     +---w user-condition-capabilities*       identityref
     |     |     +---w geographic-capabilities*           identityref
     |     +---w action-capabilities
     |     |  +---w ingress-action-capability*   identityref
     |     |  +---w egress-action-capability*    identityref
     |     |  +---w log-action-capability*       identityref
     |     +---w resolution-strategy-capabilities*   identityref
     |     +---w default-action-capabilities*        identityref
     +--ro output
        +--ro nsf-access-info
           +--ro nsf-name?              string
           +--ro ip?                    union
           +--ro port?                  inet:port-number
           +--ro management-protocol?   enumeration
           +--ro name?                  string
           +--ro password?              ianach:crypt-hash
```

Figure 6: YANG Tree of NSF Capability Query Module

Security Controller MAY require some additional capabilities to provide the security service requested by an I2NSF User, but none of the registered NSFs has the required capabilities. In this case, Security Controller makes a description of the required capabilities using this module and then queries DMS about which NSF(s) can provide these capabilities. Use NETCONF RPCs to send a NSF capability query. Input data is query-i2nsf-capability-info and output data is nsf-access-info. In Figure 6, the ietf-i2nsf-capability refers to the module defined in [I-D.ietf-i2nsf-capability-data-model].

5.2. YANG Data Module

This section provides a YANG module of the data model for the registration interface between Security Controller and Developer's Management System, as defined in Section 4.

This YANG module imports from [RFC6991], [RFC7317], and [I-D.ietf-i2nsf-capability-data-model]. It makes references to [RFC6241] [RFC8040]

```
<CODE BEGINS> file "ietf-i2nsf-registration-interface@2022-11-08.yang"

module ietf-i2nsf-registration-interface {
  yang-version 1.1;

  namespace
    "urn:ietf:params:xml:ns:yang:ietf-i2nsf-registration-interface";

  prefix
    i2nsfri;

  //RFC Ed.: replace occurences of XXXX with actual RFC number and
  //remove this note

  import ietf-inet-types {
    prefix inet;
    reference "RFC 6991";
  }
  import iana-crypt-hash {
    prefix ianach;
    reference "RFC 7317";
  }
  import ietf-i2nsf-capability {
    prefix i2nsfcap;
  // RFC Ed.: replace YYYY with actual RFC number of
  // draft-ietf-i2nsf-capability-data-model and remove this note.
    reference "RFC YYYY: I2NSF Capability YANG Data Model";
  }

  organization
   "IETF I2NSF (Interface to Network Security Functions)
    Working Group";

  contact
    "WG Web: <https://datatracker.ietf.org/wg/i2nsf>
     WG List: <mailto:i2nsf@ietf.org>

     Editor: Sangwon Hyun
     <mailto:shyun@mju.ac.kr>

     Editor: Jaehoon Paul Jeong
     <mailto:pauljeong@skku.edu>";

  description
    "This module defines a YANG data model for I2NSF
     Registration Interface.

     The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL',
     'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED',
     'NOT RECOMMENDED', 'MAY', and 'OPTIONAL' in this
```

```
   document are to be interpreted as described in BCP 14
   (RFC 2119) (RFC 8174) when, and only when, they appear
   in all capitals, as shown here.

   Copyright (c) 2022 IETF Trust and the persons
   identified as authors of the code. All rights reserved.

   Redistribution and use in source and binary forms, with or
   without modification, is permitted pursuant to, and subject
   to the license terms contained in, the Revised BSD License
   set forth in Section 4.c of the IETF Trust's Legal Provisions
   Relating to IETF Documents
   (https://trustee.ietf.org/license-info).

   This version of this YANG module is part of RFC XXXX; see
   the RFC itself for full legal notices.";

 revision "2022-11-08" {
   description "Initial revision";
   reference
     "RFC XXXX: I2NSF Registration Interface YANG Data Model";
   // RFC Ed.: replace XXXX with actual RFC number and remove
   // this note
 }

 grouping nsf-specification {
   description
     "Description of the specification of an NSF";

   container cpu {
     description
       "The Central Processing Unit (CPU) specification of the NSF";

     leaf model {
       type string;
       description
         "The model name of the CPU used in the NSF.";
     }
     leaf clock-speed {
       type uint16;
       units "MHz";
       description
         "The number of cycles the CPU executes per second,
          measured in MHz (MegaHertz).";
     }
     leaf cores {
       type uint8;
       description
         "The number of independent CPU in a single computing
```

```
       component.";
  }
  leaf threads {
    type uint16;
    description
      "The number of total threads of the CPU";
  }
}

container memory {
  description
    "Memory (i.e., Random Access Memory (RAM)) specification of
     an NSF.";

  leaf capacity {
    type uint32;
    units "MB";
    description
      "The total memory capacity in Megabytes (MB).";
  }
  leaf speed {
    type uint32;
    units "MHz";
    description
      "The data transfer rate of the memory in MegaHertz (MHz).";
  }
}

container disk {
  description
    "Disk or storage specification of an NSF";

  leaf capacity {
    type uint32;
    units "MB";
    description
      "The disk or storage maximum capacity in Megabytes (MB).";
  }
}

container bandwidth {
  description
    "Network bandwidth available on an NSF
     in the unit of Bps (Bytes per second)";

  leaf outbound {
    type uint64;
    units "Bps";
    description
```

```
        "The maximum outbound network bandwidth available to the
         NSF in bytes per second (Bps)";
    }

    leaf inbound {
      type uint64;
      units "Bps";
      description
        "The maximum inbound network bandwidth available to the
         NSF in bytes per second (Bps)";
    }
  }
}

grouping nsf-access-info {
  description
    "Information required to access an NSF";
  leaf ip {
    type union {
      type inet:ip-address-no-zone;
      type inet:domain-name;
    }
    description
      "Either an IP (IPv4 or IPv6) address or the domain name of
       this NSF";
  }
  leaf port {
    type inet:port-number;
    description
      "Port available on this NSF";
  }
  leaf management-protocol {
    type enumeration {
      enum NETCONF {
        description
          "Represents the management protocol NETCONF";
        reference
          "RFC 6241: Network Configuration Protocol (NETCONF)";
      }
      enum RESTCONF {
        description
          "Represents the management protocol RESTCONF";
        reference
          "RFC 8040: RESTCONF Protocol";
      }
    }
    description
      "The management protocol used to manage the NSF";
  }
```

```
  leaf username {
    type string;
    description
      "The user name string identifying the credentials for the
       authentication.";
  }
  leaf password {
    type ianach:crypt-hash;
    description
      "The password for the username for the authentication.
       Any plain-text password must be converted to a hashed value
       as soon as possible";
  }
}

augment "/i2nsfcap:nsf" {
  description
    "Augmented information of an NSF's capability that DMS
     registers with Security Controller";
  reference
    "draft-ietf-i2nsf-capability-data-model-32: I2NSF Capability
     YANG Data Model";
  container nsf-specification {
    description
      "The specification of an NSF";
    uses nsf-specification;
  }
  container nsf-access-info {
    description
      "Network access information of this NSF";
    uses nsf-access-info;
  }
}

rpc nsf-capability-query {
  description
    "Description of the capabilities that the
     Security Controller requests to the DMS";
  input {
    container query-nsf-capability {
      description
        "Description of the capabilities to request";
      uses i2nsfcap:nsf-capabilities;
      reference "RFC YYYY: I2NSF Capability YANG Data Model";
    //RFC Ed.: replace YYYY with actual RFC number of
    //draft-ietf-i2nsf-capability-data-model and remove this note.
    }
  }
  output {
```

```
      container nsf-access-info {
        description
          "Network access information of an NSF
           with the requested capabilities";
        leaf nsf-name {
          type string;
          description
          "The name of this registered NSF. The NSF name MUST be
           unique to identify the NSF with the capability. The name
           can be an arbitrary string including Fully Qualified
           Domain Name (FQDN).";
        }
        uses nsf-access-info;
      }
    }
  }
}

<CODE ENDS>
```

Figure 7: Registration Interface YANG Data Model

6. IANA Considerations

This document requests IANA to register the following URI in the "IETF XML Registry" [RFC3688]:

```
URI: urn:ietf:params:xml:ns:yang:ietf-i2nsf-registration-interface
Registrant Contact: The IESG.
XML: N/A; the requested URI is an XML namespace.
```

This document requests IANA to register the following YANG module in the "YANG Module Names" registry [RFC7950][RFC8525]:

```
Name: ietf-i2nsf-registration-interface
Namespace: urn:ietf:params:xml:ns:yang:ietf-i2nsf-registration-interface
Prefix: i2nsfri
Reference: RFC XXXX

// RFC Ed.: replace XXXX with actual RFC number and remove
// this note
```

7. Security Considerations

The YANG module specified in this document defines a data schema designed to be accessed through network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the required secure transport is

Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the required secure transport is TLS [RFC8446].

The NETCONF access control model [RFC8341] provides a means of restricting access to specific NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

The architecture of I2NSF Framework presents a risk to the implementation of security detection and mitigation activities. It is important to have an authentication and authorization method between the communication of the Security Controller and the DMS. The following are threats that need to be considered and mitigated:

Compromised DMS with valid credentials: It can send falsified information to the Security Controller to mislead existing detection or mitigation devices. Currently, there is no in-framework mechanism to mitigate this, and it is an issue for such infrastructures. It is important to keep confidential information from unauthorized persons to mitigate the possibility of compromising the DMS with this information.

Impersonating DMS: This involves a system trying to send false information while imitating as a DMS; client authentication would help the Security Controller to identify this invalid DMS.

The YANG module defined in this document extends the YANG module described in [I-D.ietf-i2nsf-capability-data-model]. Hence, this document shares all the security issues that are specified in Section 9 of [I-D.ietf-i2nsf-capability-data-model].

There are a number of extended data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes MAY be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

*nsf-specification: The attacker may provide incorrect information of the specification of any target NSF by illegally modifying this.

*nsf-access-info: The attacker may provide incorrect network access information of any target NSF by illegally modifying this.

Some of the readable extended data nodes in this YANG module MAY be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config,

or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

*nsf-specification: The attacker may gather the specification information of any target NSF and misuse the information for subsequent attacks.

*nsf-access-info: The attacker may gather the network access information of any target NSF and misuse the information for subsequent attacks.

The RPC operation in this YANG module MAY be considered sensitive or vulnerable in some network environments. It is thus important to control access to this operation. The following is the operation and its sensitivity/vulnerability:

*nsf-capability-query: The attacker may exploit this RPC operation to deteriorate the availability of the DMS and/or gather the information of some interested NSFs from the DMS. Some of the product capabilities provided by a vendor may be publicly known, the DMS should provide an authentication and authorization method to make sure this node cannot be used for exploitation.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol

(NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7317] Bierman, A. and M. Bjorklund, "A YANG Data Model for System Management", RFC 7317, DOI 10.17487/RFC7317, August 2014, <https://www.rfc-editor.org/info/rfc7317>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8329] Lopez, D., Lopez, E., Dunbar, L., Strassner, J., and R. Kumar, "Framework for Interface to Network Security Functions", RFC 8329, DOI 10.17487/RFC8329, February 2018, <https://www.rfc-editor.org/info/rfc8329>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

[RFC8407] Bierman, A., "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", BCP 216, RFC 8407, DOI 10.17487/RFC8407, October 2018, <https://www.rfc-editor.org/info/rfc8407>.

[RFC8431] Wang, L., Chen, M., Dass, A., Ananthakrishnan, H., Kini, S., and N. Bahadur, "A YANG Data Model for the Routing Information Base (RIB)", RFC 8431, DOI 10.17487/RFC8431, September 2018, <https://www.rfc-editor.org/info/rfc8431>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8525] Bierman, A., Bjorklund, M., Schoenwaelder, J., Watsen, K., and R. Wilton, "YANG Library", RFC 8525, DOI 10.17487/RFC8525, March 2019, <https://www.rfc-editor.org/info/rfc8525>.

[I-D.ietf-i2nsf-capability-data-model] Hares, S., Jeong, J. P., Kim, J. T., Moskowitz, R., and Q. Lin, "I2NSF Capability YANG Data Model", Work in Progress, Internet-Draft, draft-ietf-i2nsf-capability-data-model-32, 23 May 2022, <https://www.ietf.org/archive/id/draft-ietf-i2nsf-capability-data-model-32.txt>.

8.2. Informative References

[RFC3444] Pras, A. and J. Schoenwaelder, "On the Difference between Information Models and Data Models", RFC 3444, DOI 10.17487/RFC3444, January 2003, <https://www.rfc-editor.org/info/rfc3444>.

[RFC3849] Huston, G., Lord, A., and P. Smith, "IPv6 Address Prefix Reserved for Documentation", RFC 3849, DOI 10.17487/RFC3849, July 2004, <https://www.rfc-editor.org/info/rfc3849>.

[RFC5737] Arkko, J., Cotton, M., and L. Vegoda, "IPv4 Address Blocks Reserved for Documentation", RFC 5737, DOI 10.17487/RFC5737, January 2010, <https://www.rfc-editor.org/info/rfc5737>.

[RFC7348] Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3

Networks", RFC 7348, DOI 10.17487/RFC7348, August 2014, <https://www.rfc-editor.org/info/rfc7348>.

[I-D.ietf-i2nsf-nsf-monitoring-data-model]
Jeong, J. P., Lingga, P., Hares, S., Xia, L. F., and H. Birkholz, "I2NSF NSF Monitoring Interface YANG Data Model", Work in Progress, Internet-Draft, draft-ietf-i2nsf-nsf-monitoring-data-model-20, 1 June 2022, <https://www.ietf.org/archive/id/draft-ietf-i2nsf-nsf-monitoring-data-model-20.txt>.

[I-D.ietf-nvo3-vxlan-gpe] (Editor), F. M., (editor), L. K., and U. E. (editor), "Generic Protocol Extension for VXLAN (VXLAN-GPE)", Work in Progress, Internet-Draft, draft-ietf-nvo3-vxlan-gpe-12, 22 September 2021, <https://www.ietf.org/archive/id/draft-ietf-nvo3-vxlan-gpe-12.txt>.

[nfv-framework] "Network Functions Virtualisation (NFV); Architectureal Framework", ETSI GS NFV 002 ETSI GS NFV 002 V1.1.1, October 2013.

Appendix A. XML Examples of an NSF Registration with I2NSF Registration Interface Data Model

This section shows XML examples of the I2NSF Registration Interface data model for registering the capabilities in either IPv4 networks [RFC5737] or IPv6 networks [RFC3849] with Security Controller.

```
<nsf
  xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability"
  xmlns:i2nsfri="urn:ietf:params:xml:ns:yang
                 :ietf-i2nsf-registration-interface">
  <nsf-name>ipv4_general_firewall</nsf-name>
  <condition-capabilities>
    <generic-nsf-capabilities>
      <ipv4-capability>next-header</ipv4-capability>
      <ipv4-capability>source-address</ipv4-capability>
      <ipv4-capability>destination-address</ipv4-capability>
      <tcp-capability>source-port-number</tcp-capability>
      <tcp-capability>destination-port-number</tcp-capability>
    </generic-nsf-capabilities>
  </condition-capabilities>
  <action-capabilities>
    <ingress-action-capability>pass</ingress-action-capability>
    <ingress-action-capability>drop</ingress-action-capability>
    <ingress-action-capability>mirror</ingress-action-capability>
    <egress-action-capability>pass</egress-action-capability>
    <egress-action-capability>drop</egress-action-capability>
    <egress-action-capability>mirror</egress-action-capability>
  </action-capabilities>
  <i2nsfri:nsf-specification>
    <i2nsfri:cpu>
      <i2nsfri:model>
        Intel(R) Core(TM) i7-10510U
      </i2nsfri:model>
      <i2nsfri:clock-speed>1800</i2nsfri:clock-speed>
      <i2nsfri:cores>4</i2nsfri:cores>
      <i2nsfri:threads>8</i2nsfri:threads>
    </i2nsfri:cpu>
    <i2nsfri:memory>
      <i2nsfri:capacity>8192</i2nsfri:capacity>
      <i2nsfri:speed>2667</i2nsfri:speed>
    </i2nsfri:memory>
    <i2nsfri:disk>
      <i2nsfri:capacity>239000</i2nsfri:capacity>
    </i2nsfri:disk>
    <i2nsfri:bandwidth>
      <i2nsfri:outbound>1000000000</i2nsfri:outbound>
      <i2nsfri:inbound>1000000000</i2nsfri:inbound>
    </i2nsfri:bandwidth>
  </i2nsfri:nsf-specification>
  <i2nsfri:nsf-access-info>
    <i2nsfri:ip>192.0.2.11</i2nsfri:ip>
    <i2nsfri:port>49152</i2nsfri:port>
    <i2nsfri:management-protocol>
      NETCONF
    </i2nsfri:management-protocol>
```

```
    <i2nsfri:username>alice</i2nsfri:username>
    <i2nsfri:password>$0$password123</i2nsfri:password>
  </i2nsfri:nsf-access-info>
</nsf>
```

Figure 8: Configuration XML for Registration of a General Firewall in an IPv4 Network

Figure 8 shows the configuration XML for registering a general firewall in an IPv4 network [RFC5737] and its capabilities as follows.

1. The instance name of the NSF is ipv4_general_firewall.

2. The NSF can inspect IPv4 protocol header field, source address(es), and destination address(es).

3. The NSF can inspect the port number(s) for the transport layer protocol, i.e., TCP.

4. The NSF can determine whether the packets are allowed to pass, drop, or mirror.

5. The NSF's CPU model is Intel(R) Core(TM) i7-10510U. The clock speed is 1800 MHz, with 4 cores and 8 total threads.

6. The NSF's memory capacity is 8192 MB with the speed 2667 MHz.

7. The NSF's storage can hold maximum 239000 MB.

8. The network bandwidth available on the NSF is 1 GBps for both the outbound traffic and inbound traffic.

9. The IPv4 address of the NSF is 192.0.2.11.

10. The port of the NSF is 49152 using the NETCONF protocol. The credentials to access the NETCONF are "alice" as the username and "password123" for the password.

```
<nsf
  xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability"
  xmlns:i2nsfri="urn:ietf:params:xml:ns:yang
                 :ietf-i2nsf-registration-interface">
  <nsf-name>ipv6_general_firewall</nsf-name>
  <condition-capabilities>
    <generic-nsf-capabilities>
      <ipv6-capability>next-header</ipv6-capability>
      <ipv6-capability>source-address</ipv6-capability>
      <ipv6-capability>destination-address</ipv6-capability>
      <tcp-capability>source-port-number</tcp-capability>
      <tcp-capability>destination-port-number</tcp-capability>
    </generic-nsf-capabilities>
  </condition-capabilities>
  <action-capabilities>
    <ingress-action-capability>pass</ingress-action-capability>
    <ingress-action-capability>drop</ingress-action-capability>
    <ingress-action-capability>mirror</ingress-action-capability>
    <egress-action-capability>pass</egress-action-capability>
    <egress-action-capability>drop</egress-action-capability>
    <egress-action-capability>mirror</egress-action-capability>
  </action-capabilities>
  <i2nsfri:nsf-specification>
    <i2nsfri:cpu>
      <i2nsfri:model>
        Intel(R) Core(TM) i7-10510U
      </i2nsfri:model>
      <i2nsfri:clock-speed>1800</i2nsfri:clock-speed>
      <i2nsfri:cores>4</i2nsfri:cores>
      <i2nsfri:threads>8</i2nsfri:threads>
    </i2nsfri:cpu>
    <i2nsfri:memory>
      <i2nsfri:capacity>8192</i2nsfri:capacity>
      <i2nsfri:speed>2667</i2nsfri:speed>
    </i2nsfri:memory>
    <i2nsfri:disk>
      <i2nsfri:capacity>239000</i2nsfri:capacity>
    </i2nsfri:disk>
    <i2nsfri:bandwidth>
      <i2nsfri:outbound>1000000000</i2nsfri:outbound>
      <i2nsfri:inbound>1000000000</i2nsfri:inbound>
    </i2nsfri:bandwidth>
  </i2nsfri:nsf-specification>
  <i2nsfri:nsf-access-info>
    <i2nsfri:ip>2001:db8:0:1::11</i2nsfri:ip>
    <i2nsfri:port>49152</i2nsfri:port>
    <i2nsfri:management-protocol>
      NETCONF
    </i2nsfri:management-protocol>
```

```
    <i2nsfri:username>alice</i2nsfri:username>
    <i2nsfri:password>$0$password123</i2nsfri:password>
  </i2nsfri:nsf-access-info>
</nsf>
```

Figure 9: Configuration XML for Registration of a General Firewall in an IPv6 Network

In addition, Figure 9 shows the configuration XML for registering a general firewall in an IPv6 network [RFC3849] and its capabilities as follows.

1. The instance name of the NSF is ipv6_general_firewall.

2. The NSF can inspect IPv6 next header, flow direction, source address(es), and destination address(es)

3. The NSF can inspect the port number(s) and flow direction for the transport layer protocol, i.e., TCP and UDP.

4. The NSF can determine whether the packets are allowed to pass, drop, or mirror.

5. The NSF's CPU model is Intel(R) Core(TM) i7-10510U. The clock speed is 1800 MHz, with 4 cores and 8 total threads.

6. The NSF's memory capacity is 8192 MB with the speed 2667 MHz.

7. The NSF's storage can hold maximum 239 GB.

8. The network bandwidth available on the NSF is 1 GBps for both the outbound and inbound traffics.

9. The IPv6 address of the NSF is 2001:db8:0:1::11.

10. The port of the NSF is 49152 using the NETCONF protocol. The credentials to access the NETCONF are "alice" as the username and "password123" for the password.

Appendix B. XML Examples of an NSF Query with I2NSF Registration Interface Data Model

This section shows an XML example of the Security Controller requesting an additional NSF with a certain capability. In this example, an I2NSF User requests a security service that is able to block the specified websites. When the Security Controller checks that no registered NSF can provide such a service, it makes a query to the DMS with the following XML:

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <nsf-capability-query
  xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-registration-interface">
    <query-nsf-capability>
      <condition-capabilities>
        <advanced-nsf-capabilities>
          <url-filtering-capability>
            user-defined
          </url-filtering-capability>
        </advanced-nsf-capabilities>
      </condition-capabilities>
      <action-capabilities>
        <ingress-action-capability>drop</ingress-action-capability>
        <egress-action-capability>drop</egress-action-capability>
      </action-capabilities>
    </query-nsf-capability>
  </nsf-capability-query>
</rpc>
```

Figure 10: XML for NSF Query of a Web Filter

Figure 10 shows the XML for requesting an unregistered web filter with its capabilities as follows.

1. The NSF can inspect a URL matched from a user-defined URL where the user can specify their own URL.

2. The NSF can control the network by dropping the packets that match the condition. It can drop packets that are entering or leaving the target network.

After receiving a query given in Figure 10, the DMS can reply with following XML:

```
<rpc-reply message-id="101"
          xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <nsf-access-info
  xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-registration-interface">
    <nsf-name>web_filter</nsf-name>
    <ip>192.0.2.13</ip>
    <port>49152</port>
    <management-protocol>NETCONF</management-protocol>
    <username>alice</username>
    <password>$0$password123</password>
  </nsf-access-info>
</rpc-reply>
```

Figure 11: XML for the Reply of NSF Query of a Web Filter

In the reply shown in Figure 11, the additional NSF called web_filter can be used by accessing the NSF with the IPv4 address of 192.0.2.13 and the port number of 49152, using the NETCONF management protocol. The credentials to access the NETCONF are "alice" as the username and "password123" for the password.

Appendix C. NSF Lifecycle Management in NFV Environments

Network Functions Virtualization (NFV) can be used to implement I2NSF framework. In NFV environments, NSFs are deployed as virtual network functions (VNFs). Security Controller can be implemented as an Element Management (EM) of the NFV architecture, and is connected with the VNF Manager (VNFM) via the Ve-Vnfm interface [nfv-framework]. Security Controller can use this interface for the purpose of the lifecycle management of NSFs. If some NSFs need to be instantiated to enforce security policies in the I2NSF framework, Security Controller could request the VNFM to instantiate them through the Ve-Vnfm interface. Or if an NSF, running as a VNF, is not used by any traffic flows for a time period, Security Controller MAY request deinstantiating it through the interface for efficient resource utilization.

Appendix D. Acknowledgments

This document is a product by the I2NSF Working Group (WG) including WG Chairs (i.e., Linda Dunbar and Yoav Nir) and Diego Lopez. This document took advantage of the review and comments from the following people: Roman Danyliw, Reshad Rahman (YANG doctor), and Tom Petch. We authors sincerely appreciate their sincere efforts and kind help.

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea MSIT (Ministry of Science and ICT) (No. 2016-0-00078, Cloud Based Security Intelligence Technology Development for the Customized Security Service Provisioning). This work was supported in part by the IITP (2020-0-00395-003, Standard Development of Blockchain based Network Management Automation Technology).

Appendix E. Contributors

The following are co-authors of this document:

Patrick Lingga - Department of Electrical and Computer Engineering, Sungkyunkwan University, 2066 Seo-ro Jangan-gu, Suwon, Gyeonggi-do 16419, Republic of Korea. EMail: patricklink@skku.edu

Jinyong (Tim) Kim - Department of Electronic, Electrical and Computer Engineering, Sungkyunkwan University, 2066 Seo-ro Jangan-

gu, Suwon, Gyeonggi-do 16419, Republic of Korea. EMail: timkim@skku.edu

Chaehong Chung - Department of Electronic, Electrical and Computer Engineering, Sungkyunkwan University, 2066 Seo-ro Jangan-gu, Suwon, Gyeonggi-do 16419, Republic of Korea. EMail: darkhong@skku.edu

Susan Hares - Huawei, 7453 Hickory Hill, Saline, MI 48176, USA. EMail: shares@ndzh.com

Diego R. Lopez - Telefonica I+D, Jose Manuel Lara, 9, Seville, 41013, Spain. EMail: diego.r.lopez@telefonica.com

Appendix F. Changes from draft-ietf-i2nsf-registration-interface-dm-21

The following changes are made from draft-ietf-i2nsf-registration-interface-dm-21:

*The updates are made with the comments from AD Roman Danyliw.

Authors' Addresses

Sangwon Hyun (editor)
Department of Computer Engineering
Myongji University
116 Myongji-ro, Cheoin-gu
Yongin
Gyeonggi-do
17058
Republic of Korea

Email: shyun@mju.ac.kr

Jaehoon Paul Jeong (editor)
Department of Computer Science and Engineering
Sungkyunkwan University
2066 Seobu-Ro, Jangan-Gu
Suwon
Gyeonggi-Do
16419
Republic of Korea

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

Taekyun Roh
Department of Electronic, Electrical and Computer Engineering
Sungkyunkwan University
2066 Seobu-Ro, Jangan-Gu

Suwon
Gyeonggi-Do
16419
Republic of Korea

Phone: +82 31 290 7222
Email: tkroh0198@skku.edu

Sarang Wi
Department of Electronic, Electrical and Computer Engineering
Sungkyunkwan University
2066 Seobu-Ro, Jangan-Gu
Suwon
Gyeonggi-Do
16419
Republic of Korea

Phone: +82 31 290 7222
Email: dnl9795@skku.edu

Jung-Soo Park
Electronics and Telecommunications Research Institute
218 Gajeong-Ro, Yuseong-Gu
Daejeon
305-700
Republic of Korea

Phone: +82 42 860 6514
Email: pjs@etri.re.kr