

I2NSF Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 8, 2019

S. Hyun
Chosun University
J. Jeong
T. Roh
S. Wi
Sungkyunkwan University
J. Park
ETRI
November 4, 2018

I2NSF Registration Interface YANG Data Model
draft-ietf-i2nsf-registration-interface-dm-01

Abstract

This document defines an information model and a YANG data model for Interface to Network Security Functions (I2NSF) Registration Interface between Security Controller and Developer's Management System (DMS). The objective of these information and data models is to support NSF search, instantiation and registration according to required security capabilities 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 May 8, 2019.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

- [1. Introduction](#) [2](#)
- [2. Requirements Language](#) [3](#)
- [3. Terminology](#) [3](#)
- [4. Objectives](#) [4](#)
- [5. Information Model](#) [5](#)
 - [5.1. NSF Registration Mechanism](#) [5](#)
 - [5.2. NSF Access Information](#) [6](#)
 - [5.3. NSF Capability Information \(Capabilities of an NSF Instance\)](#) [6](#)
 - [5.3.1. Performance Capabilities](#) [7](#)
 - [5.4. Role-based Access Control List](#) [8](#)
- [6. Data Model](#) [9](#)
 - [6.1. High-Level YANG](#) [9](#)
 - [6.1.1. Definition of Symbols in Tree Diagrams](#) [9](#)
 - [6.1.2. Registration Interface](#) [10](#)
 - [6.1.3. Registration Request](#) [10](#)
 - [6.1.4. Instance Management Request](#) [10](#)
 - [6.1.5. NSF Capability Information](#) [11](#)
 - [6.1.6. NSF Access Information](#) [12](#)
 - [6.1.7. NSF Performance Capability](#) [12](#)
 - [6.1.8. Role-Based ACL\(Access Control List\)](#) [12](#)
 - [6.2. YANG Modules](#) [13](#)
 - [6.2.1. XML Example of Registration Interface Data Model](#) [17](#)
- [7. Security Considerations](#) [19](#)
- [8. References](#) [19](#)
 - [8.1. Normative References](#) [19](#)
 - [8.2. Informative References](#) [19](#)
- [Appendix A. NSF Lifecycle Managemet in NFV Environments](#) [21](#)
- [Appendix B. Changes from \[draft-ietf-i2nsf-registration-interface-dm-00\]\(#\)](#) [21](#)
- [Appendix C. Acknowledgments](#) [21](#)
- [Appendix D. Contributors](#) [21](#)
- [Authors' Addresses](#) [22](#)

1. Introduction

A number of virtual network security function instances typically exist in Interface to Network Security Functions (I2NSF) framework [[RFC8329](#)]. Since these NSF instances may have different security

capabilities, it is important to register the security capabilities of each NSF instance into the security controller after they have been created. In addition, it is required to search or instantiate NSFs of some required security capabilities on demand. As an example, if additional security capabilities are required to meet the new security requirements that an I2NSF user requests, the security controller should be able to request the DMS for NSFs that have the required security capabilities.

This document describes an information model (see [Section 5](#)) and a YANG [[RFC6020](#)] data model (see [Section 6](#)) for the I2NSF Registration Interface [[RFC8329](#)] between the security controller and the developer's management system (DMS) to support NSF search, instantiation and registration according to required security capabilities. It also describes the procedure which should be performed by the security controller and the DMS via the Registration Interface using the defined model.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [[RFC2119](#)].

3. Terminology

This document uses the following terms defined in [[i2nsf-terminology](#)], [[capability-im](#)], [[RFC8329](#)], [[nsf-triggered-steering](#)], [[supa-policy-data-model](#)], and [[supa-policy-info-model](#)]

- o Network Security Function (NSF): A function that is responsible for 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. [[nsf-triggered-steering](#)]
- o Advanced Inspection/Action: As like the I2NSF information model for NSF facing interface [[capability-im](#)], Advanced Inspection/Action means that a security function calls another security function for further inspection based on its own inspection result. [[nsf-triggered-steering](#)]

- o NSF Profile (NSF Capability Information): NSF Capability Information specifies the inspection capabilities of the associated NSF instance. Each NSF instance has its own NSF Capability Information to specify the type of security service it provides and its resource capacity etc. [[nsf-triggered-steering](#)]
- o Data Model: A data model is a representation of concepts of interest to an environment in a form that is dependent on data repository, data definition language, query language, implementation language, and protocol. [[supa-policy-info-model](#)]
- o Information Model: An information model is a representation of concepts of interest to an environment in a form that is independent of data repository, data definition language, query language, implementation language, and protocol. [[supa-policy-info-model](#)]

4. Objectives

- o Registering NSFs to I2NSF framework: Developer's Management System (DMS) in I2NSF framework is typically run by an NSF vendor, and uses Registration Interface to provide NSFs developed by the NSF vendor to Security Controller. DMS registers NSFs and their capabilities to I2NSF framework through Registration Interface, so that Security Controller can use those capabilities by instantiating the NSFs once they are required. Once NSFs are registered to I2NSF framework, a catalog of the NSFs and their capabilities is created and provided to Security Controller. When we consider the implementation of I2NSF framework based on NFV technology, the catalog of the NSFs may be prepared and managed by NFV MANO.
- o Updating the capabilities of registered NSFs: After an NSF is registered into I2NSF framework, some modifications on the capability of the NSF may be required later. In this case, DMS uses Registration Interface to update the capability of the NSF, and this update should be reflected on the catalog of NSFs.
- o Retrieving the catalog of NSFs: Security Controller uses Registration Interface to retrieve the catalog of available NSFs and their capabilities. Enforcing security policy requires a set of security capabilities that is provided by a set of NSFs. Once receiving a request of security policy from an I2NSF user, Security Controller figures out what capabilities are required to enforce the security policy. Security Controller then searches the catalog of NSFs for the required capabilities, and finally determines a set of NSFs that is necessary to enforce the requested policy.

- o Requesting NSF instantiation: If some NSFs need to be instantiated to enforce requested security policy, Security Controller makes a request to instantiate them through Registration Interface. Or if an NSF, running as a virtual NSF in the NFV environment, is not used by any traffic flows for a time period, Security Controller may request deinstantiating it through Registration Interface for the purpose of efficient resource utilization.

5. Information Model

The I2NSF registration interface is used by Security Controller and Developer's Management System (DMS) in I2NSF framework. The following summarizes the process typically done through the registration interface:

- 1) DMS registers NSFs to I2NSF framework through the registration interface. DMS also uses the registration interface to update the capabilities of the NSFs registered in the framework.
- 2) Once NSFs are registered to I2NSF framework, a catalog of the NSFs and their capabilities is created and provided to Security Controller via the registration interface.
- 3) Security Controller searches the catalog of NSFs for the capabilities required to enforce security policies requested by I2NSF users, and selects some of the NSFs that can provide the required capabilities.
- 4) Security Controller requests the instantiation of the selected NSFs via the registration interface.

This section clarifies the information model that is required to support the process described above.

5.1. NSF Registration Mechanism

In order to register a new NSF, DMS should generate a registration message to Security Controller. A registration message consists of an NSF capability information and an NSF Access Information. The former describes the security capability that the new NSF can provide and the latter is for enabling network access to the NSF from other components. After this registration process, as explained in [[capability-im](#)], the I2NSF capability interface can conduct controlling and monitoring the new registered NSF.

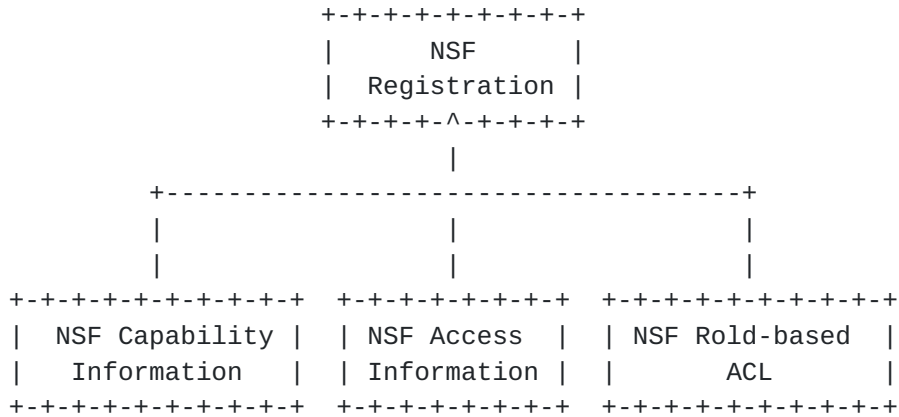


Figure 1: Registration Mechanism Sub-Model Overview

5.2. NSF Access Information

NSF Access Information contains the followings that are required to communicate with an NSF: IPv4 address, IPv6 address, port number, and supported transport protocol(s) (e.g., Virtual Extensible LAN (VXLAN) [[RFC 7348](#)], Generic Protocol Extension for VXLAN (VXLAN-GPE) [[draft-ietf-nvo3-vxlan-gpe](#)], Generic Route Encapsulation (GRE), Ethernet etc.). In this document, NSF Access Information is used to identify a specific NSF instance (i.e. NSF Access Information is the signature(unique identifier) of an NSF instance in the overall system).

5.3. NSF Capability Information (Capabilities of an NSF Instance)

NSF Profile basically describes the inspection capabilities of an NSF instance. In Figure 2, we show capability objects of an NSF instance. Following the information model of NSF capabilities defined in [[capability-im](#)], we share the same security capabilities: Network-Security Capabilities, Content-Security Capabilities, and Attack Mitigation Capabilities. Also, NSF Profile additionally contains the performance capabilities and role-Based access control list (ACL) as shown in Figure 2.

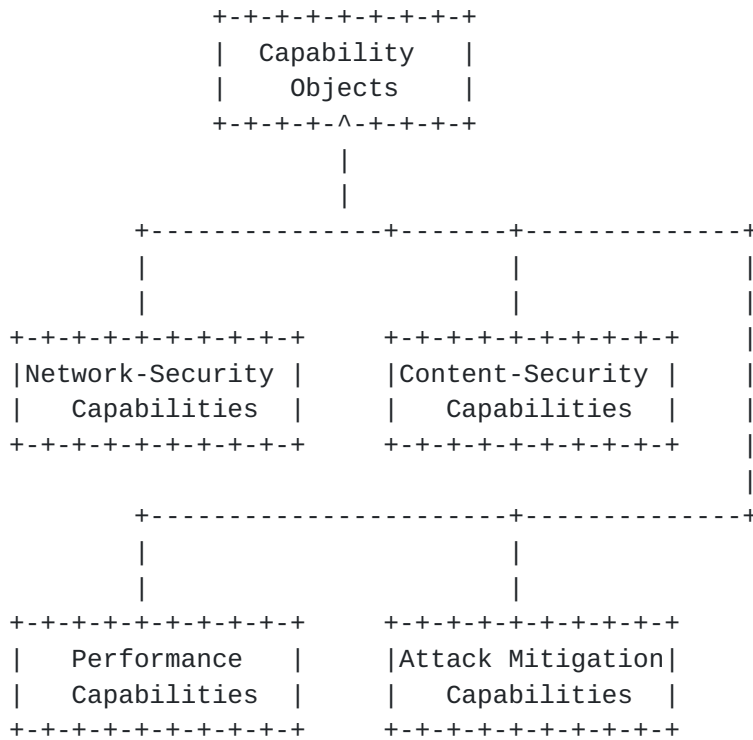


Figure 2: NSF Profile Overview

5.3.1. Performance Capabilities

This information represents the processing capability of an NSF. This information can be used to determine whether the NSF is in congestion by comparing this with the workload that the NSF currently undergoes. Moreover, this information can specify an available amount of each type of resources such as processing power which are available on the NSF. (The registration interface can control the usages and limitations of the created instance and make the appropriate request according to the status.) As illustrated in Figure 3, this information consists of two items: Processing and Bandwidth. Processing information describes the NSF's available processing power. Bandwidth describes the information about available network amount in two cases, outbound, inbound. This two information can be used for the NSF's instance request.

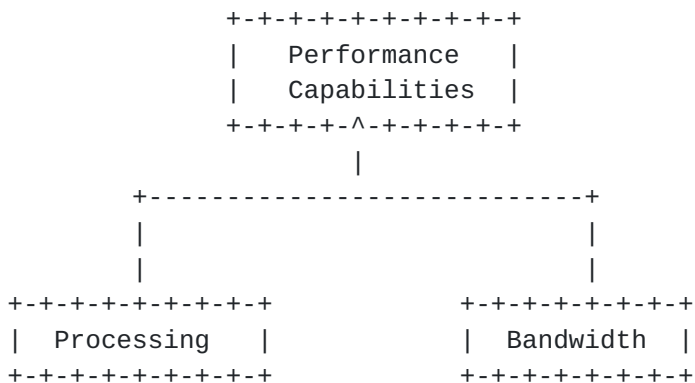


Figure 3: Performance Capability Overview

5.4. Role-based Access Control List

This information specifies access policies of an NSF to determine whether to permit or deny the access of an entity to the NSF based on the role given to the entity. Each NSF is associated with a role-based access control list (ACL) so that it can determine whether to permit or deny the access request from an entity. Figure 4 and Figure 5 show the structure of the role-based ACL, which is composed of role-id, access-type, and permit/deny. The role-id identifies roles of entities (e.g., administrator, developer etc.). The access-type identifies the specific type of access requests such as NSF rule configuration/update and NSF monitoring. Consequently, the role-based ACL in Figure 4 and Figure 5 specifies a set of access-types to be permitted and to be denied by each role-id.

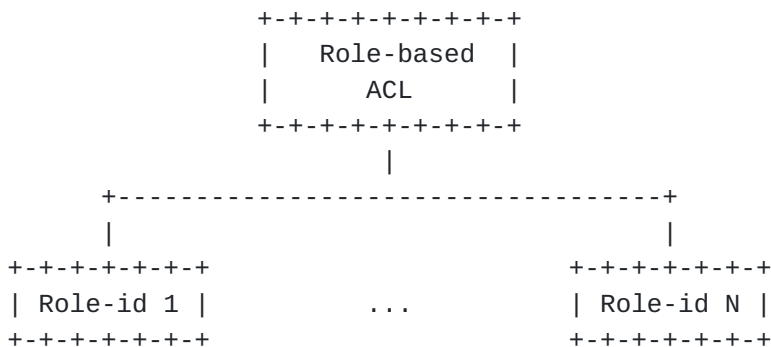


Figure 4: Role-based Access Control List

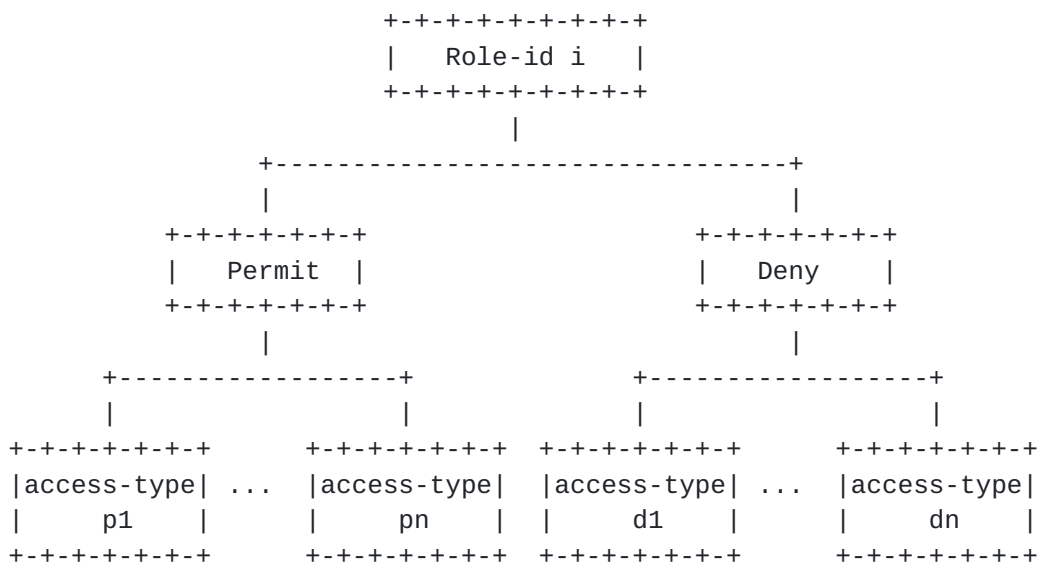


Figure 5: Role-id Subtree

6. Data Model

6.1. High-Level YANG

This section provides an overview of the high level YANG.

6.1.1. Definition 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 [[i2rs-rib-data-model](#)] 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.

6.1.2. Registration Interface

```
module : ietf-i2nsf-regs-interface-model
  +-rw regs-req
  |   uses i2nsf-regs-req
  +-rw instance-mgmt-req
  |   uses i2nsf-instance-mgmt-req
```

Figure 6: High-Level YANG of I2NSF Registration Interface

Each of these sections mirror sections of [Section 5](#).

6.1.3. Registration Request

This section expands the i2nsf-regs-req in Figure 6.

```
Registration Request
  +-rw i2nsf-regs-req
  +-rw nsf-capability-information
  |   uses i2nsf-nsf-capability-information
  +-rw nsf-access-info
  |   uses i2nsf-nsf-access-info
```

Figure 7: High-Level YANG of I2NSF Registration Request

Registration Request contains the capability information of newly created NSF to notify its capability to Security Controller. The request also contains Network Access Information so that the Security Controller can access the NSF.

6.1.4. Instance Management Request

This section expands the i2nsf-instance-mgmt-req in Figure 6.


```

Instance Management Request
+--rw i2nsf-instance-mgmt-req
  +--rw req-level uint16
  +--rw req-id uint64
  +--rw (req-type)?
    +--rw (instanciation-request)
      +--rw in-nsf-capability-information
        | uses i2nsf-nsf-capability-information
    +--rw (deinstanciation-request)
      +--rw de-nsf-access-info
        | uses i2nsf-nsf-access-info
    +--rw (updating-request)
      +--rw update-nsf-capability-information
        | uses i2nsf-nsf-capability-information

```

Figure 8: High-Level YANG of I2NSF Instance Mgmt Request

Instance management request consists of two types: `instanciation-request`, `deinstanciation-request`, and `updating-request`. The `instanciation-request` is used to request generation of a new NSF instance with NSF Capability Information which specifies required NSF capability information. The `deinstanciation-request` is used to remove an existing NSF with NSF Access Information. The `updating nsf` request is used to updating a existing NSF information with NSF capabilities.

6.1.5. NSF Capability Information

This section expands the `i2nsf-nsf-capability-information` in Figure 7 and Figure 8.

```

NSF Capability Information
+--rw i2nsf-nsf-capability-information
  +--rw i2nsf-capability
    | uses ietf-i2nsf-capability
  +--rw performance-capability
    | uses i2nsf-nsf-performance-caps

```

Figure 9: High-Level YANG of I2NSF NSF Capability Information

In Figure 9, `ietf-i2nsf-capability` refers module `ietf-i2nsf-capability` in [[i2nsf-capability-dm](#)]. We add the `performance-capability` because it is absent in [[i2nsf-capability-dm](#)] and [[netmod-acl-model](#)]

6.1.6. NSF Access Information

This section expands the `i2nsf-nsf-access-info` in Figure 7 and Figure 8.

```
NSF Access Information
  +--rw i2nsf-nsf-access-info
    +--rw nsf-address    inet:ipv4-address
    +--rw nsf-port-address inet:port-number
```

Figure 10: High-Level YANG of I2NSF NSF Access Information

This information is used by other components to access an NSF.

6.1.7. NSF Performance Capability

This section expands the `i2nsf-nsf-performance-caps` in Figure 9.

```
NSF Performance Capability
  +--rw i2nsf-nsf-performance-caps
    +--rw processing
      |   +--rw processing-average uint16
      |   +--rw processing-peak    uint16
    +--rw bandwidth
      |   +--rw outbound
      |   |   +--rw outbound-average uint16
      |   |   +--rw outbound-peak    uint16
      |   +--rw inbound
      |   |   +--rw inbound-average  uint16
      |   |   +--rw inbound-peak     uint16
```

Figure 11: High-Level YANG of I2NSF NSF Performance Capability

When the Security Controller requests the Developer Management System to create a new NSF instance, the performance capability is used to specify the performance requirements of the new instance.

6.1.8. Role-Based ACL(Access Control List)

This section expands the `ietf-netmod-acl-model` in [[netmod-acl-model](#)].

```
Role-Based ACL
  +--rw role-based-acl
    uses ietf-netmod-acl-model
```

Figure 12: Role-Based ACL

In [[netmod-acl-model](#)], ietf-netmod-acl-model refers module ietf-netmod-acl-model in [[netmod-acl-model](#)]. We add the role-based ACL because it is absent in [[i2nsf-capability-dm](#)].

6.2. YANG Modules

This section introduces a YANG module for the information model of the required data for the registration interface between Security Controller and Developer's Management System, as defined in [Section 5](#).

```
<CODE BEGINS> file "ietf-i2nsf-regs-interface@2018-11-04.yang"
  module ietf-i2nsf-regs-interface {
    namespace
      "urn:ietf:params:xml:ns:yang:ietf-i2nsf-regs-interface";
    prefix
      regs-interface;
    import ietf-inet-types{
      prefix inet;
    }

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

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

      WG Chair: Adrian Farrel
      <mailto:Adrain@olddog.co.uk>

      WG Chair: Linda Dunbar
      <mailto:Linda.dunbar@huawei.com>

      Editor: Sangwon Hyun
      <mailto:swhyun77@skku.edu>

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

      Editor: Taekyun Roh
      <mailto:tkroh0198@skku.edu>

      Editor: Sarang Wi
      <mailto:dn19795@skku.edu>

      Editor: Jung-Soo Park
```



```

    <mailto:pjs@etri.re.kr>";

description
  "It defines a YANG data module for Registration
Interface.";

revision "2018-11-04"{
  description "The second revision";
  reference
    "draft-ietf-i2nsf-capability-data-model-01";
}
list interface-container{
  key "interface-name";
  description
    "i2nsf-reg-interface-container";
  leaf interface-name{
    type string;
    description
      "interface name";
  }
  container i2nsf-regs-req {
    description
      "The capability information of newly
      created NSF to notify its
      capability to Security Controller";
    container nsf-capability-information {
      description
        "nsf-capability-information";
      uses i2nsf-nsf-capability-information;
    }
    container nsf-access-info {
      description
        "nsf-access-info";
      uses i2nsf-nsf-access-info;
    }
  }
  container ietf-netmod-acl-model{
    description
      "netmod-acl-model";
    uses ietf-netmod-acl-model;
  }
}
container i2nsf-instance-mgmt-req {
  description
    "Required information for instantiation-request,
    deinstantiation-request and updating-request";
  leaf req-level {
    type uint16;
    description
      "req-level";
  }
}
```

}

Hyun, et al.

Expires May 8, 2019

[Page 14]


```
        "processing-average";
    }
    leaf processing-peak{
        type uint16;
        description
            "processing peak";
    }
}
container bandwidth{
    description
        "bandwidth info";
    container inbound{
        description
            "inbound";
        leaf inbound-average{
            type uint16;
            description
                "inbound-average";
        }
        leaf inbound-peak{
            type uint16;
            description
                "inbound-peak";
        }
    }
    container outbound{
        description
            "outbound";
        leaf outbound-average{
            type uint16;
            description
                "outbound-average";
        }
        leaf outbound-peak{
            type uint16;
            description
                "outbound-peak";
        }
    }
}
}
grouping i2nsf-nsf-capability-information {
    description
        "Detail information of an NSF";
    container performance-capability {
        uses i2nsf-nsf-performance-caps;
        description
            "performance-capability";
    }
}
```



```

    }
    container i2nsf-capability {
      description
        "It refers draft-ietf-i2nsf-capability-data-model-01.txt
        later";
    }
  }
  grouping ietf-netmod-acl-model {
    description
      "Detail information";
    container role-based-acl {
      description
        "It refers draft-ietf-netmod-acl-model-19.txt
        later";
    }
  }
  grouping i2nsf-nsf-access-info {
    description
      "NSF access information";
    leaf nsf-address {
      type inet:ipv4-address;
      mandatory true;
      description
        "nsf-address";
    }
    leaf nsf-port-address {
      type inet:port-number;
      description
        "nsf-port-address";
    }
  }
}
<CODE ENDS>

```

Figure 13: Data Model of I2NSF Registration Interface

[6.2.1](#). XML Example of Registration Interface Data Model

Requirement: Registering the IDS NSF with VoIP/VoLTE security capability using Registration interface.

Here is the configuration xml for this Registration Interface:

```

<?xml version="1.0" encoding="UTF-8"?>
<rpc xmlns="urn:ietf:params:netconf:base:1.0" message-id="1">
  <edit-config>
    <target>

```



```
    <running/>
  </target>
<config>
  <i2nsf-regs-req>
    <i2nsf-nsf-capability-information>
      <ietf-i2nsf-capability>
        <nsf-capabilities>
          <nsf-capabilities-id>1</nsf-capabilities-id>
          <con-sec-control-capabilities>
            <content-security-control>
              <ids>
                <ids-support>true</ids-support>
                <ids-fcn nc:operation="create">
                  <ids-fcn-name>ids-service</ids-fcn-name>
                </ids-fcn>
              </ids>
            <voip-volte>
              <voip-volte-support>true</voip-volte-support>
              <voip-volte-fcn nc:operation="create">
                <voip-volte-fcn-name>
                  ips-service
                </voip-volte-fcn-name>
              </voip-volte-fcn>
            </voip-volte>
          </content-security-control>
        </con-sec-control-capabilities>
      </nsf-capabilities>
    </ietf-i2nsf-capability>
  <i2nsf-nsf-performance-caps>
    <processing>
      <processing-average>1000</processing-average>
      <processing-peak>5000</processing-peak>
    </processing>
    <bandwidth>
      <outbound>
        <outbound-average>1000</outbound-average>
        <outbound-peak>5000</outbound-peak>
      </outbound>
      <inbound>
        <inbound-average>1000</inbound-average>
        <inbound-peak>5000</inbound-peak>
      </inbound>
    </bandwidth>
  </i2nsf-nsf-performance-caps>
</i2nsf-nsf-capability-information>
<nsf-access-info>
  <nsf-address>10.0.0.1</nsf-address>
  <nsf-port-address>145</nsf-port-address>
```



```
        </nsf-access-info>
    </i2nsf-regs-req>
</config>
</edit-config>
</rpc>
```

Figure 14: Registration Interface example

7. Security Considerations

This document introduces no additional security threats and SHOULD follow the security requirements as stated in [[RFC8329](#)].

8. References

8.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [RFC 2119](#), March 1997.
- [RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", [RFC 6020](#), October 2010.

8.2. Informative References

- [capability-im] Xia, L., Strassner, J., Basile, C., and D. Lopez, "Information Model of NSFs Capabilities", [draft-i2nsf-capability-02](#) (work in progress), July 2018.
- [[draft-ietf-nvo3-vxlan-gpe](#)] Maino, Ed., F., Kreeger, Ed., L., and U. Elzur, Ed., "Generic Protocol Extension for VXLAN", [draft-ietf-nvo3-vxlan-gpe-06](#) (work in progress), April 2018.
- [i2nsf-capability-dm] Hares, S., Jeong, J., Kim, J., Moskowitz, R., and Q. Lin, "I2NSF Capability YANG Data Model", [draft-ietf-i2nsf-capability-data-model-01](#) (work in progress), July 2018.
- [i2nsf-terminology] Hares, S., Strassner, J., Lopez, D., Xia, L., and H. Birkholz, "Interface to Network Security Functions (I2NSF) Terminology", [draft-ietf-i2nsf-terminology-06](#) (work in progress), July 2018.

[i2rs-rib-data-model]

Wang, L., Chen, M., Dass, A., Ananthakrishnan, H., Kini, S., and N. Bahadur, "A YANG Data Model for Routing Information Base (RIB)", [draft-ietf-i2rs-rib-data-model-15](#) (work in progress), May 2018.

[netmod-acl-model]

Jethanandani, M., Huang, L., Agarwal, S., and D. Blair, "Network Access Control List (ACL) YANG Data Model", [draft-ietf-netmod-acl-model-19](#) (work in progress), April 2018.

[nfv-framework]

"Network Functions Virtualisation (NFV); Architectural Framework", ETSI GS NFV 002 ETSI GS NFV 002 V1.1.1, October 2013.

[nsf-triggered-steering]

Hyun, S., Jeong, J., Park, J., and S. Hares, "Service Function Chaining-Enabled I2NSF Architecture", [draft-hyun-i2nsf-nsf-triggered-steering-06](#) (work in progress), July 2018.

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

[supa-policy-data-model]

Halpern, J., Strassner, J., and S. van der Meer, "Generic Policy Data Model for Simplified Use of Policy Abstractions (SUPA)", [draft-ietf-supa-generic-policy-data-model-04](#) (work in progress), June 2017.

[supa-policy-info-model]

Strassner, J., Halpern, J., and S. van der Meer, "Generic Policy Information Model for Simplified Use of Policy Abstractions (SUPA)", [draft-ietf-supa-generic-policy-info-model-03](#) (work in progress), May 2017.

Appendix A. NSF Lifecycle Managemet 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 B. Changes from [draft-ietf-i2nsf-registration-interface-dm-00](#)

The following changes have been made from [draft-ietf-i2nsf-registration-interface-dm-00](#):

- o [Section 4](#) has been revised to clarify the major objectives of the I2NSF registration interface, considering the register-select-instantiate operation sequence that is typically performed through the registration interface in I2NSF framework based on NFV.
- o [Section 5](#) has been revised as well based on the register-select-instantiate operation sequence.
- o [Appendix A](#) has been added to clarify the lifecycle management of NSFs in I2NSF framework based on NFV.

Appendix C. 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).

Appendix D. Contributors

This document is made by the group effort of I2NSF working group. Many people actively contributed to this document. The following are considered co-authors:

- o Jinyong Tim Kim (Sungkyunkwan University)
- o Susan Hares (Huawei)

- o Diego R. Lopez (Telefonica)

Authors' Addresses

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

E-Mail: shyun@chosun.ac.kr

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

E-Mail: pauljeong@skku.edu

URI: <http://iotlab.skku.edu/people-jaehoon-jeong.php>

Taekyun Roh
Electrical Computer Engineering
Sungkyunkwan University
2066 Seobu-Ro, Jangan-Gu
Suwon, Gyeonggi-Do 16419
Republic of Korea

Phone: +82 31 290 7222

Fax: +82 31 299 6673

E-Mail: tkroh0198@skku.edu

URI: http://imtl.skku.ac.kr/xe/index.php?mid=board_YoKq57

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

Phone: +82 31 290 7222
Fax: +82 31 299 6673
EMail: dn19795@skku.edu
URI: http://imtl.skku.ac.kr/xe/index.php?mid=board_YoKq57

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

