I2NSF Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 12, 2019

S. Hyun
Chosun University
J. Jeong
T. Roh
S. Wi
Sungkyunkwan University
J. Park
ETRI

March 11, 2019

I2NSF Registration Interface YANG Data Model
draft-ietf-i2nsf-registration-interface-dm-02

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 capability registration and query via I2NSF Registration Interface.

Editorial Note (To be removed by RFC Editor)

Please update these statements within the document with the RFC number to be assigned to this document:

"This version of this YANG module is part of RFC XXXX;"

"RFC XXXX: I2NSF Registration Interface YANG Data Model"

"reference: RFC XXXX"

Please update the "revision" date of the YANG module.

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 September 12, 2019.

Table of Contents

1. Introduction

A number of network security functions may exist in Interface to Network Security Functions (I2NSF) framework [RFC8329]. Since these NSFs likely have different security capabilities, it is important to register the security capabilities of each NSF into the security controller. 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.

This document describes an information model (see Section 5) and a YANG [RFC7950] 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 capability registration and query and NSF initiation request 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.

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]

- 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]

- 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]

- 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]

- YANG: This document follows the guidelines of [RFC6087], uses the common YANG types defined in [RFC6991], and adopts the Network Management Datastore Architecture (NMDA). The meaning of the symbols in tree diagrams is defined in [RFC8340].

4. Objectives

- 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. For the registered NSFs, Security Controller maintains a catalog of the capabilities of those NSFs.

- Updating the capabilities of registered NSFs: After an NSF is registered into Security Controller, 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.

- Querying DMS about some required capabilities: Security Controller may need 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 may query DMS about NSF(s) that can provide the required capabilities via Registration Interface.

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 operations done through the registration interface:

1) DMS registers NSFs and their capabilities to Security Controller via the registration interface. DMS also uses the registration interface to update the capabilities of the NSFs registered previously.

2) In case that Security Controller fails to find any registered NSF that can provide some required capabilities, Security Controller queries DMS about NSF(s) having the required capabilities via the registration interface.

Figure 1 shows the information model of the I2NSF registration interface, which consists of three submodels: NSF capability registration, and NSF capability query. Each submodel is used for the operations listed above. The remainder of this section will provide more in-depth explanations of each submodel.

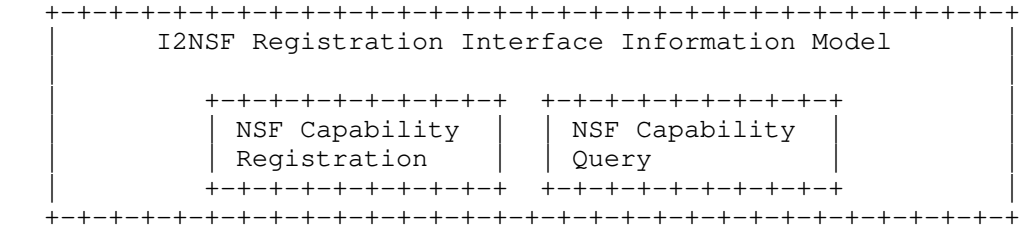

Figure 1: I2NSF Registration Interface Information Model

5.1. NSF Capability Registration

This submodel is used by DMS to register an NSF to 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. 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.

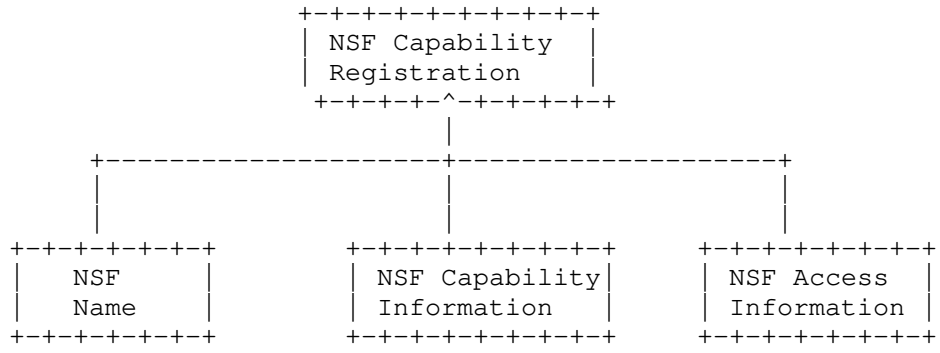

Figure 2: NSF Capability Registration Sub-Model

5.1.1. 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 defiend in [capability-im], we share the same security capabilities: Network Security Capabilities, Content Security Capabilities, and Attack Mitigation Capabilities. Also, NSF Capability Information additionally contains the performance capabilities of an NSF as shown in Figure 3.

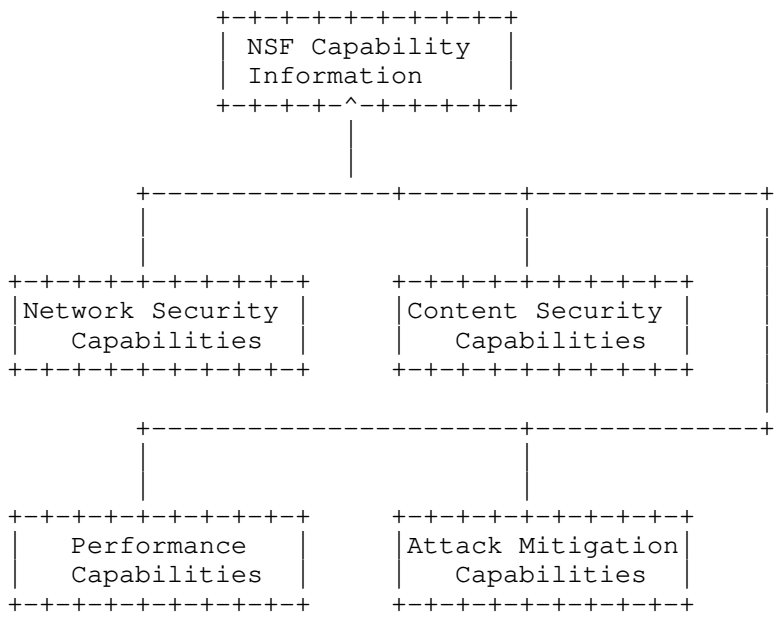

Figure 3: NSF Capability Information

5.1.1.1. 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 4, 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.

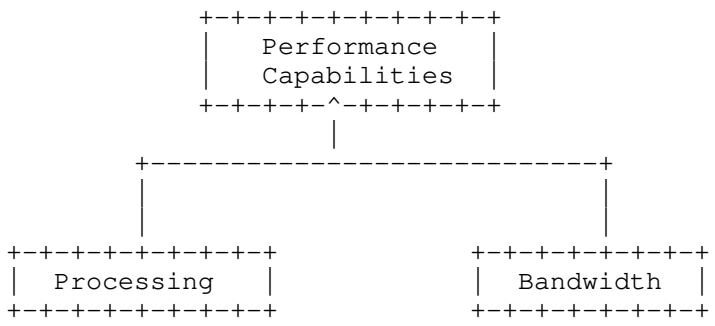

Figure 4: Performance Capability Overview

5.1.2. 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).

5.2. 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 sub-model in Section 5.1.1, and sends DMS a query about which NSF(s) can provide these capabilities.

6. Data Model

6.1. YANG Tree Diagram

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

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 [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.

6.1.2. I2NSF Registration Interface

```
module : ietf-i2nsf-reg-interface
      +--rw nsf-capability-registration
      |  uses i2nsf-nsf-registrations

rpcs :
      +---x nsf-capability-query
      |  uses i2nsf-nsf-capability-query
```

Figure 5: YANG tree of I2NSF Registration Interface

The I2NSF registration interface is used for the following purposes. Developer's Management System (DMS) registers NSFs and their capabilities into Security Controller via the registration interface. In case that 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.

6.1.2.1. NSF Capability Registration

This section expands the i2nsf-nsf-registrations in Figure 5.

```
NSF Capability Registration
 +--rw i2nsf-nsf-registrations
     +--rw i2nsf-nsf-capability-registration*  [nsf-name]
        +--rw nsf-name                              string
        +--rw nsf-capability-info
        |  uses i2nsf-nsf-capability-info
        +--rw nsf-access-info
        |  uses i2nsf-nsf-access-info
```

Figure 6: YANG tree of NSF Capability Registration

When registering an NSF to Security Controller, DMS uses this module to describe what capabilities the NSF can offer. DMS includes the network access information of the NSF which is required to make a network connection with the NSF as well as the capability description of the NSF.

6.1.2.2. NSF Capability Query

This section expands the i2nsf-nsf-capability-query in Figure 5.

```
NSF Capability Query
  +---x i2nsf-nsf-capability-query
      +---w input
      |  +---w query-i2nsf-capability-info
      |  |   uses ietf-i2nsf-capability
      +--ro output
          +--ro nsf-access-info
          |  uses i2nsf-nsf-access-info
```

Figure 7: YANG tree of NSF Capability Query

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 7, the ietf-i2nsf-capability refers to the module defined in [i2nsf-capability-dm].

6.1.3. NSF Capability Information

This section expands the i2nsf-nsf-capability-info in Figure 6 and Figure 7.

```
NSF Capability Information
  +--rw i2nsf-nsf-capability-info
    +--rw i2nsf-capability
    |  uses ietf-i2nsf-capability
    +--rw nsf-performance-capability
    |  uses i2nsf-nsf-performance-capability
```

Figure 8: YANG tree of I2NSF NSF Capability Information

In Figure 8, the ietf-i2nsf-capability refers to the module defined in [i2nsf-capability-dm]. The i2nsf-nsf-performance-capability is used to specify the performance capability of an NSF.

6.1.3.1. NSF Performance Capability

This section expands the i2nsf-nsf-performance-capability in Figure 8.

```
NSF Performance Capability
  +--rw i2nsf-nsf-performance-capability
   +--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 9: YANG tree of I2NSF NSF Performance Capability

This module is used to specify the performance capabilities of an NSF when registering or initiating the NSF.

6.1.4. NSF Access Information

This section expands the i2nsf-nsf-access-info in Figure 6.

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

Figure 10: YANG tree of I2NSF NSF Access Informantion

This module contains the network access information of an NSF that is required to enable network communications with the NSF.

6.2. YANG Data Modules

This section introduces a YANG data 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-reg-interface@2019-03-11.yang

  module ietf-i2nsf-reg-interface{
  yang-version 1.1;
  namespace
  "urn:ietf:params:xml:ns:yang:ietf-i2nsf-reg-interface";
  prefix "iiregi";

  import ietf-inet-types{
     prefix inet;
     reference "RFC 6991";
  }
  import ietf-i2nsf-capability{
    prefix capa;
    reference "draft-ietf-i2nsf-capability
            -data-model-02";
  }
  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: Linda Dunbar
    <mailto:Linda.duhbar@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:dnl9795@skku.edu>

       Editor: Jung-Soo Park
       <mailto:pjs@etri.re.kr>";

     description

      "It defines a YANG data model for Registration Interface.
      Copyright (c) 2018 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 Simplified BSD License
      set forth in Section 4.c of the IETF Trust's Legal Provisions
      Relating to IETF Documents
      (http://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 2019-03-11 {
       description "The third revision";
       reference
         "RFC XXXX: I2NSF Registration Interface YANG Data Model";
     }
     rpc i2nsf-nsf-capability-query {
        description
          "Capability information that the
           Security Controller
           sends to the DMS";
        input{
           container query-i2nsf-capability-info {
             description
             "i2nsf capability information";
             uses "capa:nsf-capabilities";
             reference
               "draft-ietf-i2nsf-capability
                     -data-model-02";
             }
```

```
        }
        output{
            container nsf-access-info {
              description
              "nsf access information";
              uses i2nsf-nsf-access-info;

             }
         }
     }
     container i2nsf-nsf-registrations{
     description
      "i2nsf-nsf-registrations";
      list i2nsf-nsf-capability-registration {
        key "nsf-name";
        description
         "Requeired information for registration";
         leaf nsf-name {
           type string;
           mandatory true;
           description
             "nsf-name";
         }
         container nsf-capability-info {
           description
             "nsf-capability-information";
             uses i2nsf-nsf-capability-info;
         }
         container nsf-access-info {
           description
             "nsf-access-info";
             uses i2nsf-nsf-access-info;
         }
       }
     }
     grouping i2nsf-nsf-performance-capability {
        description
         "NSF performance capailities";
        container processing{
          description
           "processing info";
          leaf processing-average{
           type uint16;
           description
            "processing-average";
          }
          leaf processing-peak{
           type uint16;
```

```
           description
            "processing peak";
          }
        }
        container bandwidth{
          description
            "bandwidth info";
          container outbound{
            description
               "outbound";
            leaf outbound-average{
                type uint16;
                description
                 "outbound-average";
            }
            leaf outbound-peak{
                 type uint16;
                 description
                  "outbound-peak";
             }
         }
           container inbound{
             description
              "inbound";
             leaf inbound-average{
                type uint16;
                description
                 "inbound-average";
             }
             leaf inbound-peak{
                 type uint16;
                 description
                  "inbound-peak";
             }
           }
        }
      }
        grouping i2nsf-nsf-capability-info {
          description
           "Detail information of an NSF";
          container i2nsf-capability {
            description
             "ietf i2nsf capability information";
            uses "capa:nsf-capabilities";
            reference "draft-ietf-i2nsf-capability
                 -data-model-02";
            }
            container nsf-performance-capability {
```

```
          description
           "performance capability";
          uses i2nsf-nsf-performance-capability;
        }
      }

      grouping i2nsf-nsf-access-info {
        description
         "NSF access information";
        leaf nsf-instance-name {
          type string;
          description
            "nsf-instance-name";
        }
        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 11: Registration Interface YANG Data Model

7. 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-reg-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].

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

8. Security Considerations

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

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs toIndicate Requirement Levels", RFC 2119, March 1997.

[RFC3688] Mealling, M., "The IETF XML Registry", RFC 3688, January 2004.

[RFC6087] Bierman, A., "Guidelines for Authors and Reviewers of YANG Data Model Documents", RFC 6087, January 2011.

[RFC6991] Schoenwaelder, J., "Common YANG Data Types", RFC 6991, July 2013.

[RFC7950] Bjorklund, M., "The YANG 1.1 Data Modeling Language", RFC 7950, August 2016.

[RFC8340] Bjorklund, M. and L. Berger, "YANG Tree Diagrams", RFC 8340, March 2018.

9.2. Informative References

[capability-im] Xia, L., Strassner, J., Basile, C., and D. Lopez, "Information Model of NSFs Capabilities", draft-i2nsf-capability-04 (work in progress), October 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-02 (work in progress), November 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-07 (work in progress), January 2019.

[nfv-framework]
"Network Functions Virtualisation (NFV); Architectureal 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.

[RFC8431] Wang, L., Chen, M., Dass, A., Ananthakrishnan, H., Kini, S., and N. Bahadur, "A YANG Data Model for Routing Information Base (RIB)", RFC 8431, September 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. XML Example of Registration Interface Data Model

This section describes XML examples of the I2NSF Registration Interface data model in five NSF Registration examples and one NSF Capability Query example.

A.1. Example 1: Registration for Capabilities of General Firewall

This section shows a configuration example for capabilities registration of general firewall.

```
<i2nsf-nsf-registrations
  xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-reg-interface"
  xmlns:capa="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
   <i2nsf-nsf-capability-registration>
    <nsf-name>general_firewall_capability</nsf-name>
    <nsf-capability-info>
      <i2nsf-capability>
        <condition-capabilities>
          <generic-nsf-capabilities>
           <ipv4-capa>capa:ipv4-protocol</ipv4-capa>
           <ipv4-capa>capa:exact-ipv4-address</ipv4-capa>
           <ipv4-capa>capa:range-ipv4-address</ipv4-capa>
           <tcp-capa>capa:exact-tcp-port-num</tcp-capa>
           <tcp-capa>capa:range-tcp-port-num</tcp-capa>
          </generic-nsf-capabilities>
          </condition-capabilities>
          <action-capabilities>
            <ingress-action-capa>capa:pass</ingress-action-capa>
            <ingress-action-capa>capa:drop</ingress-action-capa>
            <ingress-action-capa>capa:alert</ingress-action-capa>
            <egress-action-capa>capa:pass</egress-action-capa>
            <egress-action-capa>capa:drop</egress-action-capa>
            <egress-action-capa>capa:alert</egress-action-capa>
          </action-capabilities>
      </i2nsf-capability>
      <nsf-performance-capability>
       <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>
             </nsf-performance-capability>
            </nsf-capability-info>
            <nsf-access-info>
             <nsf-instance-name>general_firewall</nsf-instance-name>
             <nsf-address>221.159.112.100</nsf-address>
             <nsf-port-address>3000</nsf-port-address>
            </nsf-access-info>
           </i2nsf-nsf-capability-registration>
          </i2nsf-nsf-registrations>
```

Figure 12: Configuration XML for Registration of General Firewall

Figure 12 shows the configuration XML for registration of general firewall and its capabilities are as follows.

1. The instance name of the NSF is general_firewall.

2. The NSF can inspect protocol, exact IPv4 address, and range IPv4 address for IPv4 packets.

3. The NSF can inspect exact port number and range port number for tcp packets.

4. The NSF can control whether the packets are allowed to pass, drop, or alert.

5. The NSF can have processing power and bandwidth.

6. The location of the NSF is 221.159.112.100.

7. The port of the NSF is 3000.

A.2. Example 2: Registration for Capabilities of Time based Firewall

This section shows a configuration example for capabilities registration of time based firewall.

```
    <i2nsf-nsf-registrations
      xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-reg-interface"
      xmlns:capa="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
       <i2nsf-nsf-capability-registration>
        <nsf-name>time_based_firewall_capability</nsf-name>
        <nsf-capability-info>
          <i2nsf-capability>
```

```
              <time-capabilities>absolute-time</time-capabilities>
              <time-capabilities>periodic-time</time-capabilities>
              <condition-capabilities>
              <generic-nsf-capabilities>
                <ipv4-capa>capa:ipv4-protocol</ipv4-capa>
                <ipv4-capa>capa:exact-ipv4-address</ipv4-capa>
                <ipv4-capa>capa:range-ipv4-address</ipv4-capa>
              </generic-nsf-capabilities>
            </condition-capabilities>
            <action-capabilities>
              <ingress-action-capa>capa:pass</ingress-action-capa>
              <ingress-action-capa>capa:drop</ingress-action-capa>
              <ingress-action-capa>capa:alert</ingress-action-capa>
              <egress-action-capa>capa:pass</egress-action-capa>
              <egress-action-capa>capa:drop</egress-action-capa>
              <egress-action-capa>capa:alert</egress-action-capa>
            </action-capabilities>
          </i2nsf-capability>
          <nsf-performance-capability>
            <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>
          </nsf-performance-capability>
        </nsf-capability-info>
        <nsf-access-info>
          <nsf-instance-name>time_based_firewall</nsf-instance-name>
          <nsf-address>221.159.112.110</nsf-address>
          <nsf-port-address>3000</nsf-port-address>
        </nsf-access-info>
      </i2nsf-nsf-capability-registration>
    </i2nsf-nsf-registrations>
```

Figure 13: Configuration XML for Registration of Time based Firewall

Figure 13 shows the configuration XML for registration of time based firewall and its capabilities are as follows.

1. The instance name of the NSF is time_based_firewall.

2. The NSF can execute the security policy rule according to absolute time and periodic time.

3. The NSF can inspect protocol, exact IPv4 address, and range IPv4 address for IPv4 packets.

4. The NSF can control whether the packets are allowed to pass, drop, or alert.

5. The NSF can have processing power and bandwidth.

6. The location of the NSF is 221.159.112.110.

7. The port of the NSF is 3000.

A.3. Example 3: Registration for Capabilities of Web Filter

This section shows a configuration example for capabilities registration of web filter.

```
<i2nsf-nsf-registrations
  xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-reg-interface"
  xmlns:capa="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
  <i2nsf-nsf-capability-registration>
    <nsf-name>web_filter_capability</nsf-name>
    <nsf-capability-info>
      <i2nsf-capability>
        <condition-capabilities>
          <advanced-nsf-capabilities>
            <http-capa>capa:url</http-capa>
          </advanced-nsf-capabilities>
        </condition-capabilities>
        <action-capabilities>
          <ingress-action-capa>capa:pass</ingress-action-capa>
          <ingress-action-capa>capa:drop</ingress-action-capa>
          <ingress-action-capa>capa:alert</ingress-action-capa>
          <egress-action-capa>capa:pass</egress-action-capa>
          <egress-action-capa>capa:drop</egress-action-capa>
          <egress-action-capa>capa:alert</egress-action-capa>
        </action-capabilities>
      </i2nsf-capability>
      <nsf-performance-capability>
        <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>
      </nsf-performance-capability>
    </nsf-capability-info>
    <nsf-access-info>
      <nsf-instance-name>web_filter</nsf-instance-name>
      <nsf-address>221.159.112.120</nsf-address>
      <nsf-port-address>3000</nsf-port-address>
    </nsf-access-info>
  </i2nsf-nsf-capability-registration>
</i2nsf-nsf-registrations>
```

Figure 14: Configuration XML for Registration of Web Filter

Figure 14 shows the configuration XML for registration of web filter and its capabilities are as follows.

1. The instance name of the NSF is web_filter.

2. The NSF can inspect url for http and https packets.

3. The NSF can control whether the packets are allowed to pass, drop, or alert.

4. The NSF can have processing power and bandwidth.

5. The location of the NSF is 221.159.112.120.

6. The port of the NSF is 3000.

A.4. Example 4: Registration for Capabilities of VoIP/VoLTE Filter

This section shows a configuration example for capabilities registration of VoIP/VoLTE filter.

```
<i2nsf-nsf-registrations
  xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-reg-interface"
    xmlns:capa="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
  <i2nsf-nsf-capability-registration>
    <nsf-name>voip_volte_filter_capability</nsf-name>
    <nsf-capability-info>
      <i2nsf-capability>
        <condition-capabilities>
          <advanced-nsf-capabilities>
            <voip-volte-capa>capa:voice-id</voip-volte-capa>
          </advanced-nsf-capabilities>
        </condition-capabilities>
        <action-capabilities>
          <ingress-action-capa>capa:pass</ingress-action-capa>
          <ingress-action-capa>capa:drop</ingress-action-capa>
          <ingress-action-capa>capa:alert</ingress-action-capa>
          <egress-action-capa>capa:pass</egress-action-capa>
          <egress-action-capa>capa:drop</egress-action-capa>
          <egress-action-capa>capa:alert</egress-action-capa>
        </action-capabilities>
      </i2nsf-capability>
      <nsf-performance-capability>
      <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>
    </nsf-performance-capability>
  </nsf-capability-info>
  <nsf-access-info>
    <nsf-instance-name>voip_volte_filter</nsf-instance-name>
    <nsf-address>221.159.112.130</nsf-address>
    <nsf-port-address>3000</nsf-port-address>
  </nsf-access-info>
 </i2nsf-nsf-capability-registration>
</i2nsf-nsf-registrations>
```

Figure 15: Configuration XML for Registration of VoIP/VoLTE Filter

Figure 15 shows the configuration XML for registration of VoIP/VoLTE filter and its capabilities are as follows.

1. The instance name of the NSF is voip_volte_filter.

2. The NSF can inspect voice id for VoIP/VoLTE packets.

3. The NSF can control whether the packets are allowed to pass, drop, or alert.

4. The NSF can have processing power and bandwidth.

5. The location of the NSF is 221.159.112.130.

6. The port of the NSF is 3000.

A.5. Example 5: Registration for Capabilities of HTTP and HTTPS Flood Mitigation

This section shows a configuration example for capabilities registration of http and https flood mitigation.

```
<i2nsf-nsf-registrations
  xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-reg-interface"
    xmlns:capa="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
  <i2nsf-nsf-capability-registration>
    <nsf-name>
      http_and_https_flood_mitigation_capability
    </nsf-name>
     <nsf-capability-info>
      <i2nsf-capability>
           <condition-capabilities>
             <advanced-nsf-capabilities>
               <antiddos-capa>capa:http-flood-action</antiddos-capa>
               <antiddos-capa>capa:https-flood-action</antiddos-capa>
              </advanced-nsf-capabilities>
            </condition-capabilities>
            <action-capabilities>
              <ingress-action-capa>capa:pass</ingress-action-capa>
              <ingress-action-capa>capa:drop</ingress-action-capa>
              <ingress-action-capa>capa:alert</ingress-action-capa>
              <egress-action-capa>capa:pass</egress-action-capa>
              <egress-action-capa>capa:drop</egress-action-capa>
              <egress-action-capa>capa:alert</egress-action-capa>
            </action-capabilities>
    </i2nsf-capability>
    <nsf-performance-capability>
      <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>
        </nsf-performance-capability>
      </nsf-capability-info>
      <nsf-access-info>
        <nsf-instance-name>
          http_and_https_flood_mitigation
        </nsf-instance-name>
        <nsf-address>221.159.112.140</nsf-address>
        <nsf-port-address>3000</nsf-port-address>
      </nsf-access-info>
    </i2nsf-nsf-capability-registration>
  </i2nsf-nsf-registrations>
```

Figure 16: Configuration XML for Registration of of HTTP and HTTPS Flood Mitigation

Figure 16 shows the configuration XML for registration of VoIP/VoLTE filter and its capabilities are as follows.

1. The instance name of the NSF is http_and_https_flood_mitigation.

2. The NSF can control the amount of packets for http and https packets.

3. The NSF can control whether the packets are allowed to pass, drop, or alert.

4. The NSF can have processing power and bandwidth.

5. The location of the NSF is 221.159.112.140.

6. The port of the NSF is 3000.

A.6. Example 6: Query for Capabilities of Time based Firewall

This section shows a configuration example for capabilities query of Time based Firewall.

```
<rpc message-id="101"
  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <i2nsf-nsf-capability-query
    xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-reg-interface"
      xmlns:capa="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
     <query-i2nsf-capability-info>
          <time-capabilities>absolute-time</time-capabilities>
          <time-capabilities>periodic-time</time-capabilities>
          <condition-capabilities>
            <generic-nsf-capabilities>
              <ipv4-capa>capa:ipv4-protocol</ipv4-capa>
              <ipv4-capa>capa:exact-ipv4-address</ipv4-capa>
              <ipv4-capa>capa:range-ipv4-address</ipv4-capa>
            </generic-nsf-capabilities>
          </condition-capabilities>
          <action-capabilities>
            <ingress-action-capa>capa:pass</ingress-action-capa>
            <ingress-action-capa>capa:drop</ingress-action-capa>
            <ingress-action-capa>capa:alert</ingress-action-capa>
            <egress-action-capa>capa:pass</egress-action-capa>
            <egress-action-capa>capa:drop</egress-action-capa>
            <egress-action-capa>capa:alert</egress-action-capa>
          </action-capabilities>
    </query-i2nsf-capability-info>
  </i2nsf-nsf-capability-query>
</rpc>


<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-reg-interface">
    <nsf-instance-name>time-based-firewall</nsf-instance-name>
    <nsf-address>221.159.223.250</nsf-address>
    <nsf-port-address>8080</nsf-port-address>
  </nsf-access-info>
</rpc-reply>
```

Figure 17: Configuration XML for Query of Time based Firewall

Figure 17 shows the configuration of input data and output data XML for nsf capability query of time based firewall.

Appendix B. NSF Lifecycle Managmenet 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 C. Changes from draft-ietf-i2nsf-registration-interface-dm-01

The following changes have been made from draft-ietf-i2nsf-registration-interface-dm-01:

- Section 4 has been revised to clarify major objectives of the I2NSF registration interface: NSF capability registration, NSF capability query.

- Section 5 has been revised to describe the above-mentioned major operations of the I2NSF registration interface. Section 5.1 describes the information model for registering NSFs and their capabilities. Section 5.2 describes the information model for querying NSFs based on a description of required capabilities.

- In section 6, the data model has been revised according to the revised information model.

- Appendix A. has been revised to describe the XML examples of the registration interface data model in five NSF Registration examples and one NSF Capability Query example.

Appendix D. 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 E. 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:

- Jinyong Tim Kim (Sungkyunkwan University)
- Susan Hares (Huawei)
- Diego R. Lopez (Telefonica)
- Chung, Chaehong (Sungkyunkwan University)

Authors' Addresses

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

EMail: 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
EMail: 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
EMail: tkroh0198@skku.edu

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: 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