Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 3, 2019

S. Hyun
Chosun University
J. Jeong
T. Roh
S. Wi
Sungkyunkwan University
J. Park
ETRI
July 2, 2018

I2NSF Registration Interface YANG Data Model
draft-hyun-i2nsf-registration-interface-dm-04

Abstract

This document describes an YANG data model for I2NSF registration interface between Security Controller and Developer's Management System. The data model is required for NSF instance registration and dynamic life cycle management of NSF instances.

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 January 3, 2019.

Copyright Notice

Table of Contents

1. Introduction

This document provides a YANG [RFC6020] data model that defines the required data for the registration interface between Security Controller and Developer's Management System to dynamically manage a pool of NSF instances. This document defines a YANG data model based on the [i2nsf-reg-inf-im]. The terms used in this document are defined in [i2nsf-terminology].

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 terminology described 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]

- 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]

- Network Security Function 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]

- 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]

3.1. Tree Diagrams

A simplified graphical representation of the data model is used in this document. The meaning of the symbols in these 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.

4. High-Level YANG

This section provides an overview of the high level YANG.

4.1. Registration Interface

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

Figure 1: High-Level YANG of I2NSF Registration Interface

Each of these sections mirror sections of [i2nsf-reg-inf-im].

4.2. Registration Request

This section expands the i2nsf-regs-req in Figure 1.

```
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 2: 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.

4.3. Instance Management Request

This section expands the i2nsf-instance-mgnt-req in Figure 1.

```
Instance Management Request
  +--rw i2nsf-instance-mgnt-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 (reinstanciation-request)
        +--rw re-nsf-capability-information
        |  uses i2nsf-nsf-capability-information
```

Figure 3: High-Level YANG of I2NSF Instance Mgnt Request

Instance management request consists of two types: instanciation-request, deinstanciation-request, and reinstanciation-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 reinstanciation nsf request is used to updating a existing NSf information with NSF capabilities.

4.4. NSF Capability Information

This section expands the i2nsf-nsf-capability-information in Figure 2 and Figure 3.

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

Figure 4: High-Level YANG of I2NSF NSF Capability Information

In Figure 4, 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]

4.5. NSF Access Information

This section expands the i2nsf-nsf-access-info in Figure 2 and Figure 3.

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

Figure 5: High-Level YANG of I2NSF NSF Access Informantion

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

4.6. NSF Performance Capability

This section expands the i2nsf-nsf-performance-caps in Figure 4.

```
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 6: 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.

4.7. 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 7: 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].

5. 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 the [i2nsf-reg-inf-im].

```
<CODE BEGINS> file "ietf-i2nsf-regs-interface@2018-07-02.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.duhbar@huawei.com>

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

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

       Editor: Sarang Wi
```

```
          <mailto:dnl9795@skku.edu>

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

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

        description
          "It defines a YANG data module for Registration Interface.";
        revision "2018-07-02"{
          description "The second revision";
          reference
          "draft-hares-i2nsf-capability-data-model-07
          draft-hyun-i2nsf-registration-interface-im-05";
        }
      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-mgnt-req {
           description
```

```
            "Required information for instanciation-request,
            deinstanciation-request and reinstanciation-request";
            leaf req-level {
             type uint16;
             description
             "req-level";
           }
            leaf req-id {
             type uint64;
             mandatory true;
             description
             "req-id";
            }
            choice req-type {
             description
             "req-type";
             case instanciation-request {
              description
              "instanciation-request";
             container in-nsf-capability-information {
               description
               "nsf-capability-information";
               uses i2nsf-nsf-capability-information;
              }
             }
             case deinstanciation-request {
              description
              "deinstanciation-request";
              container de-nsf-access-info {
                description
                "nsf-access-info";
                uses i2nsf-nsf-access-info;
               }
             }
             case reinstanciation-request {
              description
               "reinstanciation nsf's information";
               container re-nsf-capability-information {
                description
                "nsf-capability-information";
                uses i2nsf-nsf-capability-information;
               }
             }
            }
           }
           }
       grouping i2nsf-nsf-performance-caps {
           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 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-hares-i2nsf-capability-data-model-07.txt
         later";
        }
      }
      grouping ietf-netmod-acl-model {
        description
        "Detail information";
        container role-based-acl {
         description
         "It refers draft-ietf-netmod-acl-model-15.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 8: Data Model of I2NSF Registration Interface

5.1. 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 9: Registration Interface example

6. Security Considerations

The information model of the registration interface is based on the I2NSF framework without any architectural changes. Thus, this document shares the security considerations of the I2NSF framwork architecture that are specified in [RFC8329] for the purpose of achieving secure communication among components in the proposed architecture.

7. 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).

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs toIndicate 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-00 (work in progress), September 2017.

[i2nsf-capability-dm]
Hares, S., Jeong, J., Kim, J., Moskowitz, R., and Q. Lin, "I2NSF Capability YANG Data Model", draft-hares-i2nsf-capability-data-model-07 (work in progress), March 2018.

[i2nsf-reg-inf-im]
Hyun, S., Jeong, J., Roh, T., Wi, S., and J. Park, "I2NSF Registration Interface Information Model", draft-hyun-i2nsf-registration-interface-im-04 (work in progress), October 2017.

[i2nsf-terminology]
Hares, S., Strassner, J., Lopez, D., Xia, L., and H. Birkholz, "Interface to Network Security Functions (I2NSF) Terminology", draft-ietf-i2nsf-terminology-05 (work in progress), January 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-10 (work in progress), February 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-16 (work in progress), February 2018.

[nsf-triggered-steering]
Hyun, S., Jeong, J., Park, J., and S. Hares, "Service Function Chaining-Enabled I2NSF Architecture", draft-hyun-i2nsf-nsf-triggered-steering-05 (work in progress), March 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. Changes from draft-hyun-i2nsf-registration-interface-dm-03

The following changes are made from draft-hyun-i2nsf-registration-interface-dm-03:

- We added Re-instantiation item.
- The references were updated to reflect the latest documents.

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
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: dnl9795@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