Revision Letter

I2NSF Registration Interface YANG Data Model

(Old Draft Name: draft-ietf-i2nsf-registration-interface-dm-04 and New Draft Name: draft-ietf-i2nsf-registration-interface-dm-05)

Sangwon Hyun and Jaehoon Paul Jeong

07/25/2019

Reviewer: Reshad Rahman

Review result: On the Right Track

YANG Doctor review of draft-ietf-i2nsf-registration-interface-dm-04 (by Reshad Rahman)

Dear Reshad Rahman,

Thanks for your constructive comments.

My answer for your individual question or comment starts with “=> [Sangwon].”

Major comments:

- Look at appendix B of RFC8407 for an example of how a YANG module should be structured. This document does not abide to that.

- Poor descriptions e.g."nsf-name" for leaf "nsf-name" etc

- prefix "iiregi" doesn't seem right. What about "nsfreg"? Probably needs coordination with the other I2NSF YANG modules to have consistency between the prefixes. I see that YD Acee suggested "nsfintf" for draft-ietf-i2nsf-nsf-facing-interface-dm-06

=> [Sangwon] We revised our YANG module to make it abide to the template described in appendix B of RFC8407. We added a detailed description of each component so that the purpose of each component can be better explained. As you suggested, we changed the prefix with “nsfreg.”

```
<CODE BEGINS> file "ietf-i2nsf-reg-interface@2019-07-18.yang"

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

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

```
prefix nsfreg;

import ietf-inet-types {
 prefix inet;
 reference "RFC 6991";
}
import ietf-i2nsf-capability {
 prefix capa;
 reference "draft-ietf-i2nsf-capability-data-model-04";
}

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:shyun@chosun.ac.kr>
  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
 "This module defines a YANG data model for I2NSF registration interface.

  Copyright (c) <2019> 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
  draft-ietf-i2nsf-registration-interface-dm-05; see
  the draft itself for full legal notices.";
```

```
revision 2019-07-18 {
 description "The fifth revision";
 reference
  "draft-ietf-i2nsf-registration-interface-dm-05";
}

typedef nsf-address {
 leaf nsf-ipv4-address {
  type inet:ipv4-address;
  description
   "IPv4 address of an NSF";
 }
 leaf nsf-ipv6-address {
  type inet:ipv6-address;
  description
    "IPv6 address of an NSF";
 }
}
rpc i2nsf-nsf-capability-query {
 description
  "Description of the capabilities that the\
   Security Controller requests to the DMS";
 input {
  container query-i2nsf-capability-info {
   description
    "Description of the capabilities to request";
   uses "capa:nsf-capabilities";
   reference
    "draft-ietf-i2nsf-capability-data-model-04";
   }
 }
 output {
  container nsf-access-info {
   description
    "Network access information of an NSF\
     with the requested capabilities";
   uses i2nsf-nsf-access-info;
  }
 }
}
container i2nsf-nsf-registrations {
 description
  "Information of an NSF that DMS registers to Security Controller";
 list i2nsf-nsf-capability-registration {
  key "nsf-name";
  description
   "Required information for registration";
  leaf nsf-name {
```

```
    type string;
    mandatory true;
    description
     "Unique name of this registered NSF";
    }
    container nsf-capability-info {
     description
      "Capability description of this NSF";
       uses i2nsf-nsf-capability-info;
    }
    container nsf-access-info {
     description
      "Network access information of this NSF";
     uses i2nsf-nsf-access-info;
    }
  }
}

grouping i2nsf-nsf-performance-capability {
 description
  "Description of the performance capailities\
   of an NSF";
 container processing {
  description
   "Processing power of an NSF";
  leaf processing-average {
   type uint16;
   description
    "Average processing power";
  }
  leaf processing-peak {
   type uint16;
   description
    "Peak processing power";
  }
 }
 container bandwidth {
  description
   "Network bandwidth available on an NSF";

  container outbound {
   description
    "Outbound network bandwidth";
   leaf outbound-average {
    type uint16;
    description
     "Average outbound bandwidth";
   }
```

```
   leaf outbound-peak {
    type uint16;
    description
     "Peak outbound bandwidth";
   }
  }
  container inbound {
   description
    "Inbound network bandwidth";
   leaf inbound-average {
    type uint16;
    description
     "Average inbound bandwidth";
   }
   leaf inbound-peak {
    type uint16;
    description
     "Peak inbound bandwidth";
    }
   }
  }
}
grouping i2nsf-nsf-capability-info {
 description
  "Capability description of an NSF";
 container i2nsf-capability {
   description
    "Description of the security capabilities of an NSF";
   uses "capa:nsf-capabilities";
   reference "draft-ietf-i2nsf-capability-data-model-04";
  }
  container nsf-performance-capability {
   description
    "Description of the performance capabilities of an NSF";
   uses i2nsf-nsf-performance-capability;
  }
 }

 grouping i2nsf-nsf-access-info {
  description
   "Information required to access an NSF";
  leaf nsf-instance-name {
   type string;
   description
     "Unique name of this NSF instance";
  }
  container i2nsf-nsf-address {
   uses nsf-address
     description
      "IPv4/IPv6 address of this NSF";
    }
    leaf nsf-port {
     type inet:port-number;
     description
      "Port available on this NSF";
    }
  }
}

<CODE ENDS>
```

- No unit specified for bandwidth, processing (performance)

=> [Sangwon] We have specified unit information: Gbps (gigabits per second) for bandwidth and GHz (gigahertz) for processing.

- nsf-address is IPv4 specific

=> [Sangwon] We have revised nsf-address so that it can deal with both IPv4 and IPv6 as follows.

[OLD]:

```
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

```
      }
      leaf nsf-address {
        type inet:ipv4-address;
        mandatory true;
        description
          "nsf-address";
      }
```

[NEW]:

```
NSF Access Information
  +--rw i2nsf-nsf-access-info
    +--rw nsf-instance-name      string
    +--rw i2nsf-nsf-address
      +-- rw nsf-ipv4-address       inet:ipv4-address
      +-- rw nsf-ipv6-address       inet:ipv6-address
    +--rw nsf-port-number        inet:port-number
```

Figure 10: YANG tree of I2NSF NSF Access Informantion

```
}
typedef nsf-address {
  leaf nsf-ipv4-address{
    type inet:ipv4-address
    description
    "ipv4-address"
  }
  leaf nsf-ipv6-address {
    type inet:ipv6-address
    description
    "ipv6-address"
  }
}
```

```
    container i2nsf-nsf-address {
      uses nsf-address
      description
        "ipv4 and ipv6";
    }
```

- Security considerations should list the nodes as per section 3.7 of RFC8407.

=> [Sangwon] We have added the following discussions to Section 8 of security considerations by following the guidelines described in Section 3.7 of RFC8407.

In Section 8,

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

- i2nsf-nsf-registrations: The attacker may exploit this to register a compromised or malicious NSF instead of a legitimate NSF to the Security Controller.
- i2nsf-nsf-performance-capability: The attacker may provide incorrect information of the performance capability of any target NSF by illegally modifying this.
- i2nsf-nsf-capability-info: The attacker may provide incorrect information of the security capability of any target NSF by illegally modifying this.
- i2nsf-nsf-access-info: The attacker may provide incorrect network access information of any target NSF by illegally modifying this.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

- i2nsf-nsf-registrations: The attacker may try to gather some sensitive information of a registered NSF by sniffing this.
- i2nsf-nsf-performance-capability: The attacker may gather the performance capability information of any target NSF and misuse the information for subsequent attacks.
- i2nsf-nsf-capability-info: The attacker may gather the security capability information of any target NSF and misuse the information for subsequent attacks.
- i2nsf-nsf-access-info: The attacker may gather the network access information of any target NSF and misuse the information for subsequent attacks.

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

- i2nsf-nsf-capability-query: The attacker may exploit this RPC operation to deteriorate the availability of the DMS and/or gather the information of some interested NSFs from the DMS.

- Should this document be informational since 8329 is informational?

=> [Sangwon] We authors believe that the standard track is more appropriate for this document for interoperability among multiple parties' implementations.

- Section 2 should use RFC8174 also

=> [Sangwon] As you suggested, we revised Section 2 to use RFC8174.

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

- Examples should use IPv6 as examples (use the range from RFC3849). Kudos for all the examples.
=> [Sangwon] We have revised all the examples to use IPv6 by following the guidelines in RFC3849.

```
<nsf-access-info>
 <nsf-instance-name>general_firewall</nsf-instance-name>
 <i2nsf-nsf-address>
   <nsf-ipv6-address>2001:DB8:8:4::2</nsf-ipv6-address>
 </i2nsf-nsf-address>
 <nsf-port-address>3000</nsf-port-address>
</nsf-access-info>
```

```
<nsf-access-info>
  <nsf-instance-name>time_based_firewall</nsf-instance-name>
  <i2nsf-nsf-address>
    <nsf-ipv6-address>2001:DB8:8:4::3</nsf-ipv6-address>
  </i2nsf-nsf-address>
  <nsf-port-address>3000</nsf-port-address>
```

```
  <nsf-access-info>
    <nsf-instance-name>web_filter</nsf-instance-name>
    <i2nsf-nsf-address>
      <nsf-ipv6-address>2001:DB8:8:4::4</nsf-ipv6-address>
    </i2nsf-nsf-address>
    <nsf-port-address>3000</nsf-port-address>
  </nsf-access-info>
```

```
  <nsf-access-info>
    <nsf-instance-name>voip_volte_filter</nsf-instance-name>
    <i2nsf-nsf-address>
      <nsf-ipv6-address>2001:DB8:8:4::5</nsf-ipv6-address>
    </i2nsf-nsf-address>
    <nsf-port-address>3000</nsf-port-address>
  </nsf-access-info>
<nst-access-into>
  <nsf-instance-name>
    http_and_https_flood_mitigation
  </nsf-instance-name>
  <i2nsf-nsf-address>
    <nsf-ipv6-address>2001:DB8:8:4::6</nsf-ipv6-address>
  </i2nsf-nsf-address>
  <nsf-port-address>3000</nsf-port-address>
</nsf-access-info>
```

```
  <nsf-access-info
    xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-reg-interface">
    <nsf-instance-name>time-based-firewall</nsf-instance-name>
    <i2nsf-nsf-address>
      <nsf-ipv6-address>2001:DB8:8:4::7</nsf-ipv6-address>
    </i2nsf-nsf-address>
    <nsf-port-address>8080</nsf-port-address>
  </nsf-access-info>
```

Minor comments and questions:

- The YANG trees such as Figure 6, 7 etc. don't show the contents of the groupings. So they don't help much.

=> [Sangwon] In Figures 6 and 7, we have added the contents of the groupings.

Figure 6:

```
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 i2nsf-capability
              |  uses ietf-i2nsf-capability
              +--rw nsf-performance-capability
              |  uses i2nsf-nsf-performance-capability
        +--rw nsf-access-info
        |  uses i2nsf-nsf-access-info
              +--rw nsf-instance-name
              +--rw i2nsf-nsf-address
              +-- rw nsf-ipv4-address
              +-- rw nsf-ipv6-address
              +--rw nsf-port-number
```

Figure 7:

```
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
              +--rw nsf-instance-name
              +--rw i2nsf-nsf-address
              +-- rw nsf-ipv4-address
              +-- rw nsf-ipv6-address
              +--rw nsf-port-number
```

- nsf-port-address should be nsf-port?

=> [Sangwon] “nsf-port-address” has been changed into “nsf-port.”

```
leaf nsf-port {
 type inet:port-number;
 description
  "Port available on this NSF";
}
```

- Section 4, last bullet. I am not an expert on I2NSF so not clear to me why this query is needed, is it because NSF may not re-register after their capabilities have been updated? Might be worth adding some explanation.

=> [Sangwon] We have added more explanations in Section 4 to clarify why querying DMS is necessary.

In Section 4,

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

- Have the examples been validated?

=> [Sangwon] Yes. The examples have been validated through our prototype implementation in IETF Hackathon:

- I2NSF Framework Project at the IETF-105 Hackathon:
https://trac.ietf.org/trac/ietf/meeting/wiki/105hackathon

- Slides for the IETF-105 I2NSF Framework Hackathon Project:
https://github.com/IETF-Hackathon/ietf105-project-presentations/blob/master/IETF105-I2NSF-Hackathon.pdf

- Github for the Open Source of the I2NSF Framework Project:
https://github.com/kimjinyong/i2nsf-framework

- Demonstration Video Clip for the I2NSF Framework Project:

https://www.youtube.com/watch?v=jD4ndqzN0is

Thanks.

Best Regards,

Sangwon and Paul