Workgroup: I2NSF Working Group
Internet-Draft:
draft-ietf-i2nsf-registration-interface-dm-26
Published: 10 May 2023
Intended Status: Standards Track
Expires: 11 November 2023
Authors: S. Hyun, Ed.
Myongji University
J. Jeong, Ed.
Sungkyunkwan University
T. Roh
Sungkyunkwan University
S. Wi
Sungkyunkwan University
J. Park
ETRI

I2NSF Registration Interface YANG Data Model for NSF Capability Registration

Abstract

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

Status of This Memo

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

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

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

This Internet-Draft will expire on 11 November 2023.

Table of Contents

1. Introduction

A number of Network Security Functions (NSF) may exist in the Interface to Network Security Functions (I2NSF) framework [RFC8329]. Since each of these NSFs likely has different security capabilities from each other, it is important to register the security capabilities of the NSFs to the Security Controller (i.e., Network Operator Management System in [RFC8329]). 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 Developer's Management System (DMS) for NSFs that have the required security capabilities.

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

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

2. Terminology

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

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

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

malware scanning, sandbox, Data Loss Prevention (DLP), Distributed Denial of Service (DDoS) mitigation and TLS proxy.

*Data Model: Data Models define managed objects at a lower level of abstraction, which include implementation- and protocol-specific details, e.g., rules that explain how to map managed objects onto lower-level protocol constructs [RFC3444].

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

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

3. Objectives

*Registering NSFs with the I2NSF framework: Developer's Management System (DMS) in I2NSF framework is typically run by an NSF vendor, and uses Registration Interface to provide NSFs information (i.e., capability, specification, and access information) developed by the NSF vendor to Security Controller. Since there may be multiple vendors that provide NSFs for a target network, the I2NSF Registration Interface can be used as a standard interface for the DMSs to provide NSFs capability information to the Security Controller. For the registered NSFs, Security Controller maintains a catalog of the capabilities of those NSFs to select appropriate NSFs for the requested security services. Note that the I2NSF User and the vendor should exchange information for the discovery of Security Controller and DMS during the subscription of the security service. The I2NSF User should provide the Security Controller information (e.g., access information) to the DMS for the NSFs registration, and the vendor should provide the DMS information (e.g., access information and the types of NSFs managed by the DMS) to the Security Controller for allowing such connections. The method of exchanging this information can be done either manually or dynamically (e.g., through the new options of I2NSF information in both DHCP [RFC2131] and DHCPv6 [RFC8415]). This actual method is out of the scope of this document.

*Updating the capabilities of registered NSFs: After an NSF is registered with Security Controller, some modifications on the capability of the NSF may be required later. In this case, DMS uses Registration Interface to deliver the update of the

capability of the NSF to the Security Controller, and this update MUST be reflected on the catalog of NSFs existing in the Security Controller. That is, the DMS sends the updated NSF capability information to the Security Controller through a notification mechanism. The Security Controller updates its catalog of NSFs with the updated NSF capability information.

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

4. Information Model of Registration Interface

The I2NSF registration interface is used by Security Controller and Developer's Management System (DMS) in I2NSF framework. Figure 1 shows the information model of the I2NSF registration interface, which consists of two submodels: NSF capability registration and NSF capability update. Each submodel is used for the operations listed above. The remainder of this section will provide in-depth explanation of each submodel. The consideration of the design of the data model is based on the procedure and mechanism discussed in Section 8 of [I-D.ietf-i2nsf-applicability], which discusses I2NSF Framework with Network Functions Virtualization (NFV) [nfv-framework].

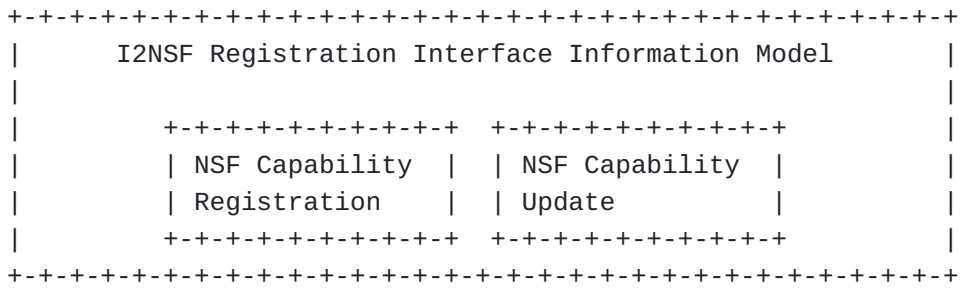

Figure 1: I2NSF Registration Interface Information Model

4.1. NSF Capability Registration

This submodel is used by the DMS to register the capabilities of NSFs with the request of the Security Controller. Figure 2 shows how this submodel is constructed. The most important part in Figure 2 is the NSF capability, and this specifies the set of capabilities that the NSF to be registered can offer. The NSF Name contains a unique

name of this NSF with the specified set of capabilities. The NSF name MUST be unique within the registered NSFs in the Security Controller to identify the NSF with the capability. The name can be an arbitrary string including Fully Qualified Domain Name (FQDN). To make sure each vendor does not provide a duplicated name, the name should include the vendor's detail (e.g., firewall-vendor-series_name-series_number). 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 sections will further explain the NSF capability information and the NSF access information in more detail.

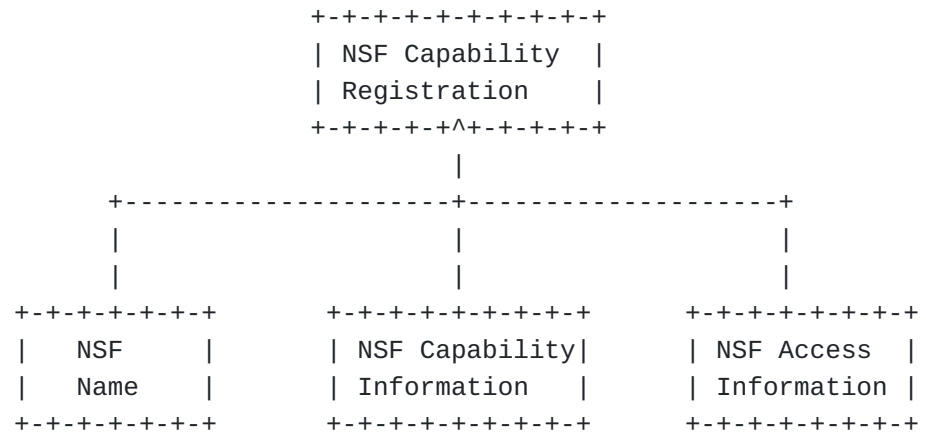

Figure 2: NSF Capability Registration Sub-Model

4.1.1. NSF Capability Information

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

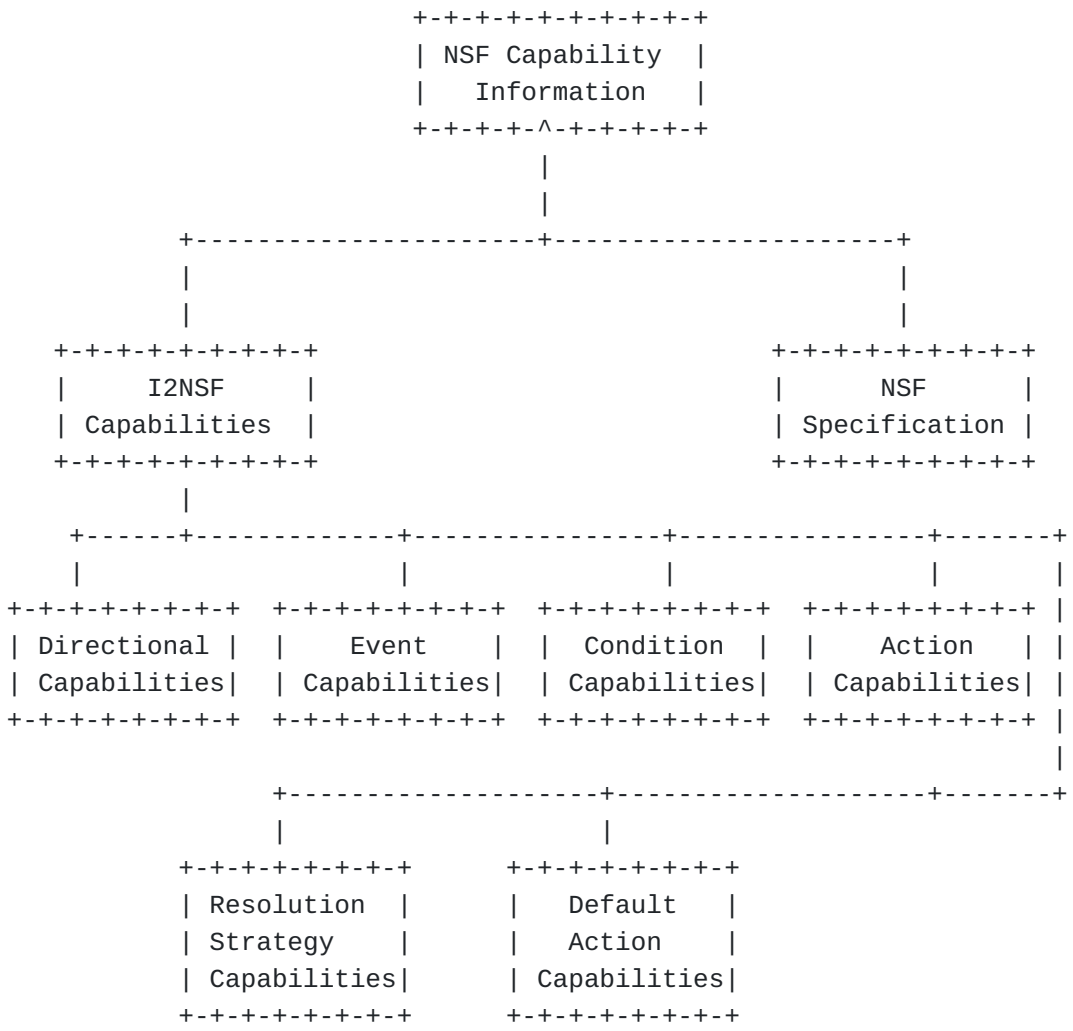

Figure 3: NSF Capability Information

4.1.1.1. NSF Specification

This information represents the specification information of an NSF. As illustrated in Figure 4, this information consists of packet processing and bandwidth capabilities.

Packet processing is the overall capability of an NSF in processing packets measured in packets per second (PPS). Bandwidth describes the information about available network amount in two cases, such as outbound and inbound. Assuming that the current throughput and packet rate statuses of each NSF are being collected through NSF monitoring [I-D.ietf-i2nsf-nsf-monitoring-data-model], these capabilities of the NSF can be used to determine whether the NSF is in congestion or not by comparing it with the current throughput of the NSF.

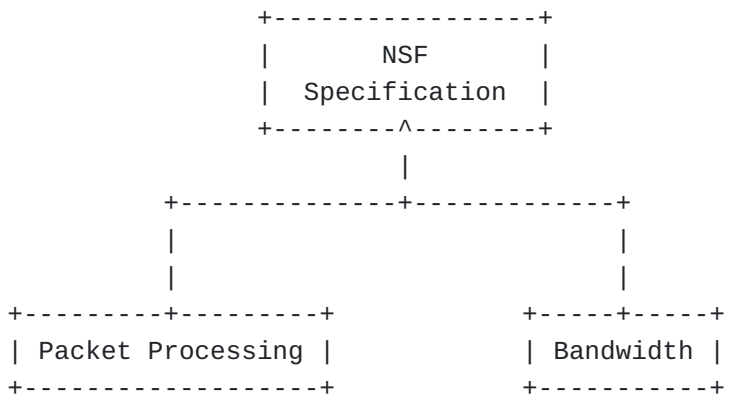

Figure 4: NSF Specification Overview

4.1.2. NSF Access Information

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

4.2. NSF Capability Update

The deployed NSFs may require to be updated to improve the quality of the security service. The Security Controller can request update information to the DMS by subscribing to the NSF capability update notification. The DMS can send the notification of NSF capability update using the NSF capability information submodel in Section 4.1.1 for updating the capabilities of the NSF.

5. YANG Data Model of Registration Interface

5.1. YANG Tree Diagrams of Registration Interface

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

Note that the YANG module in this document relies on the YANG module defined in [I-D.ietf-i2nsf-capability-data-model], hence QUIC protocol [RFC9000] and HTTP/3 are excluded in the data model. The QUIC traffic should not be treated as UDP traffic, and HTTP/3 should neither be interpreted as either HTTP/1.1 nor HTTP/2. Thus, the data model should be extended or augmented appropriately to support the handling of the QUIC protocol and HTTP/3 traffic according to the needs of the implementer.

5.1.1. NSF Capability Registration

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

```
NSF Capability Registration
rpcs:
  +---x nsf-capability-registration
  |  +---w input
  |  |  +---w query-nsf-capability
  |  |     +---w directional-capabilities*          identityref
  |  |     +---w event-capabilities
  |  |     |  +---w system-event-capability*   identityref
  |  |     |  +---w system-alarm-capability*   identityref
  |  |     +---w condition-capabilities
  |  |     |  +---w generic-nsf-capabilities
  |  |     |  |  +---w ethernet-capability*   identityref
  |  |     |  |  +---w ipv4-capability*       identityref
  |  |     |  |  +---w ipv6-capability*       identityref
  |  |     |  |  +---w icmpv4-capability*     identityref
  |  |     |  |  +---w icmpv6-capability*     identityref
  |  |     |  |  +---w tcp-capability*        identityref
  |  |     |  |  +---w udp-capability*        identityref
  |  |     |  |  +---w sctp-capability*       identityref
  |  |     |  |  +---w dccp-capability*       identityref
  |  |     |  +---w advanced-nsf-capabilities
  |  |     |  |  +---w anti-ddos-capability*            identityref
  |  |     |  |  +---w ips-capability*                  identityref
  |  |     |  |  +---w anti-virus-capability*           identityref
  |  |     |  |  +---w url-filtering-capability*        identityref
  |  |     |  |  +---w voip-vocn-filtering-capability*  identityref
  |  |     |  +---w context-capabilities
  |  |     |     +---w time-capabilities*                identityref
  |  |     |     +---w application-filter-capabilities*  identityref
  |  |     |     +---w device-type-capabilities*         identityref
  |  |     |     +---w user-condition-capabilities*      identityref
  |  |     |     +---w geographic-capabilities*          identityref
  |  |     +---w action-capabilities
  |  |     |  +---w ingress-action-capability*   identityref
  |  |     |  +---w egress-action-capability*    identityref
  |  |     |  +---w log-action-capability*       identityref
  |  |     +---w resolution-strategy-capabilities*   identityref
  |  |     +---w default-action-capabilities*        identityref
  |  +--ro output
  |     +--ro nsf* [nsf-name]
  |        +--ro nsf-name                           string
  |        +--ro version?                           string
  |        +--ro directional-capabilities*          identityref
  |        +--ro event-capabilities
  |        |  +--ro system-event-capability*   identityref
  |        |  +--ro system-alarm-capability*   identityref
  |        +--ro condition-capabilities
  |        |  +--ro generic-nsf-capabilities
  |        |  |  +--ro ethernet-capability*   identityref
```

```
|       |  |  +--ro ipv4-capability*       identityref
|       |  |  +--ro ipv6-capability*       identityref
|       |  |  +--ro icmpv4-capability*     identityref
|       |  |  +--ro icmpv6-capability*     identityref
|       |  |  +--ro tcp-capability*        identityref
|       |  |  +--ro udp-capability*        identityref
|       |  |  +--ro sctp-capability*       identityref
|       |  |  +--ro dccp-capability*       identityref
|       |  +--ro advanced-nsf-capabilities
|       |  |  +--ro anti-ddos-capability*             identityref
|       |  |  +--ro ips-capability*                   identityref
|       |  |  +--ro anti-virus-capability*            identityref
|       |  |  +--ro url-filtering-capability*         identityref
|       |  |  +--ro voip-vocn-filtering-capability*   identityref
|       |  +--ro context-capabilities
|       |     +--ro time-capabilities*                 identityref
|       |     +--ro application-filter-capabilities*   identityref
|       |     +--ro device-type-capabilities*          identityref
|       |     +--ro user-condition-capabilities*       identityref
|       |     +--ro geographic-capabilities*           identityref
|       +--ro action-capabilities
|       |  +--ro ingress-action-capability*   identityref
|       |  +--ro egress-action-capability*    identityref
|       |  +--ro log-action-capability*       identityref
|       +--ro resolution-strategy-capabilities*   identityref
|       +--ro default-action-capabilities*        identityref
|       +--ro nsf-specification
|       |  +--ro packet-processing?   uint64
|       |  +--ro bandwidth
|       |     +--ro outbound?   uint64
|       |     +--ro inbound?    uint64
|       +--ro nsf-access-info
|          +--ro ip?                    inet:ip-address-no-zone
|          +--ro port?                  inet:port-number
|          +--ro management-protocol?   enumeration
```

Figure 5: YANG Tree Diagram of NSF Capability Registration Module

When a Security Controller requests security services to the DMS, DMS uses the I2NSF Capability YANG Data Model [I-D.ietf-i2nsf-capability-data-model] to describe what capabilities the NSFs can offer. Security Controller makes a description of the required capabilities and then queries DMS about which NSF(s) can provide these capabilities. The DMS can apply a selection mechanism to select the NSFs that cover all requested capabilities effectively. This selection mechanism is out of the scope of this document. DMS includes the access information of the NSF which is required to make a network connection with the NSF as well as the specification of the NSFs. The NSF access information consists of ip, port, and management-protocol. The field of ip can have either an IPv4 address or an IPv6 address. The port field is used to get the transport protocol port number. As I2NSF uses a YANG data model, the management protocol can be either NETCONF or RESTCONF.

The credential management for accessing the NSFs is handled by pre-negotiation with every DMS. This management is out of the scope of this document.

The DMS can also include the resource information in terms of packet processing and bandwidth capabilities of the NSF. Detailed overview of NSF specification can be seen in Section 4.1.1.1.

5.1.2. NSF Capability Update

This section describes the YANG tree diagram for the NSF capability update.

```
NSF Capability Update
notifications:
  +---n nsf-capability-update
     +--ro nsf* [nsf-name]
        +--ro nsf-name                              string
        +--ro version?                              string
        +--ro directional-capabilities*             identityref
        +--ro event-capabilities
        |  +--ro system-event-capability*   identityref
        |  +--ro system-alarm-capability*   identityref
        +--ro condition-capabilities
        |  +--ro generic-nsf-capabilities
        |  |  +--ro ethernet-capability*   identityref
        |  |  +--ro ipv4-capability*       identityref
        |  |  +--ro ipv6-capability*       identityref
        |  |  +--ro icmpv4-capability*     identityref
        |  |  +--ro icmpv6-capability*     identityref
        |  |  +--ro tcp-capability*        identityref
        |  |  +--ro udp-capability*        identityref
        |  |  +--ro sctp-capability*       identityref
        |  |  +--ro dccp-capability*       identityref
        |  +--ro advanced-nsf-capabilities
        |  |  +--ro anti-ddos-capability*             identityref
        |  |  +--ro ips-capability*                   identityref
        |  |  +--ro anti-virus-capability*            identityref
        |  |  +--ro url-filtering-capability*         identityref
        |  |  +--ro voip-vocn-filtering-capability*   identityref
        |  +--ro context-capabilities
        |     +--ro time-capabilities*                 identityref
        |     +--ro application-filter-capabilities*   identityref
        |     +--ro device-type-capabilities*          identityref
        |     +--ro user-condition-capabilities*       identityref
        |     +--ro geographic-capabilities*           identityref
        +--ro action-capabilities
        |  +--ro ingress-action-capability*   identityref
        |  +--ro egress-action-capability*    identityref
        |  +--ro log-action-capability*       identityref
        +--ro resolution-strategy-capabilities*   identityref
        +--ro default-action-capabilities*        identityref
        +--ro nsf-specification
        |  +--ro packet-processing?   uint64
        |  +--ro bandwidth
        |     +--ro outbound?   uint64
        |     +--ro inbound?    uint64
        +--ro nsf-access-info
           +--ro ip?                    inet:ip-address-no-zone
           +--ro port?                  inet:port-number
           +--ro management-protocol?   enumeration
```

Figure 6: YANG Tree Diagram of NSF Capability Update Module

This YANG data model is used to update the registered NSFs. The update operation started by the Security Controller subscribing to the notification of NSFs capability updates. See [RFC8639] for the subscription mechanism. Appendix A explains the event stream for the subscription of this YANG module.

The DMS should only send the NSF(s) with updated capabilities. If an update is available, the DMS can deliver the NSF capability update notification to the Security Controller. This YANG module allows multiple NSF capability updates in a single notification. Note that the capabilities given in the update represent the full capabilities of each NSF.

5.2. YANG Module of Registration Interface

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

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

```
<CODE BEGINS> file "ietf-i2nsf-registration-interface@2023-05-10.yang"

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

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

  prefix
    i2nsfri;

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

  import ietf-inet-types {
    prefix inet;
    reference "RFC 6991";
  }

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

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

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

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

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

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

     The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL',
     'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED',
     'NOT RECOMMENDED', 'MAY', and 'OPTIONAL' in this
     document are to be interpreted as described in BCP 14
     (RFC 2119) (RFC 8174) when, and only when, they appear
     in all capitals, as shown here.
```

```
   Copyright (c) 2023 IETF Trust and the persons
   identified as authors of the code. All rights reserved.

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

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

revision "2023-05-10" {
  description "Initial revision";
  reference
    "RFC XXXX: I2NSF Registration Interface YANG Data Model";
  // RFC Ed.: replace XXXX with actual RFC number and remove
  // this note
}

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

  leaf packet-processing {
    type uint64;
    units "pps";
    description
      "The overall packet processing capability of the NSF measured in
       packets per second (pps).";
  }

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

    leaf outbound {
      type uint64;
      units "Bps";
      description
        "The maximum aggregate outbound network bandwidth across all
         interfaces available to the NSF in bytes per second (Bps).";
    }

    leaf inbound {
      type uint64;
      units "Bps";
```

```
      description
        "The maximum aggregate inbound network bandwidth across all
         interfaces available to the NSF in bytes per second (Bps).";
    }
  }

}

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

grouping nsf {
  description
    "The information of an NSF. It consists of the name of the NSF,
     NSF capabilities, NSF specifications, and NSF access
     information";
  leaf nsf-name {
    type string;
    description
```

```
      "The name of this registered NSF. The NSF name MUST be
       unique to identify the NSF with the capability. The name
       can be an arbitrary string including Fully Qualified
       Domain Name (FQDN).";
  }
  leaf version {
    type string;
    description
      "The NSF's current version level of the software in use.
       This string MAY indicate the specific software build date and
       target variant information.";
  }
  uses i2nsfcap:nsf-capabilities;
  container nsf-specification {
    description
      "The specification of an NSF.";
    uses nsf-specification;
  }
  container nsf-access-info {
    description
      "Network access information of this NSF.";
    uses nsf-access-info;
  }
}

rpc nsf-capability-registration {
  description
    "Description of the capabilities that the
     Security Controller requests to the DMS";
  input {
    container query-nsf-capability {
      description
        "The query used to request NSFs. The specified
         capabilities in this field restrict the output field.";
      uses i2nsfcap:nsf-capabilities;
      reference "RFC YYYY: I2NSF Capability YANG Data Model";
    //RFC Ed.: replace YYYY with actual RFC number of
    //draft-ietf-i2nsf-capability-data-model and remove this note.
    }
  }
  output {
    list nsf {
      key "nsf-name";
      description
        "The reply of the query to register the NSFs capabilities.
         The capabilities requested in the input field can be covered
         by multiple NSFs. This list consists of NSF(s) that cover
         every capability specified in the input field.  The
         selection method of which NSF(s) that should be listed in
```

```
            the output field depends on the implementer.  If any of
            the capabilities specified in the input field cannot be
            covered by any NSF, the reply should return an <rpc-error>
            with <error-message> of those capabilities.";
          uses nsf;
        }
      }
    }

    notification nsf-capability-update {
      description
        "This notification is sent when there are updates on the
         capabilities of one or more NSFs. The list of NSFs provided in
         this notification includes the current capabilities of each NSF.
         Note that the returned capabilities represent the full
         capabilities of each NSF.";
      list nsf {
        key "nsf-name";
        description
          "List of NSFs with capabilities updated. The returned
           capabilities are the current capabilities of the NSF.";
        uses nsf;
      }
    }
}

<CODE ENDS>
```

Figure 7: Registration Interface YANG Data Model

6. IANA Considerations

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

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

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

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

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

7. Security Considerations

The YANG module specified in this document defines a data schema designed to be accessed through network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the required secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the required secure transport is TLS [RFC8446].

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

The architecture of I2NSF Framework presents a risk to the implementation of security detection and mitigation activities. The risks of externally operated NSFs are discussed in Section 4 (Threats Associated with Externally Provided NSFs) of [RFC8329]. It is important to have an authentication and authorization method between the communication of the Security Controller and the DMS. The following are threats that need to be considered and mitigated:

Compromised DMS with valid credentials: It can send falsified information to the Security Controller to mislead existing detection or mitigation devices. Currently, there is no in-framework mechanism to mitigate this, and it is an issue for such

infrastructures. It is important to keep confidential information from unauthorized persons to mitigate the possibility of compromising the DMS with this information.

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

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

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

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

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

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

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

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

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

*nsf-capability-query: The attacker may exploit this RPC operation to deteriorate the availability of the DMS and/or gather the information of some interested NSFs from the DMS. Some of the product capabilities provided by a vendor may be publicly known,

the DMS should provide an authentication and authorization method to make sure this node cannot be used for exploitation.

8. References

8.1. Normative References

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

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, DOI 10.17487/RFC2131, March 1997, <https://www.rfc-editor.org/info/rfc2131>.

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

[RFC5277] Chisholm, S. and H. Trevino, "NETCONF Event Notifications", RFC 5277, DOI 10.17487/RFC5277, July 2008, <https://www.rfc-editor.org/info/rfc5277>.

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

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

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

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

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

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

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

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

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

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

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

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

[RFC8415] Mrugalski, T., Siodelski, M., Volz, B., Yourtchenko, A., Richardson, M., Jiang, S., Lemon, T., and T. Winters,

"Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 8415, DOI 10.17487/RFC8415, November 2018, <https://www.rfc-editor.org/info/rfc8415>.

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

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

[RFC8639] Voit, E., Clemm, A., Gonzalez Prieto, A., Nilsen-Nygaard, E., and A. Tripathy, "Subscription to YANG Notifications", RFC 8639, DOI 10.17487/RFC8639, September 2019, <https://www.rfc-editor.org/info/rfc8639>.

[I-D.ietf-i2nsf-applicability] Jeong, J. P., Hyun, S., Ahn, T., Hares, S., and D. Lopez, "Applicability of Interfaces to Network Security Functions to Network-Based Security Services", Work in Progress, Internet-Draft, draft-ietf-i2nsf-applicability-18, 16 September 2019, <https://datatracker.ietf.org/doc/html/draft-ietf-i2nsf-applicability-18>.

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

8.2. Informative References

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

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

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

[RFC8792]
Watsen, K., Auerswald, E., Farrel, A., and Q. Wu, "Handling Long Lines in Content of Internet-Drafts and RFCs", RFC 8792, DOI 10.17487/RFC8792, June 2020, <https://www.rfc-editor.org/info/rfc8792>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/info/rfc9000>.

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

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

Appendix A. I2NSF Event Stream

This section discusses the NETCONF event stream for an I2NSF Registration subscription. The YANG module in this document supports "ietf-subscribed-notifications" YANG module [RFC8639] for subscription. The reserved event stream name for this document is "I2NSF-Registration". The NETCONF Server (e.g., DMS) MUST support "I2NSF-Registration" event stream for the NETCONF Client (e.g., Security Controller). The "I2NSF-Registration" event stream contains all notifications described in this document.

The following XML example shows the capabilities of the event streams generated by the DMS (e.g., "NETCONF" and "I2NSF-Registration" event streams) for the subscription of NSF capability update. Refer to [RFC5277] for a more detailed explanation of Event Streams. The XML examples in this document follow the line breaks as per [RFC8792].

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="1"
           xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <netconf xmlns="urn:ietf:params:xml:ns:netmod:notification">
      <streams>
        <stream>
          <name>NETCONF</name>
          <description>Default NETCONF Event Stream</description>
          <replaySupport>false</replaySupport>
        </stream>
        <stream>
          <name>I2NSF-Registration</name>
          <description>
            I2NSF Registration Event Stream for Capability Updates
          </description>
          <replaySupport>true</replaySupport>
          <replayLogCreationTime>
            2023-04-13T09:37:39+00:00
          </replayLogCreationTime>
        </stream>
      </streams>
    </netconf>
  </data>
</rpc-reply>
```

Figure 8: Example of NETCONF Server supporting I2NSF-Registration Event Stream

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

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

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <nsf-capability-registration
  xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-registration-interface">
    <query-nsf-capability>
      <condition-capabilities>
        <generic-nsf-capabilities>
          <ipv4-capability>source-address</ipv4-capability>
          <ipv4-capability>destination-address</ipv4-capability>
        </generic-nsf-capabilities>
        <advanced-nsf-capabilities>
          <url-filtering-capability>
            user-defined
          </url-filtering-capability>
        </advanced-nsf-capabilities>
      </condition-capabilities>
      <action-capabilities>
        <ingress-action-capability>drop</ingress-action-capability>
        <egress-action-capability>drop</egress-action-capability>
      </action-capabilities>
    </query-nsf-capability>
  </nsf-capability-registration>
</rpc>
```

Figure 9: XML Examples of an NSF Query with I2NSF Registration Interface Data Model

```
<rpc-reply message-id="101"
  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <nsf
  xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-registration-interface">
    <nsf-name>ipv4_general_firewall</nsf-name>
    <version>1.2.0</version>
    <condition-capabilities>
      <generic-nsf-capabilities>
        <ipv4-capability>next-header</ipv4-capability>
        <ipv4-capability>source-address</ipv4-capability>
        <ipv4-capability>destination-address</ipv4-capability>
        <tcp-capability>source-port-number</tcp-capability>
        <tcp-capability>destination-port-number</tcp-capability>
      </generic-nsf-capabilities>
    </condition-capabilities>
    <action-capabilities>
      <ingress-action-capability>pass</ingress-action-capability>
      <ingress-action-capability>drop</ingress-action-capability>
      <ingress-action-capability>mirror</ingress-action-capability>
      <egress-action-capability>pass</egress-action-capability>
      <egress-action-capability>drop</egress-action-capability>
      <egress-action-capability>mirror</egress-action-capability>
    </action-capabilities>
    <nsf-specification>
      <packet-processing>14000000</packet-processing>
      <bandwidth>
        <outbound>1000000000</outbound>
        <inbound>1000000000</inbound>
      </bandwidth>
    </nsf-specification>
    <nsf-access-info>
      <ip>192.0.2.11</ip>
      <port>49152</port>
      <management-protocol>
        NETCONF
      </management-protocol>
    </nsf-access-info>
  </nsf>
  <nsf
  xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-registration-interface">
    <nsf-name>ipv4_web_filter</nsf-name>
    <version>1.1.0</version>
    <condition-capabilities>
      <advanced-nsf-capabilities>
        <url-filtering-capability>
          user-defined
        </url-filtering-capability>
      </advanced-nsf-capabilities>
    </condition-capabilities>
```

```
    <action-capabilities>
      <ingress-action-capability>pass</ingress-action-capability>
      <ingress-action-capability>drop</ingress-action-capability>
      <ingress-action-capability>mirror</ingress-action-capability>
      <egress-action-capability>pass</egress-action-capability>
      <egress-action-capability>drop</egress-action-capability>
      <egress-action-capability>mirror</egress-action-capability>
    </action-capabilities>
    <nsf-specification>
      <packet-processing>14000000</packet-processing>
      <bandwidth>
        <outbound>1000000000</outbound>
        <inbound>1000000000</inbound>
      </bandwidth>
    </nsf-specification>
    <nsf-access-info>
      <ip>192.0.2.12</ip>
      <port>49152</port>
      <management-protocol>
        NETCONF
      </management-protocol>
    </nsf-access-info>
  </nsf>
</rpc-reply>
```

Figure 10: XML Reply for the Registration of General Firewall in an IPv4 Network and Web Filter

Figure 9 shows the query for NSF(s) that can inspect IPv4 source address, destination address, and URL. Figure 10 shows the reply for the configuration XML for registering a general firewall and a web filter in an IPv4 network [RFC5737] and their capabilities.

The general firewall registered is as follows.

1. The first instance name of the NSF is ipv4_general_firewall.

2. The version used is 1.2.0.

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

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

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

6. The NSF is able to process 14,000,000 packets per second.

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

8. The IPv4 address of the NSF is 192.0.2.11.

9. The port of the NSF is 49152 using the NETCONF protocol.

The web filter registered is as follows.

1. The first instance name of the NSF is ipv4_web_filter.

2. The version used is 1.1.0.

3. The NSF can inspect a URL matched from a user-defined URL. User can specify their own URL.

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

5. The NSF is able to process 14,000,000 packets per second.

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

7. The IPv4 address of the NSF is 192.0.2.12.

8. The port of the NSF is 49152 using the NETCONF protocol.

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <nsf-capability-registration
  xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-registration-interface">
    <query-nsf-capability>
      <condition-capabilities>
        <generic-nsf-capabilities>
          <ipv6-capability>source-address</ipv6-capability>
          <ipv6-capability>destination-address</ipv6-capability>
        </generic-nsf-capabilities>
        <advanced-nsf-capabilities>
          <url-filtering-capability>
            user-defined
          </url-filtering-capability>
        </advanced-nsf-capabilities>
      </condition-capabilities>
      <action-capabilities>
        <ingress-action-capability>drop</ingress-action-capability>
        <egress-action-capability>drop</egress-action-capability>
      </action-capabilities>
    </query-nsf-capability>
  </nsf-capability-registration>
</rpc>
```

Figure 11: XML Examples of an NSF Query with I2NSF Registration Interface Data Model

```
<rpc-reply message-id="101"
  xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <nsf
  xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-registration-interface">
    <nsf-name>ipv4_general_firewall</nsf-name>
    <version>1.2.0</version>
    <condition-capabilities>
      <generic-nsf-capabilities>
        <ipv6-capability>next-header</ipv6-capability>
        <ipv6-capability>source-address</ipv6-capability>
        <ipv6-capability>destination-address</ipv6-capability>
        <tcp-capability>source-port-number</tcp-capability>
        <tcp-capability>destination-port-number</tcp-capability>
      </generic-nsf-capabilities>
    </condition-capabilities>
    <action-capabilities>
      <ingress-action-capability>pass</ingress-action-capability>
      <ingress-action-capability>drop</ingress-action-capability>
      <ingress-action-capability>mirror</ingress-action-capability>
      <egress-action-capability>pass</egress-action-capability>
      <egress-action-capability>drop</egress-action-capability>
      <egress-action-capability>mirror</egress-action-capability>
    </action-capabilities>
    <nsf-specification>
      <packet-processing>14000000</packet-processing>
      <bandwidth>
        <outbound>1000000000</outbound>
        <inbound>1000000000</inbound>
      </bandwidth>
    </nsf-specification>
    <nsf-access-info>
      <ip>2001:db8:0:1::11</ip>
      <port>49153</port>
      <management-protocol>
        NETCONF
      </management-protocol>
    </nsf-access-info>
  </nsf>
  <nsf
  xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-registration-interface">
    <nsf-name>ipv6_web_filter</nsf-name>
    <version>1.1.0</version>
    <condition-capabilities>
      <advanced-nsf-capabilities>
        <url-filtering-capability>
          user-defined
        </url-filtering-capability>
      </advanced-nsf-capabilities>
    </condition-capabilities>
```

```
    <action-capabilities>
      <ingress-action-capability>pass</ingress-action-capability>
      <ingress-action-capability>drop</ingress-action-capability>
      <ingress-action-capability>mirror</ingress-action-capability>
      <egress-action-capability>pass</egress-action-capability>
      <egress-action-capability>drop</egress-action-capability>
      <egress-action-capability>mirror</egress-action-capability>
    </action-capabilities>
    <nsf-specification>
      <packet-processing>14000000</packet-processing>
      <bandwidth>
        <outbound>1000000000</outbound>
        <inbound>1000000000</inbound>
      </bandwidth>
    </nsf-specification>
    <nsf-access-info>
      <ip>2001:db8:0:1::12</ip>
      <port>49153</port>
      <management-protocol>
        NETCONF
      </management-protocol>
    </nsf-access-info>
  </nsf>
</rpc-reply>
```

Figure 12: XML Reply for the Registration of General Firewall in an IPv4 Network and Web Filter

In addition, Figure 11 and Figure 12 shows the query and reply message for the configuration XML for registering a general firewall in an IPv6 network [RFC3849] and webfilter with their capabilities.

1. The instance name of the NSF is ipv6_general_firewall.

2. The version used is 1.2.0.

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

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

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

6. The NSF is able to process 14,000,000 packets per second.

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

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

9. The port of the NSF is 49153 using the NETCONF protocol.

The web filter registered is as follows.

1. The first instance name of the NSF is ipv6_web_filter.

2. The version used is 1.1.0.

3. The NSF can inspect a URL matched from a user-defined URL. User can specify their own URL.

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

5. The NSF is able to process 14,000,000 packets per second.

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

7. The IPv4 address of the NSF is 2001:db8:0:1::12.

8. The port of the NSF is 49153 using the NETCONF protocol.

Appendix C. XML Examples of an NSF Capability Update with I2NSF Registration Interface Data Model

This section shows an XML example of a capability update for an NSF. In this example, the registered General Firewall for the IPv4 network shown in Figure 10 is updated. The DMS can send a notification for capability update with the following XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
 <eventTime>2023-04-14T07:43:52.181088+00:00</eventTime>
 <nsf-capability-update
  xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-registration-interface">
  <nsf>
    <nsf-name>ipv4_general_firewall</nsf-name>
    <version>2.0.0</version>
    <condition-capabilities>
      <generic-nsf-capabilities>
        <ipv6-capability>next-header</ipv6-capability>
        <ipv6-capability>source-address</ipv6-capability>
        <ipv6-capability>destination-address</ipv6-capability>
        <tcp-capability>source-port-number</tcp-capability>
        <tcp-capability>destination-port-number</tcp-capability>
        <udp-capability>source-port-number</udp-capability>
        <udp-capability>destination-port-number</udp-capability>
      </generic-nsf-capabilities>
    </condition-capabilities>
    <action-capabilities>
      <ingress-action-capability>pass</ingress-action-capability>
      <ingress-action-capability>drop</ingress-action-capability>
      <ingress-action-capability>mirror</ingress-action-capability>
      <egress-action-capability>pass</egress-action-capability>
      <egress-action-capability>drop</egress-action-capability>
      <egress-action-capability>mirror</egress-action-capability>
    </action-capabilities>
    <nsf-specification>
      <packet-processing>14000000</packet-processing>
      <bandwidth>
        <outbound>1000000000</outbound>
        <inbound>1000000000</inbound>
      </bandwidth>
    </nsf-specification>
    <nsf-access-info>
      <ip>2001:db8:0:1::11</ip>
      <port>49153</port>
      <management-protocol>
        NETCONF
      </management-protocol>
    </nsf-access-info>
  </nsf>
 </nsf-capability-update>
</notification>
```

Figure 13: XML example of NSF capability update notification

Figure 13 shows the XML of an NSF capability update for the NSF named ipv4_general_firewall. In this example, the NSF has been

updated with a new version (i.e., 2.0.0) and extended capabilities (i.e., inspect the port number(s) for UDP packets).

Appendix D. NSF Lifecycle Management in NFV Environments

Network Functions Virtualization (called 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 DMS having the Ve-Vnfm interface with the VNFM. Refer to Section 8 of [I-D.ietf-i2nsf-applicability] for the detailed description on I2NSF Framework with NFV. Or if an NSF, running as a VNF, is not used by any flows for a time period, Security Controller may request deinstantiating it through the DMS having the Ve-Vnfm interface with the VNFM for efficient resource utilization.

Appendix E. Acknowledgments

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

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

Appendix F. Contributors

The following are co-authors of this document:

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

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

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

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

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

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

Appendix G. Changes from draft-ietf-i2nsf-registration-interface-dm-25

The following changes are made from draft-ietf-i2nsf-registration-interface-dm-25:

*This version replaces Network Management Operator System with Network Operator Management System according to [RFC8329]

Authors' Addresses

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

Email: shyun@mju.ac.kr

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

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

Taekyun Roh
Department of Electronic, Electrical and Computer Engineering

Sungkyunkwan University
2066 Seobu-Ro, Jangan-Gu
Suwon
Gyeonggi-Do
16419
Republic of Korea

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

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

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

Jung-Soo Park
Electronics and Telecommunications Research Institute
218 Gajeong-Ro, Yuseong-Gu
Daejeon
34129
Republic of Korea

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