I2NSF Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 6, 2021

S. Hares, Ed.
Huawei
J. Jeong, Ed.
J. Kim
Sungkyunkwan University
R. Moskowitz
HTT Consulting
Q. Lin
Huawei
November 2, 2020

I2NSF Capability YANG Data Model
draft-ietf-i2nsf-capability-data-model-13

Abstract

This document defines an information model and the corresponding YANG data model for the capabilities of various Network Security Functions (NSFs) in the Interface to Network Security Functions (I2NSF) framework to centrally manage the capabilities of the various NSFs.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

As the industry becomes more sophisticated and network devices (e.g., Internet of Things, Self-driving vehicles, and smartphone using Voice over IP (VoIP) and Voice over LTE (VoLTE)) requires advanced security protection in various scenario, service providers have a lot of problems described in [RFC8192]. To resolve these problems, this document specifies the information and data model of the capabilities of Network Security Functions (NSFs) in a framework of the Interface to Network Security Functions (I2NSF) [RFC8329].

NSFs produced by multiple security vendors provide various security capabilities to customers. Multiple NSFs can be combined together to provide security services over the given network traffic, regardless of whether the NSFs are implemented as physical or virtual functions. Security Capabilities describe the functions that Network Security Functions (NSFs) are available to provide for security policy enforcement purposes. Security Capabilities are independent of the actual security control mechanisms that will implement them.

Every NSF SHOULD be described with the set of capabilities it offers. Security Capabilities enable security functionality to be described in a vendor-neutral manner. That is, it is not needed to refer to a specific product or technology when designing the network; rather, the functions characterized by their capabilities are considered. Security Capabilities are a market enabler, providing a way to define customized security protection by unambiguously describing the security features offered by a given NSF.

This document provides an information model and the corresponding YANG data model [RFC6020][RFC7950] that defines the capabilities of NSFs to centrally manage the capabilities of those security devices. The security devices can register their own capabilities into a Network Operator Management (Mgmt) System (i.e., Security Controller) with this YANG data model through the registration interface [RFC8329]. With the database of the capabilities of those security devices maintained centrally, those security devices can be more easily managed [RFC8329].

This YANG data model uses an "Event-Condition-Action" (ECA) policy model that is used as the basis for the design of I2NSF Policy as described in [RFC8329] and Section 3.1. The "ietf-i2nsf-capability" YANG module defined in this document provides the following features:

- Definition for time capabilities of network security functions.

- Definition for event capabilities of generic network security functions.

- Definition for condition capabilities of generic network security functions.

- Definition for condition capabilities of advanced network security functions.

- Definition for action capabilities of generic network security functions.

- Definition for resolution strategy capabilities of generic network security functions.

- Definition for default action capabilities of generic network security functions.

2. Terminology

This document uses the terminology described in [RFC8329].

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

3. Capability Information Model Design

A Capability Information Model (CapIM) is a formalization of the functionality that an NSF advertises. This enables the precise specification of what an NSF can do in terms of security policy enforcement, so that computer-based tasks can unambiguously refer to, use, configure, and manage NSFs. Capabilities MUST be defined in a vendor- and technology-independent manner (e.g., regardless of the differences among vendors and individual products).

Humans are able to refer to categories of security controls and understand each other. For instance, security experts agree on what is meant by the terms "NAT", "filtering", and "VPN concentrator". As a further example, network security experts unequivocally refer to "packet filters" as stateless devices able to allow or deny packet forwarding based on various conditions (e.g., source and destination IP addresses, source and destination ports, and IP protocol type fields) [Alshaer].

However, more information is required in case of other devices, like stateful firewalls or application layer filters. These devices filter packets or communications, but there are differences in the packets and communications that they can categorize and the states they maintain. Humans deal with these differences by asking more questions to determine the specific category and functionality of the device. Machines can follow a similar approach, which is commonly referred to as question-answering [Hirschman] [Galitsky]. In this context, the CapIM and the derived Data Models provide important and rich information sources.

Analogous considerations can be applied for channel protection protocols, where we all understand that they will protect packets by means of symmetric algorithms whose keys could have been negotiated

with asymmetric cryptography, but they may work at different layers and support different algorithms and protocols. To ensure protection, these protocols apply integrity, optionally confidentiality, anti-reply protections, and authenticate peers.

The CapIM is intended to clarify these ambiguities by providing a formal description of NSF functionality. The set of functions that are advertised MAY be restricted according to the privileges of the user or application that is viewing those functions. I2NSF Capabilities enable unambiguous specification of the security capabilities available in a (virtualized) networking environment, and their automatic processing by means of computer-based techniques.

This includes enabling the security controller to properly identify and manage NSFs, and allow NSFs to properly declare their functionality, so that they can be used in the correct way.

3.1. Design Principles and ECA Policy Model Overview

This document defines an information model for representing NSF capabilities. Some basic design principles for security capabilities and the systems that manage them are:

- Independence: each security capability SHOULD be an independent function, with minimum overlap or dependency on other capabilities. This enables each security capability to be utilized and assembled together freely. More importantly, changes to one capability SHOULD NOT affect other capabilities. This follows the Single Responsibility Principle [Martin] [OODSRP].

- Abstraction: each capability MUST be defined in a vendor-independent manner.

- Advertisement: A dedicated, well-known interface MUST be used to advertise and register the capabilities of each NSF. This same interface MUST be used by other I2NSF Components to determine what Capabilities are currently available to them.

- Execution: a dedicated, well-known interface MUST be used to configure and monitor the use of a capability. This provides a standardized ability to describe its functionality, and report its processing results. This facilitates multi-vendor interoperability.

- Automation: the system MUST have the ability to auto-discover, auto-negotiate, and auto-update its security capabilities (i.e., without human intervention). These features are especially useful for the management of a large number of NSFs. They are essential

for adding smart services (e.g., refinement, analysis, capability reasoning, and optimization) to the security scheme employed. These features are supported by many design patterns, including the Observer Pattern [OODOP], the Mediator Pattern [OODMP], and a set of Message Exchange Patterns [Hohpe].

- Scalability: the management system SHOULD have the capability to scale up/down or scale in/out. Thus, it can meet various performance requirements derived from changeable network traffic or service requests. In addition, security capabilities that are affected by scalability changes SHOULD support reporting statistics to the security controller to assist its decision on whether it needs to invoke scaling or not.

Based on the above principles, this document defines a capability model that enables an NSF to register (and hence advertise) its set of capabilities that other I2NSF Components can use. These capabilities MAY have their access control restricted by policy; this is out of scope for this document. The set of capabilities provided by a given set of NSFs unambiguously define the security offered by the set of NSFs used. The security controller can compare the requirements of users and applications to the set of capabilities that are currently available in order to choose which capabilities of which NSFs are needed to meet those requirements. Note that this choice is independent of vendor, and instead relies specifically on the capabilities (i.e., the description) of the functions provided.

Furthermore, when an unknown threat (e.g., zero-day exploits and unknown malware) is reported by an NSF, new capabilities may be created, and/or existing capabilities may be updated (e.g., by updating its signature and algorithm). This results in enhancing the existing NSFs (and/or creating new NSFs) to address the new threats. New capabilities may be sent to and stored in a centralized repository, or stored separately in a vendor's local repository. In either case, a standard interface facilitates the update process. This document specifies a metadata model that MAY be used to further describe and/or prescribe the characteristics and behavior of the I2NSF capability model. For example, in this case, metadata could be used to describe the updating of the capability, and prescribe the particular version that an implementation should use. This initial version of the model covers and has been validated to describe NSFs that are designed with a set of capabilities (which covers most of the existing NSFs). Checking the behavior of the model with systems that change capabilities dynamically at runtime has been extensively explored (e.g., impact on automatic registration).

The "Event-Condition-Action" (ECA) policy model in [RFC8329] is used as the basis for the design of the capability model; definitions of

all I2NSF policy-related terms are also defined in [I-D.ietf-i2nsf-terminology]. The following three terms define the structure and behavior of an I2NSF imperative policy rule:

- Event: An Event is defined as any important occurrence in time of a change in the system being managed, and/or in the environment of the system being managed. When used in the context of I2NSF Policy Rules, it is used to determine whether the Condition clause of the I2NSF Policy Rule can be evaluated or not. Examples of an I2NSF Event include time and user actions (e.g., logon, logoff, and actions that violate an ACL).

- Condition: A condition is defined as a set of attributes, features, and/or values that are to be compared with a set of known attributes, features, and/or values in order to determine whether or not the set of Actions in that (imperative) I2NSF Policy Rule can be executed or not. Examples of I2NSF Conditions include matching attributes of a packet or flow, and comparing the internal state of an NSF to a desired state.

- Action: An action is used to control and monitor aspects of flow-based NSFs when the event and condition clauses are satisfied. NSFs provide security functions by executing various Actions. Examples of I2NSF Actions include providing intrusion detection and/or protection, web and flow filtering, and deep packet inspection for packets and flows.

An I2NSF Policy Rule is made up of three Boolean clauses: an Event clause, a Condition clause, and an Action clause. This structure is also called an ECA (Event-Condition-Action) Policy Rule. A Boolean clause is a logical statement that evaluates to either TRUE or FALSE. It may be made up of one or more terms; if more than one term is present, then each term in the Boolean clause is combined using logical connectives (i.e., AND, OR, and NOT).

An I2NSF ECA Policy Rule has the following semantics:

```
IF <event-clause> is TRUE

   IF <condition-clause> is TRUE

      THEN execute <action-clause> [constrained by metadata]

   END-IF

END-IF
```

Technically, the "Policy Rule" is really a container that aggregates the above three clauses, as well as metadata. Aggregating metadata enables business logic to be used to prescribe behavior. For example, suppose a particular ECA Policy Rule contains three actions (A1, A2, and A3, in that order). Action A2 has a priority of 10; actions A1 and A3 have no priority specified. Then, metadata may be used to restrict the set of actions that can be executed when the event and condition clauses of this ECA Policy Rule are evaluated to be TRUE; two examples are: (1) only the first action (A1) is executed, and then the policy rule returns to its caller, or (2) all actions are executed, starting with the highest priority.

The above ECA policy model is very general and easily extensible.

3.2. Matched Policy Rule

The concept of a "matched" policy rule is defined as one in which its event and condition clauses both evaluate to true. To precisely describe what an NSF can do in terms of security, the things need to describe are the events it can catch, the conditions it can evaluate, and the actions it can enforce.

Therefore, the properties that to characterize the capabilities of a NSF are as below:

- Ac is the set of Actions currently available from the NSF;
- Ec is the set of Events that an NSF can catch. Note that for NSF (e.g., a packet filter) that are not able to react to events, this set will be empty;
- Cc is the set of Conditions currently available from the NSF;
- EVc defines the set of Condition Clause Evaluation Rules that can be used at the NSF to decide when the Condition Clause is true given the result of the evaluation of the individual Conditions.

3.3. Conflict, Resolution Strategy and Default Action

Formally, two I2NSF Policy Rules conflict with each other if:

- the Event Clauses of each evaluate to TRUE;
- the Condition Clauses of each evaluate to TRUE;
- the Action Clauses affect the same object in different ways.

For example, if we have two Policy Rules in the same Policy:

R1: During 8am-6pm, if traffic is external, then run through FW

R2: During 7am-8pm, conduct anti-malware investigation

There is no conflict between R1 and R2, since the actions are different. However, consider these two rules:

R3: During 8am-6pm, John gets GoldService

R4: During 10am-4pm, FTP from all users gets BronzeService

R3 and R4 are now in conflict, between the hours of 10am and 4pm, because the actions of R3 and R4 are different and apply to the same user (i.e., John).

Conflicts theoretically compromise the correct functioning of devices (as happened for routers several year ago). However, NSFs have been designed to cope with these issues. Since conflicts are originated by simultaneously matching rules, an additional process decides the action to be applied, e.g., among the ones the matching rule would have enforced. This process is described by means of a resolution strategy

On the other hand, it may happen that, if an event is caught, none of the policy rules matches. As a simple case, no rules may match a packet arriving at border firewall. In this case, the packet is usually dropped, that is, the firewall has a default behavior to manage cases that are not covered by specific rules.

Therefore, we introduce another security capability that serves to characterize valid policies for an NSF that solve conflicts with resolution strategies and enforce default actions if no rules match:

- RSc is the set of Resolution Strategy that can be used to specify how to resolve conflicts that occur between the actions of the same or different policy rules that are matched and contained in this particular NSF;

- Dc defines the notion of a Default action. This action can be either an explicit action that has been chosen {a}, or a set of actions {F}, where F is a dummy symbol (i.e., a placeholder value) that can be used to indicate that the default action can be freely selected by the policy editor. This is denoted as {F} U {a}.

4. Overview of YANG Data Model

This section provides as overview of how the YANG data model can be used in the I2NSF framework described in [RFC8329]. Figure 1 shows the capabilities (e.g., firewall and web filter) of NSFs in the I2NSF Framework. As shown in this figure, an NSF Developer's Management System can register NSFs and the capabilities that the network security devices can support. To register NSFs in this way, the Developer's Management System utilizes this standardized capability YANG data model through the I2NSF Registration Interface [RFC8329]. That is, this Registration Interface uses the YANG module described in this document to describe the capabilities of a network security function that is registered with the Security Controller. With the capabilities of those network security devices maintained centrally, those security devices can be more easily managed, which can resolve many of the problems described in [RFC8192].

In Figure 1, a new NSF at a Developer's Management Systems has capabilities of Firewall (FW) and Web Filter (WF), which are denoted as (Cap = {FW, WF}), to support Event-Condition-Action (ECA) policy rules where 'E', 'C', and 'A' mean "Event", "Condition", and "Action", respectively. The condition involves IPv4 or IPv6 datagrams, and the action includes "Allow" and "Deny" for those datagrams.

Note that the NSF-Facing Interface [RFC8329] is used to configure the security policy rules of the generic network security functions, and the configuration of advanced security functions over the NSF-Facing Interface is used to configure the security policy rules of advanced network security functions (e.g., anti-virus and Distributed-Denial-of-Service (DDoS) attack mitigator), respectively, according to the capabilities of NSFs registered with the I2NSF Framework.

```
       +------------------------------------------------------+
       |  I2NSF User (e.g., Overlay Network Mgmt, Enterprise  |
       |  Network Mgmt, another network domain's mgmt, etc.)  |
       +--------------------+---------------------------------+
           I2NSF            ^
  Consumer-Facing Interface |
                            |
                            v                  I2NSF
          +-----------------+------------+  Registration  +-------------+
          | Network Operator Mgmt System |   Interface    | Developer's |
          | (i.e., Security Controller)  |<-------------->| Mgmt System |
          +-----------------+------------+                +-------------+
                            ^                                 New NSF
                            |                           Cap = {FW, WF}
              I2NSF         |                           E = {}
       NSF-Facing Interface |                           C = {IPv4, IPv6}
                            |                           A = {Allow, Deny}
                            v
       +---------------+----+------------+-----------------+
       |               |                 |                 |
   +---+---+       +---+---+         +---+---+         +---+---+
   | NSF-1 |  ...  | NSF-m |         | NSF-1 |   ...   | NSF-n |
   +-------+       +-------+         +-------+         +-------+
     NSF-1           NSF-m             NSF-1             NSF-n
 Cap = {FW, WF}    Cap = {FW, WF}    Cap = {FW, WF}    Cap = {FW, WF}
 E = {}            E = {user}        E = {dev}         E = {time}
 C = {IPv4}        C = {IPv6}        C = {IPv4, IPv6}  C = {IPv4}
 A = {Allow, Deny} A = {Allow, Deny} A = {Allow, Deny} A = {Allow, Deny}

   Developer's Mgmt System A           Developer's Mgmt System B
```

Figure 1: Capabilities of NSFs in I2NSF Framework

A use case of an NSF with the capabilities of firewall and web filter is described as follows.

- If a network manager wants to apply security policy rules to block malicious users with firewall and web filter, it is a tremendous burden for a network administrator to apply all of the needed rules to NSFs one by one. This problem can be resolved by managing the capabilities of NSFs in this document.

- If a network administrator wants to block malicious users for IPv4 or IPv6 traffic, he sends a security policy rule to block the users to the Network Operator Management System using the I2NSF Consumer-Facing Interface.

- When the Network Operator Management System receives the security policy rule, it automatically sends that security policy rules to appropriate NSFs (i.e., NSF-m in Developer's Management System A and NSF-1 in Developer's Management System B) which can support the capabilities (i.e., IPv6). This lets an I2NSF User not consider NSFs where the rule is applied.

- If NSFs encounter the suspicious IPv4 or IPv6 packets of malicious users, they can filter the packets out according to the configured security policy rule. Therefore, the security policy rule against the malicious users' packets can be automatically applied to appropriate NSFs without human intervention.

5. YANG Tree Diagram

This section shows a YANG tree diagram of capabilities of network security functions, as defined in the Section 3.

5.1. Network Security Function (NSF) Capabilities

This section explains a YANG tree diagram of NSF capabilities and its features. Figure 2 shows a YANG tree diagram of NSF capabilities. The NSF capabilities in the tree include time capabilities, event capabilities, condition capabilities, action capabilities, resolution strategy capabilities, and default action capabilities. Those capabilities can be tailored or extended according to a vendor's specific requirements. Refer to the NSF capabilities information model for detailed discussion Section 3.

```
module: ietf-i2nsf-capability
  +--rw nsf* [nsf-name]
     +--rw nsf-name            string
     +--rw time-capabilities*                enumeration
     +--rw event-capabilities
     |  +--rw system-event-capability*   identityref
     |  +--rw system-alarm-capability*   identityref
     +--rw condition-capabilities
     |  +--rw generic-nsf-capabilities
     |  |  +--rw ipv4-capability*   identityref
     |  |  +--rw icmp-capability*   identityref
     |  |  +--rw ipv6-capability*   identityref
     |  |  +--rw icmpv6-capability*   identityref
     |  |  +--rw tcp-capability*    identityref
     |  |  +--rw udp-capability*    identityref
     |  |  +--rw sctp-capability*    identityref
     |  +--rw advanced-nsf-capabilities
     |  |  +--rw anti-virus-capability*    identityref
     |  |  +--rw anti-ddos-capability*     identityref
     |  |  +--rw ips-capability*          identityref
     |  |  +--rw url-capability*          identityref
     |  |  +--rw voip-volte-capability*   identityref
     |  +--rw context-capabilities*        identityref
     +--rw action-capabilities
     |  +--rw ingress-action-capability*   identityref
     |  +--rw egress-action-capability*    identityref
     |  +--rw log-action-capability*       identityref
     +--rw resolution-strategy-capabilities*   identityref
     +--rw default-action-capabilities*        identityref
     +--rw ipsec-method*                       identityref
```

Figure 2: YANG Tree Diagram of Capabilities of Network Security Functions

Time capabilities are used to specify the capabilities which describe when to execute the I2NSF policy rule. The time capabilities are defined in terms of absolute time and periodic time. The absolute time means the exact time to start or end. The periodic time means repeated time like day, week, or month..

Event capabilities are used to specify the capabilities that describe the event that would trigger the evaluation of the condition clause of the I2NSF Policy Rule. The defined event capabilities are system event and system alarm.

Condition capabilities are used to specify capabilities of a set of attributes, features, and/or values that are to be compared with a

set of known attributes, features, and/or values in order to determine whether or not the set of actions in that (imperative) I2NSF policy rule can be executed. The condition capabilities are classified in terms of generic network security functions and advanced network security functions. The condition capabilities of generic network security functions are defined as IPv4 capability, IPv6 capability, TCP capability, UDP capability, SCTP capability and ICMP capability. The condition capabilities of advanced network security functions are defined as anti-virus capability, anti-DDoS capability, Intrusion Prevention System (IPS) capability, HTTP capability, and VoIP/VoLTE capability. See Section 3.1 for more information about the condition in the ECA policy model.

Action capabilities are used to specify the capabilities that describe the control and monitoring aspects of flow-based NSFs when the event and condition clauses are satisfied. The action capabilities are defined as ingress-action capability, egress-action capability, and log-action capability. See Section 3.1 for more information about the action in the ECA policy model. Also, see Section 7.2 (NSF-Facing Flow Security Policy Structure) in [RFC8329] for more information about the ingress and egress actions. In addition, see Section 9.1 (Flow-Based NSF Capability Characterization) in [RFC8329] for more information about logging at NSFs.

Resolution strategy capabilities are used to specify the capabilities that describe conflicts that occur between the actions of the same or different policy rules that are matched and contained in this particular NSF. The resolution strategy capabilities are defined as First Matching Rule (FMR), Last Matching Rule (LMR), Prioritized Matching Rule (PMR), Prioritized Matching Rule with Errors (PMRE), and Prioritized Matching Rule with No Errors (PMRN). See Section 3.3 for more information about the resolution strategy.

Default action capabilities are used to specify the capabilities that describe how to execute I2NSF policy rules when no rule matches a packet. The default action capabilities are defined as pass, drop, alert, and mirror. See Section 3.3 for more information about the default action.

IPsec method capabilities are used to specify capabilities of how to support an Internet Key Exchange (IKE) [RFC7296] for the security communication. The default action capabilities are defined as IKE or IKE-less. See [I-D.ietf-i2nsf-sdn-ipsec-flow-protection] for more information about the SDN-based IPsec flow protection in I2NSF.

6. YANG Data Model of I2NSF NSF Capability

This section introduces a YANG module for NSFs' capabilities, as defined in the Section 3.

This YANG module imports from [RFC6991]. It makes references to [RFC0768][IANA-Protocol-Numbers][RFC0791][RFC0792][RFC0793][RFC3261][RFC4443][RFC4960][RFC8200][RFC8329][I-D.ietf-i2nsf-nsf-monitoring-data-model][I-D.ietf-i2nsf-sdn-ipsec-flow-protection].

```
<CODE BEGINS> file "ietf-i2nsf-capability@2020-11-02.yang"

module ietf-i2nsf-capability {
  yang-version 1.1;
  namespace
    "urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability";
  prefix
    nsfcap;


  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>

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

     Editor: Jinyong Tim Kim
     <mailto:timkim@skku.edu>

     Editor: Susan Hares
     <mailto:shares@ndzh.com>";

  description
    "This module is a YANG module for I2NSF Network Security
     Functions (NSFs)'s Capabilities.

     Copyright (c) 2020 IETF Trust and the persons identified as
     authors of the code. All rights reserved.

     Redistribution and use in source and binary forms, with or
     without modification, is permitted pursuant to, and subject
     to the license terms contained in, the Simplified BSD License
     set forth in Section 4.c of the IETF Trust's Legal Provisions
```

```
      Relating to IETF Documents
      http://trustee.ietf.org/license-info).

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

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

   revision "2020-11-02"{
     description "Initial revision.";
     reference
       "RFC XXXX: I2NSF Capability YANG Data Model";

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

   /*
    * Identities
    */

   identity event {
     description
       "Base identity for I2NSF events.";
     reference
       "draft-ietf-i2nsf-nsf-monitoring-data-model-04: I2NSF NSF
        Monitoring YANG Data Model - Event";

     // RFC Ed.: replace the above draft with an actual RFC in the
     // YANG module and remove this note.
   }

   identity system-event-capability {
     base event;
     description
       "Identity for system event";
     reference
       "draft-ietf-i2nsf-nsf-monitoring-data-model-04: I2NSF NSF
        Monitoring YANG Data Model - System event";
   }

   identity system-alarm-capability {
     base event;
     description
       "Identity for system alarm";
     reference
       "draft-ietf-i2nsf-nsf-monitoring-data-model-04: I2NSF NSF
```

```
        Monitoring YANG Data Model - System alarm";
   }

   identity access-violation {
     base system-event-capability;
     description
       "Identity for access violation event";
     reference
       "draft-ietf-i2nsf-nsf-monitoring-data-model-04: I2NSF NSF
        Monitoring YANG Data Model - System event for access
        violation";
   }

   identity configuration-change {
     base system-event-capability;
     description
       "Identity for configuration change event";
     reference
       "draft-ietf-i2nsf-nsf-monitoring-data-model-04: I2NSF NSF
        Monitoring YANG Data Model - System event for configuration
        change";
   }

   identity memory-alarm {
     base system-alarm-capability;
     description
       "Identity for memory alarm. Alarm when memory usage
       exceed the threshold.";
     reference
       "draft-ietf-i2nsf-nsf-monitoring-data-model-04: I2NSF NSF
        Monitoring YANG Data Model - System alarm for memory";
   }

   identity cpu-alarm {
     base system-alarm-capability;
     description
       "Identity for CPU alarm. Alarm when CPU usage
       exceed the threshold.";
     reference
       "draft-ietf-i2nsf-nsf-monitoring-data-model-04: I2NSF NSF
        Monitoring YANG Data Model - System alarm for CPU";
   }

   identity disk-alarm {
     base system-alarm-capability;
     description
       "Identity for disk alarm. Alarm when disk usage
       exceed the threshold.";
```

```
     reference
       "draft-ietf-i2nsf-nsf-monitoring-data-model-04: I2NSF NSF
        Monitoring YANG Data Model - System alarm for disk";
   }

   identity hardware-alarm {
     base system-alarm-capability;
     description
       "Identity for hardware alarm. Alarm when a hardware failure
       occur.";
     reference
       "draft-ietf-i2nsf-nsf-monitoring-data-model-04: I2NSF NSF
        Monitoring YANG Data Model - System alarm for hardware";
   }

   identity interface-alarm {
     base system-alarm-capability;
     description
       "Identity for interface alarm";
     reference
       "draft-ietf-i2nsf-nsf-monitoring-data-model-04: I2NSF NSF
        Monitoring YANG Data Model - System alarm for interface";
   }

   identity condition {
     description
       "Base identity for I2NSF conditions";
   }

   identity context-capability {
     base condition;
     description
       "Base identity for context condition capabilities for an NSF.";
   }

   identity access-control-list {
     base context-capability;
     description
       "Identity for Access Control List (ACL) condition capability";
     reference
       "RFC 8519: YANG Data Model for Network Access Control Lists
        (ACLs) - A user-ordered set of rules used to configure the
        forwarding behavior in an NSF.";
   }

   identity application-layer-filter {
     base context-capability;
     description
```

```
        "Identity for application-layer-filter condition capability";
    }

    identity target {
      base context-capability;
      description
        "Identity for target condition capability";
      reference
        "RFC 8519: YANG Data Model for Network Access Control Lists
         (ACLs) - An access control for a target (e.g., the
         corresponding IP address) in an NSF.";
    }

    identity user {
      base context-capability;
      description
        "Identity for user condition capability";
      reference
        "RFC 8519: YANG Data Model for Network Access Control Lists
         (ACLs) - An access control for a user (e.g., the
         corresponding IP address) in an NSF.";
    }

    identity group {
      base context-capability;
      description
        "Identity for group condition capability";
      reference
        "RFC 8519: YANG Data Model for Network Access Control Lists
         (ACLs) - An access control for a group (e.g., the
         corresponding IP addresses) in an NSF.";
    }

    identity geography {
      base context-capability;
      description
        "Identity for geography condition capability";
      reference
        "draft-google-self-published-geofeeds-02: Self-published
         IP Geolocation Data - An access control for a geographical
         location i.e., geolocation (e.g., the corresponding IP
         address).";
    }

    identity ipv4-capability {
      base condition;
      description
        "Base identity for IPv4 condition capability";
```

```
     reference
       "RFC 791: Internet Protocol";
   }

   identity exact-ipv4-header-length {
     base ipv4-capability;
     description
       "Identity for exact-match IPv4 header-length
       condition capability";
     reference
       "RFC 791: Internet Protocol - Header Length";
   }

   identity range-ipv4-header-length {
     base ipv4-capability;
     description
       "Identity for range-match IPv4 header-length
       condition capability";
     reference
       "RFC 791: Internet Protocol - Header Length";
   }

   identity ipv4-tos {
     base ipv4-capability;
     description
       "Identity for IPv4 Type-Of-Service (TOS)
       condition capability";
     reference
       "RFC 791: Internet Protocol - Type of Service";
   }

   identity exact-ipv4-total-length {
     base ipv4-capability;
     description
       "Identity for exact-match IPv4 total length
       condition capability";
     reference
       "RFC 791: Internet Protocol - Total Length";
   }

   identity range-ipv4-total-length {
     base ipv4-capability;
     description
       "Identity for range-match IPv4 total length
       condition capability";
     reference
       "RFC 791: Internet Protocol - Total Length";
   }
```

```
   identity ipv4-id {
     base ipv4-capability;
     description
       "Identity for IPv4 identification condition capability.
        IPv4 ID Field is used for fragmentation";
     reference
       "RFC 791: Internet Protocol - Identification
        RFC 6864: Updated Specification of the IPv4 ID Field";
   }

   identity ipv4-fragment-flags {
     base ipv4-capability;
     description
       "Identity for IPv4 fragment flags condition capability";
     reference
       "RFC 791: Internet Protocol - Fragmentation Flags";
   }

   identity exact-ipv4-fragment-offset {
     base ipv4-capability;
     description
       "Identity for exact-match IPv4 fragment offset
       condition capability";
     reference
       "RFC 791: Internet Protocol - Fragmentation Offset";
   }

   identity range-ipv4-fragment-offset {
     base ipv4-capability;
     description
       "Identity for range-match IPv4 fragment offset
       condition capability";
     reference
       "RFC 791: Internet Protocol - Fragmentation Offset";
   }

   identity exact-ipv4-ttl {
     base ipv4-capability;
     description
       "Identity for exact-match IPv4 Time-To-Live (TTL)
       condition capability";
     reference
       "RFC 791: Internet Protocol - Time To Live (TTL)";
   }

   identity range-ipv4-ttl {
     base ipv4-capability;
     description
```

```
        "Identity for range-match IPv4 Time-To-Live (TTL)
        condition capability";
      reference
        "RFC 791: Internet Protocol - Time To Live (TTL)";
    }

    identity ipv4-protocol {
      base ipv4-capability;
      description
        "Identity for IPv4 protocol condition capability";
      reference
        "IANA Website: Assigned Internet Protocol Numbers
         - Protocol Number for IPv4
         RFC 791: Internet Protocol - Protocol";
    }

    identity exact-ipv4-address {
      base ipv4-capability;
      description
        "Identity for exact-match IPv4 address
        condition capability";
      reference
        "RFC 791: Internet Protocol - Address";
    }

    identity range-ipv4-address {
      base ipv4-capability;
      description
        "Identity for range-match IPv4 address condition
         capability";
      reference
        "RFC 791: Internet Protocol - Address";
    }

    identity ipv4-ip-opts {
      base ipv4-capability;
      description
        "Identity for IPv4 option condition capability";
      reference
        "RFC 791: Internet Protocol - Options";
    }

    identity ipv4-geo-ip {
      base ipv4-capability;
      description
        "Identity for geography condition capability";
    }
```

```
   identity ipv6-capability {
     base condition;
     description
       "Base identity for IPv6 condition capabilities";
     reference
       "RFC 8200: Internet Protocol, Version 6 (IPv6)
       Specification";
   }

   identity ipv6-traffic-class {
     base ipv6-capability;
     description
       "Identity for IPv6 traffic class
       condition capability";
     reference
       "RFC 8200: Internet Protocol, Version 6 (IPv6)
       Specification - Traffic Class";
   }

   identity exact-ipv6-flow-label {
     base ipv6-capability;
     description
       "Identity for exact-match IPv6 flow label
       condition capability";
     reference
       "RFC 8200: Internet Protocol, Version 6 (IPv6)
       Specification - Flow Label
       RFC 6437: IPv6 Flow Label Specification";
   }

   identity range-ipv6-flow-label {
     base ipv6-capability;
     description
       "Identity for range-match IPv6 flow label
       condition capability";
     reference
       "RFC 8200: Internet Protocol, Version 6 (IPv6)
       Specification - Flow Label
       RFC 6437: IPv6 Flow Label Specification";
   }

   identity exact-ipv6-payload-length {
     base ipv6-capability;
     description
       "Identity for exact-match IPv6 payload length
       condition capability";
     reference
       "RFC 8200: Internet Protocol, Version 6 (IPv6)
```

```
         Specification - Payload Length";
     }

     identity range-ipv6-payload-length {
       base ipv6-capability;
       description
         "Identity for range-match IPv6 payload length
         condition capability";
       reference
         "RFC 8200: Internet Protocol, Version 6 (IPv6)
         Specification - Payload Length";
     }

     identity ipv6-next-header {
       base ipv6-capability;
       description
         "Identity for IPv6 next header condition capability";
       reference
         "IANA Website: Assigned Internet Protocol Numbers
          - Protocol Number for IPv6
          RFC 8200: Internet Protocol, Version 6 (IPv6)
          Specification - Next Header";
     }

     identity exact-ipv6-hop-limit {
       base ipv6-capability;
       description
         "Identity for exact-match IPv6 hop limit condition
         capability";
       reference
         "RFC 8200: Internet Protocol, Version 6 (IPv6)
         Specification - Hop Limit";
     }

     identity range-ipv6-hop-limit {
       base ipv6-capability;
       description
         "Identity for range-match IPv6 hop limit condition
         capability";
       reference
         "RFC 8200: Internet Protocol, Version 6 (IPv6)
         Specification - Hop Limit";
     }

     identity exact-ipv6-address {
       base ipv6-capability;
       description
         "Identity for exact-match IPv6 address condition
```

```
          capability";
       reference
         "RFC 8200: Internet Protocol, Version 6 (IPv6)
         Specification - Address";
     }

     identity range-ipv6-address {
       base ipv6-capability;
       description
         "Identity for range-match IPv6 address condition
         capability";
       reference
         "RFC 8200: Internet Protocol, Version 6 (IPv6)
         Specification - Address";
     }

     identity ipv6-header-order {
       base ipv6-capability;
       description
         "Identity for header order IPv6 address condition
         capability";
       reference
         "RFC 8200: Internet Protocol, Version 6 (IPv6)
         Specification - Extension Header Order";
     }

     identity ipv6-options {
       base ipv6-capability;
       description
         "Identity for options IPv6 address condition
         capability";
       reference
         "RFC 8200: Internet Protocol, Version 6 (IPv6)
         Specification - Options";
     }

     identity ipv6-hop-by-hop {
       base ipv6-capability;
       description
         "Identity for hop by hop IPv6 address condition
         capability";
       reference
         "RFC 8200: Internet Protocol, Version 6 (IPv6)
         Specification - Options";
     }

     identity ipv6-routing-header {
       base ipv6-capability;
```

```
     description
       "Identity for routing header IPv6 address condition
       capability";
     reference
       "RFC 8200: Internet Protocol, Version 6 (IPv6)
       Specification - Routing Header";
   }

   identity ipv6-fragment-header {
     base ipv6-capability;
     description
       "Identity for fragment header IPv6 address condition
       capability";
     reference
       "RFC 8200: Internet Protocol, Version 6 (IPv6)
       Specification - Fragment Header";
   }

   identity ipv6-destination-options {
     base ipv6-capability;
     description
       "Identity for destination options IPv6 address condition
       capability";
     reference
       "RFC 8200: Internet Protocol, Version 6 (IPv6)
       Specification - Destination Options";
   }

   identity tcp-capability {
     base condition;
     description
       "Base identity for TCP condition capabilities";
     reference
       "RFC 793: Transmission Control Protocol";
   }

   identity exact-tcp-port-num {
     base tcp-capability;
     description
       "Identity for exact-match TCP port number condition
        capability";
     reference
       "RFC 793: Transmission Control Protocol - Port Number";
   }

   identity range-tcp-port-num {
     base tcp-capability;
     description
```

```
        "Identity for range-match TCP port number condition
         capability";
      reference
        "RFC 793: Transmission Control Protocol - Port Number";
    }

    identity exact-tcp-window-size {
      base tcp-capability;
      description
        "Identity for exact-match TCP window size condition capability";
      reference
        "RFC 793: Transmission Control Protocol - Window Size";
    }

    identity range-tcp-window-size {
      base tcp-capability;
      description
        "Identity for range-match TCP window size condition capability";
      reference
        "RFC 793: Transmission Control Protocol - Window Size";
    }

    identity tcp-flags {
      base tcp-capability;
      description
        "Identity for TCP flags condition capability";
      reference
        "RFC 793: Transmission Control Protocol - Flags";
    }

    identity udp-capability {
      base condition;
      description
        "Base identity for UDP condition capabilities";
      reference
        "RFC 768: User Datagram Protocol";
    }

    identity exact-udp-port-num {
      base udp-capability;
      description
        "Identity for exact-match UDP port number condition capability";
      reference
        "RFC 768: User Datagram Protocol - Port Number";
    }

    identity range-udp-port-num {
      base udp-capability;
```

```
     description
       "Identity for range-match UDP port number condition capability";
     reference
       "RFC 768: User Datagram Protocol - Port Number";
   }

   identity exact-udp-total-length {
     base udp-capability;
     description
       "Identity for exact-match UDP total-length condition capability";
     reference
       "RFC 768: User Datagram Protocol - Total Length";
   }

   identity range-udp-total-length {
     base udp-capability;
     description
       "Identity for range-match UDP total-length condition capability";
     reference
       "RFC 768: User Datagram Protocol - Total Length";
   }

   identity sctp-capability {
    description
       "Identity for SCTP condition capabilities";
     reference
       "RFC 4960: Stream Control Transmission Protocol";
   }

   identity exact-sctp-port-num {
     base sctp-capability;
     description
       "Identity for exact-match SCTP port number condition
        capability";
     reference
       "RFC 4960: Stream Control Transmission Protocol - Port Number";
   }

   identity range-sctp-port-num {
     base sctp-capability;
     description
       "Identity for range-match SCTP port number condition
        capability";
     reference
       "RFC 4960: Stream Control Transmission Protocol - Port Number";
   }

   identity sctp-chunk-type {
```

```
       base sctp-capability;
       description
         "Identity for SCTP chunk type condition capability";
       reference
         "RFC 4960: Stream Control Transmission Protocol - Chunk Type";
     }

     identity icmp-capability {
       base condition;
       description
         "Base identity for ICMP condition capability";
       reference
         "RFC 792: Internet Control Message Protocol";
     }

     identity icmp-type {
       base icmp-capability;
       description
         "Identity for ICMP type condition capability";
       reference
         "RFC 792: Internet Control Message Protocol";
     }

     identity icmp-code {
       base icmp-capability;
       description
         "Identity for ICMP code condition capability";
       reference
         "RFC 792: Internet Control Message Protocol";
     }

     identity icmpv6-capability {
       base condition;
       description
         "Base identity for ICMPv6 condition capability";
       reference
         "RFC 4443: Internet Control Message Protocol (ICMPv6)
          for the Internet Protocol Version 6 (IPv6) Specification
          - ICMPv6";
     }

     identity icmpv6-type {
       base icmpv6-capability;
       description
         "Identity for ICMPv6 type condition capability";
       reference
         "RFC 4443: Internet Control Message Protocol (ICMPv6)
          for the Internet Protocol Version 6 (IPv6) Specification
```

```
        - ICMPv6";
   }

   identity icmpv6-code {
     base icmpv6-capability;
     description
       "Identity for ICMPv6 code condition capability";
     reference
       "RFC 4443: Internet Control Message Protocol (ICMPv6)
        for the Internet Protocol Version 6 (IPv6) Specification
        - ICMPv6";
   }

   identity url-capability {
     base condition;
     description
       "Base identity for URL condition capability";
   }

   identity pre-defined {
     base url-capability;
     description
       "Identity for pre-defined URL Database condition capability.
        The NSF capable of using a predefined public URL Database.";
   }

   identity user-defined {
     base url-capability;
     description
       "Identity for user-defined URL Database condition capability.
        The NSF capable of using a URL Database that can be added
        manually by a user.";
   }

   identity log-action-capability {
     description
       "Base identity for log-action capability";
   }

   identity rule-log {
     base log-action-capability;
     description
       "Identity for rule log log-action capability.
        Log the received packet based on the rule";
   }

   identity session-log {
     base log-action-capability;
```

```
     description
       "Identity for session log log-action capability.
        Log the received packet based on the session.";
   }

   identity ingress-action-capability {
     description
       "Base identity for ingress-action capability";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Ingress action";
   }

   identity egress-action-capability {
     description
       "Base identity for egress-action capability";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Egress action";
   }

   identity default-action-capability {
     description
       "Base identity for default-action capability";
   }

   identity pass {
     base ingress-action-capability;
     base egress-action-capability;
     base default-action-capability;
     description
       "Identity for pass action capability";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Ingress, egress, and pass actions.";
   }

   identity drop {
     base ingress-action-capability;
     base egress-action-capability;
     base default-action-capability;
     description
       "Identity for drop action capability";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Ingress, egress, and drop actions.";
   }
```

```
   identity alert {
     base ingress-action-capability;
     base egress-action-capability;
     base default-action-capability;
     description
       "Identity for alert action capability";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Ingress, egress, and alert actions.
        draft-ietf-i2nsf-nsf-monitoring-data-model-04: I2NSF
        NSF Monitoring YANG Data Model - Alarm (i.e., alert).";
   }

   identity mirror {
     base ingress-action-capability;
     base egress-action-capability;
     base default-action-capability;
     description
       "Identity for mirror action capability";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Ingress, egress, and mirror actions.";
   }

   identity invoke-signaling {
     base egress-action-capability;
     description
       "Identity for invoke signaling action capability";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Invoke-signaling action";
   }

   identity forwarding {
     base egress-action-capability;
     description
       "Identity for forwarding action capability";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Forwarding action";
   }

   identity redirection {
     base egress-action-capability;
     description
       "Identity for redirection action capability";
     reference
       "RFC 8329: Framework for Interface to Network Security
```

```
          Functions - Redirection action";
   }

   identity resolution-strategy-capability {
     description
       "Base identity for resolution strategy capability";
   }

   identity fmr {
     base resolution-strategy-capability;
     description
       "Identity for First Matching Rule (FMR) resolution
        strategy capability";
   }

   identity lmr {
     base resolution-strategy-capability;
     description
       "Identity for Last Matching Rule (LMR) resolution
        strategy capability";
   }

   identity pmr {
     base resolution-strategy-capability;
     description
       "Identity for Prioritized Matching Rule (PMR) resolution
        strategy capability";
   }

   identity pmre {
     base resolution-strategy-capability;
     description
       "Identity for Prioritized Matching Rule with Errors (PMRE)
        resolution strategy capability";
   }

   identity pmrn {
     base resolution-strategy-capability;
     description
       "Identity for Prioritized Matching Rule with No Errors (PMRN)
        resolution strategy capability";
   }

   identity advanced-nsf-capability {
     description
       "Base identity for advanced Network Security Function (NSF)
        capability.  This can be used for advanced NSFs such as
        Anti-Virus, Anti-DDoS Attack, IPS, and VoIP/VoLTE Security
```

```
        Service.";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Advanced NSF capability";
   }

   identity anti-virus-capability {
     base advanced-nsf-capability;
     description
       "Identity for advanced NSF Anti-Virus capability.
        This can be used for an extension point for Anti-Virus
        as an advanced NSF.";
        reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Advanced NSF Anti-Virus capability";
   }

   identity anti-ddos-capability {
     base advanced-nsf-capability;
     description
       "Identity for advanced NSF Anti-DDoS Attack capability.
        This can be used for an extension point for Anti-DDoS
        Attack as an advanced NSF.";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Advanced NSF Anti-DDoS Attack capability";
   }

   identity ips-capability {
     base advanced-nsf-capability;
     description
       "Identity for advanced NSF IPS capabilities.  This can be
        used for an extension point for IPS as an advanced NSF.";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Advanced NSF IPS capability";
   }

   identity voip-volte-capability {
     base advanced-nsf-capability;
     description
       "Identity for advanced NSF VoIP/VoLTE Security Service
        capability.  This can be used for an extension point
        for VoIP/VoLTE Security Service as an advanced NSF.";
     reference
       "RFC 3261: SIP: Session Initiation Protocol";
   }
```

```
   identity detect {
     base anti-virus-capability;
     description
       "Identity for advanced NSF Anti-Virus Detection capability.
        This can be used for an extension point for Anti-Virus
        Detection as an advanced NSF.";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Advanced NSF Anti-Virus Detection capability";
   }

   identity allow-list {
     base anti-virus-capability;
     description
       "Identity for advanced NSF Anti-Virus Allow List capability.
        This can be used for an extension point for Anti-Virus
        Allow List as an advanced NSF.";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Advanced NSF Anti-Virus Allow List capability";
   }

   identity syn-flood-action {
     base anti-ddos-capability;
     description
       "Identity for advanced NSF Anti-DDoS SYN Flood Action
        capability.  This can be used for an extension point for
        Anti-DDoS SYN Flood Action as an advanced NSF.";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Advanced NSF Anti-DDoS SYN Flood Action
        capability";
   }

   identity udp-flood-action {
     base anti-ddos-capability;
     description
       "Identity for advanced NSF Anti-DDoS UDP Flood Action
        capability.  This can be used for an extension point for
        Anti-DDoS UDP Flood Action as an advanced NSF.";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Advanced NSF Anti-DDoS UDP Flood Action
        capability";
   }

   identity http-flood-action {
     base anti-ddos-capability;
```

```
       description
         "Identity for advanced NSF Anti-DDoS HTTP Flood Action
          capability.  This can be used for an extension point for
          Anti-DDoS HTTP Flood Action as an advanced NSF.";
       reference
         "RFC 8329: Framework for Interface to Network Security
          Functions - Advanced NSF Anti-DDoS HTTP Flood Action
          capability";
     }

     identity https-flood-action {
       base anti-ddos-capability;
       description
         "Identity for advanced NSF Anti-DDoS HTTPS Flood Action
          capability.  This can be used for an extension point for
          Anti-DDoS HTTPS Flood Action as an advanced NSF.";
       reference
         "RFC 8329: Framework for Interface to Network Security
          Functions - Advanced NSF Anti-DDoS HTTPS Flood Action
          capability";
     }

     identity dns-request-flood-action {
       base anti-ddos-capability;
       description
         "Identity for advanced NSF Anti-DDoS DNS Request Flood
          Action capability.  This can be used for an extension
          point for Anti-DDoS DNS Request Flood Action as an
          advanced NSF.";
       reference
         "RFC 8329: Framework for Interface to Network Security
          Functions - Advanced NSF Anti-DDoS DNS Request Flood
          Action capability";
     }

     identity dns-reply-flood-action {
       base anti-ddos-capability;
       description
         "Identity for advanced NSF Anti-DDoS DNS Reply Flood
          Action capability.  This can be used for an extension
          point for Anti-DDoS DNS Reply Flood Action as an
          advanced NSF.";
       reference
         "RFC 8329: Framework for Interface to Network Security
          Functions - Advanced NSF Anti-DDoS DNS Reply Flood
          Action capability";
     }
```

```
   identity icmp-flood-action {
     base anti-ddos-capability;
     description
       "Identity for advanced NSF Anti-DDoS ICMP Flood Action
        capability.  This can be used for an extension point
        for Anti-DDoS ICMP Flood Action as an advanced NSF.";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Advanced NSF Anti-DDoS ICMP Flood Action
        capability";
   }

   identity icmpv6-flood-action {
     base anti-ddos-capability;
     description
       "Identity for advanced NSF Anti-DDoS ICMPv6 Flood Action
        capability.  This can be used for an extension point
        for Anti-DDoS ICMPv6 Flood Action as an advanced NSF.";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Advanced NSF Anti-DDoS ICMPv6 Flood Action
        capability";
   }

   identity sip-flood-action {
     base anti-ddos-capability;
     description
       "Identity for advanced NSF Anti-DDoS SIP Flood Action
        capability.  This can be used for an extension point
        for Anti-DDoS SIP Flood Action as an advanced NSF.";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Advanced NSF Anti-DDoS SIP Flood Action
        capability";
   }

   identity detect-mode {
     base anti-ddos-capability;
     description
       "Identity for advanced NSF Anti-DDoS Detection Mode
        capability.  This can be used for an extension point
        for Anti-DDoS Detection Mode as an advanced NSF.";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Advanced NSF Anti-DDoS Detection Mode
        capability";
   }
```

```
   identity baseline-learning {
     base anti-ddos-capability;
     description
       "Identity for advanced NSF Anti-DDoS Baseline Learning
        capability.  This can be used for an extension point
        for Anti-DDoS Baseline Learning as an advanced NSF.";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Advanced NSF Anti-DDoS Baseline Learning
        capability";
   }

   identity signature-set {
     base ips-capability;
     description
       "Identity for advanced NSF IPS Signature Set capability.
        This can be used for an extension point for IPS Signature
        Set as an advanced NSF.";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Advanced NSF IPS Signature Set capability";
   }

   identity ips-exception-signature {
     base ips-capability;
     description
       "Identity for advanced NSF IPS Exception Signature
        capability.  This can be used for an extension point for
        IPS Exception Signature as an advanced NSF.";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - Advanced NSF IPS Exception Signature Set
        capability";
   }

   identity voip-volte-call-id {
     base voip-volte-capability;
     description
       "Identity for advanced NSF VoIP/VoLTE Call-ID capability.
        This can be used for an extension point for VoIP/VoLTE
        Voice-ID as an advanced NSF.";
     reference
       "RFC 3261: SIP: Session Initiation Protocol";

   }

   identity user-agent {
     base voip-volte-capability;
```

```
     description
       "Identity for advanced NSF VoIP/VoLTE User Agent capability.
        This can be used for an extension point for VoIP/VoLTE
        User Agent as an advanced NSF.";
     reference
       "RFC 3261: SIP: Session Initiation Protocol";
   }

   identity ipsec-capability {
     description
       "Base identity for an IPsec capability";
     reference
       "draft-ietf-i2nsf-sdn-ipsec-flow-protection-08:
        Software-Defined Networking (SDN)-based IPsec Flow
        Protection - IPsec methods such as IKE and IKE-less";
   }

   identity ike {
     base ipsec-capability;
     description
       "Identity for an IPsec Internet Key Exchange (IKE)
       capability";
     reference
       "draft-ietf-i2nsf-sdn-ipsec-flow-protection-08:
        Software-Defined Networking (SDN)-based IPsec Flow
        Protection - IPsec method with IKE.
        RFC 7296: Internet Key Exchange Protocol Version 2
        (IKEv2) - IKE as a component of IPsec used for
        performing mutual authentication and establishing and
        maintaining Security Associations (SAs).";
   }

   identity ikeless {
     base ipsec-capability;
     description
       "Identity for an IPsec without Internet Key Exchange (IKE)
       capability";
     reference
       "draft-ietf-i2nsf-sdn-ipsec-flow-protection-08:
        Software-Defined Networking (SDN)-based IPsec Flow
        Protection - IPsec method without IKE";
   }

   /*
    *  Grouping
    */

   grouping nsf-capabilities {
```

```
     description
       "Network Security Function (NSF) Capabilities";
     reference
       "RFC 8329: Framework for Interface to Network Security
        Functions - I2NSF Flow Security Policy Structure.";

     leaf-list time-capabilities {
       type enumeration {
         enum absolute-time {
           description
             "absolute time capabilities.
              If a network security function has the absolute time
              capability, the network security function supports
              rule execution according to absolute time.";
         }
         enum periodic-time {
           description
             "periodic time capabilities.
              If a network security function has the periodic time
              capability, the network security function supports
              rule execution according to periodic time.";
         }
       }
       description
         "Time capabilities";
     }

     container event-capabilities {
       description
         "Capabilities of events.
          If a network security function has the event capabilities,
          the network security function supports rule execution
          according to system event and system alarm.";

       reference
         "RFC 8329: Framework for Interface to Network Security
          Functions - I2NSF Flow Security Policy Structure.
          draft-ietf-i2nsf-nsf-monitoring-data-model-04: I2NSF
          NSF Monitoring YANG Data Model - System Alarm and
          System Events.";

       leaf-list system-event-capability {
         type identityref {
           base system-event-capability;
         }
         description
           "System event capabilities";
       }
```

```
       leaf-list system-alarm-capability {
         type identityref {
           base system-alarm-capability;
         }
         description
           "System alarm capabilities";
       }
     }

     container condition-capabilities {
       description
         "Conditions capabilities.";

       container generic-nsf-capabilities {
         description
           "Conditions capabilities.
            If a network security function has the condition
            capabilities, the network security function
            supports rule execution according to conditions of
            IPv4, IPv6, TCP, UDP, SCTP, ICMP, ICMPv6, or payload.";
         reference
           "RFC 791: Internet Protocol - IPv4.
            RFC 792: Internet Control Message Protocol - ICMP.
            RFC 793: Transmission Control Protocol - TCP.
            RFC 768: User Datagram Protocol - UDP.
            RFC 4960: Stream Control Transmission Protocol - SCTP.
            RFC 8200: Internet Protocol, Version 6 (IPv6)
            Specification - IPv6.
            RFC 4443: Internet Control Message Protocol (ICMPv6)
            for the Internet Protocol Version 6 (IPv6) Specification
            - ICMPv6.
            RFC 8329: Framework for Interface to Network Security
            Functions - I2NSF Flow Security Policy Structure.";

         leaf-list ipv4-capability {
           type identityref {
             base ipv4-capability;
           }
           description
             "IPv4 packet capabilities";
           reference
             "RFC 791: Internet Protocol";
         }

         leaf-list icmp-capability {
           type identityref {
             base icmp-capability;
           }
```

```
           description
             "ICMP packet capabilities";
           reference
             "RFC 792: Internet Control Message Protocol - ICMP";
         }

         leaf-list ipv6-capability {
           type identityref {
             base ipv6-capability;
           }
           description
             "IPv6 packet capabilities";
           reference
             "RFC 8200: Internet Protocol, Version 6 (IPv6)
              Specification - IPv6";
         }

         leaf-list icmpv6-capability {
           type identityref {
             base icmpv6-capability;
           }
           description
             "ICMPv6 packet capabilities";
           reference
             "RFC 4443: Internet Control Message Protocol (ICMPv6)
              for the Internet Protocol Version 6 (IPv6) Specification
              - ICMPv6";
         }

         leaf-list tcp-capability {
           type identityref {
             base tcp-capability;
           }
           description
             "TCP packet capabilities";
           reference
             "RFC 793: Transmission Control Protocol - TCP";
         }

         leaf-list udp-capability {
           type identityref {
             base udp-capability;
           }
           description
             "UDP packet capabilities";
           reference
             "RFC 768: User Datagram Protocol - UDP";
         }
```

```
         leaf-list sctp-capability {
           type identityref {
             base sctp-capability;
           }
           description
             "SCTP packet capabilities";
           reference
             "RFC 4960: Stream Control Transmission Protocol - SCTP";
         }
       }

       container advanced-nsf-capabilities {
         description
           "Advanced Network Security Function (NSF) capabilities,
            such as Anti-Virus, Anti-DDoS, IPS, and VoIP/VoLTE.
            This container contains the leaf-lists of advanced
            NSF capabilities";
         reference
           "RFC 8329: Framework for Interface to Network Security
            Functions - Advanced NSF capabilities";

         leaf-list anti-virus-capability {
           type identityref {
             base anti-virus-capability;
           }
           description
             "Anti-Virus capabilities";
           reference
             "RFC 8329: Framework for Interface to Network Security
              Functions - Advanced NSF Anti-Virus capabilities";
           }

         leaf-list anti-ddos-capability {
           type identityref {
             base anti-ddos-capability;
           }
           description
             "Anti-DDoS Attack capabilities";
           reference
             "RFC 8329: Framework for Interface to Network Security
              Functions - Advanced NSF Anti-DDoS Attack capabilities";
         }

         leaf-list ips-capability {
           type identityref {
             base ips-capability;
           }
           description
```

```
             "IPS capabilities";
           reference
             "RFC 8329: Framework for Interface to Network Security
              Functions - Advanced NSF IPS capabilities";
         }

         leaf-list url-capability {
           type identityref {
             base url-capability;
           }
           description
             "URL capabilities";
           reference
             "RFC 8329: Framework for Interface to Network Security
              Functions - Advanced NSF URL capabilities";
         }

         leaf-list voip-volte-capability {
           type identityref {
             base voip-volte-capability;
          }
           description
             "VoIP/VoLTE capabilities";
           reference
             "RFC 8329: Framework for Interface to Network Security
              Functions - Advanced NSF VoIP/VoLTE capabilities";
         }
       }

       leaf-list context-capabilities {
         type identityref {
           base context-capability;
         }
         description
           "Security context capabilities";
       }
     }

     container action-capabilities {
       description
         "Action capabilities.
          If a network security function has the action capabilities,
          the network security function supports the attendant
          actions for policy rules.";

       leaf-list ingress-action-capability {
         type identityref {
           base ingress-action-capability;
```

```
        }
        description
          "Ingress-action capabilities";
      }

      leaf-list egress-action-capability {
        type identityref {
          base egress-action-capability;
        }
        description
          "Egress-action capabilities";
      }

      leaf-list log-action-capability {
        type identityref {
          base log-action-capability;
        }
        description
          "Log-action capabilities";
      }
    }

    leaf-list resolution-strategy-capabilities {
      type identityref {
        base resolution-strategy-capability;
      }
      description
        "Resolution strategy capabilities.
         The resolution strategies can be used to specify how
         to resolve conflicts that occur between the actions
         of the same or different policy rules that are matched
         for the same packet and by particular NSF";
    }

    leaf-list default-action-capabilities {
      type identityref {
        base default-action-capability;
      }
      description
        "Default action capabilities.
         A default action is used to execute I2NSF policy rules
         when no rule matches a packet. The default action is
         defined as pass, drop, alert, or mirror.";
      reference
        "RFC 8329: Framework for Interface to Network Security
         Functions - Ingress and egress actions.";
    }
```

```
      leaf-list ipsec-method {
        type identityref {
          base ipsec-capability;
        }
        description
          "IPsec method capabilities";
        reference
          "draft-ietf-i2nsf-sdn-ipsec-flow-protection-08:
           Software-Defined Networking (SDN)-based IPsec Flow
           Protection - IPsec methods such as IKE and IKE-less";
      }
    }

    /*
     * Data nodes
     */

    list nsf {
      key "nsf-name";
      description
        "The list of Network Security Functions (NSFs)";
      leaf nsf-name {
        type string;
        mandatory true;
        description
          "The name of Network Security Function (NSF)";
      }
      uses nsf-capabilities;
    }
  }


  <CODE ENDS>
```

Figure 3: YANG Data Module of I2NSF Capability

7. IANA Considerations

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

ID: yang:ietf-i2nsf-capability
URI: urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability
Registrant Contact: The IESG.
XML: N/A; the requested URI is an XML namespace.
Filename: [TBD-at-Registration]
Reference: [RFC-to-be]

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

Name: ietf-i2nsf-capability
Maintained by IANA? N
Namespace: urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability
Prefix: nsfcap
Module:
Reference: [RFC-to-be]

8. 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 transport secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the required transport 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.

There are a number of data nodes defined in this YANG module that are writable, creatable, and deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations to these data nodes could have a negative effect on network and security operations.

- list nsf: An attacker could alter the security capabilities associated with an NSF whereby disabling or enabling the evasion of security mitigations.

9. References

9.1. Normative References

[I-D.google-self-published-geofeeds]
Kline, E., Duleba, K., Szamonek, Z., Moser, S., and W. Kumari, "A Format for Self-published IP Geolocation Feeds", draft-google-self-published-geofeeds-09 (work in progress), February 2020.

[I-D.ietf-i2nsf-nsf-monitoring-data-model]
Jeong, J., Lingga, P., Hares, S., Xia, L., and H. Birkholz, "I2NSF NSF Monitoring YANG Data Model", draft-ietf-i2nsf-nsf-monitoring-data-model-04 (work in progress), September 2020.

[I-D.ietf-i2nsf-sdn-ipsec-flow-protection]
Lopez, R., Lopez-Millan, G., and F. Pereniguez-Garcia, "Software-Defined Networking (SDN)-based IPsec Flow Protection", draft-ietf-i2nsf-sdn-ipsec-flow-protection-12 (work in progress), October 2020.

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI 10.17487/RFC0768, August 1980, <https://www.rfc-editor.org/info/rfc768>.

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <https://www.rfc-editor.org/info/rfc791>.

[RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, DOI 10.17487/RFC0792, September 1981, <https://www.rfc-editor.org/info/rfc792>.

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <https://www.rfc-editor.org/info/rfc793>.

[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>.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <https://www.rfc-editor.org/info/rfc3261>.

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

[RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <https://www.rfc-editor.org/info/rfc4443>.

[RFC4960] Stewart, R., Ed., "Stream Control Transmission Protocol", RFC 4960, DOI 10.17487/RFC4960, September 2007, <https://www.rfc-editor.org/info/rfc4960>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[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>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <https://www.rfc-editor.org/info/rfc7296>.

[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>.

[RFC8192] Hares, S., Lopez, D., Zarny, M., Jacquenet, C., Kumar, R., and J. Jeong, "Interface to Network Security Functions (I2NSF): Problem Statement and Use Cases", RFC 8192, DOI 10.17487/RFC8192, July 2017, <https://www.rfc-editor.org/info/rfc8192>.

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

[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>.

[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>.

[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>.

[RFC8519] Jethanandani, M., Agarwal, S., Huang, L., and D. Blair, "YANG Data Model for Network Access Control Lists (ACLs)", RFC 8519, DOI 10.17487/RFC8519, March 2019, <https://www.rfc-editor.org/info/rfc8519>.

[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>.

[RFC8805] Kline, E., Duleba, K., Szamonek, Z., Moser, S., and W. Kumari, "A Format for Self-Published IP Geolocation Feeds", RFC 8805, DOI 10.17487/RFC8805, August 2020, <https://www.rfc-editor.org/info/rfc8805>.

9.2. Informative References

[Alshaer] Shaer, Al., Hamed, E., and H. Hamed, "Modeling and management of firewall policies", 2004.

[Galitsky] Galitsky, B. and R. Pampapathi, "Can many agents answer questions better than one", First Monday http://dx.doi.org/10.5210/fm.v10i1.1204, 2005.

[Hirschman]
Hirschman, L. and R. Gaizauskas, "Natural Language Question Answering: The View from Here", Natural Language Engineering 7:4, pgs 275-300, Cambridge University Press , Nov 2001.

[Hohpe] Hohpe, G. and B. Woolf, "Enterprise Integration Patterns", ISBN 0-32-120068-3 , 2003.

[I-D.ietf-i2nsf-terminology]
Hares, S., Strassner, J., Lopez, D., Xia, L., and H. Birkholz, "Interface to Network Security Functions (I2NSF) Terminology", draft-ietf-i2nsf-terminology-08 (work in progress), July 2019.

[I-D.ietf-supa-generic-policy-info-model]
Strassner, J., Halpern, J., and S. 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.

[IANA-Protocol-Numbers]
"Assigned Internet Protocol Numbers", Available: https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml, September 2020.

[Martin] Martin, R., "Agile Software Development, Principles, Patterns, and Practices", Prentice-Hall , ISBN: 0-13-597444-5 , 2002.

[OODMP] "http://www.oodesign.com/mediator-pattern.html".

[OODOP] "http://www.oodesign.com/mediator-pattern.html".

[OODSRP] "http://www.oodesign.com/mediator-pattern.html".

Appendix A. Configuration Examples

This section shows configuration examples of "ietf-i2nsf-capability" module for capabilities registration of general firewall.

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

This section shows a configuration example for the capabilities registration of a general firewall in either an IPv4 network or an IPv6 network.

```
<nsf xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
 <nsf-name>general_firewall</nsf-name>
 <condition-capabilities>
  <generic-nsf-capabilities>
   <ipv4-capability>ipv4-protocol</ipv4-capability>
   <ipv4-capability>exact-ipv4-address</ipv4-capability>
   <ipv4-capability>range-ipv4-address</ipv4-capability>
   <tcp-capability>exact-fourth-layer-port-num</tcp-capability>
   <tcp-capability>range-fourth-layer-port-num</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>alert</ingress-action-capability>
  <egress-action-capability>pass</egress-action-capability>
  <egress-action-capability>drop</egress-action-capability>
  <egress-action-capability>alert</egress-action-capability>
 </action-capabilities>
</nsf>
```

Figure 4: Configuration XML for the Capabilities Registration of a General Firewall in an IPv4 Network

Figure 4 shows the configuration XML for the capabilities registration of a general firewall as an NSF in an IPv4 network. Its capabilities are as follows.

1. The name of the NSF is general_firewall.

2. The NSF can inspect a protocol, an exact IPv4 address, and a range of IPv4 addresses for IPv4 packets.

3. The NSF can inspect an exact port number and a range of port numbers for the fourth layer packets.

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

```
<nsf xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
 <nsf-name>general_firewall</nsf-name>
 <condition-capabilities>
  <generic-nsf-capabilities>
   <ipv6-capability>ipv6-next-header</ipv6-capability>
   <ipv6-capability>exact-ipv6-address</ipv6-capability>
   <ipv6-capability>range-ipv6-address</ipv6-capability>
   <tcp-capability>exact-fourth-layer-port-num</tcp-capability>
   <tcp-capability>range-fourth-layer-port-num</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>alert</ingress-action-capability>
  <egress-action-capability>pass</egress-action-capability>
  <egress-action-capability>drop</egress-action-capability>
  <egress-action-capability>alert</egress-action-capability>
 </action-capabilities>
</nsf>
```

Figure 5: Configuration XML for the Capabilities Registration of a General Firewall in an IPv6 Network

In addition, Figure 5 shows the configuration XML for the capabilities registration of a general firewall as an NSF in an IPv6 network. Its capabilities are as follows.

1. The name of the NSF is general_firewall.

2. The NSF can inspect a protocol (Next-Header), an exact IPv6 address, and a range of IPv6 addresses for IPv6 packets.

3. The NSF can inspect an exact port number and a range of port numbers for the fourth layer packets.

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

A.2. **Example 2: Registration for the Capabilities of a Time-based** Firewall

This section shows a configuration example for the capabilities registration of a time-based firewall in either an IPv4 network or an IPv6 network.

```
<nsf xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
 <nsf-name>time_based_firewall</nsf-name>
 <time-capabilities>absolute-time</time-capabilities>
 <time-capabilities>periodic-time</time-capabilities>
 <condition-capabilities>
  <generic-nsf-capabilities>
   <ipv4-capability>ipv4-next-header</ipv4-capability>
   <ipv4-capability>exact-ipv4-address</ipv4-capability>
   <ipv4-capability>range-ipv4-address</ipv4-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>alert</ingress-action-capability>
  <egress-action-capability>pass</egress-action-capability>
  <egress-action-capability>drop</egress-action-capability>
  <egress-action-capability>alert</egress-action-capability>
 </action-capabilities>
</nsf>
```

Figure 6: Configuration XML for the Capabilities Registration of a Time-based Firewall in an IPv4 Network

Figure 6 shows the configuration XML for the capabilities registration of a time-based firewall as an NSF in an IPv4 network. Its capabilities are as follows.

1. The name of the NSF is time_based_firewall.

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

3. The NSF can inspect a protocol (Next-Header), an exact IPv4 address, and a range of IPv4 addresses for IPv4 packets.

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

```
<nsf xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
 <nsf-name>time_based_firewall</nsf-name>
 <time-capabilities>absolute-time</time-capabilities>
 <time-capabilities>periodic-time</time-capabilities>
 <condition-capabilities>
  <generic-nsf-capabilities>
   <ipv6-capability>ipv6-next-header</ipv6-capability>
   <ipv6-capability>exact-ipv6-address</ipv6-capability>
   <ipv6-capability>range-ipv6-address</ipv6-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>alert</ingress-action-capability>
  <egress-action-capability>pass</egress-action-capability>
  <egress-action-capability>drop</egress-action-capability>
  <egress-action-capability>alert</egress-action-capability>
 </action-capabilities>
</nsf>
```

Figure 7: Configuration XML for the Capabilities Registration of a Time-based Firewall in an IPv6 Network

In addition, Figure 7 shows the configuration XML for the capabilities registration of a time-based firewall as an NSF in an IPv6 network. Its capabilities are as follows.

1. The name of the NSF is time_based_firewall.

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

3. The NSF can inspect a protocol (Next-Header), an exact IPv6 address, and a range of IPv6 addresses for IPv6 packets.

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

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

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

```
<nsf xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
 <nsf-name>web_filter</nsf-name>
 <condition-capabilities>
  <advanced-nsf-capabilities>
   <url-capability>user-defined</url-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>alert</ingress-action-capability>
  <egress-action-capability>pass</egress-action-capability>
  <egress-action-capability>drop</egress-action-capability>
  <egress-action-capability>alert</egress-action-capability>
 </action-capabilities>
</nsf>
```

Figure 8: Configuration XML for the Capabilities Registration of a Web Filter

Figure 8 shows the configuration XML for the capabilities registration of a web filter as an NSF. Its capabilities are as follows.

1. The name of the NSF is web_filter.

2. The NSF can inspect URL matched from a user-defined URL Database. User can add a new URL into the database.

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

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

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

```
<nsf xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
 <nsf-name>voip_volte_filter</nsf-name>
 <condition-capabilities>
  <advanced-nsf-capabilities>
   <voip-volte-capability>voice-id</voip-volte-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>alert</ingress-action-capability>
  <egress-action-capability>pass</egress-action-capability>
  <egress-action-capability>drop</egress-action-capability>
  <egress-action-capability>alert</egress-action-capability>
 </action-capabilities>
</nsf>
```

Figure 9: Configuration XML for the Capabilities Registration of a VoIP/VoLTE Filter

Figure 9 shows the configuration XML for the capabilities registration of a VoIP/VoLTE filter as an NSF. Its capabilities are as follows.

1. The name of the NSF is voip_volte_filter.

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

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

A.5. Example 5: Registration for the Capabilities of a HTTP and HTTPS Flood Mitigator

This section shows a configuration example for the capabilities registration of a HTTP and HTTPS flood mitigator.

```
<nsf xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
 <nsf-name>http_and_https_flood_mitigation</nsf-name>
 <condition-capabilities>
  <advanced-nsf-capabilities>
   <anti-ddos-capability>http-flood-action</anti-ddos-capability>
   <anti-ddos-capability>https-flood-action</anti-ddos-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>alert</ingress-action-capability>
  <egress-action-capability>pass</egress-action-capability>
  <egress-action-capability>drop</egress-action-capability>
  <egress-action-capability>alert</egress-action-capability>
 </action-capabilities>
</nsf>
```

Figure 10: Configuration XML for the Capabilities Registration of a HTTP and HTTPS Flood Mitigator

Figure 10 shows the configuration XML for the capabilities registration of a HTTP and HTTPS flood mitigator as an NSF. Its capabilities are as follows.

1. The name of the NSF is http_and_https_flood_mitigation.

2. The NSF can control the amount of packets for HTTP and HTTPS packets, which are routed to the NSF's IPv4 address or the NSF's IPv6 address.

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

Appendix B. Acknowledgments

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea MSIT (Ministry of Science and ICT) (R-20160222-002755, Cloud based Security Intelligence Technology Development for the Customized Security Service Provisioning). This work was supported in part by the IITP grant funded by the MSIT (2020-0-00395, Standard Development of Blockchain based Network Management Automation Technology).

Appendix C. Contributors

This document is made by the group effort of I2NSF working group. Many people actively contributed to this document, such as Acee Lindem, Roman Danyliw, and Tom Petch. The authors sincerely appreciate their contributions.

The following are co-authors of this document:

Patrick Lingga
Department of Computer Science and Engineering
Sungkyunkwan University
2066 Seo-ro Jangan-gu
Suwon, Gyeonggi-do 16419
Republic of Korea

EMail: patricklink@skku.edu

Liang Xia
Huawei
101 Software Avenue
Nanjing, Jiangsu 210012
China

EMail: Frank.Xialiang@huawei.com

Cataldo Basile
Politecnico di Torino
Corso Duca degli Abruzzi, 34
Torino, 10129
Italy

EMail: cataldo.basile@polito.it

John Strassner
Huawei
2330 Central Expressway
Santa Clara, CA 95050
USA

EMail: John.sc.Strassner@huawei.com

Diego R. Lopez
Telefonica I+D

Zurbaran, 12
Madrid, 28010
Spain

Email: diego.r.lopez@telefonica.com

Hyoungshick Kim
Department of Computer Science and Engineering
Sungkyunkwan University
2066 Seo-ro Jangan-gu
Suwon, Gyeonggi-do 16419
Republic of Korea

EMail: hyoung@skku.edu

Daeyoung Hyun
Department of Computer Science and Engineering
Sungkyunkwan University
2066 Seo-ro Jangan-gu
Suwon, Gyeonggi-do 16419
Republic of Korea

EMail: dyhyun@skku.edu

Dongjin Hong
Department of Electronic, Electrical and Computer Engineering
Sungkyunkwan University
2066 Seo-ro Jangan-gu
Suwon, Gyeonggi-do 16419
Republic of Korea

EMail: dong.jin@skku.edu

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

EMail: pjs@etri.re.kr

Tae-Jin Ahn
Korea Telecom

70 Yuseong-Ro, Yuseong-Gu
Daejeon, 305-811
Republic of Korea

EMail: taejin.ahn@kt.com

Se-Hui Lee
Korea Telecom
70 Yuseong-Ro, Yuseong-Gu
Daejeon, 305-811
Republic of Korea

EMail: sehuilee@kt.com

Authors' Addresses

Susan Hares (editor)
Huawei
7453 Hickory Hill
Saline, MI 48176
USA

Phone: +1-734-604-0332
EMail: shares@ndzh.com

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
Fax: +82 31 290 7996
EMail: pauljeong@skku.edu
URI: http://iotlab.skku.edu/people-jaehoon-jeong.php

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

Phone: +82 10 8273 0930
EMail: timkim@skku.edu

Robert Moskowitz
HTT Consulting
Oak Park, MI
USA

Phone: +1-248-968-9809
EMail: rgm@htt-consult.com

Qiushi Lin
Huawei
Huawei Industrial Base
Shenzhen, Guangdong 518129
China

EMail: linqiushi@huawei.com