Workgroup: I2NSF Working Group
Internet-Draft:
draft-ietf-i2nsf-nsf-facing-interface-dm-13
Published: 15 August 2021
Intended Status: Standards Track
Expires: 16 February 2022
A uthors: J. Kim, Ed. Sungkyunkwan University
J. Jeong, Ed. Sungkyunkwan University
J. Park ETRI
S. Hares Huawei
Q. Lin Huawei

I2NSF Network Security Function-Facing Interface YANG Data Model

Abstract

This document defines a YANG data model for configuring security policy rules on Network Security Functions (NSF) in the Interface to Network Security Functions (I2NSF) framework. The YANG data model in this document corresponds to the information model for NSF-Facing Interface in the I2NSF framework.

Status of This Memo

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

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

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

This Internet-Draft will expire on 16 February 2022.

Copyright Notice

Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

This document defines a YANG [RFC6020][RFC7950] data model for security policy rule configuration of Network Security Functions (NSF). The YANG data model in this document is based on the information and data model in [I-D.ietf-i2nsf-capability-data-model] for the NSF-Facing Interface in the Interface to Network Security Functions (I2NSF) architecture [RFC8329]. The YANG data model in this document focuses on security policy configuration for the NSFs discussed in [I-D.ietf-i2nsf-capability-data-model], i.e., generic NSF (.

This YANG data model uses an "Event-Condition-Action" (ECA) policy model that is used as the basis for the design of I2NSF Policy described in [RFC8329] and [I-D.ietf-i2nsf-capability-data-model].

The "ietf-i2nsf-policy-rule-for-nsf" YANG module defined in this document provides the configuration of the following features.

*A security policy rule of a network security function.

*An event clause of a generic network security function.

*A condition clause of a generic network security function.

*An action clause of a generic network security function.

2. Terminology

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

This document uses the 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. YANG Tree Diagram

This section shows a YANG tree diagram of policy for network security functions. [I-D.ietf-i2nsf-capability-data-model].

3.1. General I2NSF Security Policy Rule

This section shows a YANG tree diagram for a general I2NSF security policy rule for generic network security functions.

```
module: ietf-i2nsf-policy-rule-for-nsf
  +--rw i2nsf-security-policy* [system-policy-name]
     +--rw system-policy-name     string
     +--rw priority-usage?        identityref
     +--rw resolution-strategy?   identityref
     +--rw default-action?        identityref
     +--rw rules* [rule-name]
     |  +--rw rule-name             string
     |  +--rw rule-description?     string
     |  +--rw rule-priority?        uint8
     |  +--rw rule-enable?          boolean
     |  +--rw session-aging-time?   uint16
     |  +--rw long-connection
     |  |  +--rw enable?     boolean
     |  |  +--rw duration?   uint16
     |  +--rw event
     |     ...
     |  +--rw action
     |     ...
     +--rw rule-group
        +--rw groups* [group-name]
           +--rw group-name     string
           +--rw rule-range
           |  +--rw start-rule?   string
           |  +--rw end-rule?     string
           +--rw enable?        boolean
           +--rw description?   string
```

Figure 1: YANG Tree Diagram for Network Security Policy

The system policy provides for multiple system policies in one NSF, and each system policy is used by one virtual instance of the NSF/

device. The system policy includes system policy name, priority usage, resolution strategy, default action, and rules.

A resolution strategy is used to decide how to resolve conflicts that occur between the actions of the same or different policy rules that are matched and contained in a particular NSF. The resolution strategy is defined as First Matching Rule (FMR), Last Matching Rule (LMR), Prioritized Matching Rule (PMR) with Errors (PMRE), and Prioritized Matching Rule with No Errors (PMRN). The resolution strategy can be extended according to specific vendor action features. The resolution strategy is described in detail in [I-D.ietf-i2nsf-capability-data-model].

A default action is used to execute I2NSF policy rule when no rule matches a packet. The default action is defined as pass, drop, rate-limit, and mirror. The default action can be extended according to specific vendor action features. The default action is described in detail in [I-D.ietf-i2nsf-capability-data-model].

The rules include rule name, rule description, rule priority, rule enable, event, condition, and action.

3.2. Event Clause

This section shows a YANG tree diagram for an event clause for a general I2NSF security policy rule for generic network security functions.

```
module: ietf-i2nsf-policy-rule-for-nsf
  +--rw i2nsf-security-policy* [system-policy-name]
     ...
     +--rw rules* [rule-name]
     |  ...
     |  +--rw event
     |  |  +--rw event-clause-description?   string
     |  |  +--rw time
     |  |  |  +--rw start-date-time?   yang:date-and-time
     |  |  |  +--rw end-date-time?     yang:date-and-time
     |  |  |  +--rw period
     |  |  |  |  +--rw start-time?   time
     |  |  |  |  +--rw end-time?     time
     |  |  |  |  +--rw day*          identityref
     |  |  |  |  +--rw date*         int32
     |  |  |  |  +--rw month*        string
     |  |  |  +--rw frequency?         enumeration
     |  |  +--rw event-clauses
     |  |     +--rw system-event*   identityref
     |  |     +--rw system-alarm*   identityref
     |  +--rw condition
     |  |  ...
     |  +--rw action
     |     ...
     +--rw rule-group
        ...
```

Figure 2: YANG Tree Diagram for an Event Clause

An event clause is any important occurrence at a specific time of a change in the system being managed, and/or in the environment of the system being managed. An event clause is used to trigger the evaluation of the condition clause of the I2NSF Policy Rule. The event clause is defined as a system event, system alarm [I-D.ietf-i2nsf-nsf-monitoring-data-model] and time. The event clause can be extended according to specific vendor event features. The event clause is described in detail in [I-D.ietf-i2nsf-capability-data-model].

3.3. Condition Clause

This section shows a YANG tree diagram for a condition clause for a general I2NSF security policy rule for generic network security functions.

```
module: ietf-i2nsf-policy-rule-for-nsf
  +--rw i2nsf-security-policy* [system-policy-name]
     ...
     +--rw rules* [rule-name]
     |  ...
     |  +--rw event
     |     ...
     |  +--rw condition
     |  |  +--rw condition-clause-description?   string
     |  |  +--rw mac
     |  |  |  +--rw mac-description?       string
     |  |  |  +--rw source-address*        yang:mac-address
     |  |  |  +--rw destination-address*   yang:mac-address
     |  |  |  +--rw ether-type*            uint16
     |  |  +--rw ipv4
     |  |  |  +--rw description?           string
     |  |  |  +--rw header-length* [start end]
     |  |  |  |  +--rw start    uint8
     |  |  |  |  +--rw end      uint8
     |  |  |  +--rw dscp*                  inet:dscp
     |  |  |  +--rw total-length* [start end]
     |  |  |  |  +--rw start    uint16
     |  |  |  |  +--rw end      uint16
     |  |  |  +--rw identification*        uint16
     |  |  |  +--rw fragment-flags*        identityref
     |  |  |  +--rw fragment-offset* [start end]
     |  |  |  |  +--rw start    uint16
     |  |  |  |  +--rw end      uint16
     |  |  |  +--rw ttl* [start end]
     |  |  |  |  +--rw start    uint8
     |  |  |  |  +--rw end      uint8
     |  |  |  +--rw protocol*              uint8
     |  |  |  +--rw source-address
     |  |  |  |  +--rw (match-type)?
     |  |  |  |     +--:(prefix)
     |  |  |  |     |  +--rw ipv4-prefix* [ipv4]
     |  |  |  |     |     +--rw ipv4                inet:ipv4-address
     |  |  |  |     |     +--rw (subnet)?
     |  |  |  |     |        +--:(prefix-length)
     |  |  |  |     |        |  +--rw prefix-length?  uint8
     |  |  |  |     |        +--:(netmask)
     |  |  |  |     |           +--rw netmask?      yang:dotted-quad
     |  |  |  |     +--:(range)
     |  |  |  |        +--rw ipv4-range* [start end]
     |  |  |  |           +--rw start    inet:ipv4-address
     |  |  |  |           +--rw end      inet:ipv4-address
     |  |  |  +--rw destination-address
     |  |  |  |  +--rw (match-type)?
     |  |  |  |     +--:(prefix)
     |  |  |  |     |  +--rw ipv4-prefix* [ipv4]
     |  |  |  |     |     +--rw ipv4                inet:ipv4-address
     |  |  |  |     |     +--rw (subnet)?
     |  |  |  |     |        +--:(prefix-length)
     |  |  |  |     |        |  +--rw prefix-length?  uint8
     |  |  |  |     |        +--:(netmask)
     |  |  |  |     |           +--rw netmask?      yang:dotted-quad
     |  |  |  |     +--:(range)
     |  |  |  |        +--rw ipv4-range* [start end]
```

```
|  |  |  |          +--rw start    inet:ipv4-address
|  |  |  |          +--rw end      inet:ipv4-address
|  |  |  +--rw ipopts*               identityref
|  |  +--rw ipv6
|  |  |  +--rw description?          string
|  |  |  +--rw dscp*                 inet:dscp
|  |  |  +--rw flow-label* [start end]
|  |  |  |  +--rw start    inet:ipv6-flow-label
|  |  |  |  +--rw end      inet:ipv6-flow-label
|  |  |  +--rw payload-length* [start end]
|  |  |  |  +--rw start    uint16
|  |  |  |  +--rw end      uint16
|  |  |  +--rw next-header*          uint8
|  |  |  +--rw hop-limit* [start end]
|  |  |  |  +--rw start    uint8
|  |  |  |  +--rw end      uint8
|  |  |  +--rw source-address
|  |  |  |  +--rw (match-type)?
|  |  |  |     +--:(prefix)
|  |  |  |     |  +--rw ipv6-prefix* [ipv6]
|  |  |  |     |     +--rw ipv6            inet:ipv6-address
|  |  |  |     |     +--rw prefix-length?   uint8
|  |  |  |     +--:(range)
|  |  |  |        +--rw ipv6-range* [start end]
|  |  |  |           +--rw start    inet:ipv6-address
|  |  |  |           +--rw end      inet:ipv6-address
|  |  |  +--rw destination-address
|  |  |     +--rw (match-type)?
|  |  |        +--:(prefix)
|  |  |        |  +--rw ipv6-prefix* [ipv6]
|  |  |        |     +--rw ipv6            inet:ipv6-address
|  |  |        |     +--rw prefix-length?   uint8
|  |  |        +--:(range)
|  |  |           +--rw ipv6-range* [start end]
|  |  |              +--rw start    inet:ipv6-address
|  |  |              +--rw end      inet:ipv6-address
|  |  +--rw tcp
|  |  |  +--rw description?        string
|  |  |  +--rw source-port-number* [start end]
|  |  |  |  +--rw start    inet:port-number
|  |  |  |  +--rw end      inet:port-number
|  |  |  +--rw destination-port-number* [start end]
|  |  |  |  +--rw start    inet:port-number
|  |  |  |  +--rw end      inet:port-number
|  |  |  +--rw flags*              identityref
|  |  +--rw udp
|  |  |  +--rw description?              string
|  |  |  +--rw source-port-number
|  |  |  |  +--rw start?   inet:port-number
|  |  |  |  +--rw end?     inet:port-number
|  |  |  +--rw destination-port-number
|  |  |  |  +--rw start?   inet:port-number
|  |  |  |  +--rw end?     inet:port-number
|  |  |  +--rw total-length* [start end]
|  |  |     +--rw start    uint32
|  |  |     +--rw end      uint32
|  |  +--rw sctp
|  |  |  +--rw description?              string
```

```
|  |  |  +--rw source-port-number
|  |  |  |  +--rw start?   inet:port-number
|  |  |  |  +--rw end?     inet:port-number
|  |  |  +--rw destination-port-number
|  |  |  |  +--rw start?   inet:port-number
|  |  |  |  +--rw end?     inet:port-number
|  |  |  +--rw verification-tag*          uint32
|  |  |  +--rw chunk-type*                uint8
|  |  +--rw dccp
|  |  |  +--rw description?               string
|  |  |  +--rw source-port-number
|  |  |  |  +--rw start?   inet:port-number
|  |  |  |  +--rw end?     inet:port-number
|  |  |  +--rw destination-port-number
|  |  |  |  +--rw start?   inet:port-number
|  |  |  |  +--rw end?     inet:port-number
|  |  |  +--rw service-code*              uint32
|  |  +--rw icmp* [version]
|  |  |  +--rw description?   string
|  |  |  +--rw version        enumeration
|  |  |  +--rw type*          uint8
|  |  |  +--rw code*          uint8
|  |  +--rw url-category
|  |  |  +--rw description?             string
|  |  |  +--rw pre-defined-category*    string
|  |  |  +--rw user-defined-category*   string
|  |  +--rw voice
|  |  |  +--rw description?            string
|  |  |  +--rw source-voice-id*        string
|  |  |  +--rw destination-voice-id*   string
|  |  |  +--rw user-agent*             string
|  |  +--rw ddos
|  |  |  +--rw description?         string
|  |  |  +--rw alert-packet-rate?   uint32
|  |  |  +--rw alert-flow-rate?     uint32
|  |  |  +--rw alert-byte-rate?     uint32
|  |  +--rw anti-virus
|  |  |  +--rw profile?         string
|  |  |  +--rw exception-files?   string
|  |  +--rw payload
|  |  |  +--rw packet-payload-description?   string
|  |  |  +--rw payload-content*               string
|  |  +--rw context
|  |     +--rw context-description?   string
|  |     +--rw application
|  |     |  +--rw description?   string
|  |     |  +--rw object*        string
|  |     |  +--rw group*         string
|  |     |  +--rw label*         string
|  |     |  +--rw category
|  |     |     +--rw application-category* [name subcategory]
|  |     |        +--rw name           string
|  |     |        +--rw subcategory    string
|  |     +--rw target
|  |     |  +--rw description?   string
|  |     |  +--rw device*        identityref
|  |     +--rw users
|  |     |  +--rw users-description?   string
```

```
|  |     |  +--rw user* [user-id]
|  |     |  |  +--rw user-id      uint32
|  |     |  |  +--rw user-name?   string
|  |     |  +--rw group* [group-id]
|  |     |  |  +--rw group-id      uint32
|  |     |  |  +--rw group-name?   string
|  |     |  +--rw security-group?      string
|  |     +--rw geography-location
|  |        +--rw description?   string
|  |        +--rw source*        string
|  |        +--rw destination*   string
|  +--rw action
|     ...
+--rw rule-group
   ...
```

Figure 3: YANG Tree Diagram for a Condition Clause

A condition clause 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. A condition clause is classified as a condition of generic network security functions, advanced network security functions, or context. A condition clause of generic network security functions is defined as IPv4 condition, IPv6 condition, TCP condition, UDP condition, SCTP condition, DCCP condition, and ICMP (ICMPv4 and ICMPv6) condition.

Note that the data model in this document does not focus on only IP addresses, but focuses on all the fields of IPv4 and IPv6 headers. The IPv4 and IPv6 headers have similarity with some different fields. In this case, it is better to handle separately the IPv4 and IPv6 headers such that the different fields can be used to handle IPv4 and IPv6 packets.

A condition clause of advanced network security functions is defined as url category condition, voice condition, DDoS condition, or payload condition. A condition clause of context is defined as application condition, target condition, users condition, and geography condition.

Note that this document deals only with conditions of several advanced network security functions such as url filter (i.e., web filter), VoIP/VoLTE security, and DDoS-attack mitigator. A condition clause of other advanced network security functions such as Intrusion Prevention System (IPS) and Data Loss Prevention (DLP) can be defined as an extension in future. A condition clause can be extended according to specific vendor condition features. A condition clause is described in detail in [I-D.ietf-i2nsf-capability-data-model].

3.4. Action Clause

This section shows a YANG tree diagram for an action clause for a general I2NSF security policy rule for generic network security functions.

```
module: ietf-i2nsf-policy-rule-for-nsf
  +--rw i2nsf-security-policy* [system-policy-name]
     ...
     +--rw rules* [rule-name]
     |  ...
     |  +--rw event
     |     ...
     |  +--rw condition
     |     ...
     |  +--rw action
     |     +--rw action-clause-description?   string
     |     +--rw packet-action
     |     |  +--rw ingress-action?   identityref
     |     |  +--rw egress-action?    identityref
     |     |  +--rw log-action?       identityref
     |     +--rw flow-action
     |     |  +--rw ingress-action?   identityref
     |     |  +--rw egress-action?    identityref
     |     |  +--rw log-action?       identityref
     |     +--rw advanced-action
     |        +--rw content-security-control*    identityref
     |        +--rw attack-mitigation-control*   identityref
     +--rw rule-group
        ...
```

Figure 4: YANG Tree Diagram for an Action Clause

An action is used to control and monitor aspects of flow-based NSFs when the policy rule event and condition clauses are satisfied. NSFs provide security services by executing various actions. The action clause is defined as ingress action, egress action, or log action for packet action, flow action, and advanced action for additional inspection. The packet action is an action for an individual packet such as an IP datagram as a stateless process that uses the packet's header and payload. The flow action is an action of a traffic flow such as the packets of a TCP session (e.g., an HTTP/HTTPS session) as a stateful process that uses the traffic flow information such as 5-tuple information, packet counts, and byte counts. The advanced action is an action for an advanced security service (e.g., url filter, DDoS-attack mitigator, and VoIP/VoLTE filter) for either a packet or a traffic flow according to the intention of such an advanced security service. The action clause can be extended according to specific vendor action features. The action clause is described in detail in [I-D.ietf-i2nsf-capability-data-model].

4. YANG Data Model of NSF-Facing Interface

The main objective of this data model is to provide both an information model and the corresponding YANG data model of I2NSF NSF-Facing Interface. This interface can be used to deliver control and management messages between Security Controller and NSFs for the I2NSF low-level security policies.

This data model is designed to support the I2NSF framework that can be extended according to the security needs. In other words, the model design is independent of the content and meaning of specific policies as well as the implementation approach.

With the YANG data model of I2NSF NSF-Facing Interface, this document suggests use cases for security policy rules such as time-based firewall, web filter, VoIP/VoLTE security service, and DDoS-attack mitigation in Section 5.

4.1. YANG Module of NSF-Facing Interface

This section describes a YANG module of NSF-Facing Interface. This document provides identities in the data model for the configuration of an NSF. The identity has the same concept with the corresponding identity in [I-D.ietf-i2nsf-consumer-facing-interface-dm] This YANG module imports from [RFC6991]. It makes references to [RFC0768] [RFC0791] [RFC0792] [RFC0793] [RFC2474] [RFC3261] [RFC4340] [RFC4960] [RFC6335] [RFC8200] [RFC8329] [RFC8335] [RFC8344] [IEEE-802.3] [ISO-Country-Codes] [IANA-Protocol-Numbers] [IANA-ICMP-Parameters] [I-D.ietf-i2nsf-capability-data-model] [I-D.ietf-i2nsf-nsf-monitoring-data-model].

```
<CODE BEGINS> file "ietf-i2nsf-policy-rule-for-nsf@2021-08-15.yang"

module ietf-i2nsf-policy-rule-for-nsf {
  yang-version 1.1;
  namespace
    "urn:ietf:params:xml:ns:yang:ietf-i2nsf-policy-rule-for-nsf";
  prefix
    nsfintf;

  import ietf-inet-types{
    prefix inet;
    reference
      "Section 4 of RFC 6991";
  }
  import ietf-yang-types {
    prefix yang;
    reference
      "Section 3 of RFC 6991";
  }

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

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

  description
    "This module is a YANG module for Network Security Functions
     (NSF)-Facing Interface.

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

     Copyright (c) 2021 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
     (https://trustee.ietf.org/license-info).

     This version of this YANG module is part of RFC XXXX
     (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
     for full legal notices.";
```

```
revision "2021-08-15"{
  description "The latest revision.";
  reference
    "RFC XXXX: I2NSF Network Security Function-Facing Interface
     YANG Data Model";
}

/*
 * Identities
 */

identity priority-usage {
  description
    "Base identity for priority usage type.";
}

identity priority-by-order {
  base priority-usage;
  description
    "Identity for priority by order";
}

identity priority-by-number {
  base priority-usage;
  description
    "Identity for priority by number";
}

identity event {
  description
    "Base identity for policy events";
  reference
    "draft-ietf-i2nsf-nsf-monitoring-data-model-08: I2NSF NSF
     Monitoring YANG Data Model - Event";
}

identity system-event {
  base event;
  description
    "Identity for system events";
  reference
    "draft-ietf-i2nsf-nsf-monitoring-data-model-08: I2NSF NSF
     Monitoring YANG Data Model - System event";
}

identity system-alarm {
  base event;
  description
    "Identity for system alarms";
  reference
    "draft-ietf-i2nsf-nsf-monitoring-data-model-08: I2NSF NSF
     Monitoring YANG Data Model - System alarm";
}

identity access-violation {
  base system-event;
  description
```

```
    "Identity for access violation
    system events";
  reference
    "draft-ietf-i2nsf-nsf-monitoring-data-model-08: I2NSF NSF
     Monitoring YANG Data Model - System event for access
     violation";
}

identity configuration-change {
  base system-event;
  description
    "Identity for configuration change
    system events";
  reference
    "draft-ietf-i2nsf-nsf-monitoring-data-model-08: I2NSF NSF
     Monitoring YANG Data Model - System event for configuration
     change";
}

identity memory-alarm {
  base system-alarm;
  description
    "Identity for memory alarm
    system alarms";
  reference
    "draft-ietf-i2nsf-nsf-monitoring-data-model-08: I2NSF NSF
     Monitoring YANG Data Model - System alarm for memory";
}

identity cpu-alarm {
  base system-alarm;
  description
    "Identity for CPU alarm
    system alarms";
  reference
    "draft-ietf-i2nsf-nsf-monitoring-data-model-08: I2NSF NSF
     Monitoring YANG Data Model - System alarm for CPU";
}

identity disk-alarm {
  base system-alarm;
  description
    "Identity for disk alarm
    system alarms";
  reference
    "draft-ietf-i2nsf-nsf-monitoring-data-model-08: I2NSF NSF
     Monitoring YANG Data Model - System alarm for disk";
}

identity hardware-alarm {
  base system-alarm;
  description
    "Identity for hardware alarm
    system alarms";
  reference
    "draft-ietf-i2nsf-nsf-monitoring-data-model-08: I2NSF NSF
     Monitoring YANG Data Model - System alarm for hardware";
}
```

```
identity interface-alarm {
  base system-alarm;
  description
    "Identity for interface alarm
    system alarms";
  reference
    "draft-ietf-i2nsf-nsf-monitoring-data-model-08: I2NSF NSF
     Monitoring YANG Data Model - System alarm for interface";
}

identity fragmentation-flags {
  description
    "Base identity for fragmentation flags type";
  reference
    "RFC 791: Internet Protocol - Fragmentation Flags";
}

identity fragment {
  base fragmentation-flags;
  description
    "Identity for 'More fragment' flag";
  reference
    "RFC 791: Internet Protocol - Fragmentation Flags";
}

identity no-fragment {
  base fragmentation-flags;
  description
    "Identity for 'Do not fragment' flag";
  reference
    "RFC 791: Internet Protocol - Fragmentation Flags";
}

identity reserved {
  base fragmentation-flags;
  description
    "Identity for reserved flags";
  reference
    "RFC 791: Internet Protocol - Fragmentation Flags";
}

identity ipopts {
  description
    "Base identity for IP options";
  reference
    "RFC 791: Internet Protocol - Options";
}

identity rr {
  base ipopts;
  description
    "Identity for 'Record Route' IP Option";
  reference
    "RFC 791: Internet Protocol - Options";
}

identity eol {
```

```
  base ipopts;
  description
    "Identity for 'End of List' IP Option";
  reference
    "RFC 791: Internet Protocol - Options";
}

identity nop {
  base ipopts;
  description
    "Identity for 'No Operation' IP Option";
  reference
    "RFC 791: Internet Protocol - Options";
}

identity ts {
  base ipopts;
  description
    "Identity for 'Timestamp' IP Option";
  reference
    "RFC 791: Internet Protocol - Options";
}

identity sec {
  base ipopts;
  description
    "Identity for 'IP security' IP Option";
  reference
    "RFC 791: Internet Protocol - Options";
}

identity esec {
  base ipopts;
  description
    "Identity for 'IP extended security' IP Option";
  reference
    "RFC 791: Internet Protocol - Options";
}

identity lsrr {
  base ipopts;
  description
    "Identity for 'Loose Source Routing' IP Option";
  reference
    "RFC 791: Internet Protocol - Options";
}

identity ssrr {
  base ipopts;
  description
    "Identity for 'Strict Source Routing' IP Option";
  reference
    "RFC 791: Internet Protocol - Options";
}

identity satid {
  base ipopts;
  description
```

```
    "Identity for 'Stream Identifier' IP Option";
  reference
    "RFC 791: Internet Protocol - Options";
}

identity any {
  base ipopts;
  description
    "Identity for 'any IP options
    included in IPv4 packet";
  reference
    "RFC 791: Internet Protocol - Options";
}

identity tcp-flags {
  description
    "Base identity for TCP flags";
  reference
    "RFC 793: Transmission Control Protocol - Flags";
}

identity cwr {
  base tcp-flags;
  description
    "Identity for 'Congestion Window Reduced' TCP flag";
  reference
    "RFC 793: Transmission Control Protocol - Flags";
}

identity ecn {
  base tcp-flags;
  description
    "Identity for 'Explicit Congestion Notification'
    TCP flag";
  reference
    "RFC 793: Transmission Control Protocol - Flags";
}

identity urg {
  base tcp-flags;
  description
    "Identity for 'Urgent' TCP flag";
  reference
    "RFC 793: Transmission Control Protocol - Flags";
}

identity ack {
  base tcp-flags;
  description
    "Identity for 'acknowledgement' TCP flag";
  reference
    "RFC 793: Transmission Control Protocol - Flags";
}

identity psh {
  base tcp-flags;
  description
    "Identity for 'Push' TCP flag";
```

```
  reference
    "RFC 793: Transmission Control Protocol - Flags";
}

identity rst {
  base tcp-flags;
  description
    "Identity for 'Reset' TCP flag";
  reference
    "RFC 793: Transmission Control Protocol - Flags";
}

identity syn {
  base tcp-flags;
  description
    "Identity for 'Synchronize' TCP flag";
  reference
    "RFC 793: Transmission Control Protocol - Flags";
}

identity fin {
  base tcp-flags;
  description
    "Identity for 'Finish' TCP flag";
  reference
    "RFC 793: Transmission Control Protocol - Flags";
}

identity target-device {
  description
    "Base identity for target devices";
  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model";
}

identity computer {
  base target-device;
  description
    "Identity for computer such as personal computer (PC)
     and server";
}

identity mobile-phone {
  base target-device;
  description
    "Identity for mobile-phone such as smartphone and
     cellphone";
}

identity voip-volte-phone {
  base target-device;
  description
    "Identity for voip-volte-phone";
}

identity tablet {
  base target-device;
```

```
  description
    "Identity for tablet";
}

identity network-infrastructure-device {
  base target-device;
  description
    "Identity for network infrastructure devices
     such as switch, router, and access point";
}

identity iot-device {
  base target-device;
  description
    "Identity for IoT (Internet of Things) devices";
}

identity ot {
  base target-device;
  description
    "Identity for Operational Technology";
}

identity vehicle {
  base target-device;
  description
    "Identity for vehicle that connects to and shares
     data through the Internet";
}

identity advanced-nsf {
  description
    "Base identity for advanced Network Security Function (NSF)
     capability.  This can be used for advanced NSFs such as
     Anti-DDoS Attack, IPS, URL-Filtering, Antivirus,
     and VoIP/VoLTE Filter.";
  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model";
}

identity content-security-control {
  base advanced-nsf;
  description
    "Base identity for content security control";
  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model";
}

identity ips {
  base content-security-control;
  description
    "Identity for IPS (Intrusion Prevention System)
     that prevents malicious activity within a network";
}

identity url-filtering {
```

```
  base content-security-control;
  description
    "Identity for url filtering that limits access by comparing the
     web traffic's URL with the URLs for web filtering in a
     database";
}

identity anti-virus {
  base content-security-control;
  description
    "Identity for antivirus to protect the network by detecting and
     removing viruses or malwares.";
}

identity voip-volte-filter {
  base content-security-control;
  description
    "Identity for VoIP/VoLTE security service that filters out the
     packets or flows of malicious users with a deny list of
     malicious users in a database";
}

identity attack-mitigation-control {
  base advanced-nsf;
  description
    "Base identity for attack mitigation control";
  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model";
}

identity anti-ddos {
  base attack-mitigation-control;
  description
    "Identity for advanced NSF Anti-DDoS or DDoS Mitigator
     capability.";
}

identity ingress-action {
  description
    "Base identity for action";
  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Ingress Action";
}

identity egress-action {
  description
    "Base identity for egress action";
  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Egress Action";
}

identity default-action {
  description
    "Base identity for default action";
  reference
```

```
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Default Action";
}

identity pass {
  base ingress-action;
  base egress-action;
  base default-action;
  description
    "Identity for pass";
  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Actions and
     Default Action";
}

identity drop {
  base ingress-action;
  base egress-action;
  base default-action;
  description
    "Identity for drop";
  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Actions and
     Default Action";
}

identity mirror {
  base ingress-action;
  base egress-action;
  base default-action;
  description
    "Identity for mirror";
  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Actions and
     Default Action";
}

identity rate-limit {
  base ingress-action;
  base egress-action;
  base default-action;
  description
    "Identity for rate limiting action";
  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Actions and
     Default Action";
}

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

identity rule-log {
```

```
  base log-action;
  description
    "Identity for rule log";
}

identity session-log {
  base log-action;
  description
    "Identity for session log";
}

identity invoke-signaling {
  base egress-action;
  description
    "Identity for invoke signaling";
}

identity tunnel-encapsulation {
  base egress-action;
  description
    "Identity for tunnel encapsulation";
}

identity forwarding {
  base egress-action;
  description
    "Identity for forwarding";
}

identity transformation {
  base egress-action;
  description
    "Identity for transformation";
}

identity redirection {
  base egress-action;
  description
    "Identity for redirection";
}

identity resolution-strategy {
  description
    "Base identity for resolution strategy";
  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Resolution Strategy";
}

identity fmr {
  base resolution-strategy;
  description
    "Identity for First Matching Rule (FMR)";
  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Resolution Strategy";
}
```

```
identity lmr {
  base resolution-strategy;
  description
    "Identity for Last Matching Rule (LMR)";
  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Resolution Strategy";
}

identity pmr {
  base resolution-strategy;
  description
    "Identity for Prioritized Matching Rule (PMR)";
  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Resolution Strategy";
}

identity pmre {
  base resolution-strategy;
  description
    "Identity for Prioritized Matching Rule
    with Errors (PMRE)";
  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Resolution Strategy";
}

identity pmrn {
  base resolution-strategy;
  description
    "Identity for Prioritized Matching Rule
    with No Errors (PMRN)";
  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Resolution Strategy";
}

identity day {
  description
    "This represents the base for days.";
}

identity monday {
  base day;
  description
    "This represents Monday.";
}

identity tuesday {
  base day;
  description
    "This represents Tuesday.";
}

identity wednesday {
  base day;
  description
```

```
    "This represents Wednesday.";
}

identity thursday {
  base day;
  description
    "This represents Thursday.";
}

identity friday {
  base day;
  description
    "This represents Friday.";
}

identity saturday {
  base day;
  description
    "This represents Saturday.";
}

identity sunday {
  base day;
  description
    "This represents Sunday.";
}

/*
 * Typedefs
 */

typedef time {
  type string {
    pattern '(0[0-9]|1[0-9]|2[0-3]):[0-5][0-9]:[0-5][0-9](\.\d+)?'
      + '(Z|[\+\-]((1[0-3]|0[0-9]):([0-5][0-9])|14:00))?';
  }
  description
    "The time type represents an instance of time of zero-duration
     that recurs every day.";
}

/*
 * Groupings
 */

grouping ipv4-prefix {
  description
    "The list of IPv4 addresses.";
  leaf ipv4 {
    type inet:ipv4-address;
    description
      "The value of IPv4 address.";
  }
  choice subnet {
    description
      "The subnet can be specified as a prefix length or
      netmask.";
    leaf prefix-length {
```

```
      type uint8 {
        range "0..32";
      }
      description
        "The length of the subnet prefix.";
    }
    leaf netmask {
      type yang:dotted-quad;
      description
        "The subnet specified as a netmask.";
    }
  }
  reference
    "RFC  791: Internet Protocol - IPv4 address
     RFC 8344: A YANG Data Model for IP Management";
}

grouping ipv6-prefix {
  description
    "The list of IPv6 addresses.";
  leaf ipv6 {
    type inet:ipv6-address;
    description
      "The value of IPv6 address.";
  }
  leaf prefix-length {
    type uint8 {
      range "0..128";
    }
    description
      "The length of the subnet prefix.";
  }
  reference
    "RFC 8200: Internet Protocol, Version 6 (IPv6)
     Specification - IPv6 address
     RFC 8344: A YANG Data Model for IP Management";
}

grouping ipv4-range {
  description
    "Range match for the IPv4 addresses.  If only one value is
     needed, then set both start and end to the same value.
     The end IPv4 address MUST be equal or greater than the
     start IPv4 address.";
  leaf start {
    type inet:ipv4-address;
    description
      "Starting IPv4 address for a range match.";
  }
  leaf end {
    type inet:ipv4-address;
    description
      "Ending IPv4 address for a range match.";
  }
  reference
    "RFC 791: Internet Protocol - IPv4 address";
}
```

```
grouping ipv6-range {
  description
    "Range match for the IPv6 addresses.  If only one value is
     needed, then set both start and end to the same value.
     The end IPv6 address number MUST be equal to or greater than
     the start IPv6 address.";
  leaf start {
    type inet:ipv6-address;
    description
      "Starting IPv6 address for a range match.";
  }

  leaf end {
    type inet:ipv6-address;
    description
      "Ending IPv6 address for a range match.";
  }
  reference
    "RFC 8200: Internet Protocol, Version 6 (IPv6)
     Specification - IPv6 address";
}

grouping ipv4-address {
  description
    "Grouping for IPv4 address. IPv4 address can be in the form of
     prefix or range.";
  choice match-type {
    description
      "Choose between Prefix or Range";
    case prefix {
      list ipv4-prefix {
        key "ipv4";
        uses ipv4-prefix;
        description
          "The list of IPv4 addresses specified with an
           IPv4 address and a prefix-length or
           a netmask.";
      }
    }
    case range {
      list ipv4-range {
        key "start end";
        uses ipv4-range;
        description
          "The list of IPv4 address specified with a
           start IPv4 address and an end IPv4 address.
           If only one value is needed, then set both
           start and end to the same value.";
      }
    }
  }
}

grouping ipv6-address {
  description
    "Grouping for IPv6 address. IPv6 address can be in the form of
     prefix or range.";
  choice match-type {
```

```
    description
      "Choose between Prefix or Range";
    case prefix {
      list ipv6-prefix {
        key "ipv6";
        uses ipv6-prefix;
        description
          "The list of IPv6 addresses specified with an
           IPv6 address and a prefix-length.";
      }
    }
    case range {
      list ipv6-range {
        key "start end";
        uses ipv6-range;
        description
          "The list of IPv6 address specified with a
           start IPv6 address and an end IPv6 address.
           If only one value is needed, then set both
           start and end to the same value.";
      }
    }
  }
}

grouping port-range {
  leaf start {
    type inet:port-number;
    description
      "Starting port number for a range match.";
  }
  leaf end {
    type inet:port-number;
    must '. >= ../start' {
      error-message
        "The end port number MUST be equal to or greater than the
         start port number.";
    }
    description
      "Ending port number for a range match.";
  }
  description
    "Range match for the port numbers. If only one value is needed,
     then set both start and end to the same value.";
  reference
    "RFC 793: Transmission Control Protocol - Port number
     RFC 768: User Datagram Protocol - Port Number
     RFC 4960: Stream Control Transmission Protocol - Port number
     RFC 4340: Datagram Congestion Control Protocol (DCCP)
     - Port number";
}

/*
 * Data nodes
 */

list i2nsf-security-policy {
```

```
key "system-policy-name";

description
  "Container for security policy
   including a set of security rules according to certain logic,
   i.e., their similarity or mutual relations, etc. The network
   security policy can be applied to both the unidirectional
   and bidirectional traffic across the NSF.
   The I2NSF security policies use the Event-Condition-Action
   (ECA) policy model ";

reference
  "RFC 8329: Framework for Interface to Network Security
   Functions - I2NSF Flow Security Policy Structure
   draft-ietf-i2nsf-capability-data-model-17:
   I2NSF Capability YANG Data Model - Design Principles and
   ECA Policy Model Overview";

leaf system-policy-name {
  type string;
  description
    "The name of the policy.
     This must be unique.";
}

leaf priority-usage {
  type identityref {
    base priority-usage;
  }
  default priority-by-order;
  description
    "Priority usage type for security policy rule:
     priority by order and priority by number";
}

leaf resolution-strategy {
  type identityref {
    base resolution-strategy;
  }
  default fmr;
  description
    "The resolution strategies that can be used to
     specify how to resolve conflicts that occur between
     actions of the same or different policy rules that
     are matched and contained in this particular NSF";

  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Resolution strategy";
}

leaf default-action {
  type identityref {
    base default-action;
  }
  default mirror;
  description
    "This default action can be used to specify a predefined
```

```
     action when no other alternative action was matched
     by the currently executing I2NSF Policy Rule. An analogy
     is the use of a default statement in a C switch statement.";
   reference
     "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Default Action";
 }

 list rules {
   key "rule-name";
   description
     "This is a rule for network security functions.";

   leaf rule-name {
     type string;
     description
       "The name of the rule.";
   }

   leaf rule-description {
     type string;
     description
       "This description gives more information about
        rules.";
   }

   leaf rule-priority {
     type uint8 {
       range "1..255";
     }
     description
       "The priority keyword comes with a mandatory
        numeric value which can range from 1 till 255.
        Note that a higher number means a higher priority";
   }

   leaf rule-enable {
     type boolean;
     description
       "True is enable.
        False is not enable.";
   }

   leaf session-aging-time {
     type uint16;
     units "second";
     description
       "This is session aging time.";
   }

   container long-connection {
     description
       "This is long-connection";

     leaf enable {
       type boolean;
       description
         "True is enable.
```

```
        False is not enable.";
    }

    leaf duration {
      type uint16;
      description
        "This is the duration of the long-connection.";
    }
  }

  container event {
    description
      "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 policy rules for
       a flow-based NSF, it is used to determine whether the
       Condition clause of the 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 any ACL.).";

    reference
      "RFC 8329: Framework for Interface to Network Security
       Functions - I2NSF Flow Security Policy Structure
       draft-ietf-i2nsf-capability-data-model-17:
       I2NSF Capability YANG Data Model - Design Principles and
       ECA Policy Model Overview
       draft-ietf-i2nsf-nsf-monitoring-data-model-08: I2NSF
       NSF Monitoring YANG Data Model - Alarms, Events, Logs,
       and Counters";

    leaf event-clause-description {
      type string;
      description
        "Description for an event clause";
    }

    container time {
      description
        "Time to determine when the policy should be applied";
      leaf start-date-time {
        type yang:date-and-time;
        description
          "This is the start date and time for a security policy
           rule.";
      }

      leaf end-date-time {
        type yang:date-and-time;
        description
          "This is the end date and time for a policy rule.  The
           policy rule will stop working after the specified
           end-date-time.";
      }

      container period{
        when
```

```
    "../frequency!='only-once'";
  description
    "This represents the repetition time.  In the case
     where the frequency is weekly, the days can be set.";
  leaf start-time {
    type time;
    description
      "This is a period's start time for an event.";
  }
  leaf end-time {
    type time;
    description
      "This is a period's end time for an event.";
  }
  leaf-list day {
    when
      "../../frequency='weekly'";
    type identityref{
      base day;
    }
    min-elements 1;
    description
      "This represents the repeated day of every week
       (e.g., Monday and Tuesday).  More than one day can
       be specified.";
  }
  leaf-list date {
    when
      "../../frequency='monthly'";
    type int32{
      range "1..31";
    }
    min-elements 1;
    description
      "This represents the repeated date of every month.
       More than one date can be specified.";
  }
  leaf-list month {
    when
      "../../frequency='yearly'";
    type string{
      pattern '\d{2}-\d{2}';
    }
    min-elements 1;
    description
      "This represents the repeated date and month of every
       year.  More than one can be specified.  A pattern
       used here is Month and Date (MM-DD).";
  }
}

leaf frequency {
  type enumeration {
    enum only-once {
      description
        "This represents that the rule is immediately
         enforcedonly once and not repeated.  The policy
         will continuously be active from the start-time
```

```
            to the end-time.";
        }
        enum daily {
          description
            "This represents that the rule is enforced on a
             daily basis.  The policy will be repeated
             daily until the end-date.";
        }
        enum weekly {
          description
            "This represents that the rule is enforced on a
             weekly basis.  The policy will be repeated weekly
             until the end-date.  The repeated days can be
             specified.";
        }
        enum monthly {
          description
            "This represents that the rule is enforced on a
             monthly basis. The policy will be repeated monthly
             until the end-date.";
        }
        enum yearly {
          description
            "This represents that the rule is enforced on
             a yearly basis.  The policy will be repeated
             yearly until the end-date.";
        }
      }
      default only-once;
      description
        "This represents how frequently the rule
         should be enforced.";
    }
  }

  container event-clauses {
    description
      "System Event Clause - either a system event or
      system alarm";
    reference
      "RFC 8329: Framework for Interface to Network Security
       Functions - I2NSF Flow Security Policy Structure
       draft-ietf-i2nsf-capability-data-model-17:
       I2NSF Capability YANG Data Model - Design Principles and
       ECA Policy Model Overview
       draft-ietf-i2nsf-nsf-monitoring-data-model-08: I2NSF
       NSF Monitoring YANG Data Model - Alarms, Events, Logs,
       and Counters";

    leaf-list system-event {
      type identityref {
        base system-event;
      }
      description
        "The security policy rule according to
         system events.";
    }
```

```
    leaf-list system-alarm {
      type identityref {
        base system-alarm;
      }
      description
        "The security policy rule according to
         system alarms.";
    }
  }
}

container condition {
  description
    "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.";
  reference
    "RFC 8329: Framework for Interface to Network Security
     Functions - I2NSF Flow Security Policy Structure
     draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Design Principles and
     ECA Policy Model Overview";

  leaf condition-clause-description {
    type string;
    description
      "Description for a condition clause.";
  }

  container ethernet {
    description
      "The purpose of this container is to represent layer 2
       packet header information to determine the set of policy
       actions in this ECA policy rule should be executed or
       not.";
    reference
      "IEEE 802.3: IEEE Standard for Ethernet";

    leaf ethernet-description {
      type string;
      description
        "The MAC Condition description";
    }

    leaf-list source-address {
      type yang:mac-address;
      description
        "The condition for source Media Access Control (MAC)
         Address of a Layer 2 packet. Multiple source MAC
         Addresses can be given in a single rule.";
      reference
        "IEEE 802.3: IEEE Standard for Ethernet";
```

```
  }

  leaf-list destination-address {
    type yang:mac-address;
    description
      "The condition for destination Media Access Control
       (MAC) Address of a Layer 2 packet. Multiple
       destination MAC Addresses can be given in a
       single rule.";
    reference
      "IEEE 802.3: IEEE Standard for Ethernet";
  }

  leaf-list ether-type {
    type uint16;
    description
      "The condition for matching the 2-octet of IEEE 802.3
       Length/Type field. Can be specified with decimal or
       hexadecimal from 0 through 65535 (0xFFFF)

       A value from 0 through 1500 (0x05DC) specifies the
       number of MAC client data octets contained in the
       subsequent MAC Client Data Field of the basic frame

       A value greater than or equal to 1536 (0x0600)
       specifies that the Length/Type field indicates
       Ethertype of the MAC client protocol";
    reference
      "IEEE 802.3: IEEE Standard for Ethernet";
  }
}

container ipv4 {
  description
    "The purpose of this container is to represent IPv4
     packet header information to determine if the set
     of policy actions in this ECA policy rule should be
     executed or not.";
  reference
    "RFC 791: Internet Protocol";

  leaf description {
    type string;
    description
     "ipv4 condition textual description.";
  }

  list header-length {
    key "start end";
    leaf start{
      type uint8 {
        range "5..15";
      }
      description
        "Starting IPv4 header length for a range match.";
    }

    leaf end {
```

```
    type uint8 {
      range "5..15";
    }
    must '. >= ../start' {
      error-message
        "The end header length MUST be equal to or greater
         than the start header length.";
    }
    description
      "Ending IPv4 header length for a range match.";
  }
  description
    "The security policy rule according to
     IPv4 header length. If only one value is needed, then
     set both start and end to the same value.";
  reference
    "RFC 791: Internet Protocol - Header length";
}

leaf-list dscp {
  type inet:dscp;
  description
    "The security policy rule according to
     IPv4 type of service for DSCP.";
  reference
    "RFC 791: Internet Protocol - Type of service
     RFC 2474: Definition of the Differentiated
     Services Field (DS Field) in the IPv4 and
     IPv6 Headers.";
}

list total-length {
  key "start end";
  leaf start {
    type uint16;
    description
      "Starting IPv4 total length for a range match.";
  }
  leaf end {
    type uint16;
    must '. >= ../start' {
      error-message
        "The end total length MUST be equal to or greater
         than the start total length.";
    }
    description
      "Ending IPv4 total length for a range match.";
  }
  description
    "The security policy rule according to
     IPv4 total length.  If only one value is needed, then
     set both start and end to the same value.";
  reference
    "RFC 791: Internet Protocol - Total length";
}

leaf-list identification {
  type uint16;
```

```
  description
    "The security policy rule according to
     IPv4 identification.";
  reference
    "RFC 791: Internet Protocol - Identification";
}

leaf-list fragment-flags {
  type identityref {
    base fragmentation-flags;
  }
  description
    "The security policy rule according to
     IPv4 fragment flags.";
  reference
    "RFC 791: Internet Protocol - Fragment flags";
}

list fragment-offset {
  key "start end";
  leaf start {
    type uint16 {
      range "0..16383";
    }
    description
      "Starting IPv4 fragment offset for a range match.";
  }
  leaf end {
    type uint16 {
      range "0..16383";
    }
    must '. >= ../start' {
      error-message
        "The end fragment offset MUST be equal or greater
         than the start fragment offset.";
    }
    description
      "Ending IPv4 fragment offset for a range match.";
  }
  description
    "The security policy rule according to
     IPv4 fragment offset.";
  reference
    "RFC 791: Internet Protocol - Fragment offset";
}

list ttl {
  key "start end";
  leaf start {
    type uint8;
    description
      "Starting IPv4 TTL for a range match.";
  }
  leaf end {
    type uint8;
    must '. >= ../start' {
      error-message
        "The end TTL MUST be equal or greater than
```

```
            the start TTL.";
        }
        description
          "Ending IPv4 TTL for a range match.";
      }
      description
        "The security policy rule according to
         IPv4 time-to-live (TTL).  If only one value is needed,
         then set both start and end to the same value.";
      reference
        "RFC 791: Internet Protocol - Time to live";
    }

    leaf-list protocol {
      type uint8;
      description
        "The security policy rule according to
         IPv4 protocol header field.";
      reference
        "RFC 791: Internet Protocol - Protocol
         IANA: Assigned Internet Protocol Numbers";
    }

    container source-address {
      uses ipv4-address;
      description
        "The security policy rule according to
         IPv4 source address.";
      reference
        "RFC 791: Internet Protocol - IPv4 Address";
    }

    container destination-address {
      uses ipv4-address;
      description
        "The security policy rule according to
         IPv4 destination address.";
      reference
        "RFC 791: Internet Protocol - IPv4 Address";
    }

    leaf-list ipopts {
      type identityref {
        base ipopts;
      }
      description
        "The security policy rule according to
         IPv4 options.";
      reference
        "RFC 791: Internet Protocol - Options";
    }
  }

  container ipv6 {
    description
      "The purpose of this container is to represent
       IPv6 packet header information to determine
       if the set of policy actions in this ECA policy
```

```
    rule should be executed or not.";
reference
   "RFC 8200: Internet Protocol, Version 6 (IPv6)
    Specification";

leaf description {
  type string;
  description
   "This is description for ipv6 condition.";
}


leaf-list dscp {
  type inet:dscp;
  description
    "The security policy rule according to
     IPv6 traffic class for DSCP.";
  reference
    "RFC 8200: Internet Protocol, Version 6 (IPv6)
     Specification - Traffic class
     RFC 2474: Definition of the Differentiated
     Services Field (DS Field) in the IPv4 and
     IPv6 Headers.";
}

list flow-label {
  key "start end";
  leaf start {
    type inet:ipv6-flow-label;
    description
      "Starting IPv6 flow label for a range match.";
  }
  leaf end {
    type inet:ipv6-flow-label;
    must '. >= ../start' {
      error-message
        "The end flow label MUST be equal or greater than
         the start flow label.";
    }
    description
      "Ending IPv6 flow label for a range match.";
  }
  description
    "The security policy rule according to
     IPv6 flow label.  If only one value is needed,
     then set both start and end to the same value.";
  reference
    "RFC 8200: Internet Protocol, Version 6 (IPv6)
     Specification - Flow label";
}

list payload-length {
  key "start end";
  leaf start {
    type uint16;
    description
      "Starting IPv6 payload length for a range match.";
  }
```

```
  leaf end {
    type uint16;
    must '. >= ../start' {
      error-message
        "The end payload length MUST be equal or greater
         than the start payload length.";
    }
    description
      "Ending IPv6 payload length for a range match.";
  }
  description
    "The security policy rule according to
     IPv6 payload length. If only one value is needed,
     then set both start and end to the same value.";
  reference
    "RFC 8200: Internet Protocol, Version 6 (IPv6)
     Specification - Payload length";
}

leaf-list next-header {
  type uint8;
  description
    "The security policy rule according to
     IPv6 next header.";
  reference
    "RFC 8200: Internet Protocol, Version 6 (IPv6)
     Specification - Next header
     IANA: Assigned Internet Protocol Numbers";
}

list hop-limit {
  key "start end";
  leaf start {
    type uint8;
    description
      "Start IPv6 hop limit for a range match.";
  }
  leaf end {
    type uint8;
    must '. >= ../start' {
      error-message
        "The end hop limit MUST be equal or greater than
         the start hop limit.";
    }
    description
      "End IPv6 hop limit for a range match.";
  }
  description
    "The security policy rule according to
     IPv6 hop limit. If only one value is needed,
     then set both start and end to the same value.";
  reference
    "RFC 8200: Internet Protocol, Version 6 (IPv6)
     Specification - Hop limit";
}

container source-address {
  uses ipv6-address;
```

```
    description
      "The security policy rule according to
       IPv6 source address.";
    reference
      "RFC 8200: Internet Protocol, Version 6 (IPv6)
       Specification - IPv6 address";
  }

  container destination-address {
    uses ipv6-address;
    description
      "The security policy rule according to
       IPv6 destination address.";
    reference
      "RFC 8200: Internet Protocol, Version 6 (IPv6)
       Specification - IPv6 address";
  }
}

container tcp {
  description
    "The purpose of this container is to represent
     TCP packet header information to determine
     if the set of policy actions in this ECA policy
     rule should be executed or not.";
  reference
    "RFC 793: Transmission Control Protocol";

  leaf description {
    type string;
    description
     "This is description for tcp condition.";
  }

  list source-port-number {
    key "start end";
    uses port-range;
    description
      "The security policy rule according to
       tcp source port number.";
    reference
      "RFC 793: Transmission Control Protocol
       - Port number";
  }

  list destination-port-number {
    key "start end";
    uses port-range;
    description
      "The security policy rule according to
       tcp destination port number.";
    reference
      "RFC 793: Transmission Control Protocol
       - Port number";
  }

  leaf-list flags {
    type identityref {
```

```
      base tcp-flags;
    }
    description
      "The security policy rule according to
       tcp flags.";
    reference
      "RFC 793: Transmission Control Protocol
       - Flags";
  }
}

container udp {
  description
    "The purpose of this container is to represent
     UDP packet header information to determine
     if the set of policy actions in this ECA policy
     rule should be executed or not.";
  reference
    "RFC 768: User Datagram Protocol";

  leaf description {
    type string;
    description
     "This is description for udp condition.";
  }

  container source-port-number {
    uses port-range;
    description
      "The security policy rule according to
       udp source port number.";
    reference
      "RFC 768: User Datagram Protocol - Port Number";
  }

  container destination-port-number {
    uses port-range;
    description
      "The security policy rule according to
       udp destination port number.";
    reference
      "RFC 768: User Datagram Protocol - Port Number";
  }

  list total-length {
    key "start end";
    leaf start {
      type uint32;
      description
        "Start udp total length for a range match.";
    }
    leaf end {
      type uint32;
      must '. >= ../start' {
        error-message
          "The end hop limit MUST be equal or greater than
           the start hop limit.";
      }
```

```
      description
        "End udp total length for a range match.";
    }
    description
      "The security policy rule according to
       udp total length.  If only one value is needed,
       then set both start and end to the same value";
    reference
      "RFC 768: User Datagram Protocol - Total Length";
  }
}

container sctp {
  description
    "The purpose of this container is to represent
     SCTP packet header information to determine
     if the set of policy actions in this ECA policy
     rule should be executed or not.";
  leaf description {
    type string;
    description
      "This is description for sctp condition.";
  }

  container source-port-number {
    uses port-range;
    description
      "The security policy rule according to
       sctp source port number.";
    reference
      "RFC 4960: Stream Control Transmission Protocol
                 - Port number";
  }


  container destination-port-number {
    uses port-range;
    description
      "The security policy rule according to
       sctp destination port number.";
    reference
      "RFC 4960: Stream Control Transmission Protocol
                 - Port Number";
  }

  leaf-list verification-tag {
    type uint32;
    description
      "The security policy rule according to
       udp total length.";
    reference
      "RFC 4960: Stream Control Transmission Protocol
                 - Verification Tag";
  }

  leaf-list chunk-type {
    type uint8;
    description
```

```
      "The security policy rule according to
       sctp chunk type ID Value.";
    reference
      "RFC 4960: Stream Control Transmission Protocol
                 - Chunk Type";
  }
}

container dccp {
  description
    "The purpose of this container is to represent
     DCCP packet header information to determine
     if the set of policy actions in this ECA policy
     rule should be executed or not.";
  leaf description {
    type string;
    description
      "This is description for dccp condition.";
  }

  container source-port-number {
    uses port-range;
    description
      "The security policy rule according to
       dccp source port number.";
    reference
      "RFC 4340: Datagram Congestion Control Protocol (DCCP)
                 - Port number";
  }

  container destination-port-number {
    uses port-range;
    description
      "The security policy rule according to
       dccp destination port number.";
    reference
      "RFC 4340: Datagram Congestion Control Protocol (DCCP)
                 - Port number";
  }

  leaf-list service-code {
    type uint32;
    description
      "The security policy rule according to
       dccp service code.";
    reference
      "RFC 4340: Datagram Congestion Control Protocol (DCCP)
                 - Service Codes
       RFC 5595: The Datagram Congestion Control Protocol
                 (DCCP) Service Codes
       RFC 6335: Internet Assigned Numbers Authority (IANA)
                 Procedures for the Management of the Service
                 Name and Transport Protocol Port Number
                 Registry - Service Code";
  }
}

list icmp {
```

```
key "version";
description
  "The purpose of this container is to represent
   ICMP packet header information to determine
   if the set of policy actions in this ECA policy
   rule should be executed or not.";
reference
  "RFC  792: Internet Control Message Protocol
   RFC 8335: PROBE: A Utility for Probing Interfaces";

leaf description {
  type string;
  description
   "This is description for icmp condition.";
}

leaf version {
  type enumeration {
    enum icmpv4 {
      value "1";
      description
        "The ICMPv4 Protocol as defined in RFC 792";
    }
    enum icmpv6 {
      value "2";
      description
        "The ICMPv6 Protocol as defined in RFC 4443";
    }
  }
  description
    "The ICMP version to be matched. This value
     affected the type and code values.";
  reference
    "RFC  792: Internet Control Message Protocol
     RFC 4443: Internet Control Message Protocol (ICMPv6)
               for the Internet Protocol Version 6 (IPv6)
               Specification";
}

leaf-list type {
  type uint8;
  description
    "The security policy rule according to
     ICMPv4 or ICMPv6 type header field.

     The value of this leaf-list is affected by
     the value of the leaf version.

     If the version value is icmpv4, the type follows
     the IANA ICMP Parameters.

     If the version value is icmpv6, the type follows
     the IANA ICMPv6 Parameters.";
  reference
    "RFC  792: Internet Control Message Protocol
     RFC 4443: Internet Control Message Protocol (ICMPv6)
               for the Internet Protocol Version 6 (IPv6)
               Specification
```

```
        RFC 8335: PROBE: A Utility for Probing Interfaces
        IANA: Internet Control Message Protocol (ICMP)
              Parameters
        IANA: Internet Control Message Protocol version 6
              (ICMPv6) Parameters";
    }

    leaf-list code {
      type uint8;
      description
        "The security policy rule according to
         ICMPv4 or ICMPv6 code header field.

         The value of this leaf-list is affected by
         the value of the leaf version.

         If the version value is icmpv4, the code follows
         the IANA ICMP parameters.

         If the version value is icmpv6, the code follows
         the IANA ICMPv6 parameters.";
      reference
        "RFC  792: Internet Control Message Protocol
         RFC 4443: Internet Control Message Protocol (ICMPv6)
                   for the Internet Protocol Version 6 (IPv6)
                   Specification
         RFC 8335: PROBE: A Utility for Probing Interfaces
         IANA: Internet Control Message Protocol (ICMP)
               Parameters
         IANA: Internet Control Message Protocol version 6
               (ICMPv6) Parameters";
    }
  }

  container url-category {
    description
      "Condition for url category";
    leaf description {
      type string;
      description
        "This is description for the condition of a URL's
         category such as SNS sites, game sites, ecommerce
         sites, company sites, and university sites.";
    }

    leaf-list pre-defined-category {
      type string;
      description
        "This is pre-defined-category.";
    }
    leaf-list user-defined-category {
      type string;
      description
        "This user-defined-category.";
    }
  }

  container voice {
```

```
  description
    "For the VoIP/VoLTE security system, a VoIP/
     VoLTE security system can monitor each
     VoIP/VoLTE flow and manage VoIP/VoLTE
     security rules controlled by a centralized
     server for VoIP/VoLTE security service
     (called VoIP IPS). The VoIP/VoLTE security
     system controls each switch for the
     VoIP/VoLTE call flow management by
     manipulating the rules that can be added,
     deleted, or modified dynamically.";
  reference
    "RFC 3261: SIP: Session Initiation Protocol";

  leaf description {
    type string;
    description
     "This is description for voice condition.";
  }

  leaf-list source-voice-id {
    type string;
    description
      "The security policy rule according to
       a source voice ID for VoIP and VoLTE.";
  }

  leaf-list destination-voice-id {
    type string;
    description
      "The security policy rule according to
       a destination voice ID for VoIP and VoLTE.";
  }

  leaf-list user-agent {
    type string;
    description
      "The security policy rule according to
       an user agent for VoIP and VoLTE.";
  }
}

container ddos {
  description
    "Condition for DDoS attack.";

  leaf description {
    type string;
    description
     "This is description for ddos condition.";
  }

  leaf alert-packet-rate {
    type uint32;
    units "pps";
    description
      "The alert rate of flood detection for
       packets per second (PPS) of an IP address.";
```

```
  }

  leaf alert-flow-rate {
    type uint32;
    description
      "The alert rate of flood detection for
       flows per second of an IP address.";
  }

  leaf alert-byte-rate {
    type uint32;
    units "BPS";
    description
      "The alert rate of flood detection for
       bytes per second of an IP address.";
  }
}

container anti-virus {
  description
    "Condition for antivirus";

  leaf-list profile {
    type string;
    description
      "The security profile for antivirus. This is used to
       update the security profile for improving the
       security. The security profile is used to scan
       the viruses.";
  }

  leaf-list exception-files {
    type string;
    description
      "The type or name of the files to be excluded by the
       anti-virus. This can be used to keep the known
       harmless files.";
  }
}

container payload {
  description
    "Condition for packet payload";
  leaf packet-payload-description {
    type string;
    description
     "This is description for payload condition.";
  }
  leaf-list payload-content {
    type string;
    description
      "This is a condition for packet payload content.";
  }
}

container context {
  description
    "Condition for context";
```

```
leaf context-description {
  type string;
  description
    "This is description for context condition.";
}

container application {
  description
    "Condition for application";
  leaf description {
    type string;
    description
     "This is description for application condition.";
  }
  leaf-list object {
    type string;
    description
      "This is application object.";
  }
  leaf-list group {
    type string;
    description
      "This is application group.";
  }
  leaf-list label {
    type string;
    description
      "This is application label.";
  }
  container category {
    description
      "This is application category";
    list application-category {
      key "name subcategory";
      description
        "This is application category list";

      leaf name {
        type string;
        description
         "This is name for application category.";
      }
      leaf subcategory {
        type string;
        description
         "This is application subcategory.";
      }
    }
  }
}

container target {
  description
    "Condition for target";
  leaf description {
    type string;
    description
      "This is description for target condition.
```

```
      Vendors can write instructions for target condition
      that vendor made";
  }

  leaf-list device {
    type identityref {
      base target-device;
    }
    description
      "The device attribute that can identify a device,
       including the device type (i.e., router, switch,
       pc, ios, or android) and the device's owner as
       well.";
  }
}

container users {
  description
    "Condition for users";
  leaf users-description {
    type string;
    description
      "This is the description for users' condition.";
  }
  list user {
    key "user-id";
    description
      "The user with which the traffic flow is associated
       can be identified by either a user id or user name.
       The user-to-IP address mapping is assumed to be
       provided by the unified user management system via
       network.";
    leaf user-id {
      type uint32;
      description
        "The ID of the user.";
    }
    leaf user-name {
      type string;
      description
        "The name of the user.";
    }
  }
  list group {
    key "group-id";
    description
      "The user group with which the traffic flow is
       associated can be identified by either a group id
       or group name. The group-to-IP address and
       user-to-group mappings are assumed to be provided by
       the unified user management system via network.";
    leaf group-id {
      type uint32;
      description
        "The ID of the group.";
    }
    leaf group-name {
      type string;
```

```
          description
            "The name of the group.";
        }
      }

      leaf security-group {
        type string;
        description
          "security-group.";
      }
    }

    container geography-location {
      description
        "The location which network traffic flow is associated
         with. The region can be the geographical location
         such as country, province, and city,
         as well as the logical network location such as
         IP address, network section, and network domain.";

      leaf description {
        type string;
        description
          "This is description for generic context condition.
           Vendors can write instructions for generic context
           condition that vendor made";
      }

      leaf-list source {
        type string;
        description
          "The src-geography-location is a geographical
           location mapped into an IP address. It matches the
           mapped IP address to the source IP address of the
           traffic flow.";
        reference
          "ISO 3166: Codes for the representation of
           names of countries and their subdivisions";
      }

      leaf-list destination {
        type string;
        description
          "The dest-geography-location is a geographical
           location mapped into an IP address. It matches the
           mapped IP address to the destination IP address of
           the traffic flow.";
        reference
          "ISO 3166: Codes for the representation of
           names of countries and their subdivisions";
      }
    }
  }
}

container action {
  description
    "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.";
reference
  "RFC 8329: Framework for Interface to Network Security
   Functions - I2NSF Flow Security Policy Structure
   draft-ietf-i2nsf-capability-data-model-17:
   I2NSF Capability YANG Data Model - Design Principles and
   ECA Policy Model Overview";

leaf action-clause-description {
  type string;
  description
    "Description for an action clause.";
}

container packet-action {
  description
    "Action for packets";
  reference
    "RFC 8329: Framework for Interface to Network Security
     Functions - I2NSF Flow Security Policy Structure
     draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Design Principles and
     ECA Policy Model Overview";

  leaf ingress-action {
    type identityref {
      base ingress-action;
    }
    description
      "Ingress Action: pass, drop, rate-limit, and
       mirror.";
  }

  leaf egress-action {
    type identityref {
      base egress-action;
    }
    description
      "Egress action: pass, drop, rate-limit, mirror,
       invoke-signaling, tunnel-encapsulation, forwarding,
       and redirection.";
  }

  leaf log-action {
    type identityref {
      base log-action;
    }
    description
      "Log action: rule log and session log";
  }

}
```

```
container flow-action {
  description
    "Action for flows";
  reference
    "RFC 8329: Framework for Interface to Network Security
     Functions - I2NSF Flow Security Policy Structure
     draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - Design Principles and
     ECA Policy Model Overview";

  leaf ingress-action {
    type identityref {
      base ingress-action;
    }
    description
      "Action: pass, drop, rate-limit, and mirror.";
  }

  leaf egress-action {
    type identityref {
      base egress-action;
    }
    description
      "Egress action: pass, drop, rate-limit, mirror,
       invoke-signaling, tunnel-encapsulation, forwarding,
       and redirection.";
  }

  leaf log-action {
    type identityref {
      base log-action;
    }
    description
      "Log action: rule log and session log";
  }
}

container advanced-action {
  description
    "If the packet needs to be additionally inspected,
     the packet is passed to advanced network
     security functions according to the profile.
     The profile means the types of NSFs where the packet
     will be forwarded in order to additionally
     inspect the packet.
     The advanced action activates Service Function
     Chaining (SFC) for further inspection of a packet.";
  reference
    "draft-ietf-i2nsf-capability-data-model-17:
     I2NSF Capability YANG Data Model - YANG Tree
     Diagram";

  leaf-list content-security-control {
    type identityref {
      base content-security-control;
    }
    description
      "Content-security-control is the NSFs that
```

```
            inspect the payload of the packet.
            The profile for the types of NSFs for mitigation is
            divided into content security control and
            attack-mitigation-control.
            Content security control: ips, url filtering,
            anti-virus, and voip-volte-filter. This can be
            extended according to the provided NSFs.";
          reference
            "draft-ietf-i2nsf-capability-data-model-17:
             I2NSF Capability YANG Data Model - YANG Tree Diagram";
        }

        leaf-list attack-mitigation-control {
          type identityref {
            base attack-mitigation-control;
          }
          description
            "Attack-mitigation-control is the NSFs that weaken
             the attacks related to a denial of service
             and reconnaissance.
             The profile for the types of NSFs for mitigation is
             divided into content security control and
             attack-mitigation-control.
             Attack mitigation control: Anti-DDoS or DDoS
             mitigator. This can be extended according to the
             provided NSFs such as mitigators for ip sweep,
             port scanning, ping of death, teardrop, oversized
             icmp, and tracert.";
          reference
            "draft-ietf-i2nsf-capability-data-model-17:
             I2NSF Capability YANG Data Model - YANG Tree Diagram";
        }
      }
    }
  }
  container rule-group {
    description
      "This is rule group";

    list groups {
      key "group-name";
      description
        "This is a group for rules";

      leaf group-name {
        type string;
        description
          "This is a group for rules";
      }

      container rule-range {
        description
          "This is a rule range.";

        leaf start-rule {
          type string;
          description
            "This is a start rule";
```

```
          }
          leaf end-rule {
            type string;
            description
              "This is a end rule";
          }
        }
        leaf enable {
          type boolean;
          description
            "This is enable
             False is not enable.";
        }
        leaf description {
          type string;
          description
            "This is a description for rule-group";
        }
      }
    }
  }
}

<CODE ENDS>
```

Figure 5: YANG Data Module of I2NSF NSF-Facing-Interface

5. XML Configuration Examples of Low-Level Security Policy Rules

This section shows XML configuration examples of low-level security policy rules that are delivered from the Security Controller to NSFs over the NSF-Facing Interface. For security requirements, we assume that the NSFs (i.e., General firewall, Time-based firewall, URL filter, VoIP/VoLTE filter, and http and https flood mitigation) described in of [I-D.ietf-i2nsf-capability-data-model] are registered in the I2NSF framework. With the registered NSFs, we show configuration examples for security policy rules of network security functions according to the following three security requirements: (i) Block Social Networking Service (SNS) access during business hours, (ii) Block malicious VoIP/VoLTE packets coming to the company, and (iii) Mitigate http and https flood attacks on company web server.

5.1. Security Requirement 1: Block Social Networking Service (SNS) Access during Business Hours

This section shows a configuration example for blocking SNS access during business hours in IPv4 networks or IPv6 networks.

```
<i2nsf-security-policy
xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-policy-rule-for-nsf">
 <system-policy-name>sns_access</system-policy-name>
 <rules>
  <rule-name>block_sns_access_during_operation_time</rule-name>
  <event>
    <time>
      <start-date-time>2021-03-11T09:00:00.00Z</start-date-time>
      <end-date-time>2021-12-31T18:00:00.00Z</end-date-time>
      <period>
        <start-time>09:00:00Z</start-time>
        <end-time>18:00:00Z</end-time>
        <day>monday</day>
        <day>tuesday</day>
        <day>wednesday</day>
        <day>thursday</day>
        <day>friday</day>
      </period>
    </time>
    <frequency>weekly</frequency>
  </event>
  <condition>
   <ipv4>
    <source-address>
     <ipv4-range>
      <start>192.0.2.11</start>
      <end>192.0.2.90</end>
     </ipv4-range>
    </source-address>
   </ipv4>
  </condition>
  <action>
   <advanced-action>
    <content-security-control>
      url-filtering
    </content-security-control>
   </advanced-action>
  </action>
 </rules>
</i2nsf-security-policy>
```

Figure 6: Configuration XML for Time-based Firewall to Block SNS Access during Business Hours in IPv4 Networks

```
<i2nsf-security-policy
xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-policy-rule-for-nsf">
 <system-policy-name>sns_access</system-policy-name>
 <rules>
  <rule-name>block_sns_access_during_operation_time</rule-name>
  <event>
    <time>
      <start-date-time>2021-03-11T09:00:00.00Z</start-date-time>
      <end-date-time>2021-12-31T18:00:00.00Z</end-date-time>
      <period>
        <start-time>09:00:00Z</start-time>
        <end-time>18:00:00Z</end-time>
        <day>monday</day>
        <day>tuesday</day>
        <day>wednesday</day>
        <day>thursday</day>
        <day>friday</day>
      </period>
    </time>
    <frequency>weekly</frequency>
  </event>
  <condition>
   <ipv6>
    <source-address>
     <ipv6-range>
      <start>2001:DB8:0:1::11</start>
      <end>2001:DB8:0:1::90</end>
     </ipv6-range>
    </source-address>
   </ipv6>
  </condition>
  <action>
   <advanced-action>
    <content-security-control>
      url-filtering
    </content-security-control>
   </advanced-action>
  </action>
 </rules>
</i2nsf-security-policy>
```

Figure 7: Configuration XML for Time-based Firewall to Block SNS Access during Business Hours in IPv6 Networks

```
<i2nsf-security-policy
xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-policy-rule-for-nsf">
 <system-policy-name>sns_access</system-policy-name>
 <rules>
  <rule-name>block_sns_access_during_operation_time</rule-name>
  <condition>
   <url-category>
    <user-defined>SNS_1</user-defined>
    <user-defined>SNS_2</user-defined>
   </url-category>
  </condition-clause-container>
  <action-clause-container>
   <packet-action>
    <egress-action>drop</egress-action>
   </packet-action>
  </action-clause-container>
 </rules>
</i2nsf-security-policy>
```

Figure 8: Configuration XML for Web Filter to Block SNS Access during Business Hours

Figure 6 (or Figure 7) and Figure 8 show the configuration XML documents for time-based firewall and web filter to block SNS access during business hours in IPv4 networks (or IPv6 networks). For the security requirement, two NSFs (i.e., a time-based firewall and a web filter) were used because one NSF cannot meet the security requirement. The instances of XML documents for the time-based firewall and the web filter are as follows: Note that a detailed data model for the configuration of the advanced network security function (i.e., web filter) can be defined as an extension in future.

Time-based Firewall is as follows:

1. The name of the system policy is sns_access.

2. The name of the rule is block_sns_access_during_operation_time.

3. The rule is started from 2021-03-11 at 9 a.m. to 2021-12-31 at 6 p.m.

4. The rule is operated weekly every weekday (i.e., Monday, Tuesday, Wednesday, Thursday, and Friday) during the business hours (i.e., from 9 a.m. to 6 p.m.) .

5. The rule inspects a source IPv4 address (i.e., from 192.0.2.11 to 192.0.2.90) to inspect the outgoing packets of employees. For the case of IPv6 networks, the rule inspects a source IPv6 address (i.e., from 2001:DB8:0:1::11 to 2001:DB8:0:1::90) to inspect the outgoing packets of employees.

6. If the outgoing packets match the rules above, the time-based firewall sends the packets to url filtering for additional inspection because the time-based firewall can not inspect contents of the packets for the SNS URL.

Web Filter is as follows:

1. The name of the system policy is sns_access.

2. The name of the rule is block_SNS_1_and_SNS_2.

3. The rule inspects URL address to block the access packets to the SNS_1 or the SNS_2.

4. If the outgoing packets match the rules above, the packets are blocked.

5.2. Security Requirement 2: Block Malicious VoIP/VoLTE Packets Coming to a Company

This section shows a configuration example for blocking malicious VoIP/VoLTE packets coming to a company.

```
<i2nsf-security-policy
xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-policy-rule-for-nsf">
 <system-policy-name>voip_volte_inspection</system-policy-name>
 <rules>
  <rule-name>block_malicious_voice_id</rule-name>
  <condition>
   <ipv4>
    <destination-address>
     <ipv4-range>
      <start>192.0.2.11</start>
      <end>192.0.2.90</end>
     </ipv4-range>
    </destination-address>
   </ipv4>
   <tcp>
    <destination-port-number>
     <start>5060</start>
     <start>5061</end>
    </destination-port-number>
   </tcp>
  </condition>
  <action>
   <advanced-action>
    <content-security-control>
      voip-volte-filter
    </content-security-control>
   </advanced-action>
  </action>
 </rules>
</i2nsf-security-policy>
```

Figure 9: Configuration XML for General Firewall to Block Malicious VoIP/VoLTE Packets Coming to a Company

```
<i2nsf-security-policy
xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-policy-rule-for-nsf">
 <system-policy-name>voip_volte_inspection</system-policy-name>
 <rules>
  <rule-name>block_malicious_voice_id</rule-name>
  <condition>
   <voice>
    <source-voice-id>
      user1@voip.malicious.example.com
    </source-voice-id>
    <source-voice-id>
      user2@voip.malicious.example.com
    </source-voice-id>
   </voice>
  </condition>
  <action>
   <flow-action>
    <ingress-action>drop</ingress-action>
   </flow-action>
  </action>
 </rules>
</i2nsf-security-policy>
```

Figure 10: Configuration XML for VoIP/VoLTE Filter to Block Malicious VoIP/VoLTE Packets Coming to a Company

Figure 9 and Figure 10 show the configuration XML documents for general firewall and VoIP/VoLTE filter to block malicious VoIP/VoLTE packets coming to a company. For the security requirement, two NSFs (i.e., a general firewall and a VoIP/VoLTE filter) were used because one NSF can not meet the security requirement. The instances of XML documents for the general firewall and the VoIP/VoLTE filter are as follows: Note that a detailed data model for the configuration of the advanced network security function (i.e., VoIP/VoLTE filter) can be described as an extension in future.

General Firewall is as follows:

1. The name of the system policy is voip_volte_inspection.

2. The name of the rule is block_malicious_voip_volte_packets.

3. The rule inspects a destination IPv4 address (i.e., from 192.0.2.11 to 192.0.2.90) to inspect the packets coming into the company.

4. The rule inspects a port number (i.e., 5060 and 5061) to inspect VoIP/VoLTE packet.

5. If the incoming packets match the rules above, the general firewall sends the packets to VoIP/VoLTE filter for additional inspection because the general firewall can not inspect contents of the VoIP/VoLTE packets.

VoIP/VoLTE Filter is as follows:

1. The name of the system policy is malicious_voice_id.

2. The name of the rule is block_malicious_voice_id.

3. The rule inspects the voice id of the VoIP/VoLTE packets to block the malicious VoIP/VoLTE packets (i.e., user1@voip.malicious.example.com and user2@voip.malicious.example.com).

4. If the incoming packets match the rules above, the packets are blocked.

5.3. Security Requirement 3: Mitigate HTTP and HTTPS Flood Attacks on a Company Web Server

This section shows a configuration example for mitigating http and https flood attacks on a company web server.

```
<i2nsf-security-policy
xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-policy-rule-for-nsf">
 <system-policy-name>flood_attack_mitigation</system-policy-name>
 <rules>
  <rule-name>mitigate_http_and_https_flood_attack</rule-name>
  <condition>
   <ipv4>
    <destination-address>
     <ipv4-range>
      <start>192.0.2.11</start>
      <end>192.0.2.11</end>
     </ipv4-range>
    </destination-address>
   </ipv4>
   <tcp>
    <destination-port-number>
     <start>80</start>
     <end>80</end>
    </destination-port>
    <destination-port-number>
     <start>443</start>
     <end>443</end>
    </destination-port>
   </tcp>
  </condition>
  <action>
   <advanced-action>
    <attack-mitigation-control>
      anti-ddos
    </attack-mitigation-control>
   </advanced-action>
  </action>
 </rules>
</i2nsf-security-policy>
```

Figure 11: Configuration XML for General Firewall to Mitigate HTTP and HTTPS Flood Attacks on a Company Web Server

```
<i2nsf-security-policy
xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-policy-rule-for-nsf">
 <system-policy-name>flood_attack_mitigation</system-policy-name>
 <rules>
  <rule-name>mitigate_http_and_https_flood_attack</rule-name>
  <condition>
   <ddos>
    <alert-packet-rate>1000</alert-packet-rate>
   </ddos>
  </condition>
  <action>
   <flow-action>
    <ingress-action>drop</ingress-action>
   </flow-action>
  </action>
 </rules>
</i2nsf-security-policy>
```

Figure 12: Configuration XML for Anti-DDoS to Mitigate HTTP and HTTPS Flood Attacks on a Company Web Server

Figure 11 and Figure 12 show the configuration XML documents for general firewall and http and https flood attack mitigation to mitigate http and https flood attacks on a company web server. For the security requirement, two NSFs (i.e., a general firewall and a http and https flood attack mitigation) were used because one NSF can not meet the security requirement. The instances of XML documents for the general firewall and http and https flood attack mitigation are as follows: Note that a detailed data model for the configuration of the advanced network security function (i.e., http and https flood attack mitigation) can be defined as an extension in future.

General Firewall is as follows:

1. The name of the system policy is flood_attack_mitigation.

2. The name of the rule is mitigate_http_and_https_flood_attack.

3. The rule inspects a destination IPv4 address (i.e., 192.0.2.11) to inspect the access packets coming into the company web server.

4. The rule inspects a port number (i.e., 80 and 443) to inspect http and https packet.

5. If the packets match the rules above, the general firewall sends the packets to anti-DDoS for additional inspection because the general firewall can not control the amount of packets for http and https packets.

Anti DDoS for HTTP and HTTPS Flood Attack Mitigation is as follows:

1. The name of the system policy is flood_attack_mitigation.

2. The name of the rule is mitigate_http_and_https_flood_attack.

3. The rule controls the http and https packets according to the amount of incoming packets (1000 packets per second).

4. If the incoming packets match the rules above, the packets are blocked.

6. IANA Considerations

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

URI: urn:ietf:params:xml:ns:yang:ietf-i2nsf-policy-rule-for-nsf
Registrant Contact: The IESG.
XML: N/A; the requested URI is an XML namespace.

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

name: ietf-i2nsf-policy-rule-for-nsf
namespace: urn:ietf:params:xml:ns:yang:ietf-i2nsf-policy-rule-for-nsf
prefix: nsfintf
reference: RFC XXXX

7. Security Considerations

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

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

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

*ietf-i2nsf-policy-rule-for-nsf: Writing to almost any element of this YANG module would directly impact on the configuration of NSFs, e.g., completely turning off security monitoring and mitigation capabilities; altering the scope of this monitoring and mitigation; creating an overwhelming logging volume to overwhelm downstream analytics or storage capacity; creating logging patterns which are confusing; or rendering useless trained statistics or artificial intelligence models.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It

is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

*ietf-i2nsf-policy-rule-for-nsf: The attacker may gather the security policy information of any target NSFs and misuse the security policy information for subsequent attacks.

Policy rules identifying the specified users and user groups can be specified with "rules/condition/context/users". As with other data in this YANG module, this user information is provided by the Security Controller to the NSFs and is protected via the transport and access control mechanisms described above.

8. 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 (2020-0-00395, Standard Development of Blockchain based Network Management Automation Technology).

9. Contributors

This document is made by the group effort of I2NSF working group. Many people actively contributed to this document, such as Acee Lindem and Roman Danyliw. 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

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

Liang Xia Huawei 101 Software Avenue Nanjing, Jiangsu 210012 China EMail: Frank.Xialiang@huawei.com

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

10. References

10.1. Normative References

[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>.

[RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, DOI 10.17487/RFC2474, December 1998, <https://www.rfc-editor.org/info/rfc2474>.

[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>.

[RFC4340] Kohler, E., Handley, M., and S. Floyd, "Datagram Congestion Control Protocol (DCCP)", RFC 4340, DOI 10.17487/RFC4340, March 2006, <https://www.rfc-editor.org/info/rfc4340>.

[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>.

[RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, DOI 10.17487/RFC6335, August 2011, <https://www.rfc-editor.org/info/rfc6335>.

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

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

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

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

[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>.

[RFC8335] Bonica, R., Thomas, R., Linkova, J., Lenart, C., and M. Boucadair, "PROBE: A Utility for Probing Interfaces", RFC 8335, DOI 10.17487/RFC8335, February 2018, <https://www.rfc-editor.org/info/rfc8335>.

[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>.

[RFC8344] Bjorklund, M., "A YANG Data Model for IP Management", RFC 8344, DOI 10.17487/RFC8344, March 2018, <https://www.rfc-editor.org/info/rfc8344>.

[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>.

[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>.

[I-D.ietf-i2nsf-capability-data-model] Hares, S., Jeong, J. (., Kim, J. (., Moskowitz, R., and Q. Lin, "I2NSF Capability YANG Data Model", Work in Progress, Internet-Draft, draft-ietf-i2nsf-capability-data-model-17, 14 August 2021, <https://www.ietf.org/archive/id/draft-ietf-i2nsf-capability-data-model-17.txt>.

[I-D.ietf-i2nsf-nsf-monitoring-data-model] Jeong, J. (., Lingga, P., Hares, S., Xia, L. (., and H. Birkholz, "I2NSF NSF Monitoring Interface YANG Data Model", Work in Progress, Internet-Draft, draft-ietf-i2nsf-nsf-monitoring-data-model-08, 29 April 2021, <https://www.ietf.org/archive/id/draft-ietf-i2nsf-nsf-monitoring-data-model-08.txt>.

10.2. Informative References

[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>.

[I-D.ietf-i2nsf-consumer-facing-interface-dm] Jeong, J. (., Chung, C., Ahn, T., Kumar, R., and S. Hares, "I2NSF Consumer-Facing Interface YANG Data Model", Work in Progress, Internet-Draft, draft-ietf-i2nsf-consumer-facing-interface-dm-13, 8 March 2021, <https://www.ietf.org/archive/id/draft-ietf-i2nsf-consumer-facing-interface-dm-13.txt>.

[ISO-Country-Codes] "Codes for the representation of names of countries and their subdivisions", ISO 3166, September 2018, <https://www.iso.org/iso-3166-country-codes.html>.

[IANA-Protocol-Numbers] Internet Assigned Numbers Authority (IANA), "Internet Control Message Procotol (ICMP) Parameters", September 2020, <https://www.iana.org/assignments/icmp-parameters/icmp-parameters.xhtml>.

[IANA-ICMP-Parameters] Internet Assigned Numbers Authority (IANA), "Assigned Internet Protocol Numbers", February 2021, <https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml>.

[IEEE-802.3] Institute of Electrical and Electronics Engineers, "IEEE Standard for Ethernet", 2018, <https://ieeexplore.ieee.org/document/8457469/>.

Authors' Addresses

Jinyong (Tim) Kim (editor)
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

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

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

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

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

Susan Hares
Huawei
7453 Hickory Hill
Saline, MI 48176
United States of America

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

Qiushi Lin
Huawei
Huawei Industrial Base
Shenzhen
Guangdong 518129,
China

Email: linqiushi@huawei.com