I2NSF Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 25, 2019

J. Kim
J. Jeong
Sungkyunkwan University
J. Park
ETRI
S. Hares
Q. Lin
Huawei

March 24, 2019

I2NSF Network Security Function-Facing Interface YANG Data Model
draft-ietf-i2nsf-nsf-facing-interface-dm-04

Abstract

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

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 September 25, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in 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 devices. The YANG data model is corresponding to the information model [i2nsf-nsf-cap-im] for Network Security Functions (NSF) facing interface in Interface to Network Security Functions (I2NSF). The YANG data model in this document focuses on security policy configuration for generic network security functions. Note that security policy configuration for advanced network security functions are written in [i2nsf-advanced-nsf-dm].

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 [i2nsf-nsf-cap-im]. Rules.

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

- Configuration for general security policy rule of generic network security function.
- Configuration for an event clause of generic network security function.
- Configuration for a condition clause of generic network security function.
- Configuration for an action clause of generic network security function.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119][RFC8174].

3. Terminology

This document uses the terminology described in [i2nsf-nsf-cap-im][RFC8431][supa-policy-info-model]. Especially, the following terms are from [supa-policy-info-model]:

- Data Model: A data model is a representation of concepts of interest to an environment in a form that is dependent on data repository, data definition language, query language, implementation language, and protocol.
- Information Model: An information model is a representation of concepts of interest to an environment in a form that is independent of data repository, data definition language, query language, implementation language, and protocol.

3.1. Tree Diagrams

A simplified graphical representation of the data model is used in this document. The meaning of the symbols in these diagrams [RFC8340] is as follows:

- Brackets "[" and "]" enclose list keys.

- Abbreviations before data node names: "rw" means configuration (read-write) and "ro" state data (read-only).
- Symbols after data node names: "?" means an optional node and "*" denotes a "list" and "leaf-list".
- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").
- Ellipsis ("...") stands for contents of subtrees that are not shown.

4. YANG Tree Diagram

This section shows an YANG tree diagram of generic network security functions. Note that a detailed data model for the configuration of the advanced network security functions is described in [i2nsf-advanced-nsf-dm]. The section describes the following subjects:

- General I2NSF security policy rule of generic network security function.
- An event clause of generic network security function.
- A condition clause of generic network security function.
- An action clause of generic network security function.

4.1. General I2NSF Security Policy Rule

This section shows YANG tree diagram for general I2NSF security policy rule.

```
module: ietf-i2nsf-policy-rule-for-nsf
  +--rw i2nsf-security-policy
     +--rw system-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 rule-session-aging-time?            uint16
        |  +--rw rule-long-connection
        |  |  +--rw enable?   boolean
        |  |  +--rw during?   uint16
        |  +--rw time-zone
        |  |  +--rw absolute-time-zone
        |  |  |  +--rw start-time?   start-time-type
        |  |  |  +--rw end-time?     end-time-type
        |  |  +--rw periodic-time-zone
        |  |     +--rw day
        |  |     |  +--rw every-day?      boolean
        |  |     |  +--rw specific-day*   day-type
        |  |     +--rw month
        |  |        +--rw every-month?      boolean
        |  |        +--rw specific-month*   month-type
        |  +--rw event-clause-container
        |  |  ...
        |  +--rw condition-clause-container
        |  |  ...
        |  +--rw action-clause-container
        |     ...
        +--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
```

Figure 1: YANG Tree Diagram for Network Security Policy

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

The system policy represents there could be 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, resolutation 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 this 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 [i2nsf-nsf-cap-im].

A default action is used to execute I2NSF policy rule when no rule matches a packet. The default action is defined as pass, drop, reject, alert, and mirror. The default action can be extended according to specific vendor action features. The default action is described in detail in [i2nsf-nsf-cap-im].

The rules include rule name, rule description, rule priority, rule enable, time zone, event clause container, condition clause container, and action clause container.

4.2. Event Clause

This section shows YANG tree diagram for an event clause of I2NSF security policy rule.

```
module: ietf-i2nsf-policy-rule-for-nsf
  +--rw i2nsf-security-policy
     +--rw system-policy* [system-policy-name]
        ...
        +--rw rules* [rule-name]
        |  ...
        |  +--rw event-clause-container
        |  |  +--rw event-clause-description?   string
        |  |  +--rw event-clauses
        |  |     +--rw system-event*   identityref
        |  |     +--rw system-alarm*   identityref
        |  +--rw condition-clause-container
        |  |  ...
        |  +--rw action-clause-container
        |     ...
        +--rw rule-group
           ...
```

Figure 2: YANG Tree Diagram for Network Security Policy

This YANG tree diagram shows an event clause of I2NSF security policy rule for generic network security functions. An event clause is any important occurrence in 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 system event and system alarm. The event clause can be extended according to specific vendor event features. The event clause is described in detail in [i2nsf-nsf-cap-im].

4.3. Condtion Clause

This section shows YANG tree diagram for a condition clause of I2NSF security policy rule.

```
module: ietf-i2nsf-policy-rule-for-nsf
  +--rw i2nsf-security-policy
     ...
        +--rw rules* [rule-name]
           ...
           +--rw event-clause-container
           |  ...
           +--rw condition-clause-container
           |  +--rw condition-clause-description?      string
           |  +--rw packet-security-ipv4-condition
           |  |  +--rw pkt-sec-ipv4-header-length
           |  |  |  +--rw (match-type)?
```

```
|  |  |     +--:(exact-match)
|  |  |     |  +--rw ipv4-header-length*         uint8
|  |  |     +--:(range-match)
|  |  |        +--rw range-ipv4-header-length*
[start-ipv4-header-length end-ipv4-header-length]
|  |  |           +--rw start-ipv4-header-length    uint8
|  |  |           +--rw end-ipv4-header-length      uint8
|  |  +--rw pkt-sec-ipv4-tos*               identityref
|  |  +--rw pkt-sec-ipv4-total-length
|  |  |  +--rw (match-type)?
|  |  |     +--:(exact-match)
|  |  |     |  +--rw ipv4-total-length*         uint16
|  |  |     +--:(range-match)
|  |  |        +--rw range-ipv4-total-length*
[start-ipv4-total-length end-ipv4-total-length]
|  |  |           +--rw start-ipv4-total-length    uint16
|  |  |           +--rw end-ipv4-total-length      uint16
|  |  +--rw pkt-sec-ipv4-id*                uint16
|  |  +--rw pkt-sec-ipv4-fragment-flags*    identityref
|  |  +--rw pkt-sec-ipv4-fragment-offset
|  |  |  +--rw (match-type)?
|  |  |     +--:(exact-match)
|  |  |     |  +--rw ipv4-fragment-offset*         uint16
|  |  |     +--:(range-match)
|  |  |        +--rw range-ipv4-fragment-offset*
[start-ipv4-fragment-offset end-ipv4-fragment-offset]
|  |  |           +--rw start-ipv4-fragment-offset    uint16
|  |  |           +--rw end-ipv4-fragment-offset      uint16
|  |  +--rw pkt-sec-ipv4-ttl
|  |  |  +--rw (match-type)?
|  |  |     +--:(exact-match)
|  |  |     |  +--rw ipv4-ttl*         uint8
|  |  |     +--:(range-match)
|  |  |        +--rw range-ipv4-ttl*
[start-ipv4-ttl end-ipv4-ttl]
|  |  |           +--rw start-ipv4-ttl    uint8
|  |  |           +--rw end-ipv4-ttl      uint8
|  |  +--rw pkt-sec-ipv4-protocol*          identityref
|  |  +--rw pkt-sec-ipv4-src
|  |  |  +--rw (match-type)?
|  |  |     +--:(exact-match)
|  |  |     |  +--rw ipv4-address* [ipv4]
|  |  |     |     +--rw ipv4            inet:ipv4-address
|  |  |     |     +--rw (subnet)?
|  |  |     |        +--:(prefix-length)
|  |  |     |        |  +--rw prefix-length?   uint8
|  |  |     |        +--:(netmask)
|  |  |     |           +--rw netmask? yang:dotted-quad
```

```
|  |  |     +--:(range-match)
|  |  |        +--rw range-ipv4-address*
[start-ipv4-address end-ipv4-address]
|  |  |           +--rw start-ipv4-address  inet:ipv4-address
|  |  |           +--rw end-ipv4-address    inet:ipv4-address
|  |  +--rw pkt-sec-ipv4-dest
|  |  |  +--rw (match-type)?
|  |  |     +--:(exact-match)
|  |  |     |  +--rw ipv4
|  |  |     |     +--rw ipv4-address* [ipv4]
|  |  |     |        +--rw ipv4             inet:ipv4-address
|  |  |     |        +--rw (subnet)?
|  |  |     |           +--:(prefix-length)
|  |  |     |           |  +--rw prefix-length?   uint8
|  |  |     |           +--:(netmask)
|  |  |     |              +--rw netmask?    yang:dotted-quad
|  |  |     +--:(range-match)
|  |  |        +--rw range-ipv4-address*
[start-ipv4-address end-ipv4-address]
|  |  |           +--rw start-ipv4-address  inet:ipv4-address
|  |  |           +--rw end-ipv4-address    inet:ipv4-address
|  |  +--rw pkt-sec-ipv4-ipopts*            identityref
|  |  +--rw pkt-sec-ipv4-sameip?            boolean
|  |  +--rw pkt-sec-ipv4-geoip*             string
|  +--rw packet-security-ipv6-condition
|  |  +--rw pkt-sec-ipv6-traffic-class*    identityref
|  |  +--rw pkt-sec-ipv6-flow-label
|  |  |  +--rw (match-type)?
|  |  |     +--:(exact-match)
|  |  |     |  +--rw ipv6-flow-label*         uint32
|  |  |     +--:(range-match)
|  |  |        +--rw range-ipv6-flow-label*
[start-ipv6-flow-label end-ipv6-flow-label]
|  |  |           +--rw start-ipv6-flow-label    uint32
|  |  |           +--rw end-ipv6-flow-label      uint32
|  |  +--rw pkt-sec-ipv6-payload-length
|  |  |  +--rw (match-type)?
|  |  |     +--:(exact-match)
|  |  |     |  +--rw ipv6-payload-length*         uint16
|  |  |     +--:(range-match)
|  |  |        +--rw range-ipv6-payload-length*
[start-ipv6-payload-length end-ipv6-payload-length]
|  |  |           +--rw start-ipv6-payload-length    uint16
|  |  |           +--rw end-ipv6-payload-length      uint16
|  |  +--rw pkt-sec-ipv6-next-header*      identityref
|  |  +--rw pkt-sec-ipv6-hop-limit
|  |  |  +--rw (match-type)?
|  |  |     +--:(exact-match)
```

```
|  |  |     |  +--rw ipv6-hop-limit*         uint8
|  |  |     +--:(range-match)
|  |  |        +--rw range-ipv6-hop-limit*
[start-ipv6-hop-limit end-ipv6-hop-limit]
|  |  |           +--rw start-ipv6-hop-limit    uint8
|  |  |           +--rw end-ipv6-hop-limit      uint8
|  |  +--rw pkt-sec-ipv6-src
|  |  |  +--rw (match-type)?
|  |  |     +--:(exact-match)
|  |  |     |  +--rw ipv6
|  |  |     |     +--rw ipv6-address* [ipv6]
|  |  |     |        +--rw ipv6             inet:ipv6-address
|  |  |     |        +--rw prefix-length?   uint8
|  |  |     +--:(range-match)
|  |  |        +--rw range-ipv6-address*
[start-ipv6-address end-ipv6-address]
|  |  |           +--rw start-ipv6-address  inet:ipv6-address
|  |  |           +--rw end-ipv6-address    inet:ipv6-address
|  |  +--rw pkt-sec-ipv6-dest
|  |     +--rw (match-type)?
|  |        +--:(exact-match)
|  |        |  +--rw ipv6-address* [ipv6]
|  |        |     +--rw ipv6             inet:ipv6-address
|  |        |     +--rw prefix-length?   uint8
|  |        +--:(range-match)
|  |           +--rw range-ipv6-address*
[start-ipv6-address end-ipv6-address]
|  |              +--rw start-ipv6-address  inet:ipv6-address
|  |              +--rw end-ipv6-address    inet:ipv6-address
|  +--rw packet-security-tcp-condition
|  |  +--rw pkt-sec-tcp-src-port-num
|  |  |  +--rw (match-type)?
|  |  |     +--:(exact-match)
|  |  |     |  +--rw port-num*         inet:port-number
|  |  |     +--:(range-match)
|  |  |        +--rw range-port-num*
[start-port-num end-port-num]
|  |  |           +--rw start-port-num    inet:port-number
|  |  |           +--rw end-port-num      inet:port-number
|  |  +--rw pkt-sec-tcp-dest-port-num
|  |  |  +--rw (match-type)?
|  |  |     +--:(exact-match)
|  |  |     |  +--rw port-num*         inet:port-number
|  |  |     +--:(range-match)
|  |  |        +--rw range-port-num*
[start-port-num end-port-num]
|  |  |           +--rw start-port-num    inet:port-number
|  |  |           +--rw end-port-num      inet:port-number
```

```
|   |  +--rw pkt-sec-tcp-seq-num
|   |  |  +--rw (match-type)?
|   |  |     +--:(exact-match)
|   |  |     |  +--rw tcp-seq-num*         uint32
|   |  |     +--:(range-match)
|   |  |        +--rw range-tcp-seq-num*
|   [start-tcp-seq-num end-tcp-seq-num]
|   |  |           +--rw start-tcp-seq-num    uint32
|   |  |           +--rw end-tcp-seq-num      uint32
|   |  +--rw pkt-sec-tcp-ack-num
|   |  |  +--rw (match-type)?
|   |  |     +--:(exact-match)
|   |  |     |  +--rw tcp-ack-num*         uint32
|   |  |     +--:(range-match)
|   |  |        +--rw range-tcp-ack-num*
|   [start-tcp-ack-num end-tcp-ack-num]
|   |  |           +--rw start-tcp-ack-num    uint32
|   |  |           +--rw end-tcp-ack-num      uint32
|   |  +--rw pkt-sec-tcp-window-size
|   |  |  +--rw (match-type)?
|   |  |     +--:(exact-match)
|   |  |     |  +--rw tcp-window-size*         uint16
|   |  |     +--:(range-match)
|   |  |        +--rw range-tcp-window-size*
|   [start-tcp-window-size end-tcp-window-size]
|   |  |           +--rw start-tcp-window-size    uint16
|   |  |           +--rw end-tcp-window-size      uint16
|   |  +--rw pkt-sec-tcp-flags*           identityref
|   +--rw packet-security-udp-condition
|   |  +--rw pkt-sec-udp-src-port-num
|   |  |  +--rw (match-type)?
|   |  |     +--:(exact-match)
|   |  |     |  +--rw port-num*         inet:port-number
|   |  |     +--:(range-match)
|   |  |        +--rw range-port-num*
|   [start-port-num end-port-num]
|   |  |           +--rw start-port-num    inet:port-number
|   |  |           +--rw end-port-num      inet:port-number
|   |  +--rw pkt-sec-udp-dest-port-num
|   |  |  +--rw (match-type)?
|   |  |     +--:(exact-match)
|   |  |     |  +--rw port-num*         inet:port-number
|   |  |     +--:(range-match)
|   |  |        +--rw range-port-num*
|   [start-port-num end-port-num]
|   |  |           +--rw start-port-num    inet:port-number
|   |  |           +--rw end-port-num      inet:port-number
|   |  +--rw pkt-sec-udp-total-length
```

```
|  |     +--rw (match-type)?
|  |        +--:(exact-match)
|  |        |  +--rw udp-total-length*         uint32
|  |        +--:(range-match)
|  |           +--rw range-udp-total-length*
|  [start-udp-total-length end-udp-total-length]
|  |              +--rw start-udp-total-length    uint32
|  |              +--rw end-udp-total-length      uint32
|  +--rw packet-security-icmp-condition
|  |  +--rw pkt-sec-icmp-type*   identityref
|  +--rw packet-security-http-condition
|  |  +--rw pkt-sec-uri-content*   string
|  |  +--rw pkt-sec-url-content*   string
|  +--rw packet-security-voice-condition
|  |  +--rw pkt-sec-src-voice-id*     string
|  |  +--rw pkt-sec-dest-voice-id*     string
|  |  +--rw pkt-sec-user-agent*   string
|  +--rw packet-security-ddos-condition
|     +--rw pkt-sec-alert-rate?   uint32
|  |  +--rw packet-payload-condition
|  |  |  +--rw packet-payload-description?   string
|  |  |  +--rw pkt-payload-content*          string
|  |  +--rw acl-number*                        uint32
|  |  +--rw application-condition
|  |  |  +--rw application-description?   string
|  |  |  +--rw application-object*        string
|  |  |  +--rw application-group*         string
|  |  |  +--rw application-label*         string
|  |  |  +--rw category
|  |  |     +--rw application-category*
|  [name application-subcategory]
|  |  |        +--rw name                        string
|  |  |        +--rw application-subcategory    string
|  |  +--rw target-condition
|  |  |  +--rw target-description?        string
|  |  |  +--rw device-sec-context-cond
|  |  |     +--rw target-device*   identityref
|  |  +--rw users-condition
|  |  |  +--rw users-description?   string
|  |  |  +--rw user
|  |  |  |  +--rw (user-name)?
|  |  |  |     +--:(tenant)
|  |  |  |     |  +--rw tenant    uint8
|  |  |  |     +--:(vn-id)
|  |  |  |        +--rw vn-id     uint8
|  |  |  +--rw group
|  |  |  |  +--rw (group-name)?
|  |  |  |     +--:(tenant)
```

```
|  |  |  |     |  +--rw tenant    uint8
|  |  |  |     +--:(vn-id)
|  |  |  |        +--rw vn-id     uint8
|  |  |  +--rw security-grup        string
|  |  +--rw url-category-condition
|  |  |  +--rw url-category-description?   string
|  |  |  +--rw pre-defined-category*       string
|  |  |  +--rw user-defined-category*      string
|  |  +--rw context-condition
|  |  |  +--rw context-description?   string
|  |  +--rw gen-context-condition
|  |     +--rw gen-context-description?   string
|  |     +--rw geographic-location
|  |        +--rw src-geographic-location*    uint32
|  |        +--rw dest-geographic-location*   uint32
+--rw action-clause-container
   ...
```

Figure 3: YANG Tree Diagram for Network Security Policy

This YANG tree diagram shows an condition clause of I2NSF security policy rule for generic network security functions. 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. The condition clause is classified as conditions of generic network security functions and advanced network security functions. The condition clause of generic network security functions is defined as packet security IPv4 condition, packet security IPv6 condition, packet security tcp condition, and packet security icmp condition. The condition clause of advanced network security functions is defined as packet security http condition, packet security voice condition, and packet security ddos condition. Note that this document deals only with simple conditions of advanced network security functions. The condition clauses of advanced network security functions are described in detail in [i2nsf-advanced-nsf-dm]. The condition clause can be extended according to specific vendor condition features. The condition clause is described in detail in [i2nsf-nsf-cap-im].

4.4. Action Clause

This section shows YANG tree diagram for an action clause of I2NSF security policy rule.

```
module: ietf-i2nsf-policy-rule-for-nsf
 +--rw i2nsf-security-policy
    ...
       +--rw rules* [rule-name]
          ...
          +--rw event-clause-container
          |  ...
          +--rw condition-clause-container
          |  ...
          +--rw action-clause-container
             +--rw action-clause-description?   string
             +--rw packet-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
```

Figure 4: YANG Tree Diagram for Network Security Policy

This YANG tree diagram shows an action clause of I2NSF security policy rule for generic network security functions. An action is used to control and monitor aspects of flow-based NSFs when the event and condition clauses are satisfied. NSFs provide security services by executing various actions. The action clause is defined as ingress action, egress action, log action, and advanced action for additional inspection. The advanced action is described in detail in [RFC8329] and [i2nsf-nsf-cap-im]. The action clause can be extended according to specific vendor action features. The action clause is described in detail in [i2nsf-nsf-cap-im].

5. YANG Data Module

5.1. I2NSF NSF-Facing Interface YANG Data Module

This section introduces an YANG data module for configuration of security policy rules on network security functions.

<CODE BEGINS> file "ietf-i2nsf-policy-rule-for-nsf@2019-03-24.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
    iiprfn;
```

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

     WG Chair: Adrian Farrel
     <mailto:Adrain@olddog.co.uk>

     WG Chair: Linda Dunbar
     <mailto:Linda.duhbar@huawei.com>

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

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

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

  description
    "This module defines a YANG data module for network security
     functions.

    Copyright (c) 2018 IETF Trust and the persons
    identified as authors of the code.  All rights reserved.

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

    This version of this YANG module is part of RFC 8341; see
    the RFC itself for full legal notices.";
```

```
  revision "2019-03-24"{
    description "Initial revision.";
    reference
      "RFC XXXX: I2NSF Network Security Function-Facing Interface
       YANG Data Model";
  }

  /*
   * Identities
   */

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

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

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

  identity event {
    description
      "Base identity for event of policy.";
    reference
      "draft-hong-i2nsf-nsf-monitoring-data-model-06
       - Event";
  }

  identity system-event {
    base event;
    description
      "Identity for system event";
    reference
      "draft-hong-i2nsf-nsf-monitoring-data-model-06
       - System event";
  }

  identity system-alarm {
    base event;
    description
```

```
      "Identity for system alarm";
    reference
      "draft-hong-i2nsf-nsf-monitoring-data-model-06
       - System alarm";
  }

  identity access-violation {
    base system-event;
    description
      "Identity for access violation
      among system events";
    reference
      "draft-hong-i2nsf-nsf-monitoring-data-model-06
       - System event";
  }

  identity configuration-change {
    base system-event;
    description
      "Identity for configuration change
      among system events";
    reference
      "draft-hong-i2nsf-nsf-monitoring-data-model-06
       - System event";
  }

  identity memory-alarm {
    base system-alarm;
    description
      "Identity for memory alarm
      among system alarms";
    reference
      "draft-hong-i2nsf-nsf-monitoring-data-model-06
       - System alarm";
  }

  identity cpu-alarm {
    base system-alarm;
    description
      "Identity for cpu alarm
      among system alarms";
    reference
      "draft-hong-i2nsf-nsf-monitoring-data-model-06
       - System alarm";
  }

  identity disk-alarm {
    base system-alarm;
```

```
    description
      "Identity for disk alarm
      among system alarms";
    reference
      "draft-hong-i2nsf-nsf-monitoring-data-model-06
       - System alarm";
  }

  identity hardware-alarm {
    base system-alarm;
    description
      "Identity for hardware alarm
      among system alarms";
    reference
      "draft-hong-i2nsf-nsf-monitoring-data-model-06
       - System alarm";
  }

  identity interface-alarm {
    base system-alarm;
    description
      "Identity for interface alarm
      among system alarms";
    reference
      "draft-hong-i2nsf-nsf-monitoring-data-model-06
       - System alarm";
  }

  identity type-of-service {
    description
      "Base identity for type of service of IPv4";
    reference
      "RFC 791: Internet Protocol - Type of Service";
  }

  identity traffic-class {
    description
      "Base identity for traffic-class of IPv6";
    reference
      "RFC 2460: Internet Protocol, Version 6 (IPv6)
      Specification - Traffic Class";
  }

  identity normal {
    base type-of-service;
    base traffic-class;
    description
      "Identity for normal";
```

```
    reference
      "RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
      Specification - Traffic Class";
  }

  identity minimize-cost {
    base type-of-service;
    base traffic-class;
    description
      "Identity for minimize cost";
    reference
      "RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
      Specification - Traffic Class";
  }

  identity maximize-reliability {
    base type-of-service;
    base traffic-class;
    description
      "Identity for maximize reliability";
    reference
      "RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
      Specification - Traffic Class";
  }

  identity maximize-throughput {
    base type-of-service;
    base traffic-class;
    description
      "Identity for maximize throughput";
    reference
      "RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
      Specification - Traffic Class";
  }

  identity minimize-delay {
    base type-of-service;
    base traffic-class;
    description
      "Identity for minimize delay";
    reference
      "RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
      Specification - Traffic Class";
```

```
  }

  identity maximize-security {
    base type-of-service;
    base traffic-class;
    description
      "Identity for maximize security";
    reference
      "RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
      Specification - Traffic Class";
  }

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

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

  identity no-fragment {
    base fragmentation-flags-type;
    description
      "Identity for no fragment";
    reference
      "RFC 791: Internet Protocol - Fragmentation Flags";
  }

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

  identity protocol {
    description
      "Base identity for protocol of IPv4";
    reference
```

```
      "RFC 790: Assigned numbers - Assigned Internet
       Protocol Number
       RFC 791: Internet Protocol - Protocol";
  }

  identity next-header {
    description
      "Base identity for next header of IPv6";
    reference
      "RFC 2460: Internet Protocol, Version 6 (IPv6)
      Specification - Next Header";
  }

  identity icmp {
    base protocol;
    base next-header;
    description
      "Identity for icmp";
    reference
      "RFC  790: - Assigned numbers - Assigned Internet
       Protocol Number
       RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
       Specification - Next Header";
  }

  identity igmp {
    base protocol;
    base next-header;
    description
      "Identity for igmp";
    reference
      "RFC  790: - Assigned numbers - Assigned Internet
       Protocol Number
       RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
       Specification - Next Header";
  }

  identity tcp {
    base protocol;
    base next-header;
    description
      "Identity for tcp";
    reference
      "RFC  790: - Assigned numbers - Assigned Internet
       Protocol Number
       RFC  791: Internet Protocol - Type of Service
```

```
       RFC 2460: Internet Protocol, Version 6 (IPv6)
       Specification - Next Header";
  }

  identity igrp {
    base protocol;
    base next-header;
    description
      "Identity for igrp";
    reference
      "RFC  790: - Assigned numbers - Assigned Internet
       Protocol Number
       RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
       Specification - Next Header";
  }

  identity udp {
    base protocol;
    base next-header;
    description
      "Identity for udp";
    reference
      "RFC  790: - Assigned numbers - Assigned Internet
       Protocol Number
       RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
       Specification - Next Header";
  }

  identity gre {
    base protocol;
    base next-header;
    description
      "Identity for gre";
    reference
      "RFC  790: - Assigned numbers - Assigned Internet
       Protocol Number
       RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
       Specification - Next Header";
  }

  identity esp {
    base protocol;
    base next-header;
    description
      "Identity for esp";
```

```
    reference
      "RFC  790: - Assigned numbers - Assigned Internet
       Protocol Number
       RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
       Specification - Next Header";
  }

  identity ah {
    base protocol;
    base next-header;
    description
      "Identity for ah";
    reference
      "RFC  790: - Assigned numbers - Assigned Internet
       Protocol Number
       RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
       Specification - Next Header";
  }

  identity mobile {
    base protocol;
    base next-header;
    description
      "Identity for mobile";
    reference
      "RFC  790: - Assigned numbers - Assigned Internet
       Protocol Number
       RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
       Specification - Next Header";
  }

  identity tlsp {
    base protocol;
    base next-header;
    description
      "Identity for tlsp";
    reference
      "RFC  790: - Assigned numbers - Assigned Internet
       Protocol Number
       RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
       Specification - Next Header";
  }

  identity skip {
```

```
    base protocol;
    base next-header;
    description
      "Identity for skip";
    reference
      "RFC  790: - Assigned numbers - Assigned Internet
       Protocol Number
       RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
       Specification - Next Header";
  }

  identity ipv6-icmp {
    base protocol;
    base next-header;
    description
      "Identity for IPv6 icmp ";
    reference
      "RFC  790: - Assigned numbers - Assigned Internet
       Protocol Number
       RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
       Specification - Next Header";
  }


  identity eigrp {
    base protocol;
    base next-header;
    description
      "Identity for eigrp";
    reference
      "RFC  790: - Assigned numbers - Assigned Internet
       Protocol Number
       RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
       Specification - Next Header";
  }

  identity ospf {
    base protocol;
    base next-header;
    description
      "Identity for ospf";
    reference
      "RFC  790: - Assigned numbers - Assigned Internet
       Protocol Number
       RFC  791: Internet Protocol - Type of Service
```

```
       RFC 2460: Internet Protocol, Version 6 (IPv6)
       Specification - Next Header";
  }

  identity l2tp {
    base protocol;
    base next-header;
    description
      "Identity for l2tp";
    reference
      "RFC  790: - Assigned numbers - Assigned Internet
       Protocol Number
       RFC  791: Internet Protocol - Type of Service
       RFC 2460: Internet Protocol, Version 6 (IPv6)
       Specification - Next Header";
  }


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

  identity rr {
    base ipopts;
    description
      "Identity for record route";
    reference
      "RFC 791: Internet Protocol - Options";
  }

  identity eol {
    base ipopts;
    description
      "Identity for end of list";
    reference
      "RFC 791: Internet Protocol - Options";
  }

  identity nop {
    base ipopts;
    description
      "Identity for no operation";
    reference
      "RFC 791: Internet Protocol - Options";
  }
```

```
  identity ts {
    base ipopts;
    description
      "Identity for time stamp";
    reference
      "RFC 791: Internet Protocol - Options";
  }

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

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

  identity lsrr {
    base ipopts;
    description
      "Identity for loose source routing";
    reference
      "RFC 791: Internet Protocol - Options";
  }

  identity ssrr {
    base ipopts;
    description
      "Identity for strict source routing";
    reference
      "RFC 791: Internet Protocol - Options";
  }

  identity satid {
    base ipopts;
    description
      "Identity for stream identifier";
    reference
      "RFC 791: Internet Protocol - Options";
  }
```

```
  identity any {
    base ipopts;
    description
      "Identity for which any IP options are set";
    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";
    reference
      "RFC 793: Transmission Control Protocol - Flags";
  }

  identity ecn {
    base tcp-flags;
    description
      "Identity for explicit congestion notification";
    reference
      "RFC 793: Transmission Control Protocol - Flags";
  }

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

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

  identity psh {
```

```
    base tcp-flags;
    description
      "Identity for push";
    reference
      "RFC 793: Transmission Control Protocol - Flags";
  }

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

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

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

  identity icmp-type {
    description
      "Base identity for icmp types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

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

  identity destination-unreachable {
    base icmp-type;
```

```
    description
      "Identity for destination unreachable";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

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

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

  identity alternate-host-address {
    base icmp-type;
    description
      "Identity for alternate host address";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

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

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

  identity router-solicitation {
    base icmp-type;
```

```
    description
      "Identity for router solicitation";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

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

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

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

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

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

  identity information-reply {
    base icmp-type;
```

```
    description
      "Identity for information reply";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity address-mask-request {
    base icmp-type;
    description
      "Identity for address mask request";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity address-mask-reply {
    base icmp-type;
    description
      "Identity for address mask reply";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

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

  identity datagram-conversion-error {
    base icmp-type;
    description
      "Identity for datagram conversion error";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity mobile-host-redirect {
    base icmp-type;
    description
      "Identity for mobile host redirect";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity ipv6-where-are-you {
    base icmp-type;
```

```
    description
      "Identity for IPv6 where are you";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity ipv6-i-am-here {
    base icmp-type ;
    description
      "Identity for IPv6 i am here";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity mobile-registration-request {
    base icmp-type;
    description
      "Identity for mobile registration request";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity mobile-registration-reply {
    base icmp-type;
    description
      "Identity for mobile registration reply";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity domain-name-request {
    base icmp-type;
    description
      "Identity for domain name request";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity domain-name-reply {
    base icmp-type;
    description
      "Identity for domain name reply";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity iskip {
    base icmp-type;
```

```
    description
      "Identity for icmp skip";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

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

  identity experimental-mobility-protocols {
    base icmp-type;
    description
      "Identity for experimental mobility protocols";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity extended-echo-request {
    base icmp-type;
    description
      "Identity for extended echo request";
    reference
      "RFC  792: Internet Control Message Protocol
       RFC 8335: PROBE: A Utility for Probing Interfaces";
  }

  identity extended-echo-reply {
    base icmp-type;
    description
      "Identity for extended echo reply";
    reference
      "RFC  792: Internet Control Message Protocol
       RFC 8335: PROBE: A Utility for Probing Interfaces";
  }

  identity net-unreachable {
    base icmp-type;
    description
      "Identity for net unreachable
      in destination unreachable types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }
```

```
  identity host-unreachable {
    base icmp-type;
    description
      "Identity for host unreachable
      in destination unreachable types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity protocol-unreachable {
    base icmp-type;
    description
      "Identity for protocol unreachable
      in destination unreachable types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity port-unreachable {
    base icmp-type;
    description
      "Identity for port unreachable
      in destination unreachable types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity fragment-set {
    base icmp-type;
    description
      "Identity for fragmentation set
      in destination unreachable types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity source-route-failed {
    base icmp-type;
    description
      "Identity for source route failed
      in destination unreachable types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity destination-network-unknown {
    base icmp-type;
    description
```

```
      "Identity for destination network unknown
      in destination unreachable types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity destination-host-unknown {
    base icmp-type;
    description
      "Identity for destination host unknown
      in destination unreachable types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity source-host-isolated {
    base icmp-type;
    description
      "Identity for source host isolated
      in destination unreachable types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity communication-prohibited-with-destination-network {
    base icmp-type;
    description
      "Identity for which communication with destination network
      is administratively prohibited in destination unreachable
      types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity communication-prohibited-with-destination-host {
    base icmp-type;
    description
      "Identity for which communication with destination host
      is administratively prohibited in destination unreachable
      types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity destination-network-unreachable-for-tos {
    base icmp-type;
    description
      "Identity for destination network unreachable
```

```
      for type of service in destination unreachable types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity destination-host-unreachable-for-tos {
    base icmp-type;
    description
      "Identity for destination host unreachable
      for type of service in destination unreachable types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity communication-prohibited {
    base icmp-type;
    description
      "Identity for communication administratively prohibited
      in destination unreachable types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity host-precedence-violation {
    base icmp-type;
    description
      "Identity for host precedence violation
      in destination unreachable types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity precedence-cutoff-in-effect {
    base icmp-type;
    description
      "Identity for precedence cutoff in effect
      in destination unreachable types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity redirect-datagram-for-the-network {
    base icmp-type;
    description
      "Identity for redirect datagram for the network
      (or subnet) in redirect types";
    reference
      "RFC 792: Internet Control Message Protocol";
```

```
  }

  identity redirect-datagram-for-the-host {
    base icmp-type;
    description
      "Identity for redirect datagram for the host
      in redirect types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity redirect-datagram-for-the-tos-and-network {
    base icmp-type;
    description
      "Identity for redirect datagram for the type of
      service and network in redirect types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity redirect-datagram-for-the-tos-and-host {
    base icmp-type;
    description
      "Identity for redirect datagram for the type of
      service and host in redirect types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity normal-router-advertisement {
    base icmp-type;
    description
      "Identity for normal router advertisement
      in router advertisement types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity does-not-route-common-traffic {
    base icmp-type;
    description
      "Identity for does not route common traffic
      in router advertisement types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity time-to-live-exceeded-in-transit {
```

```
    base icmp-type;
    description
      "Identity for time to live exceeded in transit
      in time exceeded types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity fragment-reassembly-time-exceeded {
    base icmp-type;
    description
      "Identity for fragment reassembly time exceeded
      in time exceeded types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity pointer-indicates-the-error {
    base icmp-type;
    description
      "Identity for pointer indicates the error
      in parameter problem types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity missing-a-required-option {
    base icmp-type;
    description
      "Identity for missing a required option
      in parameter problem types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity bad-length {
    base icmp-type;
    description
      "Identity for bad length
      in parameter problem types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity bad-spi {
    base icmp-type;
    description
      "Identity for bad spi
```

```
      in photuris types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity authentication-failed {
    base icmp-type;
    description
      "Identity for authentication failed
      in photuris types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity decompression-failed {
    base icmp-type;
    description
      "Identity for decompression failed
      in photuris types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity decryption-failed {
    base icmp-type;
    description
      "Identity for decryption failed
      in photuris types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity need-authentication {
    base icmp-type;
    description
      "Identity for need authentication
      in photuris types";
    reference
      "RFC 792: Internet Control Message Protocol";
  }

  identity need-authorization {
    base icmp-type;
    description
      "Identity for need authorization
      in photuris types";
    reference
      "RFC 792: Internet Control Message Protocol";
```

```
  }

  identity req-no-error {
    base icmp-type;
    description
      "Identity for request with no error
      in extended echo request types";
    reference
      "RFC  792: Internet Control Message Protocol
       RFC 8335: PROBE: A Utility for Probing Interfaces";
  }

  identity rep-no-error {
    base icmp-type;
    description
      "Identity for reply with no error
      in extended echo reply types";
    reference
      "RFC  792: Internet Control Message Protocol
       RFC 8335: PROBE: A Utility for Probing Interfaces";
  }

  identity malformed-query {
    base icmp-type;
    description
      "Identity for malformed query
      in extended echo reply types";
    reference
      "RFC  792: Internet Control Message Protocol
       RFC 8335: PROBE: A Utility for Probing Interfaces";
  }

  identity no-such-interface {
    base icmp-type;
    description
      "Identity for no such interface
      in extended echo reply types";
    reference
      "RFC  792: Internet Control Message Protocol
       RFC 8335: PROBE: A Utility for Probing Interfaces";
  }

  identity no-such-table-entry {
    base icmp-type;
    description
      "Identity for no such table entry
      in extended echo reply types";
    reference
```

```
      "RFC  792: Internet Control Message Protocol
       RFC 8335: PROBE: A Utility for Probing Interfaces";
  }

  identity multiple-interfaces-satisfy-query {
    base icmp-type;
    description
      "Identity for multiple interfaces satisfy query
      in extended echo reply types";
    reference
      "RFC  792: Internet Control Message Protocol
       RFC 8335: PROBE: A Utility for Probing Interfaces";
  }

  identity target-device {
    description
      "Base identity for target devices";
    reference
      "draft-ietf-i2nsf-capability-04: Information Model
       of NSFs Capabilities";
  }

  identity pc {
    base target-device;
    description
      "Identity for pc";
  }

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

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

  identity tablet {
    base target-device;
    description
      "Identity for tablet";
  }

  identity iot {
    base target-device;
```

```
    description
      "Identity for IoT";
  }

  identity vehicle {
    base target-device;
    description
      "Identity for vehicle";
  }

  identity content-security-control {
    description
      "Base identity for content security control";
    reference
      "RFC 8329: Framework for Interface to
       Network Security Functions - Differences
       from ACL Data Models
       draft-ietf-i2nsf-capability-04: Information Model
       of NSFs Capabilities";
  }

  identity antivirus {
    base content-security-control;
    description
      "Identity for antivirus";
  }

  identity ips {
    base content-security-control;
    description
      "Identity for ips";
  }

  identity ids {
    base content-security-control;
    description
      "Identity for ids";
  }

  identity url-filtering {
    base content-security-control;
    description
      "Identity for url filtering";
  }

  identity mail-filtering {
    base content-security-control;
    description
```

```
      "Identity for mail filtering";
  }

  identity file-blocking {
    base content-security-control;
    description
      "Identity for file blocking";
  }

  identity file-isolate {
    base content-security-control;
    description
      "Identity for file isolate";
  }

  identity pkt-capture {
    base content-security-control;
    description
      "Identity for packet capture";
  }

  identity application-control {
    base content-security-control;
    description
      "Identity for application control";
  }

  identity voip-volte {
    base content-security-control;
    description
      "Identity for voip and volte";
  }

  identity attack-mitigation-control {
    description
      "Base identity for attack mitigation control";
    reference
      "RFC 8329: Framework for Interface to
       Network Security Functions - Differences
       from ACL Data Models
       draft-ietf-i2nsf-capability-04: Information Model
       of NSFs Capabilities";
  }

  identity syn-flood {
    base attack-mitigation-control;
    description
      "Identity for syn flood";
```

```
  }

  identity udp-flood {
    base attack-mitigation-control;
    description
      "Identity for udp flood";
  }

  identity icmp-flood {
    base attack-mitigation-control;
    description
      "Identity for icmp flood";
  }

  identity ip-frag-flood {
    base attack-mitigation-control;
    description
      "Identity for ip frag flood";
  }

  identity ipv6-related {
    base attack-mitigation-control;
    description
      "Identity for ipv6 related";
  }

  identity http-and-https-flood {
    base attack-mitigation-control;
    description
      "Identity for http and https flood";
  }

  identity dns-flood {
    base attack-mitigation-control;
    description
      "Identity for dns flood";
  }

  identity dns-amp-flood {
    base attack-mitigation-control;
    description
      "Identity for dns amp flood";
  }

  identity ssl-ddos {
    base attack-mitigation-control;
    description
      "Identity for ssl ddos";
```

```
  }

  identity ip-sweep {
    base attack-mitigation-control;
    description
      "Identity for ip sweep";
  }

  identity port-scanning {
    base attack-mitigation-control;
    description
      "Identity for port scanning";
  }

  identity ping-of-death {
    base attack-mitigation-control;
    description
      "Identity for ping of death";
  }

  identity teardrop {
    base attack-mitigation-control;
    description
      "Identity for teardrop";
  }

  identity oversized-icmp {
    base attack-mitigation-control;
    description
      "Identity for oversized icmp";
  }

  identity tracert {
    base attack-mitigation-control;
    description
      "Identity for tracert";
  }

  identity ingress-action {
    description
      "Base identity for action";
    reference
      "draft-ietf-i2nsf-capability-04: Information Model
       of NSFs Capabilities - Ingress Action";
  }

  identity egress-action {
    description
```

```
      "Base identity for egress action";
    reference
      "draft-ietf-i2nsf-capability-04: Information Model
       of NSFs Capabilities - Egress action";
  }

  identity default-action {
    description
      "Base identity for default action";
    reference
      "draft-ietf-i2nsf-capability-04: Information Model
       of NSFs Capabilities - Default action";
  }

  identity pass {
    base ingress-action;
    base egress-action;
    base default-action;
    description
      "Identity for pass";
    reference
      "draft-ietf-i2nsf-capability-04: Information Model
       of NSFs Capabilities - Actions and
       default action";
  }

  identity drop {
    base ingress-action;
    base egress-action;
    base default-action;
    description
      "Identity for drop";
    reference
      "draft-ietf-i2nsf-capability-04: Information Model
       of NSFs Capabilities - Actions and
       default action";
  }

  identity reject {
    base ingress-action;
    base egress-action;
    base default-action;
    description
      "Identity for reject";
    reference
      "draft-ietf-i2nsf-capability-04: Information Model
       of NSFs Capabilities - Actions and
       default action";
```

```
  }

  identity alert {
    base ingress-action;
    base egress-action;
    base default-action;
    description
      "Identity for alert";
    reference
      "draft-ietf-i2nsf-capability-04: Information Model
       of NSFs Capabilities - Actions and
       default action";
  }

  identity mirror {
    base ingress-action;
    base egress-action;
    base default-action;
    description
      "Identity for mirror";
    reference
      "draft-ietf-i2nsf-capability-04: Information Model
       of NSFs Capabilities - 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 redirection {
    base egress-action;
    description
      "Identity for redirection";

  }

  identity resolution-strategy {
    description
      "Base identity for resolution strategy";
    reference
      "draft-ietf-i2nsf-capability-04: Information Model
       of NSFs Capabilities - Resolution Strategy";
  }

  identity fmr {
    base resolution-strategy;
    description
      "Identity for First Matching Rule (FMR)";
    reference
      "draft-ietf-i2nsf-capability-04: Information Model
       of NSFs Capabilities - Resolution Strategy";
  }

  identity lmr {
    base resolution-strategy;
    description
      "Identity for Last Matching Rule (LMR)";
    reference
      "draft-ietf-i2nsf-capability-04: Information Model
       of NSFs Capabilities - Resolution Strategy";
  }

  identity pmr {
    base resolution-strategy;
    description
```

```
      "Identity for Prioritized Matching Rule (PMR)";
    reference
      "draft-ietf-i2nsf-capability-04: Information Model
       of NSFs Capabilities - Resolution Strategy";
  }

  identity pmre {
    base resolution-strategy;
    description
      "Identity for Prioritized Matching Rule
      with Errors (PMRE)";
    reference
      "draft-ietf-i2nsf-capability-04: Information Model
       of NSFs Capabilities - Resolution Strategy";
  }

  identity pmrn {
    base resolution-strategy;
    description
      "Identity for Prioritized Matching Rule
      with No Errors (PMRN)";
    reference
      "draft-ietf-i2nsf-capability-04: Information Model
       of NSFs Capabilities - Resolution Strategy";
  }

  /*
   * Typedefs
   */

  typedef start-time-type {
    type union {
       type string {
         pattern '\d{2}:\d{2}:\d{2}(\.\d+)?'
               + '(Z|[\+\-]\d{2}:\d{2})';
       }

      type enumeration {
        enum right-away {
          description
            "Immediate rule execution
             in the system.";
        }
      }
    }

    description
      "Start time when the rules are applied.";
```

```
  }

  typedef end-time-type {
    type union {
       type string {
         pattern '\d{2}:\d{2}:\d{2}(\.\d+)?'
               + '(Z|[\+\-]\d{2}:\d{2})';
       }

      type enumeration {
        enum infinitely {
          description
            "Infinite rule execution
             in the system.";
        }
      }
    }
    description
      "End time when the rules are applied.";
  }

   typedef day-type {
      type enumeration {
        enum sunday {
            description
              "Sunday for periodic day";
        }
        enum monday {
            description
              "Monday for periodic day";
        }
        enum tuesday {
            description
              "Tuesday for periodic day";
        }
        enum wednesday {
            description
              "Wednesday for periodic day";
        }
        enum thursday {
            description
              "Thursday for periodic day";
        }
        enum friday {
            description
              "Friday for periodic day";
        }
        enum saturday {
```

```
            description
              "Saturday for periodic day";
        }
      }
      description
        "This can be used for the rules to be applied
        according to periodic day";
  }

   typedef month-type {
      type enumeration {
        enum january {
            description
              "January for periodic month";
        }
        enum february {
            description
              "February for periodic month";
        }
        enum march {
            description
              "March for periodic month";
        }
        enum april {
            description
              "April for periodic month";
        }
        enum may {
            description
              "May for periodic month";
        }
        enum june {
            description
              "June for periodic month";
        }
        enum july {
            description
              "July for periodic month";
        }
        enum august {
            description
              "August for periodic month";
        }
        enum september {
            description
              "September for periodic month";
        }
        enum october {
```

```
            description
              "October for periodic month";
        }
        enum november {
            description
              "November for periodic month";
        }
        enum december {
            description
              "December for periodic month";
        }
      }
      description
        "This can be used for the rules to be applied
        according to periodic month";
  }

  /*
   * Groupings
   */

  grouping ipv4 {
    list ipv4-address {
      key "ipv4";
      description
        "The list of IPv4 address.";

      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.";
        }
```

```
      }
    }
    description
      "Grouping for an IPv4 address";

    reference
      "RFC  791: Internet Protocol - IPv4 address
       RFC 8344: A YANG Data Model for IP Management";
  }

  grouping ipv6 {
    list ipv6-address {
      key "ipv6";
      description
        "The list of IPv6 address.";

      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.";
      }
    }
    description
      "Grouping for an IPv6 address";

    reference
      "RFC 2460: Internet Protocol, Version 6 (IPv6)
       Specification - IPv6 address
       RFC 8344: A YANG Data Model for IP Management";
  }

  grouping pkt-sec-ipv4 {
    choice match-type {
      description
        "There are two types to configure a security policy
        for IPv4 address, such as exact match and range match.";
      case exact-match {
        uses ipv4;
        description
          "Exact match for an IPv4 address.";
```

```
      }
      case range-match {
        list range-ipv4-address {
          key "start-ipv4-address end-ipv4-address";
          leaf start-ipv4-address {
            type inet:ipv4-address;
            description
              "Start IPv4 address for a range match.";
          }

          leaf end-ipv4-address {
            type inet:ipv4-address;
            description
              "End IPv4 address for a range match.";
          }
          description
            "Range match for an IPv4 address.";
        }
      }
    }
    description
      "Grouping for an IPv4 address.";

    reference
      "RFC 791: Internet Protocol - IPv4 address";
  }

  grouping pkt-sec-ipv6 {
    choice match-type {
      description
        "There are two types to configure a security policy
        for IPv6 address, such as exact match and range match.";
      case exact-match {
        uses ipv6;
        description
          "Exact match for an IPv6 address.";
      }
      case range-match {
        list range-ipv6-address {
          key "start-ipv6-address end-ipv6-address";
          leaf start-ipv6-address {
            type inet:ipv6-address;
            description
              "Start IPv6 address for a range match.";
          }

          leaf end-ipv6-address {
            type inet:ipv6-address;
```

```
            description
              "End IPv6 address for a range match.";
          }
          description
            "Range match for an IPv6 address.";
        }
      }
    }
    description
      "Grouping for IPv6 address.";

    reference
      "RFC 2460: Internet Protocol, Version 6 (IPv6)
       Specification - IPv6 address";
  }

  grouping pkt-sec-port-number {
    choice match-type {
      description
        "There are two types to configure a security policy
        for a port number, such as exact match and range match.";
      case exact-match {
        leaf-list port-num {
          type inet:port-number;
          description
            "Exact match for a port number.";
        }
      }
      case range-match {
        list range-port-num {
          key "start-port-num end-port-num";
          leaf start-port-num {
            type inet:port-number;
            description
              "Start port number for a range match.";
          }
          leaf end-port-num {
            type inet:port-number;
            description
              "Start port number for a range match.";
          }
          description
            "Range match for a port number.";
        }
      }
    }
    description
      "Grouping for port number.";
```

```
    reference
      "RFC 793: Transmission Control Protocol - Port number
       RFC 768: User Datagram Protocol - Port Number";
  }

  /*
   * Data nodes
   */

  container i2nsf-security-policy {
    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 is able to apply over 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-04: Information Model
       of NSFs Capabilities - Design Principles and ECA Policy Model
       Overview";

    list system-policy {
      key "system-policy-name";
      description
        "The system-policy represents there could be multiple system
        policies in one NSF, and each system policy is used by
        one virtual instance of the NSF/device.";


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


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

        reference
          "draft-ietf-i2nsf-capability-04: Information Model
           of NSFs Capabilities - Resolution strategy";
      }

      leaf default-action {
        type identityref {
          base default-action;
        }
        default alert;
        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-04: Information Model
           of NSFs Capabilities - Default action";
      }


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

        leaf rule-name {
          type string;
          mandatory true;
          description
            "The name of the rule.
```

```
            This must be unique.";
        }

        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.";
        }

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

        leaf session-aging-time {
          type uint16;
          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 enbale.";
          }

          leaf during {
            type uint16;
            description
              "This is during time.";
          }
```

```
        }

        container time-zone {
          description
            "Time zone when the rules are applied";
          container absolute-time-zone {
            description
              "Rule execution according to absolute time";

            leaf start-time {
              type start-time-type;
              default right-away;
              description
                "Start time when the rules are applied";
            }
            leaf end-time {
              type end-time-type;
              default infinitely;
              description
                "End time when the rules are applied";
            }
          }

          container periodic-time-zone {
            description
              "Rule execution according to periodic time";

            container day {
              description
                "Rule execution according to day.";
              leaf every-day {
                type boolean;
                default true;
                description
                  "Rule execution every day";
              }

              leaf-list specific-day {
                when "../every-day = 'false'";
                type day-type;
                description
                 "Rule execution according
                  to specific day";
              }
            }


            container month {
```

```
            description
              "Rule execution according to month.";
            leaf every-month {
              type boolean;
              default true;
              description
                "Rule execution every day";
            }

            leaf-list specific-month {
              when "../every-month = 'false'";
              type month-type;
              description
               "Rule execution according
                to month day";
            }
          }
        }
      }

      container event-clause-container {
        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-04: Information Model
           of NSFs Capabilities - Design Principles and ECA
           Policy Model Overview
           draft-hong-i2nsf-nsf-monitoring-data-model-06: A YANG
           Data Model for Monitoring I2NSF Network Security
           Functions - System Alarm and System Events";

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

```
          container event-clauses {
            description
              "It has two event types such as
              system event and system alarm.";
            reference
              "RFC 8329: Framework for Interface to Network Security
               Functions - I2NSF Flow Security Policy Structure
               draft-ietf-i2nsf-capability-04: Information Model
               of NSFs Capabilities - Design Principles and ECA Policy
               Model Overview
               draft-hong-i2nsf-nsf-monitoring-data-model-06: A YANG
               Data Model for Monitoring I2NSF Network Security
               Functions - System Alarm and System Events";

            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-clause-container {
          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-04: Information Model
            of NSFs Capabilities - Design Principles and ECA Policy
            Model Overview";

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

          container packet-security-ipv4-condition {
            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 ipv4-description {
              type string;
              description
               "This is description for ipv4 condition.";
            }

            container pkt-sec-ipv4-header-length {
              choice match-type {
                description
                  "There are two types to configure a security
                  policy for IPv4 header length, such as exact match
                  and range match.";
                case exact-match {
                  leaf-list ipv4-header-length {
                    type uint8 {
                      range "5..15";
                    }
                    description
                      "Exact match for an IPv4 header length.";
                  }
                }
                case range-match {
                  list range-ipv4-header-length {
                    key "start-ipv4-header-length
                         end-ipv4-header-length";
                    leaf start-ipv4-header-length {
                      type uint8 {
                        range "5..15";
                      }
```

```
                    description
                      "Start IPv4 header length for a range match.";
                  }

                  leaf end-ipv4-header-length {
                    type uint8 {
                      range "5..15";
                    }
                    description
                      "End IPv4 header length for a range match.";
                  }
                  description
                    "Range match for an IPv4 header length.";
                }
              }
            }
            description
              "The security policy rule according to
               IPv4 header length.";
            reference
              "RFC 791: Internet Protocol - Header length";
          }

          leaf-list pkt-sec-ipv4-tos {
            type identityref {
              base type-of-service;
            }
            description
              "The security policy rule according to
               IPv4 type of service.";
            reference
              "RFC 791: Internet Protocol - Type of service";
          }

          container pkt-sec-ipv4-total-length {
            choice match-type {
              description
                "There are two types to configure a security
                policy for IPv4 total length, such as exact match
                and range match.";
              case exact-match {
                leaf-list ipv4-total-length {
                  type uint16;
                  description
                    "Exact match for an IPv4 total length.";
                }
              }
              case range-match {
```

```
            list range-ipv4-total-length {
              key "start-ipv4-total-length end-ipv4-total-length";
              leaf start-ipv4-total-length {
                type uint16;
                description
                  "Start IPv4 total length for a range match.";
              }
              leaf end-ipv4-total-length {
                type uint16;
                description
                  "End IPv4 total length for a range match.";
              }
              description
                "Range match for an IPv4 total length.";
            }
          }
        }
        description
          "The security policy rule according to
           IPv4 total length.";
        reference
          "RFC 791: Internet Protocol - Total length";
      }

      leaf-list pkt-sec-ipv4-id {
        type uint16;
        description
          "The security policy rule according to
           IPv4 identification.";
        reference
          "RFC 791: Internet Protocol - Identification";
      }

      leaf-list pkt-sec-ipv4-fragment-flags {
        type identityref {
          base fragmentation-flags-type;
        }
        description
          "The security policy rule according to
           IPv4 fragment flags.";
        reference
          "RFC 791: Internet Protocol - Fragment flags";
      }

      container pkt-sec-ipv4-fragment-offset {
        choice match-type {
          description
            "There are two types to configure a security
```

```
                policy for IPv4 fragment offset, such as exact match
                and range match.";
              case exact-match {
                leaf-list ipv4-fragment-offset {
                  type uint16 {
                    range "0..16383";
                  }
                  description
                    "Exact match for an IPv4 fragment offset.";
                }
              }
              case range-match {
                list range-ipv4-fragment-offset {
                  key "start-ipv4-fragment-offset
                       end-ipv4-fragment-offset";
                  leaf start-ipv4-fragment-offset {
                    type uint16 {
                      range "0..16383";
                    }
                    description
                      "Start IPv4 fragment offset for a range match.";
                  }
                  leaf end-ipv4-fragment-offset {
                    type uint16 {
                      range "0..16383";
                    }
                    description
                      "End IPv4 fragment offset for a range match.";
                  }
                  description
                    "Range match for an IPv4 fragment offset.";
                }
              }
            }
            description
              "The security policy rule according to
               IPv4 fragment offset.";
            reference
              "RFC 791: Internet Protocol - Fragment offset";
          }

          container pkt-sec-ipv4-ttl {
            choice match-type {
              description
                "There are two types to configure a security
                policy for IPv4 TTL, such as exact match
                and range match.";
              case exact-match {
```

```
            leaf-list ipv4-ttl {
              type uint8;
              description
                "Exact match for an IPv4 TTL.";
            }
          }
          case range-match {
            list range-ipv4-ttl {
              key "start-ipv4-ttl end-ipv4-ttl";
              leaf start-ipv4-ttl {
                type uint8;
                description
                  "Start IPv4 TTL for a range match.";
              }
              leaf end-ipv4-ttl {
                type uint8;
                description
                  "End IPv4 TTL for a range match.";
              }
              description
                "Range match for an IPv4 TTL.";
            }
          }
        }
        description
          "The security policy rule according to
           IPv4 time-to-live (TTL).";
        reference
          "RFC 791: Internet Protocol - Time to live";
      }


      leaf-list pkt-sec-ipv4-protocol {
        type identityref {
          base protocol;
        }
        description
          "The security policy rule according to
           IPv4 protocol.";
        reference
          "RFC 791: Internet Protocol - Protocol";
      }

      container pkt-sec-ipv4-src {
        uses pkt-sec-ipv4;
        description
          "The security policy rule according to
           IPv4 source address.";
```

```
            reference
              "RFC 791: Internet Protocol - IPv4 Address";
          }

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

          leaf-list pkt-sec-ipv4-ipopts {
            type identityref {
              base ipopts;
            }
            description
              "The security policy rule according to
               IPv4 options.";
            reference
              "RFC 791: Internet Protocol - Options";
          }

          leaf pkt-sec-ipv4-sameip {
            type boolean;
            description
              "Every packet has a source IP-address and
               a destination IP-address. It can be that
               the source IP is the same as
               the destination IP.";
          }

          leaf-list pkt-sec-ipv4-geoip {
            type string;
            description
              "The geoip keyword enables you to match on
               the source, destination or source and destination
               IP addresses of network traffic and to see to
               which country it belongs. To do this, Suricata
               uses GeoIP API with MaxMind database format.";
          }
        }

        container packet-security-ipv6-condition {
          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 2460: Internet Protocol, Version 6 (IPv6)
             Specification";

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


          leaf-list pkt-sec-ipv6-traffic-class {
            type identityref {
              base traffic-class;
            }
            description
              "The security policy rule according to
               IPv6 traffic class.";
            reference
              "RFC 2460: Internet Protocol, Version 6 (IPv6)
               Specification - Traffic class";
          }

          container pkt-sec-ipv6-flow-label {
            choice match-type {
              description
                "There are two types to configure a security
                policy for IPv6 flow label, such as exact match
                and range match.";
              case exact-match {
                leaf-list ipv6-flow-label {
                  type uint32 {
                    range "0..1048575";
                  }
                  description
                    "Exact match for an IPv6 flow label.";
                }
              }
              case range-match {
                list range-ipv6-flow-label {
                  key "start-ipv6-flow-label end-ipv6-flow-label";
                  leaf start-ipv6-flow-label {
                    type uint32 {
                      range "0..1048575";
                    }
                    description
```

```
                    "Start IPv6 flow label for a range match.";
                  }
                  leaf end-ipv6-flow-label {
                    type uint32 {
                      range "0..1048575";
                    }
                    description
                      "End IPv6 flow label for a range match.";
                  }
                  description
                    "Range match for an IPv6 flow label.";
                }
              }
            }
            description
              "The security policy rule according to
               IPv6 flow label.";
            reference
              "RFC 2460: Internet Protocol, Version 6 (IPv6)
               Specification - Flow label";
          }

          container pkt-sec-ipv6-payload-length {
            choice match-type {
              description
                "There are two types to configure a security
                policy for IPv6 payload length, such as
                exact match and range match.";
              case exact-match {
                leaf-list ipv6-payload-length {
                  type uint16;
                  description
                    "Exact match for an IPv6 payload length.";
                }
              }
              case range-match {
                list range-ipv6-payload-length {
                  key "start-ipv6-payload-length
                       end-ipv6-payload-length";
                  leaf start-ipv6-payload-length {
                    type uint16;
                    description
                      "Start IPv6 payload length for a range match.";
                  }
                  leaf end-ipv6-payload-length {
                    type uint16;
                    description
                      "End IPv6 payload length for a range match.";
```

```
            }
            description
              "Range match for an IPv6 payload length.";
          }
        }
      }
      description
        "The security policy rule according to
         IPv6 payload length.";
      reference
        "RFC 2460: Internet Protocol, Version 6 (IPv6)
         Specification - Payload length";
    }

    leaf-list pkt-sec-ipv6-next-header {
      type identityref {
        base next-header;
      }
      description
        "The security policy rule according to
         IPv6 next header.";
      reference
        "RFC 2460: Internet Protocol, Version 6 (IPv6)
         Specification - Next header";
    }

    container pkt-sec-ipv6-hop-limit {
      choice match-type {
        description
          "There are two types to configure a security
          policy for IPv6 hop limit, such as exact match
          and range match.";
        case exact-match {
          leaf-list ipv6-hop-limit {
            type uint8;
            description
              "Exact match for an IPv6 hop limit.";
          }
        }
        case range-match {
          list range-ipv6-hop-limit {
            key "start-ipv6-hop-limit end-ipv6-hop-limit";
            leaf start-ipv6-hop-limit {
              type uint8;
              description
                "Start IPv6 hop limit for a range match.";
            }
            leaf end-ipv6-hop-limit {
```

```
                    type uint8;
                    description
                      "End IPv6 hop limit for a range match.";
                  }
                  description
                    "Range match for an IPv6 hop limit.";
                }
              }
            }
            description
              "The security policy rule according to
               IPv6 hop limit.";
            reference
              "RFC 2460: Internet Protocol, Version 6 (IPv6)
               Specification - Hop limit";
          }

          container pkt-sec-ipv6-src {
            uses pkt-sec-ipv6;
            description
              "The security policy rule according to
               IPv6 source address.";
            reference
              "RFC 2460: Internet Protocol, Version 6 (IPv6)
               Specification - IPv6 address";
          }

          container pkt-sec-ipv6-dest {
            uses pkt-sec-ipv6;
            description
              "The security policy rule according to
               IPv6 destination address.";
            reference
              "RFC 2460: Internet Protocol, Version 6 (IPv6)
               Specification - IPv6 address";
          }

        }

        container packet-security-tcp-condition {
          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 tcp-description {
              type string;
              description
               "This is description for tcp condition.";
            }

            container pkt-sec-tcp-src-port-num {
              uses pkt-sec-port-number;
              description
                "The security policy rule according to
                 tcp source port number.";
              reference
                "RFC 793: Transmission Control Protocol
                 - Port number";
            }


            container pkt-sec-tcp-dest-port-num {
              uses pkt-sec-port-number;
              description
                "The security policy rule according to
                 tcp destination port number.";
              reference
                "RFC 793: Transmission Control Protocol
                 - Port number";
            }

            container pkt-sec-tcp-seq-num {
              choice match-type {
                description
                  "There are two types to configure a security
                  policy for tcp sequence number,
                  such as exact match and range match.";
                case exact-match {
                  leaf-list tcp-seq-num {
                    type uint32;
                    description
                      "Exact match for an tcp sequence number.";
                  }
                }
                case range-match {
                  list range-tcp-seq-num {
                    key "start-tcp-seq-num end-tcp-seq-num";
                    leaf start-tcp-seq-num {
                      type uint32;
                      description
                        "Start tcp sequence number for a range match.";
                    }
```

```
                leaf end-tcp-seq-num {
                  type uint32;
                  description
                    "End tcp sequence number for a range match.";
                }
                description
                  "Range match for a tcp sequence number.";
              }
            }
          }
          description
            "The security policy rule according to
             tcp sequence number.";
          reference
            "RFC 793: Transmission Control Protocol
             - Sequence number";
        }

        container pkt-sec-tcp-ack-num {
          choice match-type {
            description
              "There are two types to configure a security
              policy for tcp acknowledgement number,
              such as exact match and range match.";
            case exact-match {
              leaf-list tcp-ack-num {
                type uint32;
                description
                  "Exact match for an tcp acknowledgement number.";
              }
            }
            case range-match {
              list range-tcp-ack-num {
                key "start-tcp-ack-num end-tcp-ack-num";
                leaf start-tcp-ack-num {
                  type uint32;
                  description
                    "Start tcp acknowledgement number
                    for a range match.";
                }
                leaf end-tcp-ack-num {
                  type uint32;
                  description
                    "End tcp acknowledgement number
                    for a range match.";
                }
                description
                  "Range match for a tcp acknowledgement number.";
```

```
            }
          }
        }
        description
          "The security policy rule according to
           tcp acknowledgement number.";
        reference
          "RFC 793: Transmission Control Protocol
           - Acknowledgement number";
      }

      container pkt-sec-tcp-window-size {
        choice match-type {
          description
            "There are two types to configure a security
            policy for tcp window size,
            such as exact match and range match.";
          case exact-match {
            leaf-list tcp-window-size {
              type uint16;
              description
                "Exact match for an tcp window size.";
            }
          }
          case range-match {
            list range-tcp-window-size {
              key "start-tcp-window-size end-tcp-window-size";
              leaf start-tcp-window-size {
                type uint16;
                description
                  "Start tcp window size for a range match.";
              }
              leaf end-tcp-window-size {
                type uint16;
                description
                  "End tcp window size for a range match.";
              }
              description
                "Range match for a tcp window size.";
            }
          }
        }
        description
          "The security policy rule according to
           tcp window size.";
        reference
          "RFC 793: Transmission Control Protocol
           - Window size";
```

```
            }


            leaf-list pkt-sec-tcp-flags {
              type identityref {
                base tcp-flags;
              }
              description
                "The security policy rule according to
                 tcp flags.";
              reference
                "RFC 793: Transmission Control Protocol
                 - Flags";
            }
          }

          container packet-security-udp-condition {
            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 793: Transmission Control Protocol";

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

            container pkt-sec-udp-src-port-num {
              uses pkt-sec-port-number;
              description
                "The security policy rule according to
                 udp source port number.";
              reference
                "RFC 793: Transmission Control Protocol
                 - Port number";
            }


            container pkt-sec-udp-dest-port-num {
              uses pkt-sec-port-number;
              description
                "The security policy rule according to
                 udp destination port number.";
              reference
```

```
            "RFC 768: User Datagram Protocol
             - Total Length";
          }

          container pkt-sec-udp-total-length {
            choice match-type {
              description
                "There are two types to configure a security
                policy for udp sequence number,
                such as exact match and range match.";
              case exact-match {
                leaf-list udp-total-length {
                  type uint32;
                  description
                    "Exact match for an udp-total-length.";
                }
              }
              case range-match {
                list range-udp-total-length {
                  key "start-udp-total-length end-udp-total-length";
                  leaf start-udp-total-length {
                    type uint32;
                    description
                      "Start udp total length for a range match.";
                  }
                  leaf end-udp-total-length {
                    type uint32;
                    description
                      "End udp total length for a range match.";
                  }
                  description
                    "Range match for a udp total length.";
                }
              }
            }
            description
              "The security policy rule according to
               udp total length.";
            reference
              "RFC 768: User Datagram Protocol
               - Total Length";
          }
        }


        container packet-security-icmp-condition {
          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 icmp-description {
            type string;
            description
             "This is description for icmp condition.";
          }

          leaf-list pkt-sec-icmp-type-and-code {
            type identityref {
              base icmp-type;
            }
            description
              "The security policy rule according to
               ICMP parameters.";
            reference
              "RFC  792: Internet Control Message Protocol
               RFC 8335: PROBE: A Utility for Probing Interfaces";
          }
        }

        container packet-security-http-condition {
          description
            "Condition for http.";

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

          leaf-list pkt-sec-uri-content {
            type string;
            description
              "The security policy rule according to
               uri content.";
          }

          leaf-list pkt-sec-url-content {
            type string;
            description
              "The security policy rule according to
               url content.";
```

```
          }
        }

        container packet-security-voice-condition {
          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 voice-description {
            type string;
            description
             "This is description for voice condition.";
          }

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

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

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

        container packet-security-ddos-condition {
          description
```

```
            "Condition for DDoS attack.";

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

          leaf pkt-sec-alert-rate {
            type uint32;
            description
              "The alert rate of flood detect for
               same packets.";
          }
        }

        container packet-payload-condition {
          description
            "Condition for packet payload";
          leaf packet-payload-description {
            type string;
            description
             "This is description for payload condition.
             Vendors can write instructions for payload condition
             that vendor made";
          }
          leaf-list pkt-payload-content {
            type string;
            description
              "The content keyword is very important in
               signatures. Between the quotation marks you
               can write on what you would like the
               signature to match.";
          }
        }

        leaf-list acl-number {
          type uint32;
          description
           "This is acl-number.";
        }

        container application-condition {
          description
            "Condition for application";
          leaf application-description {
            type string;
            description
```

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

        container target-condition {
          description
            "Condition for target";
          leaf target-description {
            type string;
            description
              "This is description for target condition.
              Vendors can write instructions for target condition
              that vendor made";
          }
```

```
            container device-sec-context-cond {
              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.";

              leaf-list target-device {
                type identityref {
                  base target-device;
                }
                description
                  "Leaf list for target devices";
              }
            }
          }
          container users-condition {
            description
              "Condition for users";
            leaf users-description {
              type string;
              description
                "This is description for user condition.
                Vendors can write instructions for user condition
                that vendor made";
            }
            container user{
              description
                "The user (or user group) information with which
                 network flow is associated: The user has many
                 attributes such as name, id, password, type,
                 authentication mode and so on. Name/id is often
                 used in the security policy to identify the user.
                 Besides, NSF is aware of the IP address of the
                 user provided by a unified user management system
                 via network. Based on name-address association,
                 NSF is able to enforce the security functions
                 over the given user (or user group)";

              choice user-name {
                description
                  "The name of the user.
                   This must be unique.";

                case tenant {
                  description
                    "Tenant information.";
```

```
                leaf tenant {
                  type uint8;
                  mandatory true;
                  description
                    "User's tenant information.";
                }
              }

              case vn-id {
                description
                  "VN-ID information.";

                leaf vn-id {
                  type uint8;
                  mandatory true;
                  description
                    "User's VN-ID information.";
                }
              }
            }
          }
          container group {
            description
              "The user (or user group) information with which
               network flow is associated: The user has many
               attributes such as name, id, password, type,
               authentication mode and so on. Name/id is often
               used in the security policy to identify the user.
               Besides, NSF is aware of the IP address of the
               user provided by a unified user management system
               via network. Based on name-address association,
               NSF is able to enforce the security functions
               over the given user (or user group)";

            choice group-name {
              description
                "The name of the user.
                 This must be unique.";

              case tenant {
                description
                  "Tenant information.";

                leaf tenant {
                  type uint8;
                  mandatory true;
                  description
                    "User's tenant information.";
```

```
                }
              }

              case vn-id {
                description
                  "VN-ID information.";

                leaf vn-id {
                  type uint8;
                  mandatory true;
                  description
                    "User's VN-ID information.";
                }
              }
            }
          }
          leaf security-grup {
            type string;
            mandatory true;
            description
              "security-grup.";
          }
        }

        container url-category-condition {
          description
            "Condition for url category";
          leaf url-category-description {
            type string;
            description
              "This is description for url category condition.
              Vendors can write instructions for context condition
              that vendor made";
          }

          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 context-condition {
```

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

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

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

              leaf-list src-geographic-location {
                type uint32;
                description
                  "This is mapped to ip address. We can acquire
                   source region through ip address stored in the
                   database.";
              }
              leaf-list dest-geographic-location {
                type uint32;
                description
                  "This is mapped to ip address. We can acquire
                   destination region through ip address stored
                   in the database.";
              }
            }
          }
        }
```

```
        container action-clause-container {
          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-04: Information Model
             of NSFs Capabilities - 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-04: Information Model
               of NSFs Capabilities - Design Principles and ECA
               Policy Model Overview";

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

            leaf egress-action {
              type identityref {
                base egress-action;
              }
              description
                "Egress action: pass, drop, reject, alert, 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 need be additionally inspected,
             the packet are passed to advanced network
             security functions according to the profile.";
          reference
            "RFC 8329: Framework for Interface to Network Security
             Functions - Differences from ACL Data Models";

          leaf-list content-security-control {
              type identityref {
                base content-security-control;
              }
              description
                "The Profile is divided into content security
                 control and attack-mitigation-control.
                 Content security control: antivirus, ips, ids,
                 url filtering, mail filtering, file blocking,
                 file isolate, packet capture, application control,
                 voip and volte.";
          }

          leaf-list attack-mitigation-control {
              type identityref {
                base attack-mitigation-control;
              }
              description
                "The Profile is divided into content security
                 control and attack-mitigation-control.
                 Attack mitigation control: syn flood, udp flood,
                 icmp flood, ip frag flood, ipv6 related, http flood,
                 https flood, dns flood, dns amp flood, ssl ddos,
                 ip sweep, port scanning, ping of death, teardrop,
                 oversized icmp, tracert.";
          }
        }
```

```
        }
      }
      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 desription for rule-group";
          }
        }
      }
    }
  }
}
```

<CODE ENDS>

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

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].

name: ietf-i2nsf-policy-rule-for-nsf

namespace: urn:ietf:params:xml:ns:yang:ietf-i2nsf-policy-rule-for-nsf

prefix: iiprfn

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 transport secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the required transport secure transport is TLS [RFC8446].

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

8. References

8.1. Normative References

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

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

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

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

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

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

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

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

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

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

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

[RFC8431] Wang, L., Chen, M., Dass, A., Ananthakrishnan, H., Kini, S., and N. Bahadur, "A YANG Data Model for the Routing Information Base (RIB)", RFC 8431, DOI 10.17487/RFC8431, September 2018, <https://www.rfc-editor.org/info/rfc8431>.

[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>.

8.2. Informative References

[i2nsf-advanced-nsf-dm]
Pan, W. and L. Xia, "Configuration of Advanced Security Functions with I2NSF Security Controller", draft-dong-i2nsf-asf-config-01 (work in progress), October 2018.

[i2nsf-nsf-cap-dm]
Hares, S., Jeong, J., Kim, J., Moskowitz, R., and Q. Lin, "I2NSF Capability YANG Data Model", draft-ietf-i2nsf-capability-data-model-03 (work in progress), March 2019.

[i2nsf-nsf-cap-im]
Xia, L., Strassner, J., Basile, C., and D. Lopez, "Information Model of NSFs Capabilities", draft-ietf-i2nsf-capability-04 (work in progress), October 2018.

[supa-policy-info-model]
Strassner, J., Halpern, J., and S. Meer, "Generic Policy Information Model for Simplified Use of Policy Abstractions (SUPA)", draft-ietf-supa-generic-policy-info-model-03 (work in progress), May 2017.

Appendix A. Configuration Examples

This section shows configuration examples of "ietf-i2nsf-policy-rule-for-nsf" module for security policy rules of network security devices. 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 Appendix A. Configuration Examples of [i2nsf-nsf-cap-dm] are registered in I2NSF framework. With the registed NSFs, we show configuration examples for security policy rules of network security functions according to the following three security requirements: (i) Block 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.

A.1. Security Requirement 1: Block SNS Access during Business Hours

This section shows a configuration example for blocking SNS access during business hours.

```
<i2nsf-security-policy
xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-policy-rule-for-nsf">
<system-policy>
 <system-policy-name>sns_access</system-policy-name>
 <rules>
  <rule-name>block_sns_access_during_operation_time</rule-name>
  <time-zone>
   <absolute-time-zone>
    <start-time>09:00:00Z</start-time>
    <end-time>18:00:00Z</end-time>
   </absolute-time-zone>
  </time-zone>
  <condition-clause-container>
   <packet-security-ipv4-condition>
    <pkt-sec-ipv4-src>
     <range-ipv4-address>
      <start-ipv4-address>221.159.112.1</start-ipv4-address>
      <end-ipv4-address>221.159.112.90</end-ipv4-address>
     </range-ipv4-address>
    </pkt-sec-ipv4-src>
   </packet-security-ipv4-condition>
  </condition-clause-container>
  <action-clause-container>
   <advanced-action>
    <content-security-control>url-filtering</content-security-control>
   </advanced-action>
  </action-clause-container>
 </rules>
</system-policy>
</i2nsf-security-policy>
```

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

```
<i2nsf-security-policy
xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-policy-rule-for-nsf">
<system-policy>
 <system-policy-name>sns_access</system-policy-name>
 <rules>
  <rule-name>block_sns_access_during_operation_time</rule-name>
  <condition-clause-container>
   <packet-security-http-condition>
    <pkt-sec-url-content>facebook</pkt-sec-url-content>
    <pkt-sec-url-content>instagram</pkt-sec-url-content>
   </packet-security-http-condition>
  </condition-clause-container>
  <action-clause-container>
   <packet-action>
    <egress-action>drop</egress-action>
   </packet-action>
  </action-clause-container>
 </rules>
</system-policy>
</i2nsf-security-policy>
```

Figure 7: Configuration XML for Web Filter to Block SNS Access during Business Hours

Figure 6 and Figure 7 show the configuration XML documents for time based firewall and web filter to block SNS access during business hours. For the security requirement, two NSFs (i.e., a time based firewall and a web filter) were used because one NSF can not 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) is described in [i2nsf-advanced-nsf-dm].

Time based Firewall

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 operated during the business hours (i.e., from 9 a.m. to 6 p.m.).

4. The rule inspects a source IPv4 address (i.e., from 221.159.112.1 to 221.159.112.90) to inspect the outgoing packets of employees.

5. 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

1. The name of the system policy is sns_access.

2. The name of the rule is block_facebook_and_instagram.

3. The rule inspects URL address to block the access packets to the facebook or the instagram.

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

A.2. Security Requirement 2: Block Malicious VoIP/VoLTE Packets Coming to the Company

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

```
<i2nsf-security-policy
xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-policy-rule-for-nsf">
<system-policy>
 <system-policy-name>voip_volte_inspection</system-policy-name>
 <rules>
  <rule-name>block_malicious_voice_id</rule-name>
  <condition-clause-container>
   <packet-security-ipv4-condition>
    <pkt-sec-ipv4-dest>
     <range-ipv4-address>
      <start-ipv4-address>221.159.112.1</start-ipv4-address>
      <end-ipv4-address>221.159.112.90</end-ipv4-address>
     </range-ipv4-address>
    </pkt-sec-ipv4-dest>
   </packet-security-ipv4-condition>
   <packet-security-tcp-condition>
    <pkt-sec-tcp-dest-port-num>
     <port-num>5060</port-num>
     <port-num>5061</port-num>
    </pkt-sec-tcp-dest-port-num>
   </packet-security-tcp-condition>
  </condition-clause-container>
  <action-clause-container>
   <advanced-action>
    <content-security-control>voip-volte</content-security-control>
   </advanced-action>
  </action-clause-container>
 </rules>
</system-policy>
</i2nsf-security-policy>
```

Figure 8: Configuration XML for General Firewall to Block Malicious VoIP/VoLTE Packets Coming to the Company

```
<i2nsf-security-policy
xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-policy-rule-for-nsf">
<system-policy>
 <system-policy-name>voip_volte_inspection</system-policy-name>
 <rules>
  <rule-name>block_malicious_voice_id</rule-name>
  <condition-clause-container>
   <packet-security-voice-condition>
    <pkt-sec-src-voice-id>11111@voip.black.com</pkt-sec-src-voice-id>
    <pkt-sec-src-voice-id>22222@voip.black.com</pkt-sec-src-voice-id>
   </packet-security-voice-condition>
  </condition-clause-container>
  <action-clause-container>
   <packet-action>
    <ingress-action>drop</ingress-action>
   </packet-action>
  </action-clause-container>
 </rules>
</system-policy>
</i2nsf-security-policy>
```

Figure 9: Configuration XML for VoIP/VoLTE Filter to Block Malicious VoIP/VoLTE Packets Coming to the Company

Figure 8 and Figure 9 show the configuration XML documents for general firewall and VoIP/VoLTE filter to block malicious VoIP/VoLTE packets coming to the 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) is described in [i2nsf-advanced-nsf-dm].

General Firewall

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 221.159.112.1 to 221.159.112.90) 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

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., 11111@voip.black.com and 22222@voip.black.com).

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

A.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>
 <system-policy-name>flood_attack_mitigation</system-policy-name>
 <rules>
  <rule-name>mitigate_http_and_https_flood_attack</rule-name>
  <condition-clause-container>
   <packet-security-ipv4-condition>
    <pkt-sec-ipv4-dest>
     <ipv4-address>
      <ipv4>221.159.112.95</ipv4>
     </ipv4-address>
    </pkt-sec-ipv4-dest>
   </packet-security-ipv4-condition>
   <packet-security-tcp-condition>
    <pkt-sec-tcp-dest-port-num>
     <port-num>80</port-num>
     <port-num>443</port-num>
    </pkt-sec-tcp-dest-port-num>
   </packet-security-tcp-condition>
  </condition-clause-container>
  <action-clause-container>
   <advanced-action>
    <attack-mitigation-control>http-and-https-flood
    </attack-mitigation-control>
   </advanced-action>
  </action-clause-container>
 </rules>
</system-policy>
</i2nsf-security-policy>
```

Figure 10: 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>
 <system-policy-name>flood_attack_mitigation</system-policy-name>
 <rules>
  <rule-name>mitigate_http_and_https_flood_attack</rule-name>
  <condition-clause-container>
   <packet-security-ddos-condition>
    <pkt-sec-alert-rate>100</pkt-sec-alert-rate>
   </packet-security-ddos-condition>
  </condition-clause-container>
  <action-clause-container>
   <packet-action>
    <ingress-action>drop</ingress-action>
   </packet-action>
  </action-clause-container>
 </rules>
</system-policy>
</i2nsf-security-policy>
```

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

Figure 10 and Figure 11 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) is described in [i2nsf-advanced-nsf-dm].

General Firewall

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., 221.159.112.95) 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 http and https flood attack mitigation for additional inspection because the general firewall can not contrl the amount of packets for http and https packets.

HTTP and HTTPS Flood Attack Mitigation

1. The name of the system policy is http_and_https_flood_attack_mitigation.

2. The name of the rule is 100_per_second.

3. The rule controls the http and https packets according to the amount of incoming packets.

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

Appendix B. Changes from draft-ietf-i2nsf-nsf-facing-interface-dm-03

The following changes are made from draft-ietf-i2nsf-nsf-facing-interface-dm-04:

- We added fields for a rule (e.g., rule session aging time, rule long connection, and rule group).

- We added fields for a condition (e.g., payload, acl number, application, target, users, url category, context, and generic context)

Appendix C. Acknowledgments

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (No.R-20160222-002755, Cloud based Security Intelligence Technology Development for the Customized Security Service Provisioning).

Appendix D. Contributors

This document is made by the group effort of I2NSF working group. Many people actively contributed to this document. The following are considered co-authors:

- Hyoungshick Kim (Sungkyunkwan University)

- Daeyoung Hyun (Sungkyunkwan University)

- Dongjin Hong (Sungkyunkwan University)
- Liang Xia (Huawei)
- Tae-Jin Ahn (Korea Telecom)
- Se-Hui Lee (Korea Telecom)

Authors' Addresses

Jinyong Tim Kim
Department of 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
Department of Software
Sungkyunkwan University
2066 Seobu-Ro, Jangan-Gu
Suwon, Gyeonggi-Do 16419
Republic of Korea

Phone: +82 31 299 4957
Fax: +82 31 290 7996
EMail: pauljeong@skku.edu
URI: http://iotlab.skku.edu/people-jaehoon-jeong.php

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
USA

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

Qiushi Lin
Huawei
Huawei Industrial Base
Shenzhen, Guangdong 518129
China

EMail: linqiushi@huawei.com