Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 8, 2017

J. Jeong
J. Kim
D. Hyun
Sungkyunkwan University
J. Park
ETRI
T. Ahn
Korea Telecom

October 5, 2016

YANG Data Model of Interface to Network Security Functions Capability Interface
draft-jeong-i2nsf-capability-interface-yang-03

Abstract

This document defines a data model corresponding to the information model for Interface to Network Security Functions (I2NSF) capability interface. It describes a data model for three security capabilities (i.e., network security functions), such as network security control, content security control, and attack mitigation control, as defined in the information model for the I2NSF capability interface.

Table of Contents

1. Introduction

This document defines a YANG [RFC6020] model for security services with the information model of Interface to Network Security Functions (I2NSF) capability interface. It provides a specific information model and the corresponding data model for three security capabilities (i.e., network security functions), such as network security control, content security control, and attack mitigation control, as defined in [i2nsf-cap-interface-im].

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].

3. Terminology

This document uses the terminology described in [i2nsf-cap-interface-im][i2rs-rib-data-model] [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 [i2rs-rib-data-model] 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. Information Model Structure

Figure 1 shows an overview of a structure tree of network security control, content security control, and attack mitigation control, as defined in the [i2nsf-cap-interface-im].

```
module : ietf-i2nsf-capability-interface
  +--rw policy
    +--rw policy-name  string
    +--rw policy-id  string
    +--rw rule*  [rule-id]
      +--rw rule-name  string
      +--rw rule-id  uint 8
      +--rw event
      |  +--rw user-security-event* [usr-sec-event-id]
      |  |  +--rw usr-sec-event-id  uint 8
      |  |  +--rw usr-sec-event-content string
      |  |  +--rw usr-sec-event-format uint 8
      |  |  +--rw usr-sec-event-type uint 8
      |  +--rw device-security-event* [dev-sec-event-id]
      |  |  +--rw dev-sec-event-id  uint 8
      |  |  +--rw dev-sec-event-content string
      |  |  +--rw dev-sec-event-format uint 8
      |  |  +--rw dev-sec-event-type uint 8
      |  |  +--rw dev-sec-event-type-severity uint 8
      |  +--rw system-security-event* [sys-sec-event-id]
      |  |  +--rw sys-sec-event-id  uint 8
      |  |  +--rw sys-sec-event-content string
      |  |  +--rw sys-sec-event-format uint 8
      |  |  +--rw sys-sec-event-type uint 8
      |  +--rw time-security-event* [time-sec-event-id]
      |  |  +--rw time-sec-event-id  uint 8
      |  |  +--rw time-sec-event-period-begin yang:date-and-time
      |  |  +--rw time-sec-event-period-end yang:date-and-time
      |  |  +--rw time-sec-evnet-time-zone string
      +--rw condition
      |  +--rw packet-security-condition
      |  |  +--rw packet-security-mac-condition* [pkt-sec-cond-mac-id]
      |  |  |  +--rw pkt-sec-cond-mac-id uint 8
      |  |  |  +--rw pkt-sec-cond-mac-dest inet:port-number
      |  |  |  +--rw pkt-sec-cond-mac-src inet:port-number
      |  |  |  +--rw pkt-sec-cond-mac-8021q string
```

```
       |  |  |  +--rw pkt-sec-cond-mac-ether-type string
       |  |  |  +--rw pkt-sec-cond-mac-tci string
       |  |  +--rw packet-security-ipv4-condition* [pkt-sec-cond-ipv4-id]
       |  |  |  +--rw pkt-sec-cond-ipv4-id uint 8
       |  |  |  +--rw pkt-sec-cond-ipv4-src inet:ipv4-address
       |  |  |  +--rw pkt-sec-cond-ipv4-dest inet:ipv4-address
       |  |  |  +--rw pkt-sec-cond-ipv4-protocol string
       |  |  |  +--rw pkt-sec-cond-ipv4-dscp string
       |  |  |  +--rw pkt-sec-cond-ipv4-ecn string
       |  |  |  +--rw pkt-sec-cond-ipv4-length string
       |  |  |  +--rw pkt-sec-cond-ipv4-ttl
       |  |  +--rw packet-security-ipv6-condition* [pkt-sec-cond-ipv6-id]
       |  |  |  +--rw pkt-sec-cond-ipv6-id uint 8
       |  |  |  +--rw pkt-sec-cond-ipv6-src inet:ipv6-address
       |  |  |  +--rw pkt-sec-cond-ipv6-dest inet:ipv6-address
       |  |  |  +--rw pkt-sec-cond-ipv6-dscp string
       |  |  |  +--rw pkt-sec-cond-ipv6-ecn string
       |  |  |  +--rw pkt-sec-cond-ipv6-flow-label string
       |  |  |  +--rw pkt-sec-cond-ipv6-payload-length string
       |  |  |  +--rw pkt-sec-cond-ipv6-next-header string
       |  |  |  +--rw pkt-sec-cond-ipv6-hop-limit string
       |  |  +--rw packet-security-tcp-condition* [pkt-sec-cond-tcp-id]
       |  |  |  +--rw pkt-sec-cond-tcp-id uint 8
       |  |  |  +--rw pkt-sec-cond-tcp-src-port inet:port-number
       |  |  |  +--rw pkt-sec-cond-tcp-dest-port inet:port-number
       |  |  |  +--rw pkt-sec-cond-tcp-seq-num string
       |  |  |  +--rw pkt-sec-cond-tcp-falgs string
       |  |  +--rw packet-security-udp-condition* [pkt-sec-cond-udp-id]
       |  |     +--rw pkt-sec-cond-udp-id uint 8
       |  |     +--rw pkt-sec-cond-udp-src-port inet:port-number
       |  |     +--rw pkt-sec-cond-udp-dest-port inet:port-number
       |  |     +--rw pkt-sec-cond-udp-length string
       |  +--rw packet-payload-security-condition* [pkt-payload-id]
       |  |  +--rw pkt-payload-id uint 8
       |  +--rw target-security-condition* [target-sec-cond-id]
       |  |  +--rw target-sec-cond-id uint 8
       |  |  +--rw service-sec-context-cond?
       |  |  |  +--rw name string
       |  |  |  +--rw protocol
       |  |  |  |  +--rw TCP? boolean
       |  |  |  |  +--rw UDP? boolean
       |  |  |  |  +--rw ICMP? boolean
       |  |  |  |  +--rw ICMPv6? boolean
       |  |  |  |  +--rw IP? boolean
       |  |  |  +--rw src-port? inet:port-number
       |  |  |  +--rw dest-port? inet:port-number
       |  |  +--rw application-sec-context-cond?
       |  |  |  +--rw name string
```

```
      |  |  |  +--rw category
      |  |  |  |  +--rw business-system?  boolean
      |  |  |  |  +--rw entertainment?  boolean
      |  |  |  |  +--rw internet?  boolean
      |  |  |  |  +--rw network?  boolean
      |  |  |  |  +--rw general?  boolean
      |  |  |  +--rw subcategory
      |  |  |  |  +--rw finance?  boolean
      |  |  |  |  +--rw email?  boolean
      |  |  |  |  +--rw game?  boolean
      |  |  |  |  +--rw media-sharing?  boolean
      |  |  |  |  +--rw social-network?  boolean
      |  |  |  |  +--rw web-posting?  boolean
      |  |  |  +--rw data-transmission-model
      |  |  |  |  +--rw client-server?  boolean
      |  |  |  |  +--rw browser-based?  boolean
      |  |  |  |  +--rw networking?  boolean
      |  |  |  |  +--rw peer-to-peer?  boolean
      |  |  |  |  +--rw unassigned?  boolean
      |  |  |  +--rw risk-level
      |  |  |     +--rw exploitable?  boolean
      |  |  |     +--rw productivity-loss?  boolean
      |  |  |     +--rw evasive?  boolean
      |  |  |     +--rw data-loss?  boolean
      |  |  |     +--rw malware-vehicle?  boolean
      |  |  |     +--rw bandwidth-consuming?  boolean
      |  |  |     +--rw tunneling?  boolean
      |  |  +--rw device-sec-context-cond?
      |  |     +--rw pc?  boolean
      |  |     +--rw mobile-phone?  boolean
      |  |     +--rw tablet?  boolean
      |  |     +--rw voip-phone  boolean
      |  +--rw user-security-cond* [usr-sec-cond-id]
      |  |  +--rw usr-sec-cond-id uint 8
      |  |     +--rw user
      |  |     |  +--rw (user-name)?
      |  |     |    +--: (tenant)
      |  |     |    |  +--rw tenant uint 8
      |  |     |    +--: (vn-id)
      |  |     |       +--rw vn-id uint 8
      |  |     +--rw group
      |  |        +--rw (group-name)?
      |  |          +--: (tenant)
      |  |          |  +--rw tenant uint 8
      |  |          +--: (vn-id)
      |  |             +--rw vn-id uint 8
      |  +--rw security-context-condition* [sec-context-cond-id]
      |  |  +--rw sec-context-cond-id uint 8
```

```
      |  |  +--rw (state)?
      |  |  |  +--: (session-state)
      |  |  |  |  +--rw tcp-session-state
      |  |  |  |     +--rw new?  boolean
      |  |  |  |     +--rw established?  boolean
      |  |  |  |     +--rw related?  boolean
      |  |  |  |     +--rw invalid?  boolean
      |  |  |  |     +--rw untracked?  boolean
      |  |  |  +--: (session-aaa-state)
      |  |  |  |  +--rw session-sip-state
      |  |  |  |     +--rw auth-state?  boolean
      |  |  |  |     +--rw call-state?  boolean
      |  |  |  +--: (access-mode)
      |  |  |  |  +--rw access-mode  string
      |  +--rw generic-context-condition* [gen-context-cond-id]
      |     +--rw gen-context-cond-id uint 8
      |     +--rw geographic-location
      |     |  +--rw geographic-location-id* uint 8
     +--rw action
        +--rw (action-type)?
           +--: (ingress-action)
           |  +--rw (ingress-action-type)?
           |    +--: (permit)
           |    |  +--rw permit boolean
           |    +--: (deny)
           |    |  +--rw deny boolan
           |    +--: (mirror)
           |       +--rw mirror boolean
           +--: (egress-action)
           |  +--rw (egress-action-type)?
           |    +--: (invoke-signaling)
           |    |  +--rw invoke-signaling boolean
           |    +--: (tunnel-encapsulation)
           |    |  +--rw tunnel-encapsulation boolean
           |    +--: (forwarding)
           |       +--rw forwarding  boolean
           +--: (apply-profile-action)
              +--rw (apply-profile-action-type)?
                +--: (content-security-control)
                |  +--rw (content-security-control-type)?
                |     +--: (antivirus)
                |     |  +--rw antivirus?  boolean
                |     +--: (ips)
                |     |  +--rw ips?  boolean
                |     +--: (url-filtering)
                |     |  +--rw url-filtering?  boolean
                |     +--: (file-blocking)
                |     |  +--rw file-blocking?  boolean
```

```
|     +--: (data-filtering)
|     |  +--rw data-filtering?  boolean
|     +--: (application-control)
|     |  +--rw application-control?  boolean
|     +--: (voip-volte)
|        +--rw voip-volte-rule*  [voip-volte-rule-id]
|           +--rw voip-volte-rule-id  uint 8
|           +--rw event
|           |  +--rw called-voip  boolean
|           |  +--rw called-volte  boolean
|           +--rw condition
|           |  +--rw sip-header*  [sip-header-uri]
|           |  |  +--rw sip-header-uri string
|           |  |  +--rw sip-header-method string
|           |  |  +--rw expire-time yang:date-and-time
|           |  |  +--rw sip-header-user-agent uint32
|           |  +--rw cell-region?* [cell-id-region]
|           |     +--rw cell-id-region uint 32
|           +--rw action
|              +--rw (action-type)?
|                 +--: (ingress-action)
|                 |  +--rw (ingress-action-type)?
|                 |    +--: (permit)
|                 |    |  +--rw permit boolean
|                 |    +--: (deny)
|                 |    |  +--rw deny boolean
|                 |    +--: (mirror)
|                 |       +--rw mirror boolean
|                 +--: (egress-action)
|                    +--: (egress-action-type)?
|                      +--: (redirection)
|                         +--rw redirection? boolean
+--: (attack-mitigation-control)
   +--rw (attack-mitigation-control-type)?
      +--: (ddos-attack)
      |  +--rw (ddos-attack-type)?
      |     +--: (network-layer-ddos-attack)
      |     |  +--rw (network-layer-ddos-attack-type)?
      |     |     +--: (syn-flood-attack)
      |     |     |  +--rw syn-flood    boolean
      |     |     +--: (udp-flood-attack)
      |     |     |  +--rw udp-flood    boolean
      |     |     +--: (icmp-flood-attack)
      |     |     |  +--rw icmp-flood    boolean
      |     |     +--: (ip-fragment-flood-attack)
      |     |     |  +--rw ip-fragment-flood boolean
      |     |     +--: (ipv6-related-attacks)
      |     |        +--rw ipv6-related    boolean
```

```
                        |     +--: (app-layer-ddos-attack)
                        |        +--rw (app-layer-ddos-attack-type)?
                        |           +--: (http-flood-attack)
                        |           |  +--rw http-flood    boolean
                        |           +--: (https-flood-attack)
                        |           |  +--rw https-flood    boolean
                        |           +--: (dns-flood-attack)
                        |           |  +--rw dns-flood    boolean
                        |           +--: (dns-amp-flood-attack)
                        |           |  +--rw dns-amp-flood boolean
                        |           +--: (ssl-ddos-attack)
                        |              +--rw ssl-ddos    boolean
                        +--: (single-packet-attack)
                           +--rw (single-packet-attack-type)?
                              +--: (scan-and-sniff-attack)
                              |  +--rw (scan-and-sniff-attack-type)?
                              |  |  +--: (ip-sweep-attack)
                              |  |  |  +--rw ip-sweep    boolean
                              |  |  +--: (port-scanning-attack)
                              |  |  |  +--rw port-scanning    boolean
                              |  +--: (malformed-packet-attack)
                              |  |  +--rw (malformed-packet-attack-type)?
                              |  |  |  +--: (ping-of-death-attack)
                              |  |  |  |  +--rw ping-of-death    boolean
                              |  |  |  +--: (teardrop-attack)
                              |  |  |  |  +--rw teardrop    boolean
                              |  +--: (special-packet-attack)
                              |     +--rw (special-packet-attack-type)?
                              |        +--: (oversized-icmp-attack)
                              |        | +--rw oversized-icmp    boolean
                              |        +--: (tracert-attack)
                              |           +--rw tracert    boolean
```

Figure 1: Information Model of I2NSF Capability Interface

5. YANG Model

This section introduces a YANG model for the information model of network security functions, as defined in the [i2nsf-cap-interface-im].

<CODE BEGINS> file "ietf-i2nsf-capability-interface@2016-10-05.yang"

```
module ietf-i2nsf-capability-interface {
  namespace
    "urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability-interface";
  prefix
```

```
    capability-interface;

  import ietf-inet-types{
    prefix inet;
  }
  import ietf-yang-types{
    prefix yang;
  }


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

  description
    "This module defines a YANG data module for network security
     functions.";
  revision "2016-10-05"{
    description "Initial revision";
    reference
      "draft-xia-i2nsf-capability-interface-im-06";
  }

  //Groupings
  grouping policy {
    description
      "policy is a grouping
       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.";

      leaf policy-name {
        type string;
        mandatory true;
```

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

      list rule {
        key "rule-id";
        description
          "This is a rule for network security control.";

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

          leaf rule-id {
            type uint8;
            mandatory true;
            description
              "The ID of the rule.
               This is key for rule-list.
               This must be unique.";
          }

          container event {
            description
              " An Event is defined as any important occurrence in time
                of a change in the system being managed, and/or in the
                environment of the system being managed. When used in
                the context of policy rules for a flow-based NSF, it is
                used to determine whether the Condition clause of the
                Policy Rule can be evaluated or not. Examples of an
                I2NSF Event include time and user actions (e.g., logon,
                logoff, and actions that violate any ACL.).";
              list user-security-event {
                key usr-sec-event-id;
                description
                  "The purpose of this class is to represent Events that
```

```
                   are initiated by a user, such as logon and logoff
                   Events. Information in this Event may be used as part
                   of a test to determine if the Condition clause in
                   this ECA Policy Rule should be evaluated or not.
                   Examples include user identification data and the
                   type of connection used by the user.";

                leaf usr-sec-event-id {
                  type uint8;
                  mandatory true;
                  description
                    "The ID of the usr-sec-event.
                     This is key for usr-sec-event-list.
                     This must be unique.";
                }

                leaf usr-sec-event-content {
                  type string;
                  mandatory true;
                  description
                   "This is a mandatory string that contains the content
                    of the UserSecurityEvent. The format of the content
                    is specified in the usrSecEventFormat class
                    attribute, and the type of Event is defined in the
                    usrSecEventType class attribute. An example of the
                    usrSecEventContent attribute is a string hrAdmin
                    with the usrSecEventFormat set to 1 (GUID) and the
                    usrSecEventType attribute set to 5 (new logon).";
                }

                leaf usr-sec-event-format {
                  type uint8;
                  mandatory true;
                  description
                   "This is a mandatory uint 8 enumerated integer, which
                    is used to specify the data type of the
                    usrSecEventContent attribute. The content is
                    specified in the usrSecEventContent class attribute,
                    and the type of Event is defined in the
                    usrSecEventType class attribute. An example of the
                    usrSecEventContent attribute is string hrAdmin with
                    the usrSecEventFormat attribute set to 1 (GUID) and
                    the usrSecEventType attribute set to 5 (new logon).
                   ";
                }

                leaf usr-sec-event-type {
                  type uint8;
```

```
        mandatory true;
        description
         "This is a mandatory uint 8 enumerated integer, which
          is used to specify the type of Event that involves
          this user. The content and format are specified in
          the usrSecEventContent and usrSecEventFormat class
          attributes, respectively. An example of the
          usrSecEventContent attribute is string hrAdmin
          with the usrSecEventFormat attribute set to 1 (GUID)
          and the usrSecEventType attribute set to 5 (new
          logon).";
      }
    }

    list device-security-event {
      key dev-sec-event-id;
      description
       "The purpose of a DeviceSecurityEvent is to represent
        Events that provide information from the Device that
        are important to I2NSF Security. Information in this
        Event may be used as part of a test to determine if
        the Condition clause in this ECA Policy Rule should be
        evaluated or not. Examples include alarms and various
        device statistics (e.g., a type of threshold that was
        exceeded), which may signal the need for further
        action.";

      leaf dev-sec-event-id {
        type uint8;
        mandatory true;
        description
          "The ID of the dev-sec-event.
           This is key for dev-sec-event-list.
           This must be unique.";
      }

      leaf dev-sec-event-content {
        type string;
        mandatory true;
        description
         "This is a mandatory string that contains the content
          of the DeviceSecurityEvent. The format of the
          content is specified in the devSecEventFormat class
          attribute, and the type of Event is defined in the
          devSecEventType class attribute. An example of the
          devSecEventContent attribute is alarm with the
          devSecEventFormat attribute set to 1 (GUID) and the
          devSecEventType attribute set to 5 (new logon).";
```

```
                }

                 leaf dev-sec-event-format {
                   type uint8;
                   mandatory true;
                   description
                    "This is a mandatory uint 8 enumerated integer, which
                     is used to specify the data type of the
                     devSecEventContent attribute.";
                 }

                 leaf dev-sec-event-type {
                   type uint8;
                   mandatory true;
                   description
                    "This is a mandatory uint 8 enumerated integer, which
                     is used to specify the type of Event that was
                     generated by this device.";
                 }

                 leaf dev-sec-event-type-severity {
                   type uint8;
                   mandatory true;
                   description
                    "This is a mandatory uint 8 enumerated integer, which
                     is used to specify the perceived severity of the
                     Event generated by this Device.";
                 }
               }

               list system-security-event {
                 key sys-sec-event-id;
                 description
                   "The purpose of a SystemSecurityEvent is to represent
                    Events that are detected by the management system,
                    instead of Events that are generated by a user or a
                    device. Information in this Event may be used as part
                    of a test to determine if the Condition clause in
                    this ECA Policy Rule should be evaluated or not.
                    Examples include an event issued by an analytics
                    system that warns against a particular pattern of
                    unknown user accesses, or an Event issued by a
                    management system that represents a set of correlated
                    and/or filtered Events.";

                 leaf sys-sec-event-id {
                   type uint8;
                   mandatory true;
```

```
        description
          "The ID of the sys-sec-event.
           This is key for sys-sec-event-list.
           This must be unique.";
      }

      leaf sys-sec-event-content {
        type string;
        mandatory true;
        description
         "This is a mandatory string that contains a content
          of the SystemSecurityEvent. The format of a content
          is specified in a sysSecEventFormat class attribute,
          and the type of Event is defined in the
          sysSecEventType class attribute. An example of the
          sysSecEventContent attribute is string sysadmin3
          with the sysSecEventFormat attribute set to 1 (GUID)
          and the sysSecEventType attribute set to 2 (audit
          log cleared).";
      }

      leaf sys-sec-event-format {
        type uint8;
        mandatory true;
        description
         "This is a mandatory uint 8 enumerated integer, which
          is used to specify the data type of the
          sysSecEventContent attribute.";
      }

      leaf sys-sec-event-type {
        type uint8;
        mandatory true;
        description
         "This is a mandatory uint 8 enumerated integer, which
          is used to specify the type of Event that involves
          this device.";
      }
    }

    list time-security-event {
      key time-sec-event-id;
      description
      "Purpose of a TimeSecurityEvent is to represent Events
       that are temporal in nature (e.g., the start or end of
       a period of time). Time events signify an individual
       occurrence, or a time period, in which a significant
       event happened. Information in the Event may be used as
```

```
                part of a test to determine if the Condition clause in
                this ECA Rule should be evaluated or not. Examples
                include issuing an Event at a specific time to indicate
                that a particular resource should not be accessed, or
                that different authentication and authorization
                mechanisms should now be used (e.g., because it is now
                past regular business hours).";

              leaf time-sec-event-id {
                type uint8;
                mandatory true;
                description
                  "The ID of the time-sec-event.
                   This is key for time-sec-event-list.
                   This must be unique.";
              }

              leaf time-sec-event-period-begin {
                type yang:date-and-time;
                mandatory true;
                description
                  "This is a mandatory DateTime attribute, and
                  represents the beginning of a time period.
                  It has a value that has a date and/or a time
                  component (as in the Java or Python libraries).";
              }

              leaf time-sec-event-period-end {
                type yang:date-and-time;
                mandatory true;
                description
                  "This is a mandatory DateTime attribute, and
                   represents the end of a time period. It has
                   a value that has a date and/or a time component
                   (as in the Java or Python libraries). If this is
                   a single Event occurrence, and not a time period
                   when the Event can occur, then the
                   timeSecEventPeriodEnd attribute may be ignored.";
              }

              leaf time-sec-event-time-zone {
                type string;
                mandatory true;
                description
                  "This is a mandatory string attribute, and defines a
                   time zone that this Event occurred in using the
                   format specified in ISO8601.";
              }
```

```
        }
      }
      container condition {
        description
          "TBD";
        container packet-security-condition {
          description
            "The purpose of this Class is to represent packet header
             information that can be used as part of a test to
             determine if the set of Policy Actions in this ECA
             Policy Rule should be executed or not. This class is
             abstract, and serves as the superclass of more detailed
             conditions that involve different types of packet
             formats.";

          list packet-security-mac-condition {
            key pkt-sec-cond-mac-id;
            description
              "The purpose of this Class is to represent packet MAC
               packet header information that can be used as part of
               a test to determine if the set of Policy Actions in
               this ECA Policy Rule should be executed or not.";

            leaf pkt-sec-cond-mac-id {
              type uint8;
              mandatory true;
              description
                "The ID of the pkt-sec-cond-mac.
                 This is key for pkt-sec-cond-mac-list.
                 This must be unique.";
            }

            leaf pkt-sec-cond-mac-dest {
              type inet:port-number;
              mandatory true;
              description
                "This is a mandatory uint 32 attribute, and defines
                 the MAC destination address (6 octets long).";
            }

            leaf pkt-sec-cond-mac-src {
              type inet:port-number;
              mandatory true;
              description
                "This is a mandatory uint 32 attribute, and defines
                 the MAC source address (6 octets long).";
            }
```

```
        leaf pkt-sec-cond-mac-8021q {
          type string;
          mandatory true;
          description
            "This is an optional string attribute, and defines
             the 802.1Q tag value (2 octets long). This defines
             VLAN membership and 802.1p priority values.";
        }

        leaf pkt-sec-cond-mac-ether-type {
          type string;
          mandatory true;
          description
            "This is a mandatory string attribute, and defines
             the EtherType field (2 octets long). Values up to
             and including 1500 indicate the size of the payload
             in octets; values of 1536 and above define which
             protocol is encapsulated in the payload of the
             frame.";
        }

        leaf pkt-sec-cond-mac-tci {
          type string;
          mandatory true;
          description
            "This is an optional string attribute, and defines
             the Tag Control Information. This consists of a 3
             bit user priority field, a drop eligible indicator
             (1 bit), and a VLAN identifier (12 bits).";
        }
      }

      list packet-security-ipv4-condition {
        key pkt-sec-cond-ipv4-id;
        description
          "The purpose of this Class is to represent packet IPv4
           packet header information that can be used as part of
           a test to determine if the set of Policy Actions in
           this ECA Policy Rule should be executed or not.";

        leaf pkt-sec-cond-ipv4-id {
          type uint8;
          mandatory true;
          description
            "The ID of the pkt-sec-cond-ipv4.
             This is key for pkt-sec-cond-ipv4-list.
             This must be unique.";
        }
```

```
                leaf pkt-sec-cond-ipv4-src {
                  type inet:ipv4-address;
                  mandatory true;
                  description
                    "This is a mandatory inet:ipv4-address attribute,
                     and defines the IPv4 Source Address (32 bits).";
                }

                leaf pkt-sec-cond-ipv4-dest {
                  type inet:ipv4-address;
                  mandatory true;
                  description
                    "This is a mandatory inet:ipv4-address attribute,
                     and defines the IPv4 Destination Address
                     (32 bits).";
                }

                leaf pkt-sec-cond-ipv4-protocol {
                  type string;
                  mandatory true;
                  description
                    "This is a mandatory string attribute, and defines
                     he protocol used in the data portion of the IP
                     datagram (8 bits).";
                }

                leaf pkt-sec-cond-ipv4-dscp {
                  type string;
                  mandatory true;
                  description
                    "This is a mandatory string attribute, and defines
                     the Differentiated Services Code Point field
                     (6 bits).";
                }

                leaf pkt-sec-cond-ipv4-ecn {
                  type string;
                  mandatory true;
                  description
                    "This is an optional string attribute, and defines
                     the Explicit Congestion Notification field
                     (2 bits).";
                }

                leaf pkt-sec-cond-ipv4-length {
                  type string;
                  mandatory true;
                  description
```

```
                    "This is a mandatory string attribute, and defines
                     the total length of the packet (including header
                     and data) in bytes (16 bits).";
                  }

                  leaf pkt-sec-cond-ipv4-ttl {
                    type string;
                    mandatory true;
                    description
                      "This is a mandatory string attribute, and defines
                       the Time To Live in seconds (8 bits).";
                  }
                }

                list packet-security-ipv6-condition {
                  key pkt-sec-cond-ipv6-id;
                  description
                     "The purpose of this Class is to represent packet
                     IPv6 packet header information that can be used as
                     part of a test to determine if the set of Policy
                     Actions in this ECA Policy Rule should be executed
                     or not.";

                  leaf pkt-sec-cond-ipv6-id {
                    type uint8;
                    mandatory true;
                    description
                      "The ID of the pkt-sec-cond-ipv6.
                       This is key for pkt-sec-cond-ipv6-list.
                       This must be unique.";
                  }

                  leaf pkt-sec-cond-ipv6-src {
                    type inet:ipv6-address;
                    mandatory true;
                    description
                      "This is a mandatory inet:ipv6-address attribute,
                       and defines the IPv6 Source Address (128 bits).";
                  }

                  leaf pkt-sec-cond-ipv6-dest {
                    type inet:ipv6-address;
                    mandatory true;
                    description
                      "This is a mandatory inet:ipv6-address attribute,
                       and defines the IPv6 Destination Address
                       (128 bits).";
                  }
```

```
                leaf pkt-sec-cond-ipv6-dscp {
                  type string;
                  mandatory true;
                  description
                    "This is a mandatory string attribute, and defines
                     the Differentiated Services Code Point field
                     (6 bits). It consists of the six most significant
                     bits of the Traffic Class field in the IPv6
                     header.";
                }

                leaf pkt-sec-cond-ipv6-ecn {
                  type string;
                  mandatory true;
                  description
                   "This is a mandatory string attribute, and defines
                    the Explicit Congestion Notification field (2 bits).
                    It consists of the two least significant bits of
                    the Traffic Class field in the IPv6 header.";
                }

                leaf pkt-sec-cond-ipv6-flow-label {
                  type string;
                  mandatory true;
                  description
                    "This is a mandatory string attribute, and defines
                     an IPv6 flow label. This, in combination with the
                     Source and Destination Address fields, enables
                     efficient IPv6 flow classification by using only
                     the IPv6 main header fields (20 bits).";
                }

                leaf pkt-sec-cond-ipv6-payload-length {
                  type string;
                  mandatory true;
                  description
                    "This is a mandatory string attribute, and defines
                     the total length of the packet (including the
                     fixed and any extension headers, and data) in
                     bytes (16 bits).";
                }

                leaf pkt-sec-cond-ipv6-next-header {
                  type string;
                  mandatory true;
                  description
                    "This is a mandatory string attribute, and defines
                     the type of the next header (e.g., which extension
```

```
                    header to use) (8 bits).";
                }

                leaf pkt-sec-cond-ipv6-hop-limit {
                  type string;
                  mandatory true;
                  description
                    "This is a mandatory string attribute, and defines
                     the maximum number of hops that this packet can
                     traverse (8 bits).";
                }
              }

              list packet-security-tcp-condition {
                key pkt-sec-cond-tcp-id;
                description
                  "The purpose of this Class is to represent packet
                   TCP packet header information that can be used as
                   part of a test to determine if the set of Policy
                   Actions in this ECA Policy Rule should be executed
                   or not.";

                leaf pkt-sec-cond-tcp-id {
                  type uint8;
                  mandatory true;
                  description
                    "The ID of the pkt-sec-cond-tcp.
                     This is key for pkt-sec-cond-tcp-list.
                     This must be unique.";
                }

                leaf pkt-sec-cond-tcp-src-port {
                  type inet:port-number;
                  mandatory true;
                  description
                    "This is a mandatory port attribute, and defines
                     the Source Port (16 bits).";
                }

                leaf pkt-sec-cond-tcp-dest-port {
                  type inet:port-number;
                  mandatory true;
                  description
                    "This is a mandatory port attribute, and defines
                     the Destination Port (16 bits).";
                }

                leaf pkt-sec-cond-tcp-seq-num {
```

```
        type string;
        mandatory true;
        description
          "This is a mandatory string attribute, and defines
           the sequence number (32 bits).";
      }

      leaf pkt-sec-cond-tcp-falgs {
        type string;
        mandatory true;
        description
          "This is a mandatory string attribute, and defines
           the nine Control bit flags (9 bits).";
      }
    }

    list packet-security-udp-condition {
      key pkt-sec-cond-udp-id;
      description
        "The purpose of this Class is to represent packet UDP
         packet header information that can be used as part
         of a test to determine if the set of Policy Actions
         in this ECA Policy Rule should be executed or not.";

      leaf pkt-sec-cond-udp-id {
          type uint8;
          mandatory true;
          description
            "The ID of the pkt-sec-cond-udp.
             This is key for pkt-sec-cond-udp-list.
             This must be unique.";
      }

      leaf pkt-sec-cond-udp-src-port {
        type inet:port-number;
        mandatory true;
        description
          "This is a mandatory port attribute, and defines
           the UDP Source Port (16 bits).";
      }

      leaf pkt-sec-cond-udp-dest-port {
        type inet:port-number;
        mandatory true;
        description
          "This is a mandatory port attribute, and defines
          the UDP Destination Port (16 bits).";
      }
```

```
            leaf pkt-sec-cond-udp-length {
              type string;
              mandatory true;
              description
                "This is a mandatory string attribute, and defines
                 the length in bytes of the UDP header and data
                 (16 bits).";
            }
          }
        }
          list packet-payload-security-condition {
            key "pkt-payload-id";
            description
              "The ID of the pkt-payload.
               This is key for pkt-payload-list.
               This must be unique.";
            leaf pkt-payload-id {
              type uint8;
              mandatory true;
              description
                "The ID of the packet payload.
                 This must be unique.";
            }
          }
          list target-security-condition {
            key "target-sec-cond-id";
            description
              "Under the circumstances of network, it mainly
               refers to the service, application, and device.";
            leaf target-sec-cond-id {
              type uint8;
              mandatory true;
              description
                "The ID of the target.
                 This must be unique.";
            }
            container service-sec-context-cond{
              description
                "A service is an application identified by a
                 protocol type and port number, such as TCP,
                 UDP, ICMP, and IP.";
              leaf name {
                type string;
                mandatory true;
                description
                  "The name of the service.
                   This must be unique.";
              }
```

```
                leaf id {
                  type uint8;
                  mandatory true;
                  description
                    "The ID of the service.
                     This must be unique.";
                }
                container protocol {
                  description
                    "Protocol types:
                     TCP, UDP, ICMP, ICMPv6, IP, and etc.";
                  leaf tcp  {
                    type boolean;
                    mandatory true;
                    description
                      "TCP protocol type.";
                  }
                  leaf udp {
                    type boolean;
                    mandatory true;
                    description
                      "UDP protocol type.";
                  }
                  leaf icmp {
                    type boolean;
                    mandatory true;
                    description
                      "ICMP protocol type.";
                  }
                  leaf icmpv6 {
                    type boolean;
                    mandatory true;
                    description
                      "ICMPv6 protocol type.";
                  }
                  leaf ip {
                    type boolean;
                    mandatory true;
                    description
                      "IP protocol type.";
                  }
                }
                leaf src-port{
                  type inet:port-number;
                  description
                    "It can be used for finding programs.";
                }
                leaf dest-port{
```

```
        type inet:port-number;
        description
          "It can be used for finding programs.";
      }
    }
    container application-sec-context-cond {
      description
        "An application is a computer program for
         a specific task or purpose. It provides
         a finer granularity than service in matching
         traffic.";
      leaf name{
        type string;
        mandatory true;
        description
          "The name of the application.
           This must be unique.";
      }
      leaf id{
        type uint8;
        mandatory true;
        description
          "The ID of the application.
           This must be unique.";
      }
      container category{
        description
          "Category types: Business system, Entertainment,
           Interest, Network, General, and etc.";
        leaf business-system {
          type boolean;
          description
            "Business system category.";
        }
        leaf entertainment {
          type boolean;
          description
            "Entertainment category.";
        }
        leaf interest {
          type boolean;
          description
            "Interest category.";
        }
        leaf network {
          type boolean;
          description
            "Network category.";
```

```
    }
    leaf general {
      type boolean;
      description
        "General category.";
    }
  }
  container subcategory{
    description
      "Subcategory types: Finance, Email, Game,
       Media sharing, Social network, Web posting,
       and etc.";
    leaf finance {
      type boolean;
      description
        "Finance subcategory.";
    }
    leaf email {
      type boolean;
      description
        "Email subcategory.";
    }
    leaf game {
      type boolean;
      description
        "Game subcategory.";
      }
    leaf media-sharing {
      type boolean;
      description
        "Media sharing subcategory.";
    }
    leaf social-network {
      type boolean;
      description
        "Social network subcategory.";
    }
    leaf web-posting {
      type boolean;
      description
        "Web posting subcategory.";
    }
  }
  container data-transmission-model{
    description
      "Data transmission model types: Client-server,
       Browser-based, Networking, Peer-to-Peer,
       Unassigned, and etc.";
```

```
                  leaf client-server {
                    type boolean;
                    description
                      "client-server data transmission model.";
                  }
                  leaf browser-based {
                    type boolean;
                    description
                      "Browser-based data transmission model.";
                  }
                  leaf networking {
                    type boolean;
                    description
                      "Networking data transmission model.";
                  }
                  leaf peer-to-peer {
                    type boolean;
                    description
                      "Peer-to-Peer data transmission model.";
                  }
                  leaf unassigned {
                    type boolean;
                    description
                      "Unassigned data transmission model.";
                  }
                }
                container risk-level{
                  description
                    "Risk level types: Exploitable,
                     Productivity loss, Evasive, Data loss,
                     Malware vehicle, Bandwidth consuming,
                     Tunneling, and etc.";
                  leaf exploitable {
                    type boolean;
                    description
                      "Exploitable risk level.";
                  }
                  leaf productivity-loss {
                    type boolean;
                    description
                      "Productivity loss risk level.";
                  }
                  leaf evasive {
                    type boolean;
                    description
                      "Evasive risk level.";
                  }
                  leaf data-loss {
```

```
                    type boolean;
                    description
                      "Data loss risk level.";
                  }
                  leaf malware-vehicle {
                    type boolean;
                    description
                      "Malware vehicle risk level.";
                  }
                  leaf bandwidth-consuming {
                    type boolean;
                    description
                      "Bandwidth consuming risk level.";
                  }
                  leaf tunneling {
                    type boolean;
                    description
                      "Tunneling risk level.";
                  }
                }
              }
              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 pc {
                  type boolean;
                  description
                    "If type of a device is PC.";
                }
                leaf mobile-phone {
                  type boolean;
                  description
                    "If type of a device is mobile-phone.";
                  }
                leaf tablet {
                  type boolean;
                  description
                    "If type of a device is tablet.";
                }
                leaf voip-volte-phone {
                  type boolean;
                  description
                    "If type of a device is voip-volte-phone.";
                }
              }
```

```
            }
            list user-security-cond {
              key "usr-sec-cond-id";
              description
                "TBD";
              leaf usr-sec-cond-id {
                type uint8;
                description
                  "The ID of the user-sec-cond.
                   This is key for user-sec-cond-list.
                   This must be unique.";
              }
              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.";
            }
          }
        }
      }
      list generic-context-condition {
        key "gen-context-cond-id";
        description
          "TBD";
        leaf gen-context-cond-id {
          type uint8;
          description
```

```
                  "The ID of the gen-context-cond.
                   This is key for gen-context-cond-list.
                   This must be unique.";
                }
                container geographic-location {
                  description
                    "The location which 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 geographic-location {
                    type uint8;
                    description
                      "This is mapped to ip address. We can acquire
                       region through ip address stored the database.";
                  }
                }
              }
            }
            container action {
              description
                "TBD.";
              choice action-type {
              description
                "The flow-based NSFs realize the network security
                 functions by executing various Actions, which at least
                 includes ingress-action, egress-action, and
                 advanced-action.";
              case ingress-action {
                description
                  "The ingress actions consist of permit, deny,
                   and mirror.";
                choice ingress-action-type {
                  description
                    "Ingress action type: permit, deny, and mirror.";
                  case permit {
                    description
                      "Permit case.";
                    leaf permit {
                      type boolean;
                      mandatory true;
                      description
                        "Packet flow is permitted.";
                    }
                  }
                  case deny {
                    description
```

```
                  "Deny case.";
                leaf deny {
                  type boolean;
                  mandatory true;
                  description
                    "Packet flow is denied.";
                }
              }
              case mirror {
                description
                  "Mirror case.";
                leaf mirror {
                  type boolean;
                  mandatory true;
                  description
                    "Packet flow is mirroried.";
                }
              }
            }
          }
          case egress-action {
            description
              "The egress actions consist of invoke-signaling,
              tunnel-encapsulation, and forwarding.";
            choice egress-action-type {
              description
                "Egress-action-type: invoke-signaling,
                 tunnel-encapsulation, and forwarding.";
              case invoke-signaling {
                description
                  "Invoke-signaling case.";
                leaf invoke-signaling {
                  type boolean;
                  mandatory true;
                  description
                    "TBD.";
                }
              }
              case tunnel-encapsulation {
                description
                  "tunnel-encapsulation case.";
                leaf tunnel-encapsulation {
                  type boolean;
                  mandatory true;
                  description
                    "TBD.";
                }
              }
```

```
            case forwarding {
              description
                "forwarding case.";
              leaf forwarding {
                type boolean;
                mandatory true;
                description
                  "TBD.";
              }
            }
          }
        }
        case apply-profile-action {
          description
            "Applying a specific Functional Profile or signature
             - e.g., an IPS Profile, a signature file, an
             anti-virus file, or a URL filtering file. The
             functional profile or signature file corresponds to
             the security capability for the content security
             control and attack mitigation control which will be
             described afterwards. It is one of the key properties
             that determine the effectiveness of the NSF, and is
             mostly vendor specific today. One goal of I2NSF is
             to standardize the form and functional interface of
             those security capabilities while supporting vendor-
             specific implementations of each.";
          choice apply-profile-action-type {
            description
              "Advanced action types: Content Security Control
               and Attack Mitigation Control.";
            case content-security-control {
              description
                "Content security control is another category of
                security capabilities applied to application layer.
                Through detecting the contents carried over the
                traffic in application layer, these capabilities
                can realize various security purposes, such as
                defending against intrusion, inspecting virus,
                filtering malicious URL or junk email, and blocking
                illegal web access or data retrieval.";
              choice content-security-control-type {
                description
                 "Content Security types: Antivirus, IPS,
                  url-filtering file-blocking, data-filtering,
                  application-control, and voip-volte.";
                case antivirus {
                  leaf antivirus {
                    type boolean;
```

```
        description
          "Antivirus is computer software used to
           prevent, detect and remove malicious
           software.";
      }
    }
    case ips {
      leaf ips {
        type boolean;
        description
          "Intrusion prevention systems (IPS) are
           network security appliances that monitor
           network and/or system activities for
           malicious activities.";
      }
    }
    case url-filtering {
      leaf url-filtering {
        type boolean;
        description
          "URL filtering security service.";
      }
    }
    case file-blocking {
      leaf file-blocking {
        type boolean;
        description
          "File blocking security service.";
      }
    }
    case data-filtering {
      leaf data-filtering {
        type boolean;
        description
          "Data filtering security service.";
      }
    }
    case application-control {
      leaf application-control {
        type boolean;
        description
          "Application control security service.";
      }
    }
    case voip-volte {
      list voip-volte-rule {
        key "voip-volte-rule-id";
        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.";
                leaf voip-volte-rule-id {
                  type uint8;
                  mandatory true;
                  description
                    "The ID of the voip-volte-rule.
                     This is the key for voip-volte-rule-list.
                     This must be unique.";
                }
                container event {
                  description
                    "Event types: VoIP and VoLTE.";
                  leaf called-voip {
                    type boolean;
                    mandatory true;
                    description
                      "If content-security-control-type is
                       voip.";
                  }
                  leaf called-volte {
                    type boolean;
                    mandatory true;
                    description
                      "If content-security-control-type is
                       volte.";
                  }
                }
                container condition {
                  description
                    "TBD.";
                  list sip-header {
                    key "sip-header-uri";
                    description
                      "TBD.";
                    leaf sip-header-uri {
                      type string;
                      mandatory true;
                      description
                        "SIP header URI.";
```

```
            }
            leaf sip-header-method {
              type string;
              mandatory true;
              description
                "SIP header method.";
            }
            leaf sip-header-expire-time {
              type yang:date-and-time;
              mandatory true;
              description
                "SIP header expire time.";
            }
            leaf sip-header-user-agent {
              type uint32;
              mandatory true;
              description
                "SIP header user agent.";
            }
          }
          list cell-region {
            key "cell-id-region";
            description
              "TBD.";
            leaf cell-id-region {
              type uint32;
              mandatory true;
              description
                "Cell region.";
            }
          }
        }
        container action {
          description
            "The flow-based NSFs realize the security
             functions by executing various Actions.";
          choice action-type {
            description
              "Action type: ingress action and
               egress action.";
            case ingress-action {
              description
                "The ingress actions consist of permit,
                 deny, and mirror.";
              choice ingress-action-type {
                description
                  "Ingress-action-type: permit, deny,
                   and mirror.";
```

```
                    case permit {
                      description
                        "Permit case.";
                      leaf permit {
                        type boolean;
                        mandatory true;
                        description
                          "Packet flow is permitted.";
                      }
                    }
                    case deny {
                      description
                        "Deny case.";
                      leaf deny {
                        type boolean;
                        mandatory true;
                        description
                          "Packet flow is denied.";
                      }
                    }
                    case mirror {
                      description
                        "Mirror case.";
                      leaf mirror {
                        type boolean;
                        mandatory true;
                        description
                          "Packet flow is mirrored.";
                      }
                    }
                  }
                }
                case egress-action {
                  description
                    "The engress actions consist of
                     mirror and etc.";
                  choice egress-action-type {
                    description
                      "Engress-action-type: redirection,
                       and etc.";
                    case redirection {
                      description
                        "Redirection case.";
                      leaf redirection {
                        type boolean;
                        mandatory true;
                        description "TBD.";
                      }
```

```
                        }
                      }
                    }
                  }
                }
              }
            }
          }
        }
      case attack-mitigation-control {
        description
          "This category of security capabilities is
           specially used to detect and mitigate various
           types of network attacks.";
        choice attack-mitigation-control-type {
          description
            "Attack-mitigation types: DDoS-attack and
             Single-packet attack.";
          case ddos-attack {
            description
              "A distributed-denial-of-service (DDoS) is
               where the attack source is more than one,
               often thousands of unique IP addresses.";
            choice ddos-attack-type {
              description
                "DDoS-attack types: Network Layer DDoS Attacks
                 and Application Layer DDoS Attacks.";
              case network-layer-ddos-attack {
                description
                  "Network layer DDoS-attack.";
                choice network-layer-ddos-attack-type {
                  description
                    "Network layer DDoS attack types:
                     Syn Flood Attack, UDP Flood Attack,
                     ICMP Flood Attack, IP Fragment Flood,
                     IPv6 Related Attacks, and etc";
                  case syn-flood-attack {
                    description
                      "If the network layer DDoS-attack is
                       a syn flood attack.";
                    leaf syn-flood {
                      type boolean;
                      mandatory true;
                      description
                        "Syn Flood Attack.";
                    }
                  }
                  case udp-flood-attack {
```

```
                description
                  "If the network layer DDoS-attack is
                   a udp flood attack.";
                leaf udp-flood {
                  type boolean;
                  mandatory true;
                  description
                    "UDP Flood Attack.";
                }
              }
              case icmp-flood-attack {
                description
                  "If the network layer DDoS-attack is
                   an icmp flood attack.";
                leaf icmp-flood {
                  type boolean;
                  mandatory true;
                  description
                    "ICMP Flood Attack.";
                }
              }
              case ip-fragment-flood-attack {
                description
                  "If the network layer DDoS-attack is
                   an ip fragment flood attack.";
                leaf ip-fragment-flood {
                  type boolean;
                  mandatory true;
                  description
                    "IP Fragment Flood.";
                }
              }
              case ipv6-related-attacks {
                description
                  "If the network layer DDoS-attack is
                   ipv6 related attacks.";
                leaf ipv6-related {
                  type boolean;
                  mandatory true;
                  description
                    "IPv6 Related Attacks.";
                }
              }
            }
          }
          case app-layer-ddos-attack {
            description
              "Application layer DDoS-attack.";
```

```
                choice app-ddos-attack-type {
                  description
                    "Application layer DDoS-attack types:
                     Http Flood Attack, Https Flood Attack,
                     DNS Flood Attack, and
                     DNS Amplification Flood Attack,
                     SSL DDoS Attack, and etc.";
                  case http-flood-attack {
                    description
                      "If the application layer DDoS-attack is
                       a http flood attack.";
                    leaf http-flood {
                      type boolean;
                      mandatory true;
                      description
                        "Http Flood Attack.";
                    }
                  }
                  case https-flood-attack {
                    description
                      "If the application layer DDoS-attack is
                       a https flood attack.";
                    leaf https-flood {
                      type boolean;
                      mandatory true;
                      description
                        "Https Flood Attack.";
                    }
                  }
                  case dns-flood-attack {
                    description
                      "If the application layer DDoS-attack is
                       a dns flood attack.";
                    leaf dns-flood {
                      type boolean;
                      mandatory true;
                      description
                        "DNS Flood Attack.";
                    }
                  }
                  case dns-amp-flood-attack {
                    description
                      "If the application layer DDoS-attack is
                       a dns amplification flood attack.";
                    leaf dns-amp-flood {
                      type boolean;
                      mandatory true;
                      description
```

```
                    "DNS Amplification Flood Attack.";
                }
              }
              case ssl-ddos-attack {
                description
                  "If the application layer DDoS-attack is
                   an ssl DDoS attack.";
                leaf ssl-ddos {
                  type boolean;
                  mandatory true;
                  description
                    "SSL Flood Attack.";
                }
              }
            }
          }
        }
        case single-packet-attack {
          description
            "Single Packet Attacks.";
          choice single-packet-attack-type {
            description
              "DDoS-attack types: Scanning Attack,
               Sniffing Attack, Malformed Packet Attack,
               Special Packet Attack, and etc.";
            case scan-and-sniff-attack {
              description
                "Scanning and Sniffing Attack.";
              choice scan-and-sniff-attack-type {
                description
                  "Scanning and sniffing attack types:
                   IP Sweep attack, Port Scanning,
                   and etc.";
                case ip-sweep-attack {
                  description
                    "If the scanning and sniffing attack is
                     an ip sweep attack.";
                  leaf ip-sweep {
                    type boolean;
                    mandatory true;
                    description
                      "IP Sweep Attack.";
                  }
                }
                case port-scanning-attack {
                  description
                    "If the scanning and sniffing attack is
```

```
                    a port scanning attack.";
                  leaf port-scanning {
                    type boolean;
                    mandatory true;
                    description
                      "Port Scanning Attack.";
                  }
                }
              }
            }
            case malformed-packet-attack {
              description
                "Malformed Packet Attack.";
              choice malformed-packet-attack-type {
                description
                  "Malformed packet attack types:
                   Ping of Death Attack, Teardrop Attack,
                   and etc.";
                case ping-of-death-attack {
                  description
                    "If the malformed packet attack is
                     a ping of death attack.";
                  leaf ping-of-death {
                    type boolean;
                    mandatory true;
                    description
                      "Ping of Death Attack.";
                  }
                }
                case teardrop-attack {
                  description
                    "If the malformed packet attack is
                     a teardrop attack.";
                  leaf teardrop {
                    type boolean;
                    mandatory true;
                    description
                      "Teardrop Attack.";
                  }
                }
              }
            }
            case special-packet-attack {
              description
                "special Packet Attack.";
              choice special-packet-attack-type {
                description
                  "Special packet attack types:
```

```
                              Oversized ICMP Attack, Tracert Attack,
                              and etc.";
                            case oversized-icmp-attack {
                              description
                                "If the special packet attack is
                                 an oversized icmp attack.";
                              leaf oversized-icmp {
                                type boolean;
                                mandatory true;
                                description
                                  "Oversize ICMP Attack.";
                              }
                            }
                            case tracert-attack {
                              description
                                "If the special packet attack is
                                 a tracert attack.";
                              leaf tracert {
                                type boolean;
                                mandatory true;
                                description
                                  "Tracrt Attack.";
                              }
                            }
                          }
                        }
                      }
                    }
                  }
                }
              }
            }
          }
        }
      }
    }
  }
}

<CODE ENDS>
```

Figure 2: Data Model of I2NSF Capability Interface

6. Security Considerations

This document introduces no additional security threats and SHOULD follow the security requirements as stated in [i2nsf-framework].

7. Acknowledgements

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).

This document has greatly benefited from inputs by Hyoungshick Kim and Se-Hui Lee.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

8.2. Informative References

[i2nsf-cap-interface-im] Xia, L., Strassner, J., Li, K., Zhang, D., Lopez, E., BOUTHORS, N., and L. Fang, "Information Model of Interface to Network Security Functions Capability Interface", draft-xia-i2nsf-capability-interface-im-06 (work in progress), June 2016.

[i2rs-rib-data-model] Wang, L., Ananthakrishnan, H., Chen, M., Dass, A., Kini, S., and N. Bahadur, "A YANG Data Model for Routing Information Base (RIB)", draft-ietf-i2rs-rib-data-model-05 (work in progress), March 2016.

[supa-policy-info-model] Strassner, J. and J. Halpern, "Generic Policy Information Model for Simplified Use of Policy Abstractions (SUPA)", draft-ietf-supa-generic-policy-info-model-00 (work in progress), June 2016.

[i2nsf-framework] Lopez, E., Lopez, D., Dunbar, L., Strassner, J., Zhuang, X., Parrott, J., Krishnan, R., and S. Durbha, "Framework for

Interface to Network Security Functions", draft-ietf-i2nsf-framework-00 (work in progress), May 2016.

Appendix A. Changes from draft-jeong-i2nsf-capability-interface-yang-02

The following changes were made from draft-jeong-i2nsf-capability-interface-yang-02:

- This version reflects the information model for NSF facing interface in draft-xia-i2nsf-capability-interface-im-06.

- Event, condition, and action are updated according to the above latest information model.

Authors' Addresses

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

Jin-Yong 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: wlsdyd0930@nate.com

Dae-Young Hyun
Department of Software
Sungkyunkwan University
2066 Seobu-Ro, Jangan-Gu
Suwon, Gyeonggi-Do 16419
Republic of Korea

Phone: +82 10 4776 5672
EMail: guseodud1@naver.com

Jung-Soo Park
Electronics and Telecommunications Research Institute
218 Gajeong-Ro, Yuseong-Gu
Daejeon 305-700
Republic of Korea

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

Tae-Jin Ahn
Korea Telecom
70 Yuseong-Ro, Yuseong-Gu
Daejeon 305-811
Republic of Korea

Phone: +82 42 870 8409
EMail: taejin.ahn@kt.com