Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 3, 2018

D. Hong
J. Jeong
J. Kim
Sungkyunkwan University
S. Hares
L. Xia
Huawei

October 30, 2017

# YANG Data Model for Monitoring I2NSF Network Security Functions
draft-hong-i2nsf-nsf-monitoring-data-model-01

## Abstract

This document proposes a YANG data model for monitoring Network Security Functions (NSFs) in the Interface to Network Security Functions (I2NSF) system. If the monitoring of NSFs is performed in a comrehensive way, it is possible to detect the indication of malicious activity, anomalous behavior or the potential sign of denial of service attacks in a timely manner. This monitoring functionality is based on the monitoring information that is generated by NSFs. Thus, this document describes not only a data tree to specify an information model for monitoring NSFs, but also the corresponding YANG data model for monitoring NSFs.

## 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 May 3, 2018.

## Copyright Notice

Copyright (c) 2017 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] data model for monitoring Network Security Functions (NSFs). This monitoring means the aquisition of vital information about NSFs via notifications, events, records or counters. The data model for the monitoring presented in this document is derived from the information model for monitoring NSFs through the NSF-Facing Interface specified in [i2nsf-monitoring-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-terminology][i2nsf-framework]. Especially, the following terms are from [i2nsf-monitoring-im].

- 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.

- 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.

### 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 the overview of a structure tree of monitoring information based on the [i2nsf-monitoring-im].

```
module: ietf-i2nsf-nsf-monitoring-dm
  +--rw monitoring-message
    +--rw monitoring-messages* [message-id]
       +--rw message-id                    uint8
       +--rw message-version               uint8
       +--rw (message-type)?
       |  +--:(event)
       |  |  +--rw event-name                    string
       |  |  +--rw (event-type)?
       |  |     +--:(system-event)
       |  |     |  +--rw access-violation
       |  |     |  |  +--rw group                 string
       |  |     |  |  +--rw login-ip              inet:ipv4-address
```

```
        |  |     |  |  +--rw authentication-mode
        |  |     |  |     +--rw local-authentication              boolean
        |  |     |  |     +--rw third-part-server-authentication  boolean
        |  |     |  |     +--rw exemption-authentication       boolean
        |  |     |  |     +--rw sso-authentication             boolean
        |  |     |  +--rw config-change
        |  |     |    +--rw group                  string
        |  |     |    +--rw login-ip               inet:ipv4-address
        |  |     |    +--rw authentication-mode
        |  |     |      +--rw local-authentication              boolean
        |  |     |      +--rw third-part-server-authentication  boolean
        |  |     |      +--rw exemption-authentication       boolean
        |  |     |      +--rw sso-authentication             boolean
        |  |     +--:(nsf-event)
        |  |        +--rw user-name?                     string
        |  |        +--rw ddos-event
        |  |        |  +--rw message?            string
        |  |        |  +--rw src-ip?             inet:ipv4-address
        |  |        |  +--rw dst-ip?             inet:ipv4-address
        |  |        |  +--rw src-port?           inet:port-number
        |  |        |  +--rw dst-port?           inet:port-number
        |  |        |  +--rw src-zone?           string
        |  |        |  +--rw dst-zone?           string
        |  |        |  +--rw rule-id             uint8
        |  |        |  +--rw rule-name           string
        |  |        |  +--rw profile?            string
        |  |        |  +--rw raw-info?           string
        |  |        |  +--rw ddos-attack-type
        |  |        |  |  +--rw syn-flood?              boolean
        |  |        |  |  +--rw ack-flood?              boolean
        |  |        |  |  +--rw syn-ack-flood?          boolean
        |  |        |  |  +--rw fin-rst-flood?          boolean
        |  |        |  |  +--rw tcp-connection-flood?   boolean
        |  |        |  |  +--rw udp-flood?              boolean
        |  |        |  |  +--rw icmp-flood?             boolean
        |  |        |  |  +--rw https-flood?            boolean
        |  |        |  |  +--rw http-flood?             boolean
        |  |        |  |  +--rw dns-reply-flood?        boolean
        |  |        |  |  +--rw dns-query-flood?        boolean
        |  |        |  |  +--rw sip-flood?              boolean
        |  |        |  +--rw start-time          yang:date-and-time
        |  |        |  +--rw end-time            yang:date-and-time
        |  |        |  +--rw attack-rate?        uint32
        |  |        |  +--rw attack-speed?       uint32
        |  |        +--rw session-table-event
        |  |        |  +--rw current-session?   uint8
        |  |        |  +--rw maximum-session?   uint8
        |  |        |  +--rw threshold?         uint8
```

```
|  |        |  +--rw message?           string
|  |        +--rw virus-event
|  |        |  +--rw message?      string
|  |        |  +--rw src-ip?       inet:ipv4-address
|  |        |  +--rw dst-ip?       inet:ipv4-address
|  |        |  +--rw src-port?     inet:port-number
|  |        |  +--rw dst-port?     inet:port-number
|  |        |  +--rw src-zone?     string
|  |        |  +--rw dst-zone?     string
|  |        |  +--rw rule-id       uint8
|  |        |  +--rw rule-name     string
|  |        |  +--rw profile?      string
|  |        |  +--rw raw-info?     string
|  |        |  +--rw virus-type
|  |        |  |  +--rw trajan?   boolean
|  |        |  |  +--rw worm?     boolean
|  |        |  |  +--rw macro?    boolean
|  |        |  +--rw virus-name?   string
|  |        |  +--rw file-type?    string
|  |        |  +--rw file-name?    string
|  |        +--rw intrusion-event
|  |        |  +--rw message?                string
|  |        |  +--rw src-ip?                 inet:ipv4-address
|  |        |  +--rw dst-ip?                 inet:ipv4-address
|  |        |  +--rw src-port?               inet:port-number
|  |        |  +--rw dst-port?               inet:port-number
|  |        |  +--rw src-zone?               string
|  |        |  +--rw dst-zone?               string
|  |        |  +--rw rule-id                 uint8
|  |        |  +--rw rule-name               string
|  |        |  +--rw profile?                string
|  |        |  +--rw raw-info?               string
|  |        |  +--rw protocol
|  |        |  |  +--rw tcp?      boolean
|  |        |  |  +--rw udp?      boolean
|  |        |  |  +--rw icmp?     boolean
|  |        |  |  +--rw icmpv6?   boolean
|  |        |  |  +--rw ip?       boolean
|  |        |  |  +--rw http?     boolean
|  |        |  |  +--rw ftp?      boolean
|  |        |  +--rw intrusion-attack-type
|  |        |     +--rw brutal-force?      boolean
|  |        |     +--rw buffer-overflow?   boolean
|  |        +--rw botnet-event
|  |        |  +--rw message?       string
|  |        |  +--rw src-ip?        inet:ipv4-address
|  |        |  +--rw dst-ip?        inet:ipv4-address
|  |        |  +--rw src-port?      inet:port-number
```

```
|  |        |  +--rw dst-port?      inet:port-number
|  |        |  +--rw src-zone?      string
|  |        |  +--rw dst-zone?      string
|  |        |  +--rw rule-id        uint8
|  |        |  +--rw rule-name      string
|  |        |  +--rw profile?       string
|  |        |  +--rw raw-info?      string
|  |        |  +--rw protocol
|  |        |  |  +--rw tcp?      boolean
|  |        |  |  +--rw udp?      boolean
|  |        |  |  +--rw icmp?     boolean
|  |        |  |  +--rw icmpv6?   boolean
|  |        |  |  +--rw ip?       boolean
|  |        |  |  +--rw http?     boolean
|  |        |  |  +--rw ftp?      boolean
|  |        |  +--rw botnet-name?   string
|  |        |  +--rw role?          string
|  |        +--rw web-attack-event
|  |           +--rw message?           string
|  |           +--rw src-ip?            inet:ipv4-address
|  |           +--rw dst-ip?            inet:ipv4-address
|  |           +--rw src-port?          inet:port-number
|  |           +--rw dst-port?          inet:port-number
|  |           +--rw src-zone?          string
|  |           +--rw dst-zone?          string
|  |           +--rw rule-id            uint8
|  |           +--rw rule-name          string
|  |           +--rw profile?           string
|  |           +--rw raw-info?          string
|  |           +--rw web-attack-type
|  |           |  +--rw sql-injection?       boolean
|  |           |  +--rw command-injection?   boolean
|  |           |  +--rw xss?                 boolean
|  |           |  +--rw csrf?                boolean
|  |           +--rw req-method
|  |           |  +--rw put?   boolean
|  |           |  +--rw get?   boolean
|  |           +--rw req-url?           string
|  |           +--rw url-category?      string
|  |           +--rw filtering-type
|  |              +--rw blacklist?            boolean
|  |              +--rw whitelist?            boolean
|  |              +--rw user-defined?         boolean
|  |              +--rw balicious-category?   boolean
|  |              +--rw unknown?              boolean
|  +--:(log)
|  |  +--rw (log-type)?
|  |     +--:(system-log)
```

```
       |  |     |  +--rw access-logs
       |  |     |  |  +--rw login-ip          inet:ipv4-address
       |  |     |  |  +--rw administartor?    string
       |  |     |  |  +--rw login-mode?       login-mode
       |  |     |  |  +--rw operation-type?   operation-type
       |  |     |  |  +--rw result?           string
       |  |     |  |  +--rw content?          string
       |  |     |  +--rw resource-utiliz-logs
       |  |     |  |  +--rw system-status?       string
       |  |     |  |  +--rw cpu-usage?           uint8
       |  |     |  |  +--rw memory-usage?        uint8
       |  |     |  |  +--rw disk-usage?          uint8
       |  |     |  |  +--rw disk-left?           uint8
       |  |     |  |  +--rw session-num?         uint8
       |  |     |  |  +--rw process-num?         uint8
       |  |     |  |  +--rw in-traffic-rate?     uint32
       |  |     |  |  +--rw out-traffic-rate?    uint32
       |  |     |  |  +--rw in-traffic-speed?    uint32
       |  |     |  |  +--rw out-traffic-speed?   uint32
       |  |     |  +--rw user-activity-logs
       |  |     |     +--rw user                    string
       |  |     |     +--rw group                   string
       |  |     |     +--rw login-ip                inet:ipv4-address
       |  |     |     +--rw authentication-mode
       |  |     |     |  +--rw local-authentication              boolean
       |  |     |     |  +--rw third-part-server-authentication  boolean
       |  |     |     |  +--rw exemption-authentication       boolean
       |  |     |     |  +--rw sso-authentication             boolean
       |  |     |     +--rw access-mode
       |  |     |     |  +--rw ppp?     boolean
       |  |     |     |  +--rw svn?     boolean
       |  |     |     |  +--rw local?   boolean
       |  |     |     +--rw online-duration?       string
       |  |     |     +--rw logout-duration?       string
       |  |     |     +--rw addtional-info?        string
       |  |     +--:(nsf-log)
       |  |        +--rw ddos-logs
       |  |        |  +--rw attack-type?        string
       |  |        |  +--rw attack-ave-rate?    uint32
       |  |        |  +--rw attack-ave-speed?   uint32
       |  |        |  +--rw attack-pkt-num?     uint32
       |  |        |  +--rw attack-src-ip?      inet:ipv4-address
       |  |        |  +--rw action?             all-action
       |  |        |  +--rw os?                 string
       |  |        +--rw virus-logs
       |  |        |  +--rw protocol
       |  |        |  |  +--rw tcp?      boolean
       |  |        |  |  +--rw udp?      boolean
```

```
|  |        |  |  +--rw icmp?     boolean
|  |        |  |  +--rw icmpv6?   boolean
|  |        |  |  +--rw ip?       boolean
|  |        |  |  +--rw http?     boolean
|  |        |  |  +--rw ftp?      boolean
|  |        |  +--rw attack-type?   string
|  |        |  +--rw action?        all-action
|  |        |  +--rw os?            string
|  |        |  +--rw time           yang:date-and-time
|  |        +--rw intrusion-logs
|  |        |  +--rw attack-type?    string
|  |        |  +--rw action?         all-action
|  |        |  +--rw time            yang:date-and-time
|  |        |  +--rw attack-rate?    uint32
|  |        |  +--rw attack-speed?   uint32
|  |        +--rw botnet-logs
|  |        |  +--rw attack-type?      string
|  |        |  +--rw botnet-pkt-num?   uint8
|  |        |  +--rw action?           all-action
|  |        |  +--rw os?               string
|  |        +--rw dpi-logs
|  |        |  +--rw dpi-type?      dpi-type
|  |        |  +--rw src-ip?        inet:ipv4-address
|  |        |  +--rw dst-ip?        inet:ipv4-address
|  |        |  +--rw src-port?      inet:port-number
|  |        |  +--rw dst-port?      inet:port-number
|  |        |  +--rw src-zone?      string
|  |        |  +--rw dst-zone?      string
|  |        |  +--rw src-region?    string
|  |        |  +--rw dst-region?    string
|  |        |  +--rw policy-id      uint8
|  |        |  +--rw policy-name    string
|  |        |  +--rw src-user?      string
|  |        |  +--rw protocol
|  |        |  |  +--rw tcp?      boolean
|  |        |  |  +--rw udp?      boolean
|  |        |  |  +--rw icmp?     boolean
|  |        |  |  +--rw icmpv6?   boolean
|  |        |  |  +--rw ip?       boolean
|  |        |  |  +--rw http?     boolean
|  |        |  |  +--rw ftp?      boolean
|  |        |  +--rw file-type?     string
|  |        |  +--rw file-name?     string
|  |        +--rw vulnerability-scanning-logs* [vulnerability-id]
|  |        |  +--rw vulnerability-id    uint8
|  |        |  +--rw victim-ip?          inet:ipv4-address
|  |        |  +--rw protocol
|  |        |  |  +--rw tcp?      boolean
```

```
       |  |        |  |  +--rw udp?      boolean
       |  |        |  |  +--rw icmp?     boolean
       |  |        |  |  +--rw icmpv6?   boolean
       |  |        |  |  +--rw ip?       boolean
       |  |        |  |  +--rw http?     boolean
       |  |        |  |  +--rw ftp?      boolean
       |  |        |  +--rw port-num?           inet:port-number
       |  |        |  +--rw level?              severity
       |  |        |  +--rw os?                 string
       |  |        |  +--rw addtional-info?     string
       |  |        +--rw web-attack-logs
       |  |           +--rw attack-type?     string
       |  |           +--rw rsp-code?        string
       |  |           +--rw req-clientapp?   string
       |  |           +--rw req-cookies?     string
       |  |           +--rw req-host?        string
       |  |           +--rw raw-info?        string
       |  +--:(counters)
       |     +--rw (counter-type)?
       |        +--:(system-counter)
       |        |  +--rw interface-counters
       |        |     +--rw interface-name?            string
       |        |     +--rw in-total-traffic-pkts?     uint32
       |        |     +--rw out-total-traffic-pkts?    uint32
       |        |     +--rw in-total-traffic-bytes?    uint32
       |        |     +--rw out-total-traffic-bytes?   uint32
       |        |     +--rw in-drop-traffic-pkts?      uint32
       |        |     +--rw out-drop-traffic-pkts?     uint32
       |        |     +--rw in-drop-traffic-bytes?     uint32
       |        |     +--rw out-drop-traffic-bytes?    uint32
       |        |     +--rw total-traffic?             uint32
       |        |     +--rw in-traffic-ave-rate?       uint32
       |        |     +--rw in-traffic-peak-rate?      uint32
       |        |     +--rw in-traffic-ave-speed?      uint32
       |        |     +--rw in-traffic-peak-speed?     uint32
       |        |     +--rw out-traffic-ave-rate?      uint32
       |        |     +--rw out-traffic-peak-rate?     uint32
       |        |     +--rw out-traffic-ave-speed?     uint32
       |        |     +--rw out-traffic-peak-speed?    uint32
       |        +--:(nsf-counter)
       |        |  +--rw firewall-counters
       |        |     +--rw src-ip?                    inet:ipv4-address
       |        |     +--rw dst-ip?                    inet:ipv4-address
       |        |     +--rw src-port?                  inet:port-number
       |        |     +--rw dst-port?                  inet:port-number
       |        |     +--rw src-zone?                  string
       |        |     +--rw dst-zone?                  string
       |        |     +--rw src-region?                string
```

```
|        |     +--rw dst-region?              string
|        |     +--rw policy-id                uint8
|        |     +--rw policy-name              string
|        |     +--rw src-user?                string
|        |     +--rw protocol
|        |     |  +--rw tcp?      boolean
|        |     |  +--rw udp?      boolean
|        |     |  +--rw icmp?     boolean
|        |     |  +--rw icmpv6?   boolean
|        |     |  +--rw ip?       boolean
|        |     |  +--rw http?     boolean
|        |     |  +--rw ftp?      boolean
|        |     +--rw total-traffic?           uint32
|        |     +--rw in-traffic-ave-rate?     uint32
|        |     +--rw in-traffic-peak-rate?    uint32
|        |     +--rw in-traffic-ave-speed?    uint32
|        |     +--rw in-traffic-peak-speed?   uint32
|        |     +--rw out-traffic-ave-rate?    uint32
|        |     +--rw out-traffic-peak-rate?   uint32
|        |     +--rw out-traffic-ave-speed?   uint32
|        |     +--rw out-traffic-peak-speed?  uint32
|        |     +--rw bound
|        |        +--rw in-interface?    boolean
|        |        +--rw out-interface?   boolean
|        +--:(policy-hit-counters)
|           +--rw policy-hit-counters
|              +--rw src-ip?                 inet:ipv4-address
|              +--rw dst-ip?                 inet:ipv4-address
|              +--rw src-port?               inet:port-number
|              +--rw dst-port?               inet:port-number
|              +--rw src-zone?               string
|              +--rw dst-zone?               string
|              +--rw src-region?             string
|              +--rw dst-region?             string
|              +--rw policy-id               uint8
|              +--rw policy-name             string
|              +--rw src-user?               string
|              +--rw protocol
|              |  +--rw tcp?      boolean
|              |  +--rw udp?      boolean
|              |  +--rw icmp?     boolean
|              |  +--rw icmpv6?   boolean
|              |  +--rw ip?       boolean
|              |  +--rw http?     boolean
|              |  +--rw ftp?      boolean
|              +--rw total-traffic?          uint32
|              +--rw in-traffic-ave-rate?    uint32
|              +--rw in-traffic-peak-rate?   uint32
```

```
      |              +--rw in-traffic-ave-speed?     uint32
      |              +--rw in-traffic-peak-speed?    uint32
      |              +--rw out-traffic-ave-rate?     uint32
      |              +--rw out-traffic-peak-rate?    uint32
      |              +--rw out-traffic-ave-speed?    uint32
      |              +--rw out-traffic-peak-speed?   uint32
      |              +--rw hit-times?                uint32
      +--rw message                          string
      +--rw time-stamp                       yang:date-and-time
      +--rw severity                         severity
```

Figure 1: Information Model for NSF Monitoring

## 5. YANG Data Model

This section introduces a YANG data model for the information model of monitoring inforamtion based on [i2nsf-monitoring-im].

<CODE BEGINS> file "ietf-i2nsf-nsf-monitoring-dm@2017-10-30.yang"

```
module ietf-i2nsf-nsf-monitoring-dm {
  namespace
    "urn:ietf:params:xml:ns:yang:ietf-i2nsf-nsf-monitoring-dm";
  prefix
    monitoring-information;
  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: Linda Dunbar
     <mailto:Linda.duhbar@huawei.com>

     Editor: Dongjin Hong
     <mailto:dong.jin@skku.edu>

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

```
  description
    "This module defines a YANG data module for monitoring NSFs.";

  revision "2017-10-29" {
    description "Initial revision";
    reference
      "draft-zhang-i2nsf-info-model-monitoring-04";
  }

  typedef severity {
    type enumeration {
      enum high {
        description
          "high-level";
      }
      enum middle {
        description
          "middle-level";
      }
      enum low {
        description
          "low-level";
      }
    }
    description
      "This is used for indicating the severity";
  }
  typedef all-action {
    type enumeration {
      enum allow {
        description
          "TBD";
      }
      enum alert {
        description
          "TBD";
      }
      enum block {
        description
          "TBD";
      }
      enum discard {
        description
          "TBD";
      }
      enum declare {
        description
          "TBD";
```

```
        }
        enum block-ip {
          description
            "TBD";
        }
        enum block-service{
          description
            "TBD";
        }
      }
      description
        "This is used for protocol";
    }
    typedef dpi-type{
      type enumeration {
        enum file-blocking{
          description
            "TBD";
        }
        enum data-filtering{
          description
            "TBD";
        }
        enum application-behavior-control{
          description
            "TBD";
        }
      }
      description
        "This is used for dpi type";
    }
    typedef operation-type{
      type enumeration {
        enum login{
          description
            "TBD";
        }
        enum logout{
          description
            "TBD";
        }
        enum configuration{
          description
            "TBD";
        }
      }
      description
        "This is used for operation type";
```

```
    }
    typedef login-mode{
      type enumeration {
        enum root{
          description
            "TBD";
        }
        enum user{
          description
            "TBD";
        }
        enum guest{
          description
            "TBD";
        }
      }
      description
        "This is used for login mode";
    }
    grouping protocol {
      description
        "A set of protocols";
      container protocol {
        description
        "Protocol types:
         TCP, UDP, ICMP, ICMPv6, IP, HTTP, FTP and etc.";
        leaf tcp {
          type boolean;
          description
            "TCP protocol type.";
        }
        leaf udp {
          type boolean;
          description
            "UDP protocol type.";
        }
        leaf icmp {
          type boolean;
          description
            "ICMP protocol type.";
        }
        leaf icmpv6 {
          type boolean;
          description
            "ICMPv6 protocol type.";
        }
        leaf ip {
          type boolean;
```

```
          description
            "IP protocol type.";
        }
        leaf http {
          type boolean;
          description
            "HTTP protocol type.";
        }
        leaf ftp {
          type boolean;
          description
            "ftp protocol type.";
        }
      }
    }
    grouping traffic-rates {
      description
        "A set of traffic rates
        for statistics data";
      leaf total-traffic {
        type uint32;
        description
          "Total traffic";
      }
      leaf in-traffic-ave-rate {
        type uint32;
        description
          "Inbound traffic average rate in pps";
      }
      leaf in-traffic-peak-rate {
        type uint32;
        description
          "Inbound traffic peak rate in pps";
      }
      leaf in-traffic-ave-speed {
        type uint32;
        description
          "Inbound traffic average speed in bps";
      }
      leaf in-traffic-peak-speed {
        type uint32;
        description
          "Inbound traffic peak speed in bps";
      }
      leaf out-traffic-ave-rate {
        type uint32;
        description
          "Outbound traffic average rate in pps";
```

```
      }
      leaf out-traffic-peak-rate {
        type uint32;
        description
          "Outbound traffic peak rate in pps";
      }
      leaf out-traffic-ave-speed {
        type uint32;
        description
          "Outbound traffic average speed in bps";
      }
      leaf out-traffic-peak-speed {
        type uint32;
        description
          "Outbound traffic peak speed in bps";
      }
    }
    grouping i2nsf-system-event-type-content {
      description
        "A set of system event type contents";
      leaf group {
        type string;
        mandatory true;
        description
          "Group to which a user belongs.";
      }
      leaf login-ip {
        type inet:ipv4-address;
        mandatory true;
        description
          "Login IP address of a user.";
      }
      container authentication-mode {
        description
          "User authentication mode. e.g., Local Authentication,
          Third-Party Server Authentication,
          Authentication Exemption, SSO Authentication.";
        leaf local-authentication {
          type boolean;
          mandatory true;
          description
            "Authentication-mode : local authentication.";
        }
        leaf third-part-server-authentication {
          type boolean;
          mandatory true;
          description
            "TBD";
```

```
        }
        leaf exemption-authentication {
          type boolean;
          mandatory true;
          description
            "TBD";
        }
        leaf sso-authentication {
          type boolean;
          mandatory true;
          description
            "TBD";
        }
      }
    }
    grouping i2nsf-nsf-event-type-content {
      description
        "A set of nsf event type contents";
      leaf message {
        type string;
        description
          "The message for nsf events";
      }
      leaf src-ip {
        type inet:ipv4-address;
        description
          "The source IP address of the packet";
      }
      leaf dst-ip {
        type inet:ipv4-address;
        description
          "The destination IP address of the packet";
      }
      leaf src-port {
        type inet:port-number;
        description
          "The source port of the packet";
      }
      leaf dst-port {
        type inet:port-number;
        description
          "The destination port of the packet";
      }
      leaf src-zone {
        type string;
        description
          "The source security zone of the packet";
      }
```

```
        leaf dst-zone {
          type string;
          description
            "The destination security zone of the packet";
        }
        leaf rule-id {
          type uint8;
          mandatory true;
          description
            "The ID of the rule being triggered";
        }
        leaf rule-name {
          type string;
          mandatory true;
          description
            "The name of the rule being triggered";
        }
        leaf profile {
          type string;
          description
            "Security profile that traffic matches.";
        }
        leaf raw-info {
          type string;
          description
            "The information describing the packet
            triggering the event.";
        }
      }
      grouping i2nsf-system-counter-type-content{
        description
          "A set of system counter type contents";
        leaf interface-name {
          type string;
          description
            "Network interface name configured in NSF";
        }
        leaf in-total-traffic-pkts {
          type uint32;
          description
            "Total inbound packets";
        }
        leaf out-total-traffic-pkts {
          type uint32;
          description
            "Total outbound packets";
        }
        leaf in-total-traffic-bytes {
```

```
        type uint32;
        description
          "Total inbound bytes";
      }
      leaf out-total-traffic-bytes {
        type uint32;
        description
          "Total outbound bytes";
      }
      leaf in-drop-traffic-pkts {
        type uint32;
        description
          "Total inbound drop packets";
      }
      leaf out-drop-traffic-pkts {
        type uint32;
        description
          "Total outbound drop packets";
      }
      leaf in-drop-traffic-bytes {
        type uint32;
        description
          "Total inbound drop bytes";
      }
      leaf out-drop-traffic-bytes {
        type uint32;
        description
          "Total outbound drop bytes";
      }
      uses traffic-rates;
    }
    grouping i2nsf-nsf-counters-type-content{
      description
        "A set of nsf counters type contents";
      leaf src-ip {
        type inet:ipv4-address;
        description
          "The source IP address of the packet";
      }
      leaf dst-ip {
        type inet:ipv4-address;
        description
          "The destination IP address of the packet";
      }
      leaf src-port {
        type inet:port-number;
        description
          "The source port of the packet";
```

```
      }
      leaf dst-port {
        type inet:port-number;
        description
          "The destination port of the packet";
      }
      leaf src-zone {
        type string;
        description
          "The source security zone of the packet";
      }
      leaf dst-zone {
        type string;
        description
          "The destination security zone of the packet";
      }
      leaf src-region {
        type string;
        description
          "Source region of the traffic";
      }
      leaf dst-region{
        type string;
        description
          "Destination region of the traffic";
      }
      leaf policy-id {
        type uint8;
        mandatory true;
        description
          "The ID of the policy being triggered";
      }
      leaf policy-name {
        type string;
        mandatory true;
        description
          "The name of the policy being triggered";
      }
      leaf src-user{
        type string;
        description
          "User who generates traffic";
      }
      uses protocol;
      uses traffic-rates;
    }

    container monitoring-message {
```

```
      description
        "The message for monitoring information";
      list monitoring-messages {
        key message-id;
        description
          "The messages according to monitoring information";
        leaf message-id {
          type uint8;
          mandatory true;
          description
            "This is message ID
             This is key for monitoring messages";
        }
        leaf message-version {
          type uint8;
          mandatory true;
          description
            "The version of message";
        }
        choice message-type {
          description
            "The type of message";
          case event {
            description
              "If the message type is event";
            leaf event-name {
              type string;
              mandatory true;
              description
                "The name of the event";
            }
            choice event-type {
              description
                "This is event type such as system event
                and nsf event.";
              case system-event {
                description
                  "If the event type is system event";
                container access-violation {
                  description
                    "If the system event is
                    access violation";
                  uses i2nsf-system-event-type-content;
                }
                container config-change {
                  description
                    "If the system event is
                    config change violation";
```

```
            uses i2nsf-system-event-type-content;
          }
        }
        case nsf-event {
          description
            "If the event type is nsf event";
          leaf user-name {
            type string;
            description
              "This is user name for NSF event";
          }
          container ddos-event {
            description
              "If the event type is DDoS event";
            uses i2nsf-nsf-event-type-content;
            container ddos-attack-type{
              description
                "Type of DDoS attack";
              leaf syn-flood{
                type boolean;
                description
                  "If the DDoS attack is
                  syn flood";
              }
              leaf ack-flood{
                type boolean;
                description
                  "If the DDoS attack is
                  ack flood";
              }
              leaf syn-ack-flood{
                type boolean;
                description
                  "If the DDoS attack is
                  syn ack flood";
              }
              leaf fin-rst-flood{
                type boolean;
                description
                  "If the DDoS attack is
                  fin rst flood";
              }
              leaf tcp-connection-flood{
                type boolean;
                description
                  "If the DDoS attack is
                  tcp connection flood";
              }
```

```
          leaf udp-flood{
            type boolean;
            description
              "If the DDoS attack is
              udp flood";
          }
          leaf icmp-flood{
            type boolean;
            description
              "If the DDoS attack is
              icmp flood";
          }
          leaf https-flood{
            type boolean;
            description
              "If the DDoS attack is
              https flood";
          }
          leaf http-flood{
            type boolean;
            description
              "If the DDoS attack is
              http flood";
          }
          leaf dns-reply-flood{
            type boolean;
            description
              "If the DDoS attack is
              dns reply flood";
          }
          leaf dns-query-flood{
            type boolean;
            description
              "If the DDoS attack is
              dns query flood";
          }
          leaf sip-flood{
            type boolean;
            description
              "If the DDoS attack is
              sip flood";
          }
        }
        leaf start-time {
          type yang:date-and-time;
          mandatory true;
          description
            "The time stamp indicating
```

```
            when the attack started";
          }
          leaf end-time {
            type yang:date-and-time;
            mandatory true;
            description
              "The time stamp indicating
              when the attack ended";
          }
          leaf attack-rate {
            type uint32;
            description
              "The PPS of attack traffic";
          }
          leaf attack-speed {
            type uint32;
            description
              "the bps of attack traffic";
          }
        }
        container session-table-event {
          description
            "If the event type is session
            table event";
            leaf current-session {
              type uint8;
              description
               "The number of concurrent
               sessions";
            }
            leaf maximum-session {
              type uint8;
              description
               "The maximum number of sessions
               that the session table can
               support";
            }
            leaf threshold {
              type uint8;
              description
               "The threshold triggering
               the event";
            }
            leaf message {
              type string;
              description
               "The number of session table
                exceeded the threshold";
```

```
      }
    }
    container virus-event {
      description
        "If the event type is virus event";
      uses i2nsf-nsf-event-type-content;
      container virus-type {
        description
          "The type of virus";
        leaf trajan {
          type boolean;
          description
           "If the virus type is trajan";
        }
        leaf worm {
          type boolean;
          description
            "If the virus type is worm";
        }
        leaf macro {
          type boolean;
          description
            "If the virus type is macro";
        }
      }
      leaf virus-name {
        type string;
        description
          "The name of virus";
      }
      leaf file-type {
        type string;
        description
          "The type of file";
      }
      leaf file-name {
        type string;
        description
          "The name of file";
      }
    }
    container intrusion-event {
      description
       "If the event type is intrusion event";
      uses i2nsf-nsf-event-type-content;
      uses protocol;
      container intrusion-attack-type {
        description
```

```
          "The attack type of intrusion";
        leaf brutal-force {
          type boolean;
          description
            "The intrusion type is
            brutal force";
        }
        leaf buffer-overflow {
          type boolean;
          description
            "The intrusion type is
            buffer overflow";
        }
      }
    }
    container botnet-event {
      description
        "If the event type is botnet event";
      uses i2nsf-nsf-event-type-content;
      uses protocol;
      leaf botnet-name {
        type string;
        description
         "The name of the detected botnet";
      }
      leaf role {
        type string;
        description
         "The role of the communicating
          parties within the botnet";
      }
    }
    container web-attack-event {
      description
        "If the event type is web
        attack event";
      uses i2nsf-nsf-event-type-content;
      container web-attack-type {
        description
          "To determine the attack
          type";
        leaf sql-injection {
          type boolean;
          description
            "If the web attack type is
            sql injection";
        }
        leaf command-injection {
```

```
        type boolean;
        description
          "If the web attack type is
          command injection";
      }
      leaf xss {
        type boolean;
        description
          "If the web attack type is
          xss injection";
      }
      leaf csrf {
        type boolean;
        description
          "If the web attack type is
          csrf injection";
      }
    }
    container req-method {
      description
       "The method of requirement.
        For instance, PUT or GET
        in HTTP";
      leaf put{
        type boolean;
        description
          "If req method is PUT";
      }
      leaf get {
       type boolean;
        description
          "If req method is GET";
      }
    }
    leaf req-url {
      type string;
      description
        "Requested URL";
    }
    leaf url-category {
      type string;
      description
        "Matched URL category";
    }
    container filtering-type {
      description
       "URL filtering type,
        e.g., Blacklist, Whitelist,
```

```
                    User-Defined, Predefined,
                    Malicious Category, Unknown";
                  leaf blacklist {
                    type boolean;
                    description
                      "The filtering type is
                      blacklist";
                  }
                  leaf whitelist {
                    type boolean;
                    description
                      "The filtering type is
                      whitelist";
                  }
                  leaf user-defined {
                    type boolean;
                    description
                      "The filtering type is
                      user defined";
                  }
                  leaf balicious-category{
                    type boolean;
                    description
                      "The filtering type is
                      balicious category";
                  }
                  leaf unknown {
                    type boolean;
                    description
                      "The filtering type is
                      unknown";
                  }
                }
              }
            }
          }
          case log {
            description
              "If the message type is log";
            choice log-type {
              description
                "The type of log";
              case system-log{
                description
                  "If the log type is system log";
                container access-logs {
                  description
```

```
        "If the log is access logs
        in system log";
      leaf login-ip {
        type inet:ipv4-address;
        mandatory true;
        description
          "Login IP address of a user.";
      }
      leaf administartor {
        type string;
        description
          "Administrator that
          operates on the device";
      }
      leaf login-mode {
        type login-mode;
        description
          "Specifies the
          administrator logs in mode";
      }
      leaf operation-type {
        type operation-type;
        description
          "The operation type that
          the administrator execute";
      }
      leaf result {
        type string;
        description
          "Command execution result";
      }
      leaf content {
        type string;
        description
          "Operation performed by
          an administrator after login.";
      }
    }
    container resource-utiliz-logs {
      description
        "If the log is resource utilize
        logs in system log";
      leaf system-status {
        type string;
        description
          "TBD";
      }
      leaf cpu-usage {
```

```
        type uint8;
        description
          "specifies the amount of
          cpu usage";
      }
      leaf memory-usage {
        type uint8;
        description
          "specifies the amount of
          memory usage";
      }
      leaf disk-usage {
        type uint8;
        description
          "specifies the amount of
          disk usage";
      }
      leaf disk-left {
        type uint8;
        description
          "specifies the amount of
          disk left";
      }
      leaf session-num {
        type uint8;
        description
          "The total number of
          sessions";
      }
      leaf process-num {
        type uint8;
        description
          "The total number of
          process";
      }
      leaf in-traffic-rate {
        type uint32;
        description
          "The total inbound
          traffic rate in pps";
      }
      leaf out-traffic-rate {
        type uint32;
        description
          "The total outbound
          traffic rate in pps";
      }
      leaf in-traffic-speed {
```

```
            type uint32;
            description
              "The total inbound
              traffic speed in bps";
          }
          leaf out-traffic-speed {
            type uint32;
            description
              "The total outbound
              traffic speed in bps";
          }
        }
        container user-activity-logs {
          description
            "If the log is user activity
            logs in system log";
          leaf user {
            type string;
            mandatory true;
            description
              "Name of a user";
          }
          leaf group {
            type string;
            mandatory true;
            description
              "Group to which a user belongs.";
          }
          leaf login-ip {
            type inet:ipv4-address;
            mandatory true;
            description
              "Login IP address of a user.";
          }
          container authentication-mode {
            description
              "User authentication mode. e.g.,
              Local Authentication,
              Third-Party Server Authentication,
              Authentication Exemption, SSO Authentication.";
            leaf local-authentication {
              type boolean;
              mandatory true;
              description
                "Authentication-mode : local authentication.";
            }
            leaf third-part-server-authentication {
              type boolean;
```

```
        mandatory true;
        description
          "TBD";
      }
      leaf exemption-authentication {
        type boolean;
        mandatory true;
        description
          "TBD";
      }
      leaf sso-authentication {
        type boolean;
        mandatory true;
        description
          "TBD";
      }
    }
    container access-mode {
      description
        "TBD";
      leaf ppp{
        type boolean;
        description
          "TBD";
      }
      leaf svn{
        type boolean;
        description
          "TBD";
      }
      leaf local{
        type boolean;
        description
          "TBD";
      }
    }
    leaf online-duration {
      type string;
      description
        "TBD";
    }
    leaf logout-duration {
      type string;
      description
        "TBD";
    }
    leaf addtional-info {
      type string;
```

```
                    description
                      "TBD";
                  }
                }
              }
              case nsf-log{
                description
                  "If the log type is nsf log";
                container ddos-logs {
                  description
                    "If the log is DDoS logs
                    in nsf log";
                  leaf attack-type{
                    type string;
                    description
                      "DDoS";
                  }
                  leaf attack-ave-rate {
                    type uint32;
                    description
                      "The ave PPS of
                      attack traffic";
                  }
                  leaf attack-ave-speed {
                    type uint32;
                    description
                      "the ave bps of
                      attack traffic";
                  }
                  leaf attack-pkt-num{
                    type uint32;
                    description
                      "the number of
                      attack packets";
                  }
                  leaf attack-src-ip {
                    type inet:ipv4-address;
                    description
                      "TBD";
                  }
                  leaf action {
                    type all-action;
                    description
                      "TBD";
                  }
                  leaf os  {
                    type string;
                    description
```

```
        "simple os information";
    }
  }
  container virus-logs {
    description
      "If the log is virus logs
      in nsf log";
    uses protocol;
    leaf attack-type{
      type string;
      description
        "Virus";
    }
    leaf action{
      type all-action;
      description
        "TBD";
    }
    leaf os{
      type string;
      description
        "simple os information";
    }
    leaf time {
      type yang:date-and-time;
      mandatory true;
      description
        "Indicate the time when the
        message is generated";
    }
  }
  container intrusion-logs {
    description
      "If the log is intrusion logs
      in nsf log";
    leaf attack-type{
      type string;
      description
        "Intrusion";
    }
    leaf action{
      type all-action;
      description
        "TBD";
    }
    leaf time {
      type yang:date-and-time;
      mandatory true;
```

```
        description
          "Indicate the time when the
          message is generated";
      }
      leaf attack-rate {
        type uint32;
        description
          "The PPS of attack traffic";
      }
      leaf attack-speed {
        type uint32;
        description
          "the bps of attack traffic";
      }
    }
    container botnet-logs {
      description
        "If the log is botnet logs
        in nsf log";
      leaf attack-type{
        type string;
        description
          "Botnet";
      }
      leaf botnet-pkt-num{
        type uint8;
        description
          "The number of the packets
          sent to or from the
          detected botnet";
      }
      leaf action{
        type all-action;
        description
          "TBD";
      }
      leaf os{
        type string;
        description
          "simple os information";
      }
    }
    container dpi-logs {
      description
        "If the log is dpi logs
        in nsf log";
      leaf dpi-type{
        type dpi-type;
```

```
      description
        "The type of dpi";
    }
    leaf src-ip {
      type inet:ipv4-address;
      description
        "The source IP address of the packet";
    }
    leaf dst-ip {
      type inet:ipv4-address;
      description
        "The destination IP address of the packet";
    }
    leaf src-port {
      type inet:port-number;
      description
        "The source port of the packet";
    }
    leaf dst-port {
      type inet:port-number;
      description
        "The destination port of the packet";
    }
    leaf src-zone {
      type string;
      description
        "The source security zone of the packet";
    }
    leaf dst-zone {
      type string;
      description
        "The destination security zone of the packet";
    }
    leaf src-region {
      type string;
      description
        "Source region of the traffic";
    }
    leaf dst-region{
      type string;
      description
        "Destination region of the traffic";
    }
    leaf policy-id {
      type uint8;
      mandatory true;
      description
        "The ID of the policy being triggered";
```

```
      }
      leaf policy-name {
        type string;
        mandatory true;
        description
          "The name of the policy being triggered";
      }
      leaf src-user{
        type string;
        description
          "User who generates traffic";
      }
      uses protocol;
      leaf file-type {
        type string;
        description
          "The type of file";
      }
      leaf file-name {
        type string;
        description
          "The name of file";
      }
    }
    list vulnerability-scanning-logs {
      key vulnerability-id;
      description
        "If the log is vulnerability
        scanning logs in nsf log";
      leaf vulnerability-id{
        type uint8;
        description
          "The vulnerability id";
      }
      leaf victim-ip {
        type inet:ipv4-address;
        description
          "IP address of the victim
          host which has vulnerabilities";
      }
      uses protocol;
      leaf port-num{
        type inet:port-number;
        description
          "The port number";
      }
      leaf level{
        type severity;
```

```
        description
          "The vulnerability severity";
      }
      leaf os{
        type string;
        description
          "simple os information";
      }
      leaf addtional-info{
        type string;
        description
          "TBD";
      }
    }
    container web-attack-logs {
      description
        "If the log is web attack
        logs in nsf log";
      leaf attack-type{
        type string;
        description
          "Web Attack";
      }
      leaf rsp-code{
        type string;
        description
          "Response code";
      }
      leaf req-clientapp{
        type string;
        description
          "The client application";
      }
      leaf req-cookies{
        type string;
        description
          "Cookies";
      }
      leaf req-host{
        type string;
        description
          "The domain name of the
          requested host";
      }
      leaf raw-info{
        type string;
        description
          "The information describing
```

```
                    the packet triggering the
                    event.";
                }
              }
            }
          }
        }
        case counters {
          description
            "If the message type is counters";
          choice counter-type {
            description
              "The type of counter";
            case system-counter {
              container interface-counters {
                description
                  "The system counter type is
                  interface counter";
                uses i2nsf-system-counter-type-content;
              }
            }
            case nsf-counter{
              container firewall-counters {
                description
                  "The nsf counter type is
                  firewall counter";
                uses i2nsf-nsf-counters-type-content;
                container bound{
                  description
                  "Inbound or Outbound";
                  leaf in-interface {
                    type boolean;
                    description
                      "If the bound is inbound";
                  }
                  leaf out-interface {
                    type boolean;
                    description
                      "If the bound is outbound";
                  }
                }
              }
            }
            container policy-hit-counters {
              description
                "The counters of policy hit";
              uses i2nsf-nsf-counters-type-content;
              leaf hit-times{
```

```
                    type uint32;
                    description
                      "The hit times for policy";
                  }
                }
              }
            }
          }
          leaf message {
            type string;
            mandatory true;
            description
              "This is a message for monitoring information";
          }
          leaf time-stamp {
            type yang:date-and-time;
            mandatory true;
            description
              "Indicate the time when the message is generated";
          }
          leaf severity {
            type severity;
            mandatory true;
            description
              "The severity of the alarm such as
              critical, high, middle, low.";
          }
        }
      }
    }
    <CODE ENDS>
```

Figure 2: Data Model of Monitoring

## 6. Acknowledgments

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (R-20160222-002755, Cloud based Security Intelligence Technology Development for the Customized Security Service Provisioning).

This document has greatly benefited from inputs by Daeyoung Hyun.

# 7. References

## 7.1. 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.

## 7.2. Informative References

[i2nsf-framework]
Hares,, S., Lopez,, J., Dunbar, L., Strassner, J., and R. Kumar, "Framework for Interface to Network Security Functions", draft-ietf-i2nsf-framework-05 (work in progress), May 2017.

[i2nsf-monitoring-im]
Xia,, L., Zhang,, D., Wu, Y., Kumar, R., Lohiya, A., and H. Birkholz, "An Information Model for the Monitoring of Network Security Functions (NSF)", draft-zhang-i2nsf-info-model-monitoring-04 (work in progress), July 2017.

[i2nsf-terminology]
Hares,, S., Strassner,, J., Lopez,, D., Xia,, L., and H. Birkholz,, "Interface to Network Security Functions (I2NSF) Terminology", draft-ietf-i2nsf-terminology-04 (work in progress), July 2017.

[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-07 (work in progress), January 2017.

## Appendix A. draft-hong-i2nsf-nsf-monitoring-data-model-01

The following changes are made from draft-hong-i2nsf-nsf-monitoring-data-model-00:

1. The YANG data model is defined in more detail based on the information model for monitoring NSFs.

2. Typos and grammatical errors are corrected.

## Authors' Addresses

Dongjin Hong
Department of Computer Engineering
Sungkyunkwan University
2066 Seobu-Ro, Jangan-Gu
Suwon, Gyeonggi-Do 16419
Republic of Korea

Phone: +82 10 7630 5473
EMail: dong.jin@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

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

Susan Hares
Huawei
7453 Hickory Hill
Saline, MI 48176
USA

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

Liang Xia (Frank)
Huawei
101 Software Avenue, Yuhuatai District
Nanjing, Jiangsu
China

EMail: Frank.xialiang@huawei.com