Network Working Group J. Jeong, Ed.
Internet-Draft P. Lingga
Intended status: Standards Track Sungkyunkwan University
Expires: October 2, 2021 S. Hares
L. Xia
Huawei
H. Birkholz
Fraunhofer SIT
March 31, 2021

I2NSF NSF Monitoring Interface YANG Data Model
draft-ietf-i2nsf-nsf-monitoring-data-model-07

Abstract

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

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

According to [RFC8329], the interface provided by a Network Security Function (NSF) (e.g., Firewall, IPS, Anti-DDoS, or Anti-Virus function) to administrative entities (e.g., Security Controller) to enable remote management (i.e., configuring and monitoring) is referred to as an I2NSF Monitoring Interface. Monitoring procedures intent to acquire vital types of data with respect to NSFs, (e.g., alarms, records, and counters) via data in motion (e.g., queries, notifications, and events). The monitoring of NSF plays an important role in an overall security framework, if it is done in a timely and comprehensive way. The monitoring information generated by an NSF can be a good, early indication of anomalous behavior or malicious activity, such as denial of service attacks (DoS).

This document defines a comprehensive information model of an NSF monitoring interface that provides visibility for an NSF for an NSF data collector (e.g., Security Controller and NSF Data Analyzer). Note that an NSF data collector is defined as an entity to collect NSF monitoring data from an NSF, such as Security Controller and NSF

Data Analyzer. It specifies the information and illustrates the methods that enable an NSF to provide the information required in order to be monitored in a scalable and efficient way via the NSF Monitoring Interface. The information model for the NSF monitoring interface presented in this document is a complementary information model to the information model for the security policy provisioning functionality of the NSF-Facing Interface specified in [I-D.ietf-i2nsf-nsf-facing-interface-dm].

This document also defines a YANG [RFC7950] data model for the NSF monitoring interface, which is derived from the information model for the NSF monitoring interface.

2. Terminology

This document uses the terminology described in [RFC8329].

This document follows the guidelines of [RFC8407], uses the common YANG types defined in [RFC6991], and adopts the Network Management Datastore Architecture (NMDA) [RFC8342]. The meaning of the symbols in tree diagrams is defined in [RFC8340].

3. Use Cases for NSF Monitoring Data

As mentioned earlier, monitoring plays a critical role in an overall security framework. The monitoring of the NSF provides very valuable information to an NSF data collector (e.g., Security Controller and NSF data analyzer) in maintaining the provisioned security posture. Besides this, there are various other reasons to monitor the NSF as listed below:

- The security administrator with I2NSF User can configure a policy that is triggered on a specific event occurring in the NSF or the network [RFC8329] [I-D.ietf-i2nsf-consumer-facing-interface-dm]. If an NSF data collector detects the specified event, it configures additional security functions as defined by policies.

- The events triggered by an NSF as a result of security policy violation can be used by Security Information and Event Management (SIEM) to detect any suspicious activity in a larger correlation context.

- The events and activity logs from an NSF can be used to build advanced analytics, such as behavior and predictive models to improve security posture in large deployments.

- The NSF data collector can use events from the NSF for achieving high availability. It can take corrective actions such as restarting a failed NSF and horizontally scaling up the NSF.

- The events and activity logs from the NSF can aid in the root cause analysis of an operational issue, so it can improve debugging.

- The activity logs from the NSF can be used to build historical data for operational and business reasons.

4. Classification of NSF Monitoring Data

In order to maintain a strong security posture, it is not only necessary not only to configure an NSF's security policies but also to continuously monitor the NSF by consuming acquirable and observable information. This enables security administrators to assess the state of the network topology in a timely fashion. It is not possible to block all the internal and external threats based on static security posture. A more practical approach is supported by enabling dynamic security measures, for which continuous visibility is required. This document defines a set of information elements (and their scope) that can be acquired from an NSF and can be used as NSF monitoring information. In essence, these types of monitoring information can be leveraged to support constant visibility on multiple levels of granularity and can be consumed by the corresponding functions.

Three basic domains about the monitoring information originating from a system entity [RFC4949] or an NSF are highlighted in this document.

- Retention and Emission

- Notifications and Events

- Unsolicited Poll and Solicited Push

The Alarm Management Framework in [RFC3877] defines an Event as something that happens as a thing of of interest. It defines a fault as a change in status, crossing a threshold, or an external input to the system. In the I2NSF domain, I2NSF events are created and the scope of the Alarm Management Framework's Events is still applicable due to its broad definition. The model presented in this document elaborates on the workflow of creating I2NSF events in the context of NSF monitoring and on the way initial I2NSF events are created.

As with I2NSF components, every generic system entity can include a set of capabilities that creates information about the context,

composition, configuration, state or behavior of that system entity. This information is intended to be provided to other consumers of information and in the scope of this document, which deals with NSF information monitoring in an automated fashion.

4.1. Retention and Emission

Typically, a system entity populates standardized interface, such as SNMP, NETCONF, RESTCONF or CoMI to provide and emit created information directly via NSF Monitoring Interface. Alternatively, the created information is retained inside the system entity (or a hierarchy of system entities in a composite device) via records or counters that are not exposed directly via NSF Monistoring Interface.

Information emitted via standardized interfaces can be consumed by an I2NSF User that includes the capability to consume information not only via an I2NSF Interface (e.g., Consumer-Facing Interface [I-D.ietf-i2nsf-consumer-facing-interface-dm]), but also via interfaces complementary to the standardized interfaces a generic system entity provides.

Information retained on a system entity requires a corresponding I2NSF User to access aggregated records of information, typically in the form of log-files or databases. There are ways to aggregate records originating from different system entities over a network, for examples via Syslog Protocol [RFC5424] or Syslog over TCP [RFC6587]. But even if records are conveyed, the result is the same kind of retention in form of a bigger aggregate of records on another system entity.

An I2NSF User is required to process fresh [RFC4949] records created by I2NSF Functions in order to provide them to other I2NSF Components via the corresponding I2NSF Interfaces in a timely manner. This process is effectively based on homogenizing functions, which can access and convert specific kinds of records into information that can be provided and emitted via I2NSF interfaces.

When retained or emitted, the information required to support monitoring processes has to be processed by an I2NSF User at some point in the workflow. Typical locations of these I2NSF Users are:

- a system entity that creates the information

- a system entity that retains an aggregation of records

- an I2NSF Component that includes the capabilities of using standardized interfaces provided by other system entities that are not I2NSF Components

- o an I2NSF Component that creates the information

4.2. Notifications and Events

A specific task of I2NSF User is to process I2NSF Policy Rules. The rules of a policy are composed of three clauses: Events, Conditions, and Actions. In consequence, an I2NSF Event is specified to trigger an I2NSF Policy Rule. Such an I2NSF Event is defined as any important occurrence over time in the system being managed, and/or in the environment of the system being managed, which aligns well with the generic definition of Event from [RFC3877].

The model illustrated in this document introduces a complementary type of information that can be a conveyed notification.

Notification: An occurrence of a change of context, composition, configuration, state or behavior of a system entity that can be directly or indirectly observed by an I2NSF User and can be used as input for an event-clause in I2NSF Policy Rules.

A notification is similar to an I2NSF Event with the exception that it is created by a system entity that is not an I2NSF Component and that its importance is yet to be assessed. Semantically, a notification is not an I2NSF Event in the context of I2NSF, although they can potentially use the exact same information or data model. In respect to [RFC3877], a Notification is a specific subset of events, because they convey information about something that happens as a thing of of interest. In consequence, Notifications may contain information with very low expressiveness or relevance. Hence, additional post-processing functions, such as aggregation, correlation or simple anomaly detection, might have to be employed to satisfy a level of expressiveness that is required for an event-clause of an I2NSF Policy Rule.

It is important to note that the consumer of a notification (the observer) assesses the importance of a notification and not the producer. The producer can include metadata in a notification that supports the observer in assessing the importance (even metadata about severity), but the deciding entity is an I2NSF User.

4.3. Unsolicited Poll and Solicited Push

The freshness of the monitored information depends on the acquisition method. Ideally, an I2NSF User is accessing every relevant information about the I2NSF Component and is emitting I2NSF Events to an NSF data collector (e.g., Security Controller and NSF data analyzer) in a timely manner. Publication of events via a pubsub/

broker model, peer-2-peer meshes, or static defined channels are only a few examples on how a solicited push of I2NSF Events can be facilitated. The actual mechanic implemented by an I2NSF Component is out of the scope of this document.

Often, the corresponding management interfaces have to be queried in intervals or on-demand if required by an I2NSF Policy rule. In some cases, a collection of information has to be conducted via login mechanics provided by a system entity. Accessing records of information via this kind of unsolicited polls can introduce a significant latency in regard to the freshness of the monitored information. The actual definition of intervals implemented by an I2NSF Component is also out of scope of this document.

4.4. I2NSF Monitoring Terminology for Retained Information

Records: Unlike information emitted via notifications and events, records do not require immediate attention from an analyst but may be useful for visibility and retroactive cyber forensic. Depending on the record format, there are different qualities in regard to structure and detail. Records are typically stored in log-files or databases on a system entity or NSF. Records in the form of log-files usually include less structures but potentially more detailed information in regard to the changes of a system entity's characteristics. In contrast, databases often use more strict schemas or data models, therefore enforcing a better structure. However, they inhibit storing information that do not match those models ("closed world assumption"). Records can be continuously processed by I2NSF Agents that act as I2NSF Producer and emit events via functions specifically tailored to a certain type of record. Typically, records are information generated either by an NSF or a system entity about operational and informational data, or various changes in system characteristics, such as user activities, network/traffic status, and network activity. They are important for debugging, auditing and security forensic.

Counters: A specific representation of continuous value changes of information elements that potentially occur in high frequency. Prominent example are network interface counters, e.g., PDU amount or byte amount, drop counters, and error counters. Counters are useful in debugging and visibility into operational behavior of an NSF. An I2NSF Agent that observes the progression of counters can act as an I2NSF Producer and emit events in respect to I2NSF Policy Rules.

5. Conveyance of NSF Monitoring Information

As per the use cases of NSF monitoring data, information needs to be conveyed to various I2NSF Consumers based on requirements imposed by I2NSF Capabilities and workflows. There are multiple aspects to be considered in regard to the emission of monitoring information to requesting parties as listed below:

- Pull-Push Model: A set of data can be pushed by an NSF to a requesting party or pulled by a requesting party from an NSF. Specific types of information might need both the models at the same time if there are multiple I2NSF Consumers with varying requirements. In general, any I2NSF Event including a high severity assessment is considered to be of great importance and should be processed as soon as possible (push-model). Records, in contrast, are typically not as critical (pull-model). The I2NSF Architecture does not mandate a specific scheme for each type of information and is therefore out of scope of this document.

- Pub-Sub Model: In order for an I2NSF Provider to push monitoring information to multiple appropriate I2NSF Consumers, a subscription can be maintained by both I2NSF Components. Discovery of available monitoring information can be supported by an I2NSF Controller that takes the role of a broker and therefore includes I2NSF Capabilities that support registration.

- Export Frequency: Monitoring information can be emitted immediately upon generation by an NSF to requesting I2NSF Consumers or can be pushed periodically. The frequency of exporting the data depends upon its size and timely usefulness. It is out of the scope of I2NSF and left to each NSF implementation.

- Authentication: There may be a need for authentication between an I2NSF Producer of monitoring information and its corresponding I2NSF Consumer to ensure that critical information remains confidential. Authentication in the scope of I2NSF can also require its corresponding content authorization. This may be necessary, for example, if an NSF emits monitoring information to an I2NSF Consumer outside its administrative domain. The I2NSF Architecture does not mandate when and how specific authentication has to be implemented.

- Data-Transfer Model: Monitoring information can be pushed by an NSF using a connection-less model that does require a persistent connection or streamed over a persistent connection. An appropriate model depends on the I2NSF Consumer requirements and the semantics of the information to be conveyed.

- Data Model and Interaction Model for Data in Motion: There are a lot of transport mechanisms such as IP, UDP, and TCP. There are also open source implementations for specific set of data such as systems counter, e.g. IPFIX [RFC7011] and NetFlow [RFC3954]. The I2NSF does not mandate any specific method for a given data set, so it is up to each implementation.

5.1. Information Types and Acquisition Methods

In this document, most defined information types defined benefit from high visibility with respect to value changes, e.g., alarms and records. In contrast, values that change monotonically in a continuous way do not benefit from this high visibility. On the contrary, emitting each change would result in a useless amount of value updates. Hence, values, such as counter, are best acquired in periodic intervals.

The mechanisms provided by YANG Push [I-D.ietf-netconf-yang-push] and YANG Subscribed Notifications [I-D.ietf-netconf-subscribed-notifications] address exactly these set of requirements. YANG also enables semantically well-structured information, as well as subscriptions to datastores or event streams - by changes or periodically.

In consequence, this information model in this document is intended to support data models used in solicited or unsolicited event streams that potentially are facilitated by a subscription mechanism. A subset of information elements defined in the information model address this domain of application.

6. Basic Information Model for All Monitoring Data

As explained in the above section, there is a wealth of data available from the NSF that can be monitored. Firstly, there must be some general information with each monitoring message sent from an NSF that helps a consumer to identify meta data with that message, which are listed as below:

- message: Event, Alert, Alarm, Log, Counter, etc.

- vendor-name: The name of the NSF vendor.

- nsf-name: The name (or IP) of the NSF generating the message.

- severity: It indicates the severity level. There are total four levels, from 0 to 3. The smaller the numeral is, the higher the severity is.

7. Extended Information Model for Monitoring Data

This section covers the additional information associated with the system messages. The extended information model is only for the structured data such as alarm. Any unstructured data is specified with basic information model only.

7.1. System Alarms

Characteristics:

- o acquisition-method: subscription
- o emission-type: on-change
- o dampening-type: on-repetition

7.1.1. Memory Alarm

The following information should be included in a Memory Alarm:

- o event-name: mem-usage-alarm
- o usage: specifies the size of memory used.
- o threshold: The threshold triggering the alarm
- o severity: The severity of the alarm such as critical, high, medium, low
- o message: The memory usage exceeded the threshold

7.1.2. CPU Alarm

The following information should be included in a CPU Alarm:

- o event-name: cpu-usage-alarm
- o usage: Specifies the size of CPU used.
- o threshold: The threshold triggering the event
- o severity: The severity of the alarm such as critical, high, medium, low
- o message: The CPU usage exceeded the threshold.

7.1.3. Disk Alarm

The following information should be included in a Disk Alarm:

- event-name: disk-usage-alarm
- usage: Specifies the size of disk space used.
- threshold: The threshold triggering the event
- severity: The severity of the alarm such as critical, high, medium, low
- message: The disk usage exceeded the threshold.

7.1.4. Hardware Alarm

The following information should be included in a Hardware Alarm:

- event-name: hw-failure-alarm
- component-name: It indicates the HW component responsible for generating this alarm.
- severity: The severity of the alarm such as critical, high, medium, low
- message: The HW component has failed or degraded.

7.1.5. Interface Alarm

The following information should be included in an Interface Alarm:

- event-name: ifnet-state-alarm
- interface-name: The name of interface
- interface-state: up, down, congested
- threshold: The threshold triggering the event
- severity: The severity of the alarm such as critical, high, medium, low
- message: Current interface state

7.2. System Events

Characteristics:

- acquisition-method: subscription

- emission-type: on-change

- dampening-type: on-repetition

7.2.1. Access Violation

The following information should be included in this event:

- event-name: access-denied

- user: Name of a user

- group: Group to which a user belongs

- login-ip-address: Login IP address of a user

- authentication: User authentication mode. e.g., Local Authentication, Third-Party Server Authentication, Authentication Exemption, Single Sign-On (SSO) Authentication

- message: access is denied.

7.2.2. Configuration Change

The following information should be included in this event:

- event-name: config-change

- user: Name of a user

- group: Group to which a user belongs

- login-ip-address: Login IP address of a user

- authentication: User authentication mode. e.g., Local Authentication, Third-Party Server Authentication, Authentication Exemption, SSO Authentication

- message: Configuration is modified.

7.2.3. Traffic flows

The following information should be included in this event:

- src-ip: The source IPv4 or IPv6 address of the flows

- dst-ip: The destination IPv4 or IPv6 address of the flows

- src-port: The source port of the flows

- dst-port: The destination port of the flows

- protocol: The protocol of the packet flows.

- arrival-rate: Arrival rate of the same flow.

7.3. NSF Events

Characteristics:

- acquisition-method: subscription

- emission-type: on-change

- dampening-type: on-repetition

7.3.1. DDoS Detection

The following information should be included in a DDoS Event:

- event-name: detection-ddos

- attack-type: Any one of SYN flood, ACK flood, SYN-ACK flood, FIN/RST flood, TCP Connection flood, UDP flood, ICMP flood, HTTPS flood, HTTP flood, DNS query flood, DNS reply flood, SIP flood, and etc.

- dst-ip: The IP address of a victim under attack

- dst-port: The port number that the attack traffic aims at.

- start-time: The time stamp indicating when the attack started

- end-time: The time stamp indicating when the attack ended. If the attack is still undergoing when sending out the alarm, this field can be empty.

- attack-rate: The PPS of attack traffic

- attack-speed: the bps of attack traffic

- rule-name: The name of the rule being triggered

- profile: Security profile that traffic matches.

7.3.2. Session Table Event

The following information should be included in a Session Table Event:

- event-name: session-table

- current-session: The number of concurrent sessions

- maximum-session: The maximum number of sessions that the session table can support

- threshold: The threshold triggering the event

- message: The number of session table exceeded the threshold.

7.3.3. Virus Event

The following information should be included in a Virus Event:

- event-name: detection-virus

- virus: Type of the virus. e.g., trojan, worm, macro virus type

- virus-name: Name of the virus

- dst-ip: The destination IP address of the packet where the virus is found

- src-ip: The source IP address of the packet where the virus is found

- src-port: The source port of the packet where the virus is found

- dst-port: The destination port of the packet where the virus is found

- src-zone: The source security zone of the packet where the virus is found

- dst-zone: The destination security zone of the packet where the virus is found

- file-type: The type of the file where the virus is hided within

- file-name: The name of the file where the virus is hided within

- raw_info: The information describing the packet triggering the event.

- rule_name: The name of the rule being triggered

7.3.4. Intrusion Event

The following information should be included in an Intrusion Event:

- event-name: The name of event. e.g., detection-intrusion

- attack-type: Attack type, e.g., brutal force and buffer overflow

- src-ip: The source IP address of the packet

- dst-ip: The destination IP address of the packet

- src-port:The source port number of the packet

- dst-port: The destination port number of the packet

- src-zone: The source security zone of the packet

- dst-zone: The destination security zone of the packet

- protocol: The employed transport layer protocol. e.g.,TCP and UDP

- app: The employed application layer protocol. e.g.,HTTP and FTP

- rule-name: The name of the rule being triggered

- raw-info: The information describing the packet triggering the event

7.3.5. Botnet Event

The following information should be included in a Botnet Event:

- event-name: The name of event. e.g., detection-botnet

- botnet-name: The name of the detected botnet

- src-ip: The source IP address of the packet

- dst-ip: The destination IP address of the packet

- src-port: The source port number of the packet

- dst-port: The destination port number of the packet

- src-zone: The source security zone of the packet

- dst-zone: The destination security zone of the packet

- protocol: The employed transport layer protocol. e.g.,TCP and UDP

- role: The role of the communicating parties within the botnet:

    1. The packet from the zombie host to the attacker

    2. The packet from the attacker to the zombie host

    3. The packet from the IRC/WEB server to the zombie host

    4. The packet from the zombie host to the IRC/WEB server

    5. The packet from the attacker to the IRC/WEB server

    6. The packet from the IRC/WEB server to the attacker

    7. The packet from the zombie host to the victim

- rule-name: The name of the rule being triggered

- raw-info: The information describing the packet triggering the event.

7.3.6. Web Attack Event

The following information should be included in a Web Attack Alarm:

- event-name: The name of event. e.g., detection-web-attack

- attack-type: Concrete web attack type. e.g., SQL injection, command injection, XSS, CSRF

- src-ip: The source IP address of the packet

- dst-ip: The destination IP address of the packet

- src-port: The source port number of the packet

- dst-port: The destination port number of the packet

- src-zone: The source security zone of the packet

- dst-zone: The destination security zone of the packet

- request-method: The method of requirement. For instance, "PUT" and "GET" in HTTP

- req-uri: Requested URI

- rsp-code: Response code

- req-clientapp: The client application

- req-cookies: Cookies

- req-host: The domain name of the requested host

- uri-category: Matched URI category

- filtering-type: URL filtering type. e.g., Blacklist, Whitelist, User-Defined, Predefined, Malicious Category, and Unknown

- rule-name: The name of the rule being triggered

- profile: Security profile that traffic matches

7.4. System Logs

Characteristics:

- acquisition-method: subscription

- emission-type: on-change

- dampening-type: on-repetition

7.4.1. Access Log

Access logs record administrators' login, logout, and operations on a device. By analyzing them, security vulnerabilities can be identified. The following information should be included in an operation report:

- Administrator: Administrator that operates on the device

- login-ip-address: IP address used by an administrator to log in

- login-mode: Specifies the administrator logs in mode e.g. root, user

- operation-type: The operation type that the administrator execute, e.g., login, logout, and configuration.

- result: Command execution result

- content: Operation performed by an administrator after login.

7.4.2. Resource Utilization Log

Running reports record the device system's running status, which is useful for device monitoring. The following information should be included in running report:

- system-status: The current system's running status

- cpu-usage: Specifies the CPU usage.

- memory-usage: Specifies the memory usage.

- disk-usage: Specifies the disk usage.

- disk-left: Specifies the available disk space left.

- session-number: Specifies total concurrent sessions.

- process-number: Specifies total number of systems processes.

- in-traffic-rate: The total inbound traffic rate in pps

- out-traffic-rate: The total outbound traffic rate in pps

- in-traffic-speed: The total inbound traffic speed in bps

- out-traffic-speed: The total outbound traffic speed in bps

7.4.3. User Activity Log

User activity logs provide visibility into users' online records (such as login time, online/lockout duration, and login IP addresses) and the actions that users perform. User activity reports are helpful to identify exceptions during a user's login and network access activities.

- user: Name of a user

- group: Group to which a user belongs

- login-ip-addr: Login IP address of a user

- authentication: User authentication mode. e.g., Local Authentication, Third-Party Server Authentication, Authentication Exemption, SSO Authentication

- access: User access mode. e.g., PPP, SVN, LOCAL

- online-duration: Online duration

- logout-duration: Logout duration

- additional-info: Additional Information for login:

    1. type: User activities. e.g., Successful User Login, Failed Login attempts, User Logout, Successful User Password Change, Failed User Password Change, User Lockout, User Unlocking, Unknown

    2. cause: Cause of a failed user activity

7.5. NSF Logs

Characteristics:

- acquisition-method: subscription

- emission-type: on-change

- dampening-type: on-repetition

7.5.1. DPI Log

DPI Logs provide statistics on uploaded and downloaded files and data, sent and received emails, and alert and block records on websites. It is helpful to learn risky user behaviors and why access to some URLs is blocked or allowed with an alert record.

- attack-type: DPI action types. e.g., File Blocking, Data Filtering, and Application Behavior Control

- src-user: User source who generates the policy

- policy-name: Security policy name that traffic matches

- action: Action defined in the file blocking rule, data filtering rule, or application behavior control rule that traffic matches.

7.5.2. Vulnerability Scanning Log

Vulnerability scanning logs record the victim host and its related vulnerability information that should to be fixed. The following information should be included in the report:

- victim-ip: IP address of the victim host which has vulnerabilities

- vulnerability-id: The vulnerability id

- level: The vulnerability level. e.g., high, middle, and low

- OS: The operating system of the victim host

- service: The service which has vulnerability in the victim host

- protocol: The protocol type. e.g., TCP and UDP

- port-num: The port number

- vulnerability-info: The information about the vulnerability

- fix-suggestion: The fix suggestion to the vulnerability.

7.6. System Counter

Characteristics:

- acquisition-method: subscription or query

- emission-type: periodical

- dampening-type: none

7.6.1. Interface Counter

Interface counters provide visibility into traffic into and out of an NSF, and bandwidth usage.

- interface-name: Network interface name configured in NSF

- in-total-traffic-pkts: Total inbound packets

- out-total-traffic-pkts: Total outbound packets

- in-total-traffic-bytes: Total inbound bytes

- out-total-traffic-bytes: Total outbound bytes

- in-drop-traffic-pkts: Total inbound drop packets

- out-drop-traffic-pkts: Total outbound drop packets

- in-drop-traffic-bytes: Total inbound drop bytes

- out-drop-traffic-bytes: Total outbound drop bytes

- in-traffic-average-rate: Inbound traffic average rate in pps

- in-traffic-peak-rate: Inbound traffic peak rate in pps

- in-traffic-average-speed: Inbound traffic average speed in bps

- in-traffic-peak-speed: Inbound traffic peak speed in bps

- out-traffic-average-rate: Outbound traffic average rate in pps

- out-traffic-peak-rate: Outbound traffic peak rate in pps

- out-traffic-average-speed: Outbound traffic average speed in bps

- out-traffic-peak-speed: Outbound traffic peak speed in bps

7.7. NSF Counters

Characteristics:

- acquisition-method: subscription or query

- emission-type: periodical

- dampening-type: none

7.7.1. Firewall Counter

Firewall counters provide visibility into traffic signatures, bandwidth usage, and how the configured security and bandwidth policies have been applied.

- src-zone: Source security zone of traffic

- dst-zone: Destination security zone of traffic

- src-region: Source region of traffic

- dst-region: Destination region of traffic

- src-ip: Source IP address of traffic

- src-user: User who generates traffic

- dst-ip: Destination IP address of traffic

- src-port: Source port of traffic

- dst-port: Destination port of traffic

- protocol: Protocol type of traffic

- app: Application type of traffic

- policy-id: Security policy id that traffic matches

- policy-name: Security policy name that traffic matches

- in-interface: Inbound interface of traffic

- out-interface: Outbound interface of traffic

- total-traffic: Total traffic volume

- in-traffic-average-rate: Inbound traffic average rate in pps

- in-traffic-peak-rate: Inbound traffic peak rate in pps

- in-traffic-average-speed: Inbound traffic average speed in bps

- in-traffic-peak-speed: Inbound traffic peak speed in bps

- out-traffic-average-rate: Outbound traffic average rate in pps

- out-traffic-peak-rate: Outbound traffic peak rate in pps

- out-traffic-average-speed: Outbound traffic average speed in bps

- out-traffic-peak-speed: Outbound traffic peak speed in bps.

7.7.2. Policy Hit Counter

Policy Hit Counters record the security policy that traffic matches and its hit count. It can check if policy configurations are correct.

- src-zone: Source security zone of traffic

- dst-zone: Destination security zone of traffic

- src-region: Source region of the traffic

- dst-region: Destination region of the traffic

- src-ip: Source IP address of traffic

- src-user: User who generates traffic

- dst-ip: Destination IP address of traffic

- src-port: Source port of traffic

- dst-port: Destination port of traffic

- protocol: Protocol type of traffic

- app: Application type of traffic

- policy-id: Security policy id that traffic matches

- policy-name: Security policy name that traffic matches

- hit-times: The hit times that the security policy matches the specified traffic.

8. NSF Monitoring Management in I2NSF

A standard model for monitoring data is required for an administrator to check the monitoring data generated by an NSF. The administrator can check the monitoring data through the following process. When the NSF monitoring data that is under the standard format is generated, the NSF forwards it to an NSF data collector via the I2NSF NSF Monitoring Interface. The NSF data collector delivers it to I2NSF Consumer or Developer's Management System (DMS) so that the administrator can know the state of the I2NSF framework.

In order to communicate with other components, an I2NSF framework [RFC8329] requires the interfaces. The three main interfaces in I2NSF framework are used for sending monitoring data as follows:

- I2NSF Consumer-Facing Interface [I-D.ietf-i2nsf-consumer-facing-interface-dm]: When an I2NSF User makes a security policy and forwards it to the Security Controller via Consumer-Facing Interface, it can specify the threat-feed for threat prevention, the custom list, the malicious code scan group, and the event map group. They can be used as an event to be monitored by an NSF.

- I2NSF Registration Interface [I-D.ietf-i2nsf-registration-interface-dm]: The Network Functions Virtualization (NFV) architecture provides the lifecycle management of a Virtual Network Function (VNF) via the Ve-Vnfm interface. The role of Ve-Vnfm is to request VNF lifecycle management (e.g., the instantiation and de-instantiation of an NSF, and load balancing among NSFs), exchange configuration information, and exchange status information for a network service. In the I2NSF framework, the DMS manages data about resource states and network traffic for the lifecycle management of an NSF. Therefore, the generated monitoring data from NSFs are delivered from the NSF data collector to the DMS via either Registration Interface or a new interface (e.g., NSF Monitoring Interface). These data are delivered from the DMS to the VNF Manager in the Management and Orchestration (MANO) in the NFV system [I-D.ietf-i2nsf-applicability].

- I2NSF NSF Monitoring Interface [RFC8329]: After a high-level security policy from I2NSF User is translated by security policy translator [I-D.yang-i2nsf-security-policy-translation] in the Security Controller, the translated security policy (i.e., low-level policy) is applied to an NSF via NSF-Facing Interface. The monitoring data model for an NSF specifies the list of events that can trigger Event-Condition-Action (ECA) policies via NSF Monitoring Interface.

9. Tree Structure

The tree structure of the NSF monitoring YANG module is provided below:

```
module: ietf-i2nsf-nsf-monitoring
  +--ro i2nsf-counters
  |  +--ro system-interface* [interface-name]
  |  |  +--ro acquisition-method?           identityref
  |  |  +--ro emission-type?                identityref
```

```
  |  |  +--ro dampening-type?                identityref
  |  |  +--ro interface-name                 string
  |  |  +--ro in-total-traffic-pkts?         yang:counter32
  |  |  +--ro out-total-traffic-pkts?        yang:counter32
  |  |  +--ro in-total-traffic-bytes?        uint64
  |  |  +--ro out-total-traffic-bytes?       uint64
  |  |  +--ro in-drop-traffic-pkts?          yang:counter32
  |  |  +--ro out-drop-traffic-pkts?         yang:counter32
  |  |  +--ro in-drop-traffic-bytes?         uint64
  |  |  +--ro out-drop-traffic-bytes?        uint64
  |  |  +--ro total-traffic?                 yang:counter32
  |  |  +--ro in-traffic-average-rate?       uint32
  |  |  +--ro in-traffic-peak-rate?          uint32
  |  |  +--ro in-traffic-average-speed?      uint32
  |  |  +--ro in-traffic-peak-speed?         uint32
  |  |  +--ro out-traffic-average-rate?      uint32
  |  |  +--ro out-traffic-peak-rate?         uint32
  |  |  +--ro out-traffic-average-speed?     uint32
  |  |  +--ro out-traffic-peak-speed?        uint32
  |  |  +--ro message?                       string
  |  |  +--ro vendor-name?                   string
  |  |  +--ro nsf-name?                      string
  |  |  +--ro severity?                      severity
  |  +--ro nsf-firewall* [policy-name]
  |  |  +--ro acquisition-method?       identityref
  |  |  +--ro emission-type?            identityref
  |  |  +--ro dampening-type?           identityref
  |  |  +--ro policy-name
    -> /nsfi:i2nsf-security-policy/system-policy/system-policy-name
  |  |  +--ro src-user?                     string
  |  |  +--ro total-traffic?                yang:counter32
  |  |  +--ro in-traffic-average-rate?      uint32
  |  |  +--ro in-traffic-peak-rate?         uint32
  |  |  +--ro in-traffic-average-speed?     uint32
  |  |  +--ro in-traffic-peak-speed?        uint32
  |  |  +--ro out-traffic-average-rate?     uint32
  |  |  +--ro out-traffic-peak-rate?        uint32
  |  |  +--ro out-traffic-average-speed?    uint32
  |  |  +--ro out-traffic-peak-speed?       uint32
  |  |  +--ro message?                      string
  |  |  +--ro vendor-name?                  string
  |  |  +--ro nsf-name?                     string
  |  |  +--ro severity?                     severity
  |  +--ro nsf-policy-hits* [policy-name]
  |     +--ro acquisition-method?   identityref
  |     +--ro emission-type?        identityref
  |     +--ro dampening-type?       identityref
  |     +--ro policy-name
```

```
    -> /nsfi:i2nsf-security-policy/system-policy/system-policy-name
  |     +--ro src-user?             string
  |     +--ro message?              string
  |     +--ro vendor-name?          string
  |     +--ro nsf-name?             string
  |     +--ro severity?             severity
  |     +--ro hit-times?            yang:counter32
  +--rw i2nsf-monitoring-configuration
     +--rw i2nsf-system-detection-alarm
     |  +--rw enabled?        boolean
     |  +--rw system-alarm* [alarm-type]
     |     +--rw alarm-type          enumeration
     |     +--rw threshold?          uint8
     |     +--rw dampening-period?   uint32
     +--rw i2nsf-system-detection-event
     |  +--rw enabled?            boolean
     |  +--rw dampening-period?   uint32
     +--rw i2nsf-traffic-flows
     |  +--rw dampening-period?   uint32
     |  +--rw enabled?            boolean
     +--rw i2nsf-nsf-detection-ddos {i2nsf-nsf-detection-ddos}?
     |  +--rw enabled?            boolean
     |  +--rw dampening-period?   uint32
     +--rw i2nsf-nsf-detection-session-table-configuration
     |  +--rw enabled?            boolean
     |  +--rw dampening-period?   uint32
     +--rw i2nsf-nsf-detection-virus {i2nsf-nsf-detection-virus}?
     |  +--rw enabled?            boolean
     |  +--rw dampening-period?   uint32
     +--rw i2nsf-nsf-detection-intrusion
        {i2nsf-nsf-detection-intrusion}?
     |  +--rw enabled?            boolean
     |  +--rw dampening-period?   uint32
     +--rw i2nsf-nsf-detection-botnet {i2nsf-nsf-detection-botnet}?
     |  +--rw enabled?            boolean
     |  +--rw dampening-period?   uint32
     +--rw i2nsf-nsf-detection-web-attack
        {i2nsf-nsf-detection-web-attack}?
     |  +--rw enabled?            boolean
     |  +--rw dampening-period?   uint32
     +--rw i2nsf-nsf-system-access-log
     |  +--rw enabled?            boolean
     |  +--rw dampening-period?   uint32
     +--rw i2nsf-system-res-util-log
     |  +--rw enabled?            boolean
     |  +--rw dampening-period?   uint32
     +--rw i2nsf-system-user-activity-log
     |  +--rw enabled?            boolean
```

```
     |  +--rw dampening-period?   uint32
     +--rw i2nsf-nsf-log-dpi {i2nsf-nsf-log-dpi}?
     |  +--rw enabled?            boolean
     |  +--rw dampening-period?   uint32
     +--rw i2nsf-nsf-log-vuln-scan {i2nsf-nsf-log-vuln-scan}?
     |  +--rw enabled?            boolean
     |  +--rw dampening-period?   uint32
     +--rw i2nsf-counter
        +--rw period?   uint16

  notifications:
    +---n i2nsf-event
    |  +--ro (sub-event-type)?
    |     +--:(i2nsf-system-detection-alarm)
    |     |  +--ro i2nsf-system-detection-alarm
    |     |     +--ro alarm-category?       identityref
    |     |     +--ro component-name?       string
    |     |     +--ro interface-name?       string
    |     |     +--ro interface-state?      enumeration
    |     |     +--ro acquisition-method?   identityref
    |     |     +--ro emission-type?        identityref
    |     |     +--ro dampening-type?       identityref
    |     |     +--ro usage?                uint8
    |     |     +--ro threshold?            uint8
    |     |     +--ro message?              string
    |     |     +--ro vendor-name?          string
    |     |     +--ro nsf-name?             string
    |     |     +--ro severity?             severity
    |     +--:(i2nsf-system-detection-event)
    |     |  +--ro i2nsf-system-detection-event
    |     |     +--ro event-category?       identityref
    |     |     +--ro acquisition-method?   identityref
    |     |     +--ro emission-type?        identityref
    |     |     +--ro dampening-type?       identityref
    |     |     +--ro user                  string
    |     |     +--ro group                 string
    |     |     +--ro login-ip-addr         inet:ip-address
    |     |     +--ro authentication?       identityref
    |     |     +--ro message?              string
    |     |     +--ro vendor-name?          string
    |     |     +--ro nsf-name?             string
    |     |     +--ro severity?             severity
    |     +--:(i2nsf-traffic-flows)
    |     |  +--ro i2nsf-traffic-flows
    |     |     +--ro src-ip?               inet:ip-address
    |     |     +--ro dst-ip?               inet:ip-address
    |     |     +--ro protocol?             identityref
    |     |     +--ro src-port?             inet:port-number
```

```
    |     |     +--ro dst-port?             inet:port-number
    |     |     +--ro arrival-rate?         uint32
    |     |     +--ro acquisition-method?   identityref
    |     |     +--ro emission-type?        identityref
    |     |     +--ro dampening-type?       identityref
    |     |     +--ro message?              string
    |     |     +--ro vendor-name?          string
    |     |     +--ro nsf-name?             string
    |     |     +--ro severity?             severity
    |     +--:(i2nsf-nsf-detection-session-table)
    |        +--ro i2nsf-nsf-detection-session-table
    |           +--ro current-session?   uint32
    |           +--ro maximum-session?   uint32
    |           +--ro threshold?         uint32
    |           +--ro message?           string
    |           +--ro vendor-name?       string
    |           +--ro nsf-name?          string
    |           +--ro severity?          severity
    +---n i2nsf-log
    |  +--ro (sub-logs-type)?
    |     +--:(i2nsf-nsf-system-access-log)
    |     |  +--ro i2nsf-nsf-system-access-log
    |     |     +--ro login-ip              inet:ip-address
    |     |     +--ro administrator?        string
    |     |     +--ro login-mode?           login-mode
    |     |     +--ro operation-type?       operation-type
    |     |     +--ro result?               string
    |     |     +--ro content?              string
    |     |     +--ro acquisition-method?   identityref
    |     |     +--ro emission-type?        identityref
    |     |     +--ro dampening-type?       identityref
    |     |     +--ro message?              string
    |     |     +--ro vendor-name?          string
    |     |     +--ro nsf-name?             string
    |     |     +--ro severity?             severity
    |     +--:(i2nsf-system-res-util-log)
    |     |  +--ro i2nsf-system-res-util-log
    |     |     +--ro system-status?        string
    |     |     +--ro cpu-usage?            uint8
    |     |     +--ro memory-usage?         uint8
    |     |     +--ro disk-usage?           uint8
    |     |     +--ro disk-left?            uint8
    |     |     +--ro session-num?          uint8
    |     |     +--ro process-num?          uint8
    |     |     +--ro in-traffic-rate?      uint32
    |     |     +--ro out-traffic-rate?     uint32
    |     |     +--ro in-traffic-speed?     uint32
    |     |     +--ro out-traffic-speed?    uint32
```

```
    |     |     +--ro acquisition-method?   identityref
    |     |     +--ro emission-type?        identityref
    |     |     +--ro dampening-type?       identityref
    |     |     +--ro message?              string
    |     |     +--ro vendor-name?          string
    |     |     +--ro nsf-name?             string
    |     |     +--ro severity?             severity
    |     +--:(i2nsf-system-user-activity-log)
    |        +--ro i2nsf-system-user-activity-log
    |           +--ro acquisition-method?   identityref
    |           +--ro emission-type?        identityref
    |           +--ro dampening-type?       identityref
    |           +--ro user                  string
    |           +--ro group                 string
    |           +--ro login-ip-addr         inet:ip-address
    |           +--ro authentication?       identityref
    |           +--ro message?              string
    |           +--ro vendor-name?          string
    |           +--ro nsf-name?             string
    |           +--ro severity?             severity
    |           +--ro access?               identityref
    |           +--ro online-duration?      string
    |           +--ro logout-duration?      string
    |           +--ro additional-info?      string
    +---n i2nsf-nsf-event
       +--ro (sub-event-type)?
          +--:(i2nsf-nsf-detection-ddos) {i2nsf-nsf-detection-ddos}?
          |  +--ro i2nsf-nsf-detection-ddos
          |     +--ro dst-ip?               inet:ip-address
          |     +--ro dst-port?             inet:port-number
          |     +--ro rule-name
           -> /nsfi:i2nsf-security-policy/system-policy/rules/rule-name
          |     +--ro raw-info?             string
          |     +--ro attack-type?          identityref
          |     +--ro start-time            yang:date-and-time
          |     +--ro end-time              yang:date-and-time
          |     +--ro attack-src-ip?        inet:ip-address
          |     +--ro attack-dst-ip?        inet:ip-address
          |     +--ro attack-rate?          uint32
          |     +--ro attack-speed?         uint32
          |     +--ro action?               log-action
          |     +--ro acquisition-method?   identityref
          |     +--ro emission-type?        identityref
          |     +--ro dampening-type?       identityref
          |     +--ro message?              string
          |     +--ro vendor-name?          string
          |     +--ro nsf-name?             string
          |     +--ro severity?             severity
```

```
          +--:(i2nsf-nsf-detection-virus) {i2nsf-nsf-detection-virus}?
          |  +--ro i2nsf-nsf-detection-virus
          |     +--ro dst-ip?               inet:ip-address
          |     +--ro dst-port?             inet:port-number
          |     +--ro rule-name
           -> /nsfi:i2nsf-security-policy/system-policy/rules/rule-name
          |     +--ro raw-info?             string
          |     +--ro src-ip?               inet:ip-address
          |     +--ro src-port?             inet:port-number
          |     +--ro src-zone?             string
          |     +--ro dst-zone?             string
          |     +--ro virus?                identityref
          |     +--ro virus-name?           string
          |     +--ro file-type?            string
          |     +--ro file-name?            string
          |     +--ro os?                   string
          |     +--ro action?               log-action
          |     +--ro acquisition-method?   identityref
          |     +--ro emission-type?        identityref
          |     +--ro dampening-type?       identityref
          |     +--ro message?              string
          |     +--ro vendor-name?          string
          |     +--ro nsf-name?             string
          |     +--ro severity?             severity
          +--:(i2nsf-nsf-detection-intrusion)
              {i2nsf-nsf-detection-intrusion}?
          |  +--ro i2nsf-nsf-detection-intrusion
          |     +--ro dst-ip?               inet:ip-address
          |     +--ro dst-port?             inet:port-number
          |     +--ro rule-name
           -> /nsfi:i2nsf-security-policy/system-policy/rules/rule-name
          |     +--ro raw-info?             string
          |     +--ro src-ip?               inet:ip-address
          |     +--ro src-port?             inet:port-number
          |     +--ro src-zone?             string
          |     +--ro dst-zone?             string
          |     +--ro protocol?             identityref
          |     +--ro app?                  string
          |     +--ro attack-type?          identityref
          |     +--ro action?               log-action
          |     +--ro attack-rate?          uint32
          |     +--ro attack-speed?         uint32
          |     +--ro acquisition-method?   identityref
          |     +--ro emission-type?        identityref
          |     +--ro dampening-type?       identityref
          |     +--ro message?              string
          |     +--ro vendor-name?          string
          |     +--ro nsf-name?             string
```

```
          |     +--ro severity?             severity
          +--:(i2nsf-nsf-detection-botnet)
            {i2nsf-nsf-detection-botnet}?
          |  +--ro i2nsf-nsf-detection-botnet
          |     +--ro dst-ip?               inet:ip-address
          |     +--ro dst-port?             inet:port-number
          |     +--ro rule-name
           -> /nsfi:i2nsf-security-policy/system-policy/rules/rule-name
          |     +--ro raw-info?             string
          |     +--ro src-ip?               inet:ip-address
          |     +--ro src-port?             inet:port-number
          |     +--ro src-zone?             string
          |     +--ro dst-zone?             string
          |     +--ro attack-type?          identityref
          |     +--ro protocol?             identityref
          |     +--ro botnet-name?          string
          |     +--ro role?                 string
          |     +--ro action?               log-action
          |     +--ro botnet-pkt-num?       uint8
          |     +--ro os?                   string
          |     +--ro acquisition-method?   identityref
          |     +--ro emission-type?        identityref
          |     +--ro dampening-type?       identityref
          |     +--ro message?              string
          |     +--ro vendor-name?          string
          |     +--ro nsf-name?             string
          |     +--ro severity?             severity
          +--:(i2nsf-nsf-detection-web-attack)
              {i2nsf-nsf-detection-web-attack}?
          |  +--ro i2nsf-nsf-detection-web-attack
          |     +--ro dst-ip?               inet:ip-address
          |     +--ro dst-port?             inet:port-number
          |     +--ro rule-name
           -> /nsfi:i2nsf-security-policy/system-policy/rules/rule-name
          |     +--ro raw-info?             string
          |     +--ro src-ip?               inet:ip-address
          |     +--ro src-port?             inet:port-number
          |     +--ro src-zone?             string
          |     +--ro dst-zone?             string
          |     +--ro attack-type?          identityref
          |     +--ro request-method?       identityref
          |     +--ro req-uri?              string
          |     +--ro uri-category?         string
          |     +--ro filtering-type*       identityref
          |     +--ro rsp-code?             string
          |     +--ro req-clientapp?        string
          |     +--ro req-cookies?          string
          |     +--ro req-host?             string
```

```
          |     +--ro acquisition-method?   identityref
          |     +--ro emission-type?        identityref
          |     +--ro dampening-type?       identityref
          |     +--ro action?               log-action
          |     +--ro message?              string
          |     +--ro vendor-name?          string
          |     +--ro nsf-name?             string
          |     +--ro severity?             severity
          +--:(i2nsf-nsf-log-vuln-scan) {i2nsf-nsf-log-vuln-scan}?
          |  +--ro i2nsf-nsf-log-vuln-scan
          |     +--ro vulnerability-id?     uint8
          |     +--ro victim-ip?            inet:ip-address
          |     +--ro protocol?             identityref
          |     +--ro port-num?             inet:port-number
          |     +--ro level?                severity
          |     +--ro os?                   string
          |     +--ro vulnerability-info?   string
          |     +--ro fix-suggestion?       string
          |     +--ro service?              string
          |     +--ro acquisition-method?   identityref
          |     +--ro emission-type?        identityref
          |     +--ro dampening-type?       identityref
          |     +--ro message?              string
          |     +--ro vendor-name?          string
          |     +--ro nsf-name?             string
          |     +--ro severity?             severity
          +--:(i2nsf-nsf-log-dpi) {i2nsf-nsf-log-dpi}?
             +--ro i2nsf-nsf-log-dpi
                +--ro attack-type?          dpi-type
                +--ro acquisition-method?   identityref
                +--ro emission-type?        identityref
                +--ro dampening-type?       identityref
                +--ro policy-name
           -> /nsfi:i2nsf-security-policy/system-policy/system-policy-name
                +--ro src-user?             string
                +--ro message?              string
                +--ro vendor-name?          string
                +--ro nsf-name?             string
                +--ro severity?             severity
```

Figure 1: Information Model for NSF Monitoring

10. YANG Data Model

This section describes a YANG module of I2NSF NSF Monitoring. This YANG module imports from [RFC6991], and makes references to [RFC0768][RFC0791][RFC0792][RFC0793][RFC0956][RFC2616][RFC4443][RFC8200][RFC8641].

```
<CODE BEGINS> file "ietf-i2nsf-nsf-monitoring@2021-03-31.yang"

module ietf-i2nsf-nsf-monitoring {
  yang-version 1.1;
  namespace
    "urn:ietf:params:xml:ns:yang:ietf-i2nsf-nsf-monitoring";
  prefix
    nsfmi;
  import ietf-inet-types{
    prefix inet;
    reference
      "Section 4 of RFC 6991";
  }
  import ietf-yang-types {
    prefix yang;
    reference
      "Section 3 of RFC 6991";
  }
  import ietf-i2nsf-policy-rule-for-nsf {
    prefix nsfi;
  }
  organization
    "IETF I2NSF (Interface to Network Security Functions)
     Working Group";
  contact
    "WG Web: <http://tools.ietf.org/wg/i2nsf>
     WG List: <mailto:i2nsf@ietf.org>

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

     Editor: Patrick Lingga
     <mailto:patricklink@skku.edu>";

  description
    "This module is a YANG module for I2NSF NSF Monitoring.

     Copyright (c) 2021 IETF Trust and the persons identified as
     authors of the code.  All rights reserved.

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

     This version of this YANG module is part of RFC XXXX
```

```
     (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
     for full legal notices.";

  revision "2021-03-31" {
    description "Initial revision";
    reference
      "RFC XXXX: I2NSF NSF Monitoring YANG Data Model";

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

  /*
   * Typedefs
   */

  typedef severity {
    type enumeration {
      enum critical {
        description
          "The 'critical' severity level indicates that
           an immediate corrective action is required.
           A 'critical' severity is reported when a service
           becomes totally out of service and must be restored.";
      }
      enum high {
        description
          "The 'high' severity level indicates that
           an urgent corrective action is required.
           A 'high' severity is reported when there is
           a severe degradation in the capability of the
           service and its full capability must be restored.";
      }
      enum middle {
        description
          "The 'middle' severity level indicates the
           existence of a non-service-affecting fault
           condition and corrective action should be done
           to prevent a more serious fault. The 'middle'
           severity is reported when the detected problem
           is not degrading the capability of the service but
           might happen if not prevented.";
      }
      enum low {
        description
          "The 'low' severity level indicates the detection
           of a potential fault before any effect is felt.
           The 'low' severity is reported when an action should
```

```
          be done before a fault happen.";
      }
    }
    description
      "An indicator representing severity level. The severity level
       starting from the highest are critical, high, middle, and
       low.";
    reference
      "RFC 8632: A YANG Data Model for Alarm Management -
       The severity levels are defined.";
  }

  typedef log-action {
    type enumeration {
      enum allow {
        description
          "If action is allowed";
      }
      enum alert {
        description
          "If action is alert";
      }
      enum block {
        description
          "If action is block";
      }
      enum discard {
        description
          "If action is discarded";
      }
      enum declare {
        description
          "If action is declared";
      }
      enum block-ip {
        description
          "If action is block-ip";
      }
      enum block-service{
        description
          "If action is block-service";
      }
    }
    description
      "The type representing action for logging.";
  }

  typedef dpi-type{
```

```
    type enumeration {
      enum file-blocking{
        description
          "DPI for blocking file";
      }
      enum data-filtering{
        description
          "DPI for filtering data";
      }
      enum application-behavior-control{
        description
          "DPI for controlling application behavior";
      }
    }
    description
      "The type of deep packet inspection.";
  }

  typedef operation-type{
    type enumeration {
      enum login{
        description
          "Login operation";
      }
      enum logout{
        description
          "Logout operation";
      }
      enum configuration{
        description
          "Configuration operation";
      }
    }
    description
      "The type of operation done by a user
       during a session.";
  }

  typedef login-mode{
    type enumeration {
      enum root{
        description
          "Root login-mode";
      }
      enum user{
        description
          "User login-mode";
      }
```

```
      enum guest{
        description
          "Guest login-mode";
      }
    }
    description
      "The authorization login-mode done by a user.";
  }

  /*
   * Identity
   */

  identity characteristics {
    description
      "Base identity for monitoring information
       characteristics";
  }
  identity acquisition-method {
    base characteristics;
    description
      "The type of acquisition-method. It can be multiple
       types at once.";
  }
  identity subscription {
    base acquisition-method;
    description
      "The acquisition-method type is subscription.";
  }
  identity query {
    base acquisition-method;
    description
      "The acquisition-method type is query.";
  }
  identity emission-type {
    base characteristics;
    description
      "The type of emission-type.";
  }
  identity periodical {
    base emission-type;
    description
      "The emission-type type is periodical.";
  }
  identity on-change {
    base emission-type;
    description
      "The emission-type type is on-change.";
```

```
  }
  identity dampening-type {
    base characteristics;
    description
      "The type of dampening-type.";
  }
  identity no-dampening {
    base dampening-type;
    description
      "The dampening-type is no-dampening.";
  }
  identity on-repetition {
    base dampening-type;
    description
      "The dampening-type is on-repetition.";
  }
  identity none {
    base dampening-type;
    description
      "The dampening-type is none.";
  }
  identity authentication-mode {
    description
      "User authentication mode types:
       e.g., Local Authentication,
       Third-Party Server Authentication,
       Authentication Exemption, or Single Sign-On (SSO)
       Authentication.";
  }
  identity local-authentication {
    base authentication-mode;
    description
      "Authentication-mode : local authentication.";
  }
  identity third-party-server-authentication {
    base authentication-mode;
    description
      "If authentication-mode is
       third-party-server-authentication";
  }
  identity exemption-authentication {
    base authentication-mode;
    description
      "If authentication-mode is
       exemption-authentication";
  }
  identity sso-authentication {
    base authentication-mode;
```

```
    description
      "If authentication-mode is
       sso-authentication";
  }
  identity alarm-type {
    description
      "Base identity for detectable alarm types";
  }
  identity mem-usage-alarm {
    base alarm-type;
    description
      "A memory alarm is alerted.";
  }
  identity cpu-usage-alarm {
    base alarm-type;
    description
      "A CPU alarm is alerted.";
  }
  identity disk-usage-alarm {
    base alarm-type;
    description
      "A disk alarm is alerted.";
  }
  identity hw-failure-alarm {
    base alarm-type;
    description
      "A hardware alarm is alerted.";
  }
  identity ifnet-state-alarm {
    base alarm-type;
    description
      "An interface alarm is alerted.";
  }
  identity event-type {
    description
      "Base identity for detectable event types";
  }
  identity access-denied {
    base event-type;
    description
      "The system event is access-denied.";
  }
  identity config-change {
    base event-type;
    description
      "The system event is config-change.";
  }
  identity attack-type {
```

```
    description
      "The root ID of attack-based notification
       in the notification taxonomy";
  }
  identity system-attack-type {
    base attack-type;
    description
      "This ID is intended to be used
       in the context of system events.";
  }
  identity nsf-attack-type {
    base attack-type;
    description
      "This ID is intended to be used
       in the context of NSF event.";
  }
  identity botnet-attack-type {
    base nsf-attack-type;
    description
      "This indicates that this attack type is botnet.
       The usual semantic and taxonomy is missing
       and a name is used.";
  }
  identity virus-type {
    base nsf-attack-type;
    description
      "The type of virus. It caan be multiple types at once.
       This attack type is associated with a detected
       system-log virus-attack.";
  }
  identity trojan {
    base virus-type;
    description
      "The detected virus type is trojan.";
  }
  identity worm {
    base virus-type;
    description
      "The detected virus type is worm.";
  }
  identity macro {
    base virus-type;
    description
      "The detected virus type is macro.";
  }
  identity intrusion-attack-type {
    base nsf-attack-type;
    description
```

```
      "The attack type is associated with a detected
       system-log intrusion.";
  }
  identity brute-force {
    base intrusion-attack-type;
    description
      "The intrusion type is brute-force.";
  }
  identity buffer-overflow {
    base intrusion-attack-type;
    description
      "The intrusion type is buffer-overflow.";
  }
  identity web-attack-type {
    base nsf-attack-type;
    description
      "The attack type is associated with a detected
       system-log web-attack.";
  }
  identity command-injection {
    base web-attack-type;
    description
      "The detected web attack type is command injection.";
  }
  identity xss {
    base web-attack-type;
    description
      "The detected web attack type is XSS.";
  }
  identity csrf {
    base web-attack-type;
    description
      "The detected web attack type is CSRF.";
  }
  identity flood-type {
    base nsf-attack-type;
    description
      "Base identity for detectable flood types";
  }
  identity syn-flood {
    base flood-type;
    description
      "A SYN flood is detected.";
  }
  identity ack-flood {
    base flood-type;
    description
      "An ACK flood is detected.";
```

```
  }
  identity syn-ack-flood {
    base flood-type;
    description
      "A SYN-ACK flood is detected.";
  }
  identity fin-rst-flood {
    base flood-type;
    description
      "A FIN-RST flood is detected.";
  }
  identity tcp-con-flood {
    base flood-type;
    description
      "A TCP connection flood is detected.";
  }
  identity udp-flood {
    base flood-type;
    description
      "A UDP flood is detected.";
  }
  identity icmp-flood {
    base flood-type;
    description
      "Either an ICMPv4 or ICMPv6 flood is detected.";
  }
  identity icmpv4-flood {
    base flood-type;
    description
      "An ICMPv4 flood is detected.";
  }
  identity icmpv6-flood {
    base flood-type;
    description
      "An ICMPv6 flood is detected.";
  }
  identity http-flood {
    base flood-type;
    description
      "An HTTP flood is detected.";
  }
  identity https-flood {
    base flood-type;
    description
      "An HTTPS flood is detected.";
  }
  identity dns-query-flood {
    base flood-type;
```

```
    description
      "A DNS query flood is detected.";
  }
  identity dns-reply-flood {
    base flood-type;
    description
     "A DNS reply flood is detected.";
  }
  identity sip-flood {
    base flood-type;
    description
      "An SIP flood is detected.";
  }

  identity req-method {
    description
      "A set of request types (if applicable).
       For instance, PUT or GET in HTTP.";
  }
  identity put-req {
    base req-method;
    description
      "The detected request type is PUT.";
  }
  identity get-req {
    base req-method;
    description
      "The detected request type is GET.";
  }
  identity filter-type {
    description
      "The type of filter used to detect an attack,
       for example, a web-attack.  It can be applicable to
       more than web-attacks.  It can be more than one type.";
  }
  identity whitelist {
    base filter-type;
    description
      "The applied filter type is whitelist.";
  }
  identity blacklist {
    base filter-type;
    description
      "The applied filter type is blacklist.";
  }
  identity user-defined {
   base filter-type;
    description
```

```
      "The applied filter type is user-defined.";
  }
  identity malicious-category {
    base filter-type;
    description
      "The applied filter is malicious category.";
  }
  identity unknown-filter {
    base filter-type;
    description
      "The applied filter is unknown.";
  }

  identity access-mode {
    description
      "Base identity for detectable access mode.";
  }
  identity ppp {
    base access-mode;
    description
      "Access-mode: ppp";
  }
  identity svn {
    base access-mode;
    description
      "Access-mode: svn";
  }
  identity local {
    base access-mode;
    description
      "Access-mode: local";
  }

  identity protocol-type {
    description
      "An identity used to enable type choices in leaves
       and leaflists with respect to protocol metadata.";
  }
  identity tcp {
    base ipv4;
    base ipv6;
    description
      "TCP protocol type.";
    reference
      "RFC 793: Transmission Control Protocol";
  }
  identity udp {
    base ipv4;
```

```
      base ipv6;
      description
        "UDP protocol type.";
      reference
        "RFC 768: User Datagram Protocol";
    }
    identity icmp {
      base ipv4;
      base ipv6;
      description
        "General ICMP protocol type.";
      reference
        "RFC 792: Internet Control Message Protocol
         RFC 4443: Internet Control Message Protocol
         (ICMPv6) for the Internet Protocol Version 6
         (IPv6) Specification";
    }
    identity icmpv4 {
      base ipv4;
      description
        "ICMPv4 protocol type.";
      reference
        "RFC 791: Internet Protocol
         RFC 792: Internet Control Message Protocol";
    }
    identity icmpv6 {
      base ipv6;
      description
        "ICMPv6 protocol type.";
      reference
        "RFC 8200: Internet Protocol, Version 6 (IPv6)
         RFC 4443: Internet Control Message Protocol (ICMPv6)
         for the Internet Protocol Version 6 (IPv6)
         Specification";
    }
    identity ip {
      base protocol-type;
      description
        "General IP protocol type.";
      reference
        "RFC 791: Internet Protocol
         RFC 8200: Internet Protocol, Version 6 (IPv6)";
    }
    identity ipv4 {
      base ip;
      description
        "IPv4 protocol type.";
      reference
```

```
        "RFC 791: Internet Protocol";
    }
    identity ipv6 {
      base ip;
      description
        "IPv6 protocol type.";
      reference
        "RFC 8200: Internet Protocol, Version 6 (IPv6)";
    }
    identity http {
      base tcp;
      description
        "HTPP protocol type.";
      reference
        "RFC 2616: Hypertext Transfer Protocol";
    }
    identity ftp {
      base tcp;
      description
        "FTP protocol type.";
      reference
        "RFC 959: File Transfer Protocol";
    }

    /*
     * Grouping
     */

    grouping common-monitoring-data {
      description
        "A set of common monitoring data that is needed
        as the basic information.";
      leaf message {
        type string;
        description
          "This is a freetext annotation for
           monitoring a notification's content.";
      }
      leaf vendor-name {
        type string;
        description
          "The name of the NSF vendor";
      }
      leaf nsf-name {
        type string;
        description
          "The name (or IP) of the NSF generating the message.";
      }
```

```
    leaf severity {
      type severity;
      description
        "The severity of the alarm such as critical, high,
         middle, low.";
    }
  }
  grouping characteristics {
    description
      "A set of characteristics of a notification.";
    leaf acquisition-method {
      type identityref {
        base acquisition-method;
      }
      description
        "The acquisition-method for characteristics";
    }
    leaf emission-type {
      type identityref {
        base emission-type;
      }
     description
        "The emission-type for characteristics";
    }
    leaf dampening-type {
      type identityref {
        base dampening-type;
      }
      description
        "The dampening-type for characteristics";
    }
  }
  grouping i2nsf-system-alarm-type-content {
    description
      "A set of contents for alarm type notification.";
    leaf usage {
      type uint8 {
        range "0..100";
      }
      units "percent";
      description
        "Specifies the used percentage";
    }
    leaf threshold {
      type uint8 {
        range "0..100";
      }
      units "percent";
```

```
      description
        "The threshold percentage triggering the alarm or
         the event";
    }
  }
  grouping i2nsf-system-event-type-content {
    description
      "System event metadata associated with system events
       caused by user activity.";
    leaf user {
      type string;
      mandatory true;
      description
        "The name of a user";
    }
    leaf group {
      type string;
      mandatory true;
      description
        "The group to which a user belongs.";
    }
    leaf login-ip-addr {
      type inet:ip-address;
      mandatory true;
      description
        "The login IPv4 (or IPv6) address of a user.";
    }
    leaf authentication {
      type identityref {
        base authentication-mode;
      }
      description
        "The authentication-mode for authentication";
    }
  }
  grouping i2nsf-nsf-event-type-content {
    description
      "A set of common IPv4 (or IPv6)-related NSF event
       content elements";
    leaf dst-ip {
      type inet:ip-address;
      description
        "The destination IPv4 (IPv6) address of the packet";
    }
    leaf dst-port {
      type inet:port-number;
      description
        "The destination port of the packet";
```

```
    }
    leaf rule-name {
      type leafref {
        path
          "/nsfi:i2nsf-security-policy/nsfi:system-policy/nsfi:rules/nsfi:rule-
name";
      }
      mandatory true;
      description
        "The name of the rule being triggered";
    }
    leaf raw-info {
      type string;
      description
        "The information describing the packet
         triggering the event.";
    }
  }
  grouping i2nsf-nsf-event-type-content-extend {
    description
      "A set of extended common IPv4 (or IPv6)-related NSF
       event content elements";
    uses i2nsf-nsf-event-type-content;
    leaf src-ip {
      type inet:ip-address;
      description
        "The source IPv4 (or IPv6) address of the packet";
    }
    leaf src-port {
      type inet:port-number;
      description
        "The source port of the packet";
    }
    leaf src-zone {
      type string {
        length "1..100";
        pattern "[0-9a-zA-Z ]*";
      }
      description
        "The source security zone of the packet";
    }
    leaf dst-zone {
      type string {
        length "1..100";
        pattern "[0-9a-zA-Z ]*";
      }
      description
        "The destination security zone of the packet";
    }
```

```
  }
  grouping log-action {
    description
      "A grouping for logging action.";
    leaf action {
      type log-action;
      description
        "Action type: allow, alert, block, discard, declare,
         block-ip, block-service";
    }
  }
  grouping attack-rates {
    description
      "A set of traffic rates for monitoring attack traffic
       data";
    leaf attack-rate {
      type uint32;
      units "pps";
      description
        "The PPS rate of attack traffic";
    }
    leaf attack-speed {
      type uint32;
      units "bps";
      description
        "The BPS speed of attack traffic";
    }
  }
  grouping traffic-rates {
    description
      "A set of traffic rates for statistics data";
    leaf total-traffic {
      type yang:counter32;
      description
        "Total traffic";
    }
    leaf in-traffic-average-rate {
      type uint32;
      units "pps";
      description
        "Inbound traffic average rate in packets per second (pps)";
    }
    leaf in-traffic-peak-rate {
      type uint32;
      units "pps";
      description
        "Inbound traffic peak rate in packets per second (pps)";
    }
```

```
    leaf in-traffic-average-speed {
      type uint32;
      units "bps";
      description
        "Inbound traffic average speed in bits per second (bps)";
    }
    leaf in-traffic-peak-speed {
      type uint32;
      units "bps";
      description
        "Inbound traffic peak speed in bits per second (bps)";
    }
    leaf out-traffic-average-rate {
      type uint32;
      units "pps";
      description
        "Outbound traffic average rate in packets per second (pps)";
    }
    leaf out-traffic-peak-rate {
      type uint32;
      units "pps";
      description
       "Outbound traffic peak rate in packets per Second (pps)";
    }
    leaf out-traffic-average-speed {
      type uint32;
      units "bps";
      description
        "Outbound traffic average speed in bits per second (bps)";
    }
    leaf out-traffic-peak-speed {
      type uint32;
      units "bps";
      description
        "Outbound traffic peak speed in bits per second (bps)";
    }
  }
  grouping i2nsf-system-counter-type-content{
    description
      "A set of counters for an interface traffic data.";
    leaf interface-name {
      type string;
      description
        "Network interface name configured in an NSF";
    }
    leaf in-total-traffic-pkts {
      type yang:counter32;
      description
```

```
          "Total inbound packets";
      }
      leaf out-total-traffic-pkts {
        type yang:counter32;
        description
          "Total outbound packets";
      }
      leaf in-total-traffic-bytes {
        type uint64;
        units "bytes";
        description
          "Total inbound bytes";
      }
      leaf out-total-traffic-bytes {
        type uint64;
        units "bytes";
        description
          "Total outbound bytes";
      }
      leaf in-drop-traffic-pkts {
        type yang:counter32;
        description
          "Total inbound drop packets";
      }
      leaf out-drop-traffic-pkts {
        type yang:counter32;
        description
          "Total outbound drop packets";
      }
      leaf in-drop-traffic-bytes {
        type uint64;
        units "bytes";
        description
          "Total inbound drop bytes";
      }
      leaf out-drop-traffic-bytes {
        type uint64;
        units "bytes";
        description
          "Total outbound drop bytes";
      }
      uses traffic-rates;
    }
    grouping i2nsf-nsf-counters-type-content{
      description
        "A set of contents of a policy in an NSF.";
      leaf policy-name {
        type leafref {
```

```
        path
          "/nsfi:i2nsf-security-policy/nsfi:system-policy/nsfi:system-policy-
name";
      }
      mandatory true;
      description
        "The name of the policy being triggered";
    }
    leaf src-user{
      type string;
      description
        "User who generates the policy";
    }
  }

  grouping enable-notification {
    description
      "A grouping for enabling or disabling notification";
    leaf enabled {
      type boolean;
      default "true";
      description
        "Enables or Disables the notification.
         If 'true', then the notification is enabled.
         If 'false, then the notification is disabled.";
    }
  }

  grouping dampening {
    description
      "A grouping for dampening period of notification.";
    leaf dampening-period {
      type uint32;
      units "centiseconds";
      default "0";
      description
        "Specifies the minimum interval between the assembly of
         successive update records for a single receiver of a
         subscription.  Whenever subscribed objects change and
         a dampening-period interval (which may be zero) has
         elapsed since the previous update record creation for
         a receiver, any subscribed objects and properties
         that have changed since the previous update record
         will have their current values marshalled and placed
         in a new update record.";
      reference
        "RFC 8641:  Subscription to YANG Notifications for
         Datastore Updates - Section 5.";
    }
```

```
  }

  /*
   * Feature Nodes
   */

  feature i2nsf-nsf-detection-ddos {
    description
      "This feature means it supports I2NSF nsf-detection-ddos
       notification";
  }
  feature i2nsf-nsf-detection-virus {
    description
      "This feature means it supports I2NSF nsf-detection-virus
       notification";
  }
  feature i2nsf-nsf-detection-intrusion {
    description
      "This feature means it supports I2NSF nsf-detection-intrusion
       notification";
  }
  feature i2nsf-nsf-detection-botnet {
    description
      "This feature means it supports I2NSF nsf-detection-botnet
       notification";
  }
  feature i2nsf-nsf-detection-web-attack {
    description
      "This feature means it supports I2NSF nsf-detection-web-attack
       notification";
  }
  feature i2nsf-nsf-log-dpi {
    description
      "This feature means it supports I2NSF nsf-log-dpi
       notification";
  }
  feature i2nsf-nsf-log-vuln-scan {
    description
      "This feature means it supports I2NSF nsf-log-vuln-scan
       notification";
  }



  /*
   * Notification nodes
   */
```

```
  notification i2nsf-event {
    description
      "Notification for I2NSF Event.";
    choice sub-event-type {
      description
        "This choice must be augmented with cases for each allowed
         sub-event. Only 1 sub-event will be instantiated in each
         i2nsf-event message. Each case is expected to define one
         container with all the sub-event fields.";
      case i2nsf-system-detection-alarm {
        container i2nsf-system-detection-alarm{
          description
            "This notification is sent, when a system alarm
             is detected.";
          leaf alarm-category {
            type identityref {
             base alarm-type;
            }
            description
              "The alarm category for
               system-detection-alarm notification";
          }
          leaf component-name {
            type string;
            description
              "The hardware component responsible for generating
               the message. Applicable for Hardware Failure
               Alarm.";
          }
          leaf interface-name {
            type string;
            description
              "The interface name responsible for generating
               the message. Applicable for Network Interface
               Failure Alarm.";
          }
          leaf interface-state {
            type enumeration {
              enum down {
                description
                  "The interface state is down.";
              }
              enum up {
                description
                  "The interface state is up.";
              }
              enum congested {
                description
```

```
                  "The interface state is congested.";
              }
            }
            description
              "The state of the interface (i.e., up, down, congested).
              Applicable for Network Interface Failure Alarm.";
          }
          uses characteristics;
          uses i2nsf-system-alarm-type-content;
          uses common-monitoring-data;
        }
      }

      case i2nsf-system-detection-event {
        container i2nsf-system-detection-event {
          description
            "This notification is sent when a security-sensitive
             authentication action fails.";
          leaf event-category {
           type identityref {
              base event-type;
            }
            description
              "The event category for system-detection-event";
          }
          uses characteristics;
          uses i2nsf-system-event-type-content;
          uses common-monitoring-data;
        }
      }

      case i2nsf-traffic-flows {
        container i2nsf-traffic-flows {
          description
            "This notification is sent to inform about the traffic
             flows.";
          leaf src-ip {
            type inet:ip-address;
            description
              "The source IPv4 (or IPv6) address of the packet";
          }
          leaf dst-ip {
            type inet:ip-address;
            description
              "The destination IPv4 (or IPv6) address of the packet";
          }
          leaf protocol {
            type identityref {
```

```
              base protocol-type;
            }
            description
              "The protocol type for nsf-detection-intrusion
               notification";
          }
          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 arrival-rate {
            type uint32;
            units "pps";
            description
              "The arrival rate of the packet in packets
              per second";
          }
          uses characteristics;
          uses common-monitoring-data;
        }
      }

      case i2nsf-nsf-detection-session-table {
        container i2nsf-nsf-detection-session-table {
          description
            "This notification is sent, when a session table
             event is detected.";
          leaf current-session {
            type uint32;
            description
              "The number of concurrent sessions";
          }
          leaf maximum-session {
            type uint32;
            description
              "The maximum number of sessions that the session
               table can support";
          }
          leaf threshold {
            type uint32;
            description
              "The threshold triggering the event";
```

```
          }
          uses common-monitoring-data;
        }
      }
    }
  }

  notification i2nsf-log {
    description
      "Notification for I2NSF log. The notification is generated
       from the logs of the NSF.";
    choice sub-logs-type {
      description
        "This choice must be augmented with cases for each allowed
         sub-logs. Only 1 sub-event will be instantiated in each
         i2nsf-logs message. Each case is expected to define one
         container with all the sub-logs fields.";
      case i2nsf-nsf-system-access-log {
        container i2nsf-nsf-system-access-log {
          description
            "The notification is sent, if there is a new system
             log entry about a system access event.";
          leaf login-ip {
            type inet:ip-address;
            mandatory true;
            description
              "Login IP address of a user";
          }
          leaf administrator {
            type string;
            description
              "Administrator that maintains the device";
          }
          leaf login-mode {
            type login-mode;
            description
              "Specifies the administrator log-in mode";
          }
          leaf operation-type {
            type operation-type;
            description
              "The operation type that the administrator executes";
          }
          leaf result {
            type string;
            description
              "Command execution result";
          }
```

```
          leaf content {
            type string;
            description
              "The Operation performed by an administrator after
               login";
          }
          uses characteristics;
          uses common-monitoring-data;
        }
      }

      case i2nsf-system-res-util-log {
        container i2nsf-system-res-util-log {
          description
            "This notification is sent, if there is a new log
             entry representing resource utilization updates.";
          leaf system-status {
             type string;
             description
              "The current systems running status";
          }
          leaf cpu-usage {
            type uint8;
            description
              "Specifies the relative size of CPU usage with
               respect to platform resources";
          }
          leaf memory-usage {
            type uint8;
            description
              "Specifies the size of memory usage.";
          }
          leaf disk-usage {
            type uint8;
            description
              "Specifies the size of disk usage";
          }
          leaf disk-left {
            type uint8;
            description
              "Specifies the size 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;
            units "pps";
            description
              "The total inbound traffic rate in pps";
          }
          leaf out-traffic-rate {
            type uint32;
            units "pps";
            description
               "The total outbound traffic rate in pps";
          }
          leaf in-traffic-speed {
            type uint32;
            units "bps";
            description
              "The total inbound traffic speed in bps";
          }
          leaf out-traffic-speed {
            type uint32;
            units "bps";
            description
              "The total outbound traffic speed in bps";
          }
          uses characteristics;
          uses common-monitoring-data;
        }
      }

      case i2nsf-system-user-activity-log {
        container i2nsf-system-user-activity-log {
          description
            "This notification is sent, if there is a new user
             activity log entry.";
          uses characteristics;
          uses i2nsf-system-event-type-content;
          uses common-monitoring-data;
          leaf access {
            type identityref {
              base access-mode;
            }
            description
              "The access type for system-user-activity-log
               notification";
```

```
          }
          leaf online-duration {
            type string;
            description
              "Online duration";
          }
          leaf logout-duration {
            type string;
            description
              "Lockout duration";
          }
          leaf additional-info {
            type string;
            description
              "User activities, e.g., Successful User Login,
               Failed Login attempts, User Logout, Successful User
               Password Change, Failed User Password Change, User
               Lockout, User Unlocking, and Unknown.";
          }
        }
      }
    }
  }

  notification i2nsf-nsf-event {
    description
      "Notification for I2NSF NSF Event. This notification is
       used for a specific NSF that supported such feature.";
    choice sub-event-type {
      description
      "This choice must be augmented with cases for each allowed
       sub-event. Only 1 sub-event will be instantiated in each
       i2nsf-event message. Each case is expected to define one
       container with all the sub-event fields.";
      case i2nsf-nsf-detection-ddos {
        if-feature "i2nsf-nsf-detection-ddos";
        container i2nsf-nsf-detection-ddos {
          description
            "This notification is sent, when a specific flood type
             is detected.";
          uses i2nsf-nsf-event-type-content;
          leaf attack-type {
            type identityref {
              base flood-type;
            }
            description
              "Any one of Syn flood, ACK flood, SYN-ACK flood,
               FIN/RST flood, TCP Connection flood, UDP flood,
```

```
               ICMP (i.e., ICMPv4 or ICMPv6) flood, HTTP flood,
               HTTPS flood, DNS query flood, DNS reply flood, SIP
               flood, etc.";
          }
          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-src-ip {
            type inet:ip-address;
            description
              "The source IPv4 (or IPv6) addresses of attack
               traffic. If there are a large number of IPv4
               (or IPv6) addresses, then pick a certain number
               of resources according to different rules.";
          }
          leaf attack-dst-ip {
            type inet:ip-address;
            description
              "The destination IPv4 (or IPv6) addresses of attack
               traffic. If there are a large number of IPv4
               (or IPv6) addresses, then pick a certain number
               of resources according to different rules.";
          }
          uses attack-rates;
          uses log-action;
          uses characteristics;
          uses common-monitoring-data;
        }
      }
      case i2nsf-nsf-detection-virus {
        if-feature "i2nsf-nsf-detection-virus";
        container i2nsf-nsf-detection-virus {
          description
            "This notification is sent, when a virus is detected.";
          uses i2nsf-nsf-event-type-content-extend;
          leaf virus {
            type identityref {
              base virus-type;
            }
```

```
             description
                "The virus type for nsf-detection-virus notification";
            }
            leaf virus-name {
              type string;
              description
                "The name of the detected virus";
            }
            leaf file-type {
              type string;
              description
                "The type of file virus code is found in (if
                 applicable).";
            }
            leaf file-name {
              type string;
              description
                "The name of file virus code is found in (if
                 applicable).";
            }
            leaf os {
              type string;
              description
                "Simple OS information";
            }
            uses log-action;
            uses characteristics;
            uses common-monitoring-data;
          }
        }
        case i2nsf-nsf-detection-intrusion {
          if-feature "i2nsf-nsf-detection-intrusion";
          container i2nsf-nsf-detection-intrusion {
            description
              "This notification is sent, when an intrusion event
               is detected.";
            uses i2nsf-nsf-event-type-content-extend;
            leaf protocol {
              type identityref {
                base protocol-type;
              }
              description
                "The protocol type for nsf-detection-intrusion
                 notification";
            }
            leaf app {
              type string;
              description
```

```
              "The employed application layer protocol";
          }
          leaf attack-type {
            type identityref {
              base intrusion-attack-type;
            }
            description
              "The sub attack type for intrusion attack";
          }
          uses log-action;
          uses attack-rates;
          uses characteristics;
          uses common-monitoring-data;
        }
      }
      case i2nsf-nsf-detection-botnet {
        if-feature "i2nsf-nsf-detection-botnet";
        container i2nsf-nsf-detection-botnet {
          description
            "This notification is sent, when a botnet event is
             detected.";
          uses i2nsf-nsf-event-type-content-extend;
          leaf attack-type {
            type identityref {
              base botnet-attack-type;
            }
           description
              "The attack type for botnet attack";
          }
          leaf protocol {
            type identityref {
              base protocol-type;
            }
            description
              "The protocol type for nsf-detection-botnet notification";
          }
          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";
          }
          uses log-action;
```

```
          leaf botnet-pkt-num{
            type uint8;
            description
              "The number of the packets sent to or from the detected botnet";
          }
          leaf os{
            type string;
            description
              "Simple OS information";
          }
          uses characteristics;
          uses common-monitoring-data;
        }
      }
      case i2nsf-nsf-detection-web-attack {
        if-feature "i2nsf-nsf-detection-web-attack";
        container i2nsf-nsf-detection-web-attack {
          description
            "This notification is sent, when an attack event is
             detected.";
          uses i2nsf-nsf-event-type-content-extend;
          leaf attack-type {
            type identityref {
              base web-attack-type;
            }
            description
              "Concrete web attack type, e.g., SQL injection,
               command injection, XSS, and CSRF.";
          }
          leaf request-method {
            type identityref {
              base req-method;
            }
            description
              "The method of requirement. For instance, PUT or
               GET in HTTP.";
          }
          leaf req-uri {
            type string;
            description
              "Requested URI";
          }
          leaf uri-category {
            type string;
            description
              "Matched URI category";
          }
          leaf-list filtering-type {
```

```
            type identityref {
              base filter-type;
            }
            description
              "URL filtering type, e.g., Blacklist, Whitelist,
               User-Defined, Predefined, Malicious Category,
               and Unknown";
          }
          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";
          }
          uses characteristics;
          uses log-action;
          uses common-monitoring-data;
        }
      }
      case i2nsf-nsf-log-vuln-scan {
        if-feature "i2nsf-nsf-log-vuln-scan";
        container i2nsf-nsf-log-vuln-scan {
          description
            "This notification is sent, if there is a new
             vulnerability-scan report in the NSF log.";
          leaf vulnerability-id {
            type uint8;
            description
              "The vulnerability ID";
          }
          leaf victim-ip {
            type inet:ip-address;
            description
              "IPv4 (or IPv6) address of the victim host which
```

```
                 has vulnerabilities";
            }
            leaf protocol {
              type identityref {
                base protocol-type;
              }
              description
                "The protocol type for nsf-log-vuln-scan
                 notification";
            }
            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 vulnerability-info {
              type string;
              description
                "The information about the vulnerability";
            }
            leaf fix-suggestion {
              type string;
              description
                "The fix suggestion to the vulnerability";
            }
            leaf service {
              type string;
              description
                "The service which has vulnerability in the victim
                 host";
            }
            uses characteristics;
            uses common-monitoring-data;
          }
        }
        case i2nsf-nsf-log-dpi {
          if-feature "i2nsf-nsf-log-dpi";
          container i2nsf-nsf-log-dpi {
```

```
          description
            "This notification is sent, if there is a new DPI
             event in the NSF log.";
          leaf attack-type {
            type dpi-type;
            description
              "The type of the DPI";
          }
          uses characteristics;
          uses i2nsf-nsf-counters-type-content;
          uses common-monitoring-data;
        }
      }
    }
  }
  /*
   * Data nodes
   */
  container i2nsf-counters {
    config false;
    description
      "This is probably better covered by an import as this
       will not be notifications.  Counters are not very
       suitable as telemetry, maybe via periodic
       subscriptions, which would still violate the principle
       of least surprise.";
    list system-interface {
      key interface-name;
      description
        "Interface counters provide the visibility of traffic into and
         out of an NSF, and bandwidth usage.";
      uses characteristics;
      uses i2nsf-system-counter-type-content;
      uses common-monitoring-data;
    }
    list nsf-firewall {
      key policy-name;
      description
        "Firewall counters provide the visibility of traffic signatures,
         bandwidth usage, and how the configured security and bandwidth
         policies have been applied.";
      uses characteristics;
      uses i2nsf-nsf-counters-type-content;
      uses traffic-rates;
      uses common-monitoring-data;
    }
    list nsf-policy-hits {
      key policy-name;
```

```
      description
        "Policy Hit Counters record the number of hits that traffic
         packets match a security policy. It can check if policy
         configurations are correct or not.";
      uses characteristics;
      uses i2nsf-nsf-counters-type-content;
      uses common-monitoring-data;
      leaf hit-times {
        type yang:counter32;
        description
          "The number of times a policy is hit";
      }
    }
  }

  container i2nsf-monitoring-configuration {
    description
      "The container for configuring I2NSF monitoring.";
    container i2nsf-system-detection-alarm {
      description
        "The container for configuring I2NSF system-detection-alarm
         notification";
      uses enable-notification;
      list system-alarm {
        key alarm-type;
        description
          "Configuration for system alarm (i.e., CPU, Memory,
           and Disk Usage)";
        leaf alarm-type {
          type enumeration {
            enum CPU {
              description
                "To configure the CPU usage threshold to trigger the
                 CPU-USAGE-ALARM";
            }
            enum Memory {
              description
                "To configure the Memory usage threshold to trigger the
                 MEM-USAGE-ALARM";
            }
            enum Disk {
              description
                "To configure the Disk (storage) usage threshold to
                 trigger the DISK-USAGE-ALARM";
            }
          }
          description
            "Type of alarm to be configured";
```

```
        }
        leaf threshold {
          type uint8 {
            range "1..100";
          }
          units "percent";
          description
            "The configuration for threshold percentage to trigger
             the alarm. The alarm will be triggered if the usage
             is exceeded the threshold.";
        }
        uses dampening;
      }
    }
    container i2nsf-system-detection-event {
      description
        "The container for configuring I2NSF system-detection-event
         notification";
      uses enable-notification;
      uses dampening;
    }
    container i2nsf-traffic-flows {
      description
        "The container for configuring I2NSF traffic-flows
         notification";
      uses dampening;
      uses enable-notification;
    }
    container i2nsf-nsf-detection-ddos {
      if-feature "i2nsf-nsf-detection-ddos";
      description
        "The container for configuring I2NSF nsf-detection-ddos
         notification";
      uses enable-notification;
      uses dampening;
    }
    container i2nsf-nsf-detection-session-table-configuration {
      description
        "The container for configuring I2NSF nsf-detection-session-table
         notification";
      uses enable-notification;
      uses dampening;
    }
    container i2nsf-nsf-detection-virus {
      if-feature "i2nsf-nsf-detection-virus";
      description
        "The container for configuring I2NSF nsf-detection-virus
         notification";
```

```
        uses enable-notification;
        uses dampening;
      }
      container i2nsf-nsf-detection-intrusion {
        if-feature "i2nsf-nsf-detection-intrusion";
        description
          "The container for configuring I2NSF nsf-detection-intrusion
           notification";
        uses enable-notification;
        uses dampening;
      }
      container i2nsf-nsf-detection-botnet {
        if-feature "i2nsf-nsf-detection-botnet";
        description
          "The container for configuring I2NSF nsf-detection-botnet
           notification";
        uses enable-notification;
        uses dampening;
      }
      container i2nsf-nsf-detection-web-attack {
        if-feature "i2nsf-nsf-detection-web-attack";
        description
          "The container for configuring I2NSF nsf-detection-web-attack
           notification";
        uses enable-notification;
        uses dampening;
      }
      container i2nsf-nsf-system-access-log {
        description
          "The container for configuring I2NSF system-access-log
           notification";
        uses enable-notification;
        uses dampening;
      }
      container i2nsf-system-res-util-log {
        description
          "The container for configuring I2NSF system-res-util-log
           notification";
        uses enable-notification;
        uses dampening;
      }
      container i2nsf-system-user-activity-log {
        description
          "The container for configuring I2NSF system-user-activity-log
           notification";
        uses enable-notification;
        uses dampening;
      }
```

```
    container i2nsf-nsf-log-dpi {
      if-feature "i2nsf-nsf-log-dpi";
      description
        "The container for configuring I2NSF nsf-log-dpi
         notification";
      uses enable-notification;
      uses dampening;
    }
    container i2nsf-nsf-log-vuln-scan {
      if-feature "i2nsf-nsf-log-vuln-scan";
      description
        "The container for configuring I2NSF nsf-log-vuln-scan
         notification";
      uses enable-notification;
      uses dampening;
    }
    container i2nsf-counter {
      description
        "This is used to configure the counters
         for monitoring an NSF";
      leaf period {
        type uint16;
        units "minutes";
        default 0;
        description
          "The configuration for the period interval of reporting
           the counter. If 0, then the counter period is disabled.
           If value is not 0, then the counter will be reported
           following the period value.";
      }
    }
  }
}
<CODE ENDS>
```

Figure 2: Data Model of Monitoring

11. I2NSF Event Stream

This section discusses the NETCONF event stream for I2NSF NSF Monitoring subscription. The YANG module in this document supports "ietf-subscribed-notifications" YANG module [RFC8639] for subscription. The reserved event stream name for this document is "I2NSF-Monitoring". The NETCONF Server (e.g., an NSF) MUST support "I2NSF-Monitoring" event stream for an NSF data collector (e.g., Security Controller and NSF data analyzer). The "I2NSF-Monitoring" event stream contains all I2NSF events described in this document. The following example shows the capabilities of the event streams of

an NSF (e.g., "NETCONF" and "I2NSF-Monitoring" event streams) by the subscription of an NSF data collector; note that this example XML file is delivered by an NSF to an NSF data collector:

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <data>
    <netconf xmlns="urn:ietf:params:xml:ns:netmod:notification">
      <streams>
        <stream>
          <name>NETCONF</name>
          <description>Default NETCONF Event Stream</description>
          <replaySupport>false</replaySupport>
        </stream>
        <stream>
          <name>I2NSF-Monitoring</name>
          <description>I2NSF Monitoring Event Stream</description>
          <replaySupport>true</replaySupport>
          <replayLogCreationTime>2021-03-31T09:37:39+00:00</
replayLogCreationTime>
        </stream>
      </streams>
    </netconf>
  </data>
</rpc-reply>
```

Figure 3: Example of NETCONF Server supporting I2NSF-Monitoring Event Stream

12. XML Examples for I2NSF NSF Monitoring

This section shows the XML examples of I2NSF NSF Monitoring data delivered via Monitoring Interface from an NSF.

12.1. I2NSF System Detection Alarm

The following example shows an alarm triggered by Memory Usage of the server; note that this example XML file is delivered by an NSF to an NSF data collector:

```
<?xml version="1.0" encoding="UTF-8"?>
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2021-03-31T07:43:52.181088+00:00</eventTime>
  <i2nsf-event xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-nsf-monitoring">
    <i2nsf-system-detection-alarm>
      <alarm-category xmlns:nsfmi="urn:ietf:params:xml:ns:yang:ietf-i2nsf-nsf-
monitoring">
        nsfmi:mem-usage-alarm
      </alarm-category>
      <acquisition-method xmlns:nsfmi="urn:ietf:params:xml:ns:yang:ietf-i2nsf-
nsf-monitoring">
        nsfmi:subscription
      </acquisition-method>
      <emission-type xmlns:nsfmi="urn:ietf:params:xml:ns:yang:ietf-i2nsf-nsf-
monitoring">
        nsfmi:on-change
      </emission-type>
      <dampening-type xmlns:nsfmi="urn:ietf:params:xml:ns:yang:ietf-i2nsf-nsf-
monitoring">
        nsfmi:on-repetition
      </dampening-type>
      <usage>91</usage>
      <threshold>90</threshold>
      <message>Memory Usage Exceeded The Threshold</message>
      <nsf-name>time_based_firewall</nsf-name>
      <severity>high</severity>
    </i2nsf-system-detection-alarm>
  </i2nsf-event>
</notification>
```

Figure 4: Example of I2NSF System Detection Alarm triggered by Memory Usage

The XML data above shows:

1. The NSF that sends the information is named "time_based_firewall".

2. The memory usage of the NSF triggered the alarm.

3. The monitoring information is received by subscription method.

4. The monitoring information is emitted "on-change".

5. The monitoring information is dampened "on-repetition".

6. The memory usage of the NSF is 91 percent.

7. The memory threshold to trigger the alarm is 90 percent.

8. The severity level of the notification is high.

12.2. I2NSF Interface Counters

To get the I2NSF system interface counters information by query, NETCONF Client (e.g., NSF data collector) needs to initiate GET connection with NETCONF Server (e.g., NSF). The following XML file can be used to get the state data and filter the information.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <get>
    <filter xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-nsf-monitoring">
      <i2nsf-counters>
        <system-interface/>
      </i2nsf-counters>
    </filter>
  </get>
</rpc>
```

Figure 5: XML Example for NETCONF GET with System Interface Filter

The following XML file shows the reply from the NETCONF Server (e.g., NSF):

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <data>
    <i2nsf-counters xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-nsf-
monitoring">
      <system-interface>
        <interface-name>ens3</interface-name>
        <acquisition-method xmlns:nsfmi="urn:ietf:params:xml:ns:yang:ietf-
i2nsf-nsf-monitoring">
          nsfmi:query
        </acquisition-method>
        <in-total-traffic-bytes>549050</in-total-traffic-bytes>
        <out-total-traffic-bytes>814956</out-total-traffic-bytes>
        <in-drop-traffic-bytes>0</in-drop-traffic-bytes>
        <out-drop-traffic-bytes>5078</out-drop-traffic-bytes>
        <nsf-name>time_based_firewall</nsf-name>
      </system-interface>
      <system-interface>
        <interface-name>lo</interface-name>
        <acquisition-method xmlns:nsfmi="urn:ietf:params:xml:ns:yang:ietf-
i2nsf-nsf-monitoring">
          nsfmi:query
        </acquisition-method>
        <in-total-traffic-bytes>48487</in-total-traffic-bytes>
        <out-total-traffic-bytes>48487</out-total-traffic-bytes>
        <in-drop-traffic-bytes>0</in-drop-traffic-bytes>
        <out-drop-traffic-bytes>0</out-drop-traffic-bytes>
        <nsf-name>time_based_firewall</nsf-name>
      </system-interface>
    </i2nsf-counters>
  </data>
</rpc-reply>
```

Figure 6: Example of I2NSF System Interface Counters XML Information

13. IANA Considerations

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

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

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

name: ietf-i2nsf-nsf-monitoring
namespace: urn:ietf:params:xml:ns:yang:ietf-i2nsf-nsf-monitoring
prefix: nsfmi
reference: RFC XXXX

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

14. Security Considerations

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

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

All data nodes defined in the YANG module which can be created, modified and deleted (i.e., config true, which is the default) are considered sensitive. Write operations (e.g., edit-config) applied to these data nodes without proper protection can negatively affect framework operations. The monitoring YANG module should be protected by the secure communication channel, to ensure its confidentiality and integrity. In another side, the NSF and NSF data collector can all be faked, which lead to undesirable results (i.e., leakage of an NSF's important operational information, and faked NSF sending false information to mislead the NSF data collector). The mutual authentication is essential to protected against this kind of attack. The current mainstream security technologies (i.e., TLS, DTLS, IPsec, and X.509 PKI) can be employed appropriately to provide the above security functions.

In addition, to defend against the DDoS attack caused by a lot of NSFs sending massive notifications to the NSF data collector, the rate limiting or similar mechanisms should be considered in both an NSF and NSF data collector, whether in advance or just in the process of DDoS attack.

15. Acknowledgments

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea MSIT (Ministry of Science and ICT) (R-20160222-002755, Cloud based Security Intelligence Technology Development for the Customized Security Service Provisioning). This work was supported in part by the IITP (2020-0-00395, Standard Development of Blockchain based Network Management Automation Technology). This work was supported in part by the MSIT under the Information Technology Research Center (ITRC) support program (IITP-2020-2017-0-01633) supervised by the IITP.

16. Contributors

This document is made by the group effort of I2NSF working group. Many people actively contributed to this document. The authors sincerely appreciate their contributions.

The following are co-authors of this document:

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

EMail: darkhong@skku.edu

Jinyong (Tim) Kim
Department of Electronic, Electrical and Computer Engineering
Sungkyunkwan University
2066 Seo-ro Jangan-gu
Suwon, Gyeonggi-do 16419
Republic of Korea

EMail: timkim@skku.edu

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

EMail: dong.jin@skku.edu

Dacheng Zhang
Huawei

EMail: dacheng.zhang@huawei.com

Yi Wu
Aliababa Group

EMail: anren.wy@alibaba-inc.com

Rakesh Kumar
Juniper Networks
1133 Innovation Way
Sunnyvale, CA 94089
USA

EMail: rkkumar@juniper.net

Anil Lohiya
Juniper Networks

EMail: alohiya@juniper.net

17. References

17.1. Normative References

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

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

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

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

[RFC0956] Mills, D., "Algorithms for synchronizing network clocks", RFC 956, DOI 10.17487/RFC0956, September 1985, <https://www.rfc-editor.org/info/rfc956>.

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

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, DOI 10.17487/RFC2616, June 1999, <https://www.rfc-editor.org/info/rfc2616>.

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

[RFC3877] Chisholm, S. and D. Romascanu, "Alarm Management Information Base (MIB)", RFC 3877, DOI 10.17487/RFC3877, September 2004, <https://www.rfc-editor.org/info/rfc3877>.

[RFC3954] Claise, B., Ed., "Cisco Systems NetFlow Services Export Version 9", RFC 3954, DOI 10.17487/RFC3954, October 2004, <https://www.rfc-editor.org/info/rfc3954>.

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

[RFC4949] Shirey, R., "Internet Security Glossary, Version 2", FYI 36, RFC 4949, DOI 10.17487/RFC4949, August 2007, <https://www.rfc-editor.org/info/rfc4949>.

[RFC5424] Gerhards, R., "The Syslog Protocol", RFC 5424, DOI 10.17487/RFC5424, March 2009, <https://www.rfc-editor.org/info/rfc5424>.

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

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

[RFC6587] Gerhards, R. and C. Lonvick, "Transmission of Syslog Messages over TCP", RFC 6587, DOI 10.17487/RFC6587, April 2012, <https://www.rfc-editor.org/info/rfc6587>.

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

[RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <https://www.rfc-editor.org/info/rfc7011>.

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

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

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

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

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

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

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

[RFC8407] Bierman, A., "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", BCP 216, RFC 8407, DOI 10.17487/RFC8407, October 2018, <https://www.rfc-editor.org/info/rfc8407>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8525] Bierman, A., Bjorklund, M., Schoenwaelder, J., Watsen, K., and R. Wilton, "YANG Library", RFC 8525, DOI 10.17487/RFC8525, March 2019, <https://www.rfc-editor.org/info/rfc8525>.

[RFC8639] Voit, E., Clemm, A., Gonzalez Prieto, A., Nilsen-Nygaard, E., and A. Tripathy, "Subscription to YANG Notifications", RFC 8639, DOI 10.17487/RFC8639, September 2019, <https://www.rfc-editor.org/info/rfc8639>.

[RFC8641] Clemm, A. and E. Voit, "Subscription to YANG Notifications for Datastore Updates", RFC 8641, DOI 10.17487/RFC8641, September 2019, <https://www.rfc-editor.org/info/rfc8641>.

17.2. Informative References

[I-D.ietf-i2nsf-applicability] Jeong, J., Hyun, S., Ahn, T., Hares, S., and D. Lopez, "Applicability of Interfaces to Network Security Functions to Network-Based Security Services", draft-ietf-i2nsf-applicability-18 (work in progress), September 2019.

[I-D.ietf-i2nsf-capability] Xia, L., Strassner, J., Basile, C., and D. Lopez, "Information Model of NSFs Capabilities", draft-ietf-i2nsf-capability-05 (work in progress), April 2019.

[I-D.ietf-i2nsf-consumer-facing-interface-dm]
Jeong, J., Chung, C., Ahn, T., Kumar, R., and S. Hares, "I2NSF Consumer-Facing Interface YANG Data Model", draft-ietf-i2nsf-consumer-facing-interface-dm-12 (work in progress), September 2020.

[I-D.ietf-i2nsf-nsf-facing-interface-dm]
Kim, J., Jeong, J., J., J., PARK, P., Hares, S., and Q. Lin, "I2NSF Network Security Function-Facing Interface YANG Data Model", draft-ietf-i2nsf-nsf-facing-interface-dm-10 (work in progress), August 2020.

[I-D.ietf-i2nsf-registration-interface-dm]
Hyun, S., Jeong, J., Roh, T., Wi, S., J., J., and P. PARK, "I2NSF Registration Interface YANG Data Model", draft-ietf-i2nsf-registration-interface-dm-09 (work in progress), August 2020.

[I-D.ietf-netconf-subscribed-notifications]
Voit, E., Clemm, A., Prieto, A., Nilsen-Nygaard, E., and A. Tripathy, "Subscription to YANG Event Notifications", draft-ietf-netconf-subscribed-notifications-26 (work in progress), May 2019.

[I-D.ietf-netconf-yang-push]
Clemm, A. and E. Voit, "Subscription to YANG Datastores", draft-ietf-netconf-yang-push-25 (work in progress), May 2019.

[I-D.yang-i2nsf-security-policy-translation]
Jeong, J., Yang, J., Chung, C., and J. Kim, "Security Policy Translation in Interface to Network Security Functions", draft-yang-i2nsf-security-policy-translation-07 (work in progress), November 2020.

Appendix A. Changes from draft-ietf-i2nsf-nsf-monitoring-data-model-06

The following changes are made from draft-ietf-i2nsf-nsf-monitoring-data-model-06:

- This version is revised according to the comments of Andy Bierman who is a YANG doctor.

- This version updates its title as "I2NSF NSF Monitoring Interface YANG Data Model". It clarifies the NSF Monitoring Interface to deliver NSF monitoring data to an NSF data collector (e.g., Security Controller and NSF data analyzer).

- This version adds an attack destination IP address for DDoS-attack event to provide I2NSF Analyser with more information about the destination of DDoS-attack packets.

- This version supports a notification for monitoring traffic flows.

Authors' Addresses

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

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

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

Phone: +82 31 299 4957
EMail: patricklink@skku.edu

Susan Hares
Huawei
7453 Hickory Hill
Saline, MI 48176
USA

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

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

EMail: Frank.xialiang@huawei.com

Henk Birkholz
Fraunhofer Institute for Secure Information Technology
Rheinstrasse 75
Darmstadt 64295
Germany

EMail: henk.birkholz@sit.fraunhofer.de