I2NSF Capability YANG Data Model
draft-ietf-i2nsf-capability-data-model-05

Abstract

This document defines a YANG data model for the capabilities of various Network Security Functions (NSFs) in the Interface to Network Security Functions (I2NSF) framework to centrally manage the capabilities of the various NSFs.

Status of This Memo

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

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

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

This Internet-Draft will expire on January 26, 2020.

Copyright Notice

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

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

As the industry becomes more sophisticated and network devices (e.g., Internet of Things, Self-driving vehicles, and VoIP/VoLTE smartphones), service providers have a lot of problems described in [RFC8192]. To resolve these problems, [draft-ietf-i2nsf-capability] specifies the information model of the capabilities of Network Security Functions (NSFs).
This document provides a YANG data model [RFC6020][RFC7950] that defines the capabilities of NSFs to centrally manage the capabilities of those security devices. The security devices can register their own capabilities into a Network Operator Management (Mgmt) System (i.e., Security Controller) with this YANG data model through the registration interface [RFC8329]. With the capabilities of those security devices maintained centrally, those security devices can be easily managed [RFC8329]. This YANG data model is based on the information model for I2NSF NSF capabilities [draft-ietf-i2nsf-capability].

This YANG data model uses an "Event-Condition-Action" (ECA) policy model that is used as the basis for the design of I2NSF Policy as described in [RFC8329] and [draft-ietf-i2nsf-capability]. The "ietf-i2nsf-capability" YANG module defined in this document provides the following features:

- Definition for general capabilities of network security functions.
- Definition for event capabilities of generic network security functions.
- Definition for condition capabilities of generic network security functions.
- Definition for condition capabilities of advanced network security functions.
- Definition for action capabilities of generic network security functions.
- Definition for resolution strategy capabilities of generic network security functions.
- Definition for default action capabilities of generic network security functions.

2. Requirements Language

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

3. Terminology

This document uses the terminology described in [draft-ietf-i2nsf-terminology][draft-ietf-i2nsf-capability][RFC8431][draft-ietf-supa-generic-policy-info-model]. Especially,
the following terms are from
[draft-ietf-sup-a-generic-policy-info-model]:

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

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

3.1.  Tree Diagrams

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

4.  Overview

This section provides an overview of how the YANG data model can be
used in the I2NSF framework described in [RFC8329]. Figure 1 shows
the capabilities of NSFs in I2NSF Framework. As shown in this
figure, an NSF Developer’s Mgmt System can register NSFs and the
capabilities that the network security device can support. To
register NSFs in this way, the Developer’s Mgmt System utilizes this
standardized capabilities YANG data model through its registration
interface. With the capabilities of those network security devices
maintained centrally, those security devices can be easily managed,
which can resolve many of the problems described in [RFC8192]. The
use cases are described below.

Note that the NSF-Facing Interface is used to configure the security
policy rules of the generic network security functions
[draft-ietf-i2nsf-nsf-facing-interface-dm], and the NSF Monitoring
Interface is used to configure the security policy rules of advanced
network security functions [draft-dong-i2nsf-asf-config],
respectively, according to the capabilities of NSFs registered with
the I2NSF Framework.
If a network manager wants to apply security policy rules to block malicious users, it is a tremendous burden to apply all of the needed rules to NSFs one-by-one. This problem can be resolved by managing the capabilities of NSFs. If network manager wants to block malicious users with IPv6, the network manager sends the security policy rules to block the users to the Network Operator Mgmt System using I2NSF user (i.e., a web browser or a software). When the Network Operator Mgmt System receives the security policy rules, it automatically sends that security policy rules to appropriate NSFs (i.e., NSF-m in Developer’s Mgmt System A and NSF-1 in Developer’s Mgmt System B) which can support the capabilities (i.e., IPv6). Therefore, an I2NSF User need not consider NSFs where to which NSFs the rules apply.

If NSFs encounter the malicious packets, it is a tremendous burden for the network manager to apply the rule to block the malicious packets to NSFs one-by-one. This problem can be resolved by
managing the capabilities of NSFs. If NSFs encounter the suspicious IPv4 packets, they can ask the Network Operator Mgmt System for information about the suspicious IPv4 packets in order to alter specific rules and/or configurations. When the Network Operator Mgmt System receives information, it inspects the information about the suspicious IPv4 packets. If the suspicious packets are determined to be malicious packets, the Network Operator Mgmt System creates and sends the security policy rules blocking malicious packets to appropriate NSFs (i.e., NSF-1 in Developer’s Mgmt System A and NSF-1 and NSF-n in Developer’s Mgmt System B) which can support the capabilities (i.e., IPv4). Therefore, the new security policy rules blocking malicious packets can be applied to appropriate NSFs without humans intervention.

5. YANG Tree Diagram

This section shows an YANG tree diagram of capabilities for network security functions, as defined in the [draft-ietf-i2nsf-capability].

5.1. Network Security Function (NSF) Capabilities

This section shows YANG tree diagram for NSF capabilities.
module: ietf-i2nsf-capability
   +--rw nsf* [nsf-name]
      +--rw nsf-name            string
      +--rw time-capabilities*                  enumeration
      +--rw event-capabilities
         +--rw system-event-capability*   identityref
         +--rw system-alarm-capability*   identityref
      +--rw condition-capabilities
         +--rw generic-nsf-capabilities
            +--rw ipv4-capability*   identityref
            +--rw ipv6-capability*   identityref
            +--rw tcp-capability*    identityref
            +--rw udp-capability*    identityref
            +--rw icmp-capability*   identityref
         +--rw advanced-nsf-capabilities
            +--rw anti-virus-capability*    identityref
            +--rw anti-ddos-capability*   identityref
            +--rw ips-capability*          identityref
            +--rw url-capability*         identityref
            +--rw voip-volte-capability*   identityref
         +--rw context-capabilities*        identityref
      +--rw action-capabilities
         +--rw ingress-action-capability*   identityref
         +--rw egress-action-capability*    identityref
         +--rw log-action-capability*       identityref
      +--rw resolution-strategy-capabilities*   identityref
      +--rw default-action-capabilities*        identityref
      +--rw ipsec-method*                       identityref

Figure 2: YANG Tree Diagram for Capabilities of Network Security Functions

This YANG tree diagram shows NSF capabilities.

The model includes NSF capabilities. The NSF capabilities include time capabilities, event capabilities, condition capabilities, action capabilities, resolution strategy capabilities, and default action capabilities.

Time capabilities are used to specify the capabilities to specify when to execute the I2NSF policy rule. The time capabilities are defined in terms of absolute time and periodic time. The absolute time means the exact time to start or end. The periodic time means repeated time like day, week, or month.

Event capabilities are used to specify capabilities how to trigger the evaluation of the condition clause of the I2NSF Policy Rule. The
defined event capabilities are defined as system event and system alarm. The event capability can be extended according to specific vendor condition features. The event capability is described in detail in [draft-ietf-i2nsf-capability].

Condition capabilities are used to specify capabilities of a set of attributes, features, and/or values that are to be compared with a set of known attributes, features, and/or values in order to determine whether or not the set of actions in that (imperative) I2NSF policy rule can be executed. The condition capabilities are classified in terms of generic network security functions and advanced network security functions. The condition capabilities of generic network security functions are defined as IPv4 capability, IPv6 capability, TCP capability, UDP capability, and ICMP capability. The condition capabilities of advanced network security functions are defined as anti-virus capability, anti-ddos capability, IPS capability, HTTP capability, and VoIP/VoLTE capability. The condition capability can be extended according to specific vendor condition features. The condition capability is described in detail in [draft-ietf-i2nsf-capability].

Action capabilities are used to specify capabilities of how to control and monitor aspects of flow-based NSFs when the event and condition clauses are satisfied. The action capabilities are defined as ingress-action capability, egress-action capability, and log-action capability. The action capability can be extended according to specific vendor action features. The action capability is described in detail in [draft-ietf-i2nsf-capability].

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

Default action capabilities are used to specify capabilities of how to execute I2NSF policy rules when no rule matches a packet. The default action capabilities are defined as pass, drop, reject, alert, and mirror. The default action capability can be extended according to specific vendor action features. The default action capability is described in detail in [draft-ietf-i2nsf-capability].
IPsec method capabilities are used to specify capabilities of how to support an Internet Key Exchange (IKE) for the security communication. The default action capabilities are defined as IKE and IKE-less. The default action capability can be extended according to specific vendor action features. The default action capability is described in detail in [draft-ietf-i2nsf-sdn-ipsec-flow-protection].

6. YANG Data Modules

6.1. I2NSF Capability YANG Data Module

This section introduces a YANG data module for network security functions capabilities, as defined in the [draft-ietf-i2nsf-capability].

<CODE BEGINS> file "ietf-i2nsf-capability@2019-07-24.yang"

module ietf-i2nsf-capability {
  yang-version 1.1;
  namespace
  prefix
    nsfcap;

  organization
    "IETF I2NSF (Interface to Network Security Functions)
    Working Group";

  contact
    "WG Web: <http://tools.ietf.org/wg/i2nsf>
    WG List: <mailto:i2nsf@ietf.org>

    WG Chair: Linda Dunbar
    <mailto:ldunbar@futurewei.com>

    WG Chair: Yoav Nir
    <mailto:ynir.ietf@gmail.com>

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

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

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

This module describes a capability model for I2NSF devices.

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

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

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

revision "2019-07-24"{
    description "Initial revision.";
    reference
        "RFC XXXX: I2NSF Capability YANG Data Model";
}

/*
 * Identities
 */

identity event {
    description
        "Base identity for I2NSF policy events.";
    reference
        "draft-ietf-i2nsf-nsf-monitoring-data-model-01
         - Event";
}

identity system-event-capability {
    base event;
    description
        "Identity for system events";
    reference
        "draft-ietf-i2nsf-nsf-monitoring-data-model-01
         - System alarm";
}

identity system-alarm-capability {
    base event;
    description
        "Identity for system alarms";
reference
"draft-ietf-i2nsf-nsf-monitoring-data-model-01
 - System alarm";
}

identity access-violation {
    base system-event-capability;
    description
    "Identity for access violation events";
    reference
    "draft-ietf-i2nsf-nsf-monitoring-data-model-01
    - System event";
}

identity configuration-change {
    base system-event-capability;
    description
    "Identity for configuration change events";
    reference
    "draft-ietf-i2nsf-nsf-monitoring-data-model-01
    - System event";
}

identity memory-alarm {
    base system-alarm-capability;
    description
    "Identity for memory alarm events";
    reference
    "draft-ietf-i2nsf-nsf-monitoring-data-model-01
    - System alarm";
}

identity cpu-alarm {
    base system-alarm-capability;
    description
    "Identity for CPU alarm events";
    reference
    "draft-ietf-i2nsf-nsf-monitoring-data-model-01
    - System alarm";
}

identity disk-alarm {
    base system-alarm-capability;
    description
    "Identity for disk alarm events";
    reference
    "draft-ietf-i2nsf-nsf-monitoring-data-model-01
    - System alarm";
identity hardware-alarm {
    base system-alarm-capability;
    description
        "Identity for hardware alarm events";
    reference
        "draft-ietf-i2nsf-nsf-monitoring-data-model-01
          - System alarm";
}

identity interface-alarm {
    base system-alarm-capability;
    description
        "Identity for interface alarm events";
    reference
        "draft-ietf-i2nsf-nsf-monitoring-data-model-01
          - System alarm";
}

identity condition {
    description
        "Base identity for policy conditions";
}

identity context-capability {
    base condition;
    description
        "Identity for context condition capabilities";
}

identity acl-number {
    base context-capability;
    description
        "Identity for ACL number condition capability";
}

identity application {
    base context-capability;
    description
        "Identity for application condition capability";
}

identity target {
    base context-capability;
    description
        "Identity for target condition capability";
}
identity user {
    base context-capability;
    description
        "Identity for user condition capability";
}

identity group {
    base context-capability;
    description
        "Identity for group condition capability";
}

identity geography {
    base context-capability;
    description
        "Identity for geography condition capability";
}

identity ipv4-capability {
    base condition;
    description
        "Identity for IPv4 condition capabilities";
    reference
        "RFC 791: Internet Protocol";
}

identity exact-ipv4-header-length {
    base ipv4-capability;
    description
        "Identity for exact-match IPv4 header-length condition capability";
    reference
        "RFC 791: Internet Protocol - Header Length";
}

identity range-ipv4-header-length {
    base ipv4-capability;
    description
        "Identity for range-match IPv4 header-length condition capability";
    reference
        "RFC 791: Internet Protocol - Header Length";
}

identity ipv4-tos {
    base ipv4-capability;
    description
        "Identity for IPv4 Type-Of-Service (TOS)
identity exact-ipv4-total-length {
    base ipv4-capability;
    description "Identity for exact-match IPv4 total length condition capability";
    reference "RFC 791: Internet Protocol - Total Length";
}

identity range-ipv4-total-length {
    base ipv4-capability;
    description "Identity for range-match IPv4 total length condition capability";
    reference "RFC 791: Internet Protocol - Total Length";
}

identity ipv4-id {
    base ipv4-capability;
    description "Identity for identification condition capability";
    reference "RFC 791: Internet Protocol - Identification";
}

identity ipv4-fragment-flags {
    base ipv4-capability;
    description "Identity for IPv4 fragment flags condition capability";
    reference "RFC 791: Internet Protocol - Fragmentation Flags";
}

identity exact-ipv4-fragment-offset {
    base ipv4-capability;
    description "Identity for exact-match IPv4 fragment offset condition capability";
    reference "RFC 791: Internet Protocol - Fragmentation Offset";
}
identity range-ipv4-fragment-offset {
  base ipv4-capability;
  description "Identity for range-match IPv4 fragment offset condition capability";
  reference "RFC 791: Internet Protocol - Fragmentation Offset";
}

identity exact-ipv4-ttl {
  base ipv4-capability;
  description "Identity for exact-match IPv4 Time-To-Live (TTL) condition capability";
  reference "RFC 791: Internet Protocol - Time To Live (TTL)";
}

identity range-ipv4-ttl {
  base ipv4-capability;
  description "Identity for range-match IPv4 Time-To-Live (TTL) condition capability";
  reference "RFC 791: Internet Protocol - Time To Live (TTL)";
}

identity ipv4-protocol {
  base ipv4-capability;
  description "Identity for IPv4 protocol condition capability";
}

identity exact-ipv4-address {
  base ipv4-capability;
  description "Identity for exact-match IPv4 address condition capability";
  reference "RFC 791: Internet Protocol - Address";
}

identity range-ipv4-address {
  base ipv4-capability;
description
"Identity for range-match IPv4 address condition capability";
reference
"RFC 791: Internet Protocol - Address";
}

identity ipv4-ip-opts {
  base ipv4-capability;
  description
"Identity for IPv4 option condition capability";
  reference
"RFC 791: Internet Protocol - Options";
}

identity ipv4-geo-ip {
  base ipv4-capability;
  description
"Identity for geography condition capability";
  reference
"draft-ietf-i2nsf-capability-04: Information Model of NSFs Capabilities - Geo-IP";
}

identity ipv6-capability {
  base condition;
  description
"Identity for IPv6 condition capabilities";
  reference
}

identity ipv6-traffic-class {
  base ipv6-capability;
  description
"Identity for IPv6 traffic class condition capability";
  reference
}

identity exact-ipv6-flow-label {
  base ipv6-capability;
  description
"Identity for exact-match IPv6 flow label condition capability";
}
identity range-ipv6-flow-label {
  base ipv6-capability;
  description
    "Identity for range-match IPv6 flow label condition capability";
  reference
}

identity exact-ipv6-payload-length {
  base ipv6-capability;
  description
    "Identity for exact-match IPv6 payload length condition capability";
  reference
    "RFC 8200: Internet Protocol, Version 6 (IPv6) Specification - Payload Length";
}

identity range-ipv6-payload-length {
  base ipv6-capability;
  description
    "Identity for range-match IPv6 payload length condition capability";
  reference
    "RFC 8200: Internet Protocol, Version 6 (IPv6) Specification - Payload Length";
}

identity ipv6-next-header {
  base ipv6-capability;
  description
    "Identity for IPv6 next header condition capability";
  reference
}

identity exact-ipv6-hop-limit {
  base ipv6-capability;
  description
    "Identity for exact-match IPv6 hop limit
condition capability";
reference
"RFC 8200: Internet Protocol, Version 6 (IPv6)
Specification - Hop Limit";
}

identity range-ipv6-hop-limit {
  base ipv6-capability;
  description
    "Identity for range-match IPv6 hop limit
collection capability";
  reference
    "RFC 8200: Internet Protocol, Version 6 (IPv6)
Specification - Hop Limit";
}

identity exact-ipv6-address {
  base ipv6-capability;
  description
    "Identity for exact-match IPv6 address
collection capability";
  reference
    "RFC 8200: Internet Protocol, Version 6 (IPv6)
Specification - Address";
}

identity range-ipv6-address {
  base ipv6-capability;
  description
    "Identity for range-match IPv6 address
collection capability";
  reference
    "RFC 8200: Internet Protocol, Version 6 (IPv6)
Specification - Address";
}

identity tcp-capability {
  base condition;
  description
    "Identity for TCP condition capabilities";
  reference
    "RFC 793: Transmission Control Protocol";
}

identity exact-tcp-port-num {
  base tcp-capability;
  description
    "Identity for exact-match TCP port number

identity range-tcp-port-num {
    base tcp-capability;
    description
        "Identity for range-match TCP port number
collection capability";
    reference
        "RFC 793: Transmission Control Protocol - Port Number";
}

identity exact-tcp-seq-num {
    base tcp-capability;
    description
        "Identity for exact-match TCP sequence number
collection capability";
    reference
        "RFC 793: Transmission Control Protocol - Sequence Number";
}

identity range-tcp-seq-num {
    base tcp-capability;
    description
        "Identity for range-match TCP sequence number
collection capability";
    reference
        "RFC 793: Transmission Control Protocol - Sequence Number";
}

identity exact-tcp-ack-num {
    base tcp-capability;
    description
        "Identity for exact-match TCP acknowledgement number
collection capability";
    reference
        "RFC 793: Transmission Control Protocol - Acknowledgement Number";
}

identity range-tcp-ack-num {
    base tcp-capability;
    description
        "Identity for range-match TCP acknowledgement number
collection capability";
    reference
        "RFC 793: Transmission Control Protocol - Acknowledgement Number";
identity exact-tcp-window-size {
    base tcp-capability;
    description
        "Identity for exact-match TCP window size
         condition capability";
    reference
        "RFC 793: Transmission Control Protocol - Window Size";
}

identity range-tcp-window-size {
    base tcp-capability;
    description
        "Identity for range-match TCP window size
         condition capability";
    reference
        "RFC 793: Transmission Control Protocol - Window Size";
}

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

identity udp-capability {
    base condition;
    description
        "Identity for UDP condition capabilities";
    reference
        "RFC 768: User Datagram Protocol";
}

identity exact-udp-port-num {
    base udp-capability;
    description
        "Identity for exact-match UDP port number
         condition capability";
    reference
        "RFC 768: User Datagram Protocol - Port Number";
}

identity range-udp-port-num {
    base udp-capability;
    description

"Identity for range-match UDP port number condition capability";
reference
"RFC 768: User Datagram Protocol - Port Number";
}

identity exact-udp-total-length {
  base udp-capability;
  description
    "Identity for exact-match UDP total-length condition capability";
  reference
    "RFC 768: User Datagram Protocol - Total Length";
}

identity range-udp-total-length {
  base udp-capability;
  description
    "Identity for range-match UDP total-length condition capability";
  reference
    "RFC 768: User Datagram Protocol - Total Length";
}

identity icmp-capability {
  base condition;
  description
    "Identity for ICMP condition capabilities";
  reference
    "RFC 792: Internet Control Message Protocol";
}

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

identity url-capability {
  base condition;
  description
    "Identity for URL condition capabilities";
}

identity pre-defined {
  base url-capability;
}
description
  "Identity for URL pre-defined condition capabilities";
}

identity user-defined {
  base url-capability;
  description
    "Identity for URL user-defined condition capabilities";
}

identity log-action-capability {
  description
    "Identity for log-action capabilities";
}

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

identity anti-virus-capability {
  base advanced-nsf-capability;
  description
    "Identity for advanced NSF anti-virus capabilities";
  reference
}

identity anti-ddos-capability {
  base advanced-nsf-capability;
  description
    "Identity for advanced NSF anti-ddos capabilities";
  reference
}

identity ips-capability {
  base advanced-nsf-capability;
  description
    "Identity for advanced NSF Intrusion Prevention System (IPS) capabilities";
  reference
}
identity voip-volte-capability {
  base advanced-nsf-capability;
  description
    "Identity for advanced NSF VoIP/VoLTE capabilities";
  reference
    "RFC 3261: SIP: Session Initiation Protocol
    RFC 8329: Framework for Interface to Network Security
    Functions - Differences from ACL Data Models
draft-dong-i2nsf-asf-config-01: Configuration of
    Advanced Security Functions with I2NSF Security
    Controller";
}

identity detect {
  base anti-virus-capability;
  description
    "Identity for advanced NSF anti-virus detect capability";
  reference
    "draft-dong-i2nsf-asf-config-01: Configuration of
    Advanced Security Functions with I2NSF Security
    Controller - Anti-virus";
}

identity exception-application {
  base anti-virus-capability;
  description
    "Identity for advanced NSF anti-virus exception
    application capability";
  reference
    "draft-dong-i2nsf-asf-config-01: Configuration of
    Advanced Security Functions with I2NSF Security
    Controller - Anti-virus";
}

identity exception-signature {
  base anti-virus-capability;
  description
    "Identity for advanced NSF anti-virus exception
    signature capability";
  reference
    "draft-dong-i2nsf-asf-config-01: Configuration of
    Advanced Security Functions with I2NSF Security
    Controller - Anti-virus";
}

identity whitelists {
  base anti-virus-capability;
  description

"Identity for advanced NSF anti-virus whitelists capability";
reference
"draft-dong-i2nsf-asf-config-01: Configuration of Advanced Security Functions with I2NSF Security Controller - Anti-virus";
}

identity syn-flood-action {
    base anti-ddos-capability;
    description
        "Identity for advanced NSF anti-DDoS syn flood action capability";
    reference
        "draft-dong-i2nsf-asf-config-01: Configuration of Advanced Security Functions with I2NSF Security Controller - Anti-DDoS";
}

identity udp-flood-action {
    base anti-ddos-capability;
    description
        "Identity for advanced NSF anti-DDoS UDP flood action capability";
    reference
        "draft-dong-i2nsf-asf-config-01: Configuration of Advanced Security Functions with I2NSF Security Controller - Anti-DDoS";
}

identity http-flood-action {
    base anti-ddos-capability;
    description
        "Identity for advanced NSF anti-DDoS http flood action capability";
    reference
        "draft-dong-i2nsf-asf-config-01: Configuration of Advanced Security Functions with I2NSF Security Controller - Anti-DDoS";
}

identity https-flood-action {
    base anti-ddos-capability;
    description
        "Identity for advanced NSF anti-DDoS https flood action capability";
    reference
        "draft-dong-i2nsf-asf-config-01: Configuration of Advanced Security Functions with I2NSF Security Controller - Anti-DDoS";
}

Advanced Security Functions with I2NSF Security Controller - Anti-DDoS;
}

identity dns-request-flood-action {
  base anti-ddos-capability;
  description "Identity for advanced NSF anti-DDoS dns request flood action capability";
}

identity dns-reply-flood-action {
  base anti-ddos-capability;
  description "Identity for advanced NSF anti-DDoS dns reply flood action capability";
}

identity icmp-flood-action {
  base anti-ddos-capability;
  description "Identity for advanced NSF anti-DDoS icmp flood action capability";
}

identity sip-flood-action {
  base anti-ddos-capability;
  description "Identity for advanced NSF anti-DDoS sip flood action capability";
}
identity detect-mode {
    base anti-ddos-capability;
    description "Identity for advanced NSF anti-DDoS detect mode capability";
}

identity baseline-learning {
    base anti-ddos-capability;
    description "Identity for advanced NSF anti-DDoS baseline learning capability";
}

identity signature-set {
    base ips-capability;
    description "Identity for advanced NSF IPS signature set capability";
    reference "draft-dong-i2nsf-asf-config-01: Configuration of Advanced Security Functions with I2NSF Security Controller - Intrusion Prevention System";
}

identity ips-exception-signature {
    base ips-capability;
    description "Identity for advanced NSF IPS exception signature capability";
    reference "draft-dong-i2nsf-asf-config-01: Configuration of Advanced Security Functions with I2NSF Security Controller - Intrusion Prevention System";
}

identity voice-id {
    base voip-volte-capability;
    description "Identity for advanced NSF VoIP/VoLTE voice-id"
identity user-agent {
  base voip-volte-capability;
  description "Identity for advanced NSF VoIP/VoLTE user agent capability";
  reference "RFC 3261: SIP: Session Initiation Protocol";
}

identity ipsec-capability {
  description "Base identity for an IPsec capabilities";
}

identity ike {
  base ipsec-capability;
  description "Identity for an IPsec Internet Key Exchange (IKE) capability";
}

identity ikeless {
  base ipsec-capability;
  description "Identity for an IPsec without Internet Key Exchange (IKE) capability";
}

/*
 * Grouping
 */

grouping nsf-capabilities {
  description "Network Security Funtion (NSF) Capabilities";

leaf-list time-capabilities {
  type enumeration {
    enum absolute-time {
      description
        "absolute time capabilities.
        If network security function has the absolute time
        capability, the network security function
        supports rule execution according to absolute time.";
    }
    enum periodic-time {
      description
        "periodic time capabilities.
        If network security function has the periodic time
        capability, the network security function
        supports rule execution according to periodic time.";
    }
  }
  description
    "Time capabilities";
}

container event-capabilities {
  description
    "Capabilities of events.
    If network security function has
    the event capabilities, the network security functions
    supports rule execution according to system event
    and system alarm.";
  reference
    "RFC 8329: Framework for Interface to Network Security
    Functions - I2NSF Flow Security Policy Structure
draft-ietf-i2nsf-capability-04: Information Model
of NSFs Capabilities - Design Principles and ECA
Policy Model Overview
draft-ietf-i2nsf-nsf-monitoring-data-model-01: A YANG
Data Model for Monitoring I2NSF Network Security
Functions - System Alarm and System Events";

leaf-list system-event-capability {
  type identityref {
    base system-event-capability;
  }
  description
    "System event capabilities";
}

leaf-list system-alarm-capability {
}
type identityref {
  base system-alarm-capability;
}
description
  "System alarm Capabilities";
}

container condition-capabilities {
  description
  "Conditions capabilities.";
}

container generic-nsf-capabilities {
  description
  "Conditions capabilities.
   If a network security function has
   the condition capabilities, the network security function
   supports rule execution according to conditions of IPv4,
   IPv6, TCP, UDP, ICMP, and payload.";
  reference
    "RFC 791: Internet Protocol
RFC 792: Internet Control Message Protocol
RFC 793: Transmission Control Protocol
RFC 8200: Internet Protocol, Version 6 (IPv6)
Specification - Next Header
RFC 8329: Framework for Interface to Network Security
Functions - I2NSF Flow Security Policy Structure
draft-ietf-i2nsf-capability-04: Information Model
of NSFs Capabilities - Design Principles and ECA Policy
Model Overview";

  leaf-list ipv4-capability {
    type identityref {
      base ipv4-capability;
    }
    description
      "IPv4 packet capabilities";
    reference
      "RFC 791: Internet Protocol";
  }

  leaf-list ipv6-capability {
    type identityref {
      base ipv6-capability;
    }
    description
      "IPv6 packet capabilities";
    reference
  }

leaf-list tcp-capability {
    type identityref {
        base tcp-capability;
    }
    description
        "TCP packet capabilities";
    reference
        "RFC 793: Transmission Control Protocol";
}

leaf-list udp-capability {
    type identityref {
        base udp-capability;
    }
    description
        "UDP packet capabilities";
    reference
        "RFC 768: User Datagram Protocol";
}

leaf-list icmp-capability {
    type identityref {
        base icmp-capability;
    }
    description
        "ICMP packet capabilities";
    reference
        "RFC 8200: Internet Protocol, Version 6 (IPv6)";
}

container advanced-nsf-capabilities {
    description
        "Advanced Network Security Function (NSF) capabilities, such as anti-virus, anti-DDoS, IPS, and VoIP/VoLTE.";
    reference

    leaf-list anti-virus-capability {
        type identityref {
            base anti-virus-capability;
        }
        description
            "Anti-virus packet capabilities";
        reference
            "RFC 8200: Internet Protocol, Version 6 (IPv6)";
    }
}
base anti-virus-capability;
}
description
"Anti-virus capabilities";
reference
"draft-dong-i2nsf-asf-config-01: Configuration of Advanced Security Functions with I2NSF Security Controller";
}

leaf-list anti-ddos-capability {
type identityref {
  base anti-ddos-capability;
}
description
"Anti-ddos capabilities";
reference
"draft-dong-i2nsf-asf-config-01: Configuration of Advanced Security Functions with I2NSF Security Controller";
}

leaf-list ips-capability {
type identityref {
  base ips-capability;
}
description
"Intrusion Prevention System (IPS) capabilities";
reference
"draft-dong-i2nsf-asf-config-01: Configuration of Advanced Security Functions with I2NSF Security Controller";
}

leaf-list url-capability {
type identityref {
  base url-capability;
}
description
"URL capabilities";
reference
"draft-dong-i2nsf-asf-config-01: Configuration of Advanced Security Functions with I2NSF Security Controller";
}

leaf-list voip-volte-capability {
type identityref {

base voip-volte-capability;

}  
description
"VoIP and VoLTE capabilities";
reference
"draft-dong-i2nsf-asf-config-01: Configuration of
Advanced Security Functions with I2NSF Security
Controller";

}

leaf-list context-capabilities {  
type identityref {  
  base context-capability;
}

description
"Security context capabilities";

}

container action-capabilities {  
description
"Action capabilities. 
If network security function has
the action capabilities, it supports
the attendant actions for policy rules.";

leaf-list ingress-action-capability {  
type identityref {  
  base ingress-action-capability;
}

description
"Ingress-action capabilities";

}

leaf-list egress-action-capability {  
type identityref {  
  base egress-action-capability;
}

description
"Egress-action capabilities";

}

leaf-list log-action-capability {  
type identityref {  
  base log-action-capability;
}

description
"Log-action capabilities";

}
leaf-list resolution-strategy-capabilities {
  type identityref {
    base resolution-strategy-capability;
  }
  description
    "Resolution strategy capabilities. The resolution strategies can be used to specify how to resolve conflicts that occur between the actions of the same or different policy rules that are matched for the same packet and by particular NSF";
  reference
    "draft-ietf-i2nsf-capability-04: Information Model of NSFs Capabilities - Resolution strategy";
}

leaf-list default-action-capabilities {
  type identityref {
    base default-action-capability;
  }
  description
    "Default action capabilities. A default action is used to execute I2NSF policy rules when no rule matches a packet. The default action is defined as pass, drop, reject, alert, or mirror.";
  reference
    "draft-ietf-i2nsf-capability-04: Information Model of NSFs Capabilities - Default action";
}

leaf-list ipsec-method {
  type identityref {
    base ipsec-capability;
  }
  description
    "IPsec method capabilities";
  reference
    "draft-ietf-i2nsf-sdn-ipsec-flow-protection-04";
}

/*
* Data nodes
*/
list nsf {
  key "nsf-name";
  description
    "The list of Network security Function (NSF) capabilities";
  leaf nsf-name {
    type string;
    mandatory true;
    description
      "The name of network security function";
  }
}

<CODE ENDS>

Figure 3: YANG Data Module of I2NSF Capability

7. IANA Considerations

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


Registrant Contact: The IESG.

XML: N/A; the requested URI is an XML namespace.

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

  name: ietf-i2nsf-capability
  prefix: nsfcap
  reference: RFC XXXX

8. Security Considerations

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

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

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

- ietf-i2nsf-capability: The attacker may provide incorrect information of the security capability of any target NSF by illegally modifying this.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

- ietf-i2nsf-capability: The attacker may gather the security capability information of any target NSF and misuse the information for subsequent attacks.

9. References

9.1. Normative References


9.2. Informative References

[draft-dong-i2nsf-asf-config]

[draft-ietf-i2nsf-capability]

[draft-ietf-i2nsf-nsf-facing-interface-dm]
Internet-Draft    I2NSF Capability YANG Data Model    July 2019

[draft-ietf-i2nsf-nsf-monitoring-data-model]
Jeong, J., Chung, C., Hares, S., Xia, L., and H. Birkholz,
"I2NSF NSF Monitoring YANG Data Model", draft-ietf-i2nsf-
nsf-monitoring-data-model-01 (work in progress), July
2019.

[draft-ietf-i2nsf-sdn-ipsec-flow-protection]
Marin-Lopez, R., Lopez-Millan, G., and F. Pereniguez-
Garcia, "Software-Defined Networking (SDN)-based IPsec
Flow Protection", draft-ietf-i2nsf-sdn-ipsec-flow-
protection-05 (work in progress), July 2019.

[draft-ietf-i2nsf-terminology]
Hares, S., Strassner, J., Lopez, D., Xia, L., and H.
Birkholz, "Interface to Network Security Functions (I2NSF)
Terminology", draft-ietf-i2nsf-terminology-08 (work in
progress), July 2019.

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

This section shows configuration examples of "ietf-i2nsf-capability" module for capabilities registration of general firewall.

A.1. Example 1: Registration for Capabilities of General Firewall

This section shows a configuration example for capabilities registration of general firewall.

```xml
<nsf xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
  <nsf-name>general_firewall</nsf-name>
  <condition-capabilities>
    <generic-nsf-capabilities>
      <ipv4-capability>ipv4-protocol</ipv4-capability>
      <ipv4-capability>exact-ipv4-address</ipv4-capability>
      <ipv4-capability>range-ipv4-address</ipv4-capability>
      <tcp-capability>exact-fourth-layer-port-num</tcp-capability>
      <tcp-capability>range-fourth-layer-port-num</tcp-capability>
    </generic-nsf-capabilities>
  </condition-capabilities>
  <action-capabilities>
    <ingress-action-capability>pass</ingress-action-capability>
    <ingress-action-capability>drop</ingress-action-capability>
    <ingress-action-capability>alert</ingress-action-capability>
    <egress-action-capability>pass</egress-action-capability>
    <egress-action-capability>drop</egress-action-capability>
    <egress-action-capability>alert</egress-action-capability>
  </action-capabilities>
</nsf>
```

Figure 4: Configuration XML for Capabilities Registration of General Firewall

Figure 4 shows the configuration XML for capabilities registration of general firewall and its capabilities are as follows.

1. The name of the NSF is general_firewall.

2. The NSF can inspect protocol, exact IPv4 address, and range IPv4 address for IPv4 packets.

3. The NSF can inspect exact port number and range port number for fourth layer packets.

4. The NSF can control whether the packets are allowed to pass, drop, or alert.

A.2. Example 2: Registration for Capabilities of Time based Firewall

This section shows a configuration example for capabilities registration of time based firewall.

```xml
<nf xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
  <nf-name>time_based_firewall</nf-name>
  <time-capabilities>absolute-time</time-capabilities>
  <time-capabilities>periodic-time</time-capabilities>
  <condition-capabilities>
    <generic-nsf-capabilities>
      <ipv4-capability>ipv4-protocol</ipv4-capability>
      <ipv4-capability>exact-ipv4-address</ipv4-capability>
      <ipv4-capability>range-ipv4-address</ipv4-capability>
    </generic-nsf-capabilities>
  </condition-capabilities>
  <action-capabilities>
    <ingress-action-capability>pass</ingress-action-capability>
    <ingress-action-capability>drop</ingress-action-capability>
    <ingress-action-capability>alert</ingress-action-capability>
    <egress-action-capability>pass</egress-action-capability>
    <egress-action-capability>drop</egress-action-capability>
    <egress-action-capability>alert</egress-action-capability>
  </action-capabilities>
</nf>
```

Figure 5: Configuration XML for Capabilities Registration of Time based Firewall

Figure 5 shows the configuration XML for capabilities registration of time based firewall and its capabilities are as follows.

1. The name of the NSF is `time_based_firewall`.
2. The NSF can execute the security policy rule according to absolute time and periodic time.
3. The NSF can inspect protocol, exact IPv4 address, and range IPv4 address for IPv4 packets.
4. The NSF can control whether the packets are allowed to pass, drop, or alert.
A.3.  Example 3: Registration for Capabilities of Web Filter

This section shows a configuration example for capabilities registration of web filter.

```xml
<nf xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
  <nf-name>web_filter</nf-name>
  <condition-capabilities>
    <advanced-nsf-capabilities>
      <url-capability>user-defined</url-capability>
    </advanced-nsf-capabilities>
  </condition-capabilities>
  <action-capabilities>
    <ingress-action-capability>pass</ingress-action-capability>
    <ingress-action-capability>drop</ingress-action-capability>
    <ingress-action-capability>alert</ingress-action-capability>
    <egress-action-capability>pass</egress-action-capability>
    <egress-action-capability>drop</egress-action-capability>
    <egress-action-capability>alert</egress-action-capability>
  </action-capabilities>
</nf>
```

Figure 6: Configuration XML for Capabilities Registration of Web Filter

Figure 6 shows the configuration XML for capabilities registration of web filter and its capabilities are as follows.

1. The name of the NSF is web_filter.
2. The NSF can inspect url for http and https packets.
3. The NSF can control whether the packets are allowed to pass, drop, or alert.

A.4.  Example 4: Registration for Capabilities of VoIP/VoLTE Filter

This section shows a configuration example for capabilities registration of VoIP/VoLTE filter.
<nsf xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
  <nsf-name>voip_volte_filter</nsf-name>
  <condition-capabilities>
    <advanced-nsf-capabilities>
      <voip-volte-capability>voice-id</voip-volte-capability>
    </advanced-nsf-capabilities>
  </condition-capabilities>
  <action-capabilities>
    <ingress-action-capability>pass</ingress-action-capability>
    <ingress-action-capability>drop</ingress-action-capability>
    <ingress-action-capability>alert</ingress-action-capability>
    <egress-action-capability>pass</egress-action-capability>
    <egress-action-capability>drop</egress-action-capability>
    <egress-action-capability>alert</egress-action-capability>
  </action-capabilities>
</nsf>

Figure 7: Configuration XML for Capabilities Registration of VoIP/VoLTE Filter

Figure 7 shows the configuration XML for capabilities registration of VoIP/VoLTE filter and its capabilities are as follows.

1. The name of the NSF is voip_volte_filter.
2. The NSF can inspect voice id for VoIP/VoLTE packets.
3. The NSF can control whether the packets are allowed to pass, drop, or alert.

A.5. Example 5: Registration for Capabilities of HTTP and HTTPS Flood Mitigation

This section shows a configuration example for capabilities registration of http and https flood mitigation.
<nsf xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-capability">
  <nsf-name>http_and_https_flood_mitigation</nsf-name>
  <condition-capabilities>
    <advanced-nsf-capabilities>
      <anti-ddos-capability>http-flood-action</anti-ddos-capability>
      <anti-ddos-capability>https-flood-action</anti-ddos-capability>
    </advanced-nsf-capabilities>
  </condition-capabilities>
  <action-capabilities>
    <ingress-action-capability>pass</ingress-action-capability>
    <ingress-action-capability>drop</ingress-action-capability>
    <ingress-action-capability>alert</ingress-action-capability>
    <egress-action-capability>pass</egress-action-capability>
    <egress-action-capability>drop</egress-action-capability>
    <egress-action-capability>alert</egress-action-capability>
  </action-capabilities>
</nsf>

Figure 8: Configuration XML for Capabilities Registration of HTTP and HTTPS Flood Mitigation

Figure 8 shows the configuration XML for capabilities registration of http and https flood mitigation and its capabilities are as follows.

1. The name of the NSF is http_and_https_flood_mitigation.
2. The location of the NSF is 221.159.112.140.
3. The NSF can control the amount of packets for http and https packets.
4. The NSF can control whether the packets are allowed to pass, drop, or alert.

Appendix B. Changes from draft-ietf-i2nsf-capability-data-model-04

The following changes are made from draft-ietf-i2nsf-capability-data-model-04:

- The version is revised according to the comments from Acee Lindem and Carl Moberg who are YANG doctors for review.

Appendix C. 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).

Appendix D. Contributors

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

- Hyoungshick Kim (Sungkyunkwan University)
- Daeyoung Hyun (Sungkyunkwan University)
- Dongjin Hong (Sungkyunkwan University)
- Liang Xia (Huawei)
- Jung-Soo Park (ETRI)
- Tae-Jin Ahn (Korea Telecom)
- Se-Hui Lee (Korea Telecom)

Authors’ Addresses

Susan Hares
Huawei
7453 Hickory Hill
Saline, MI  48176
USA

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

Jaehoon Paul Jeong
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
Jinyong Tim Kim
Department of Electronic, Electrical and Computer Engineering
Sungkyunkwan University
2066 Seobu-Ro, Jangan-Gu
Suwon, Gyeonggi-Do 16419
Republic of Korea
Phone: +82 10 8273 0930
EMail: timkim@skku.edu

Robert Moskowitz
HTT Consulting
Oak Park, MI
USA
Phone: +1-248-968-9809
EMail: rgm@htt-consult.com

Qiushi Lin
Huawei
Huawei Industrial Base
Shenzhen, Guangdong 518129
China
EMail: linqiushi@huawei.com
Abstract

This document describes an information model and a YANG data model for the Consumer-Facing Interface between an Interface to Network Security Functions (I2NSF) User and Security Controller in an I2NSF system in a Network Functions Virtualization (NFV) environment. The information model defines various types of managed objects and the relationship among them needed to build the interface. The information model is organized based on the "Event-Condition-Action" (ECA) policy model defined by a capability information model for I2NSF [i2nsf-capability-im], and the data model is defined for enabling different users of a given I2NSF system to define, manage, and monitor security policies for specific flows within an administrative domain.

Status of This Memo

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

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

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

This Internet-Draft will expire on January 25, 2020.
Internet-Draft  Consumer-Facing Interface YANG Data Model      July 2019

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust’s Legal
Provisions Relating to IETF Documents
(https://trustee.ietf.org/license-info) in effect on the date of
publication of this document. Please review these documents
carefully, as they describe your rights and restrictions with respect
to this document. Code Components extracted from this document must
include Simplified BSD License text as described in Section 4.e of
the Trust Legal Provisions and are provided without warranty as
described in the Simplified BSD License.

Table of Contents

1. Introduction .................................................. 3
2. Requirements Language ........................................ 5
3. Terminology ..................................................... 5
4. Information Model for Policy .................................. 5
  4.1. Event Sub-model ........................................... 7
  4.2. Condition Sub-model ....................................... 8
  4.3. Action Sub-model .......................................... 9
5. Information Model for Multi-Tenancy .......................... 10
  5.1. Policy Domain .............................................. 10
  5.2. Policy Tenant .............................................. 11
  5.3. Policy Role ................................................ 11
  5.4. Policy User ................................................ 12
  5.5. Policy Management Authentication Method .................. 13
6. Information Model for Policy Endpoint Groups ............... 14
  6.1. User Group ................................................ 15
  6.2. Device Group .............................................. 16
  6.3. Location Group ............................................. 16
7. Information Model for Threat Prevention ..................... 17
  7.1. Threat Feed ................................................. 18
  7.2. Payload Content ........................................... 18
8. Role-based Access Control (RBAC) ........................... 19
9. YANG Data Model for Security Policies for Consumer-Facing
   Interface ....................................................... 20
10. Example XML Output for Various Scenarios .................. 49
  10.1. DB Registration: Information of Positions and Devices
       (Endpoint Group) ........................................... 49
  10.2. Scenario 1: Block SNS Access during Business Hours .... 50
  10.3. Scenario 2: Block Malicious VoIP/VoLTE Packets Coming to
       a Company ................................................ 52
  10.4. Scenario 3: Mitigate HTTP and HTTPS Flood Attacks on a
       Company Web Server ......................................... 53
1. Introduction

In a framework of Interface to Network Security Functions (I2NSF), each vendor can register their NSFs using a Developer’s Management System (DMS). Assuming that vendors also provide the front-end web applications registered with an I2NSF User, the Consumer-Facing Interface is required because the web applications developed by each vendor need to have a standard interface specifying the data types used when the I2NSF User and Security Controller communicate using this interface. Therefore, this document specifies the required information, their data types, and encoding schemes so that high-level security policies (or configuration information for security policies) can be transferred to the Security Controller through the Consumer-Facing Interface. These policies can easily be translated by the Security Controller into low-level security policies. The Security Controller delivers the translated policies to Network Security Functions (NSFs) according to their respective security capabilities for the required security enforcement.

The Consumer-Facing Interface would be built using a set of objects, with each object capturing a unique set of information from Security Administrator (i.e., I2NSF User [RFC8329]) needed to express a Security Policy. An object may have relationship with various other objects to express a complete set of requirements. An information model captures the managed objects and relationship among these objects. The information model proposed in this document is structured in accordance with the "Event-Condition-Action" (ECA) policy model.

An NSF Capability model is proposed in [i2nsf-capability-im] as the basic model for both the NSF-Facing interface and Consumer-Facing Interface security policy model of this document.

[RFC3444] explains differences between an information and data model. This document uses the guidelines in [RFC3444] to define both the information and data model for Consumer-Facing Interface. Figure 1 shows a high-level abstraction of Consumer-Facing Interface. A data
model, which represents an implementation of the information model in a specific data representation language, is also defined in this document.

![Diagram]

**Figure 1: Diagram for High-level Abstraction of Consumer-Facing Interface**

Data models are defined at a lower level of abstraction and provide many details. They provide details about the implementation of a protocol's specification, e.g., rules that explain how to map managed objects onto lower-level protocol constructs. Since conceptual models can be implemented in different ways, multiple data models can be derived from a single information model.

The efficient and flexible provisioning of network functions by a Network Functions Virtualization (NFV) system leads to a rapid advance in the network industry. As practical applications, Network Security Functions (NSFs), such as firewall, Intrusion Detection System (IDS)/Intrusion Prevention System (IPS), and attack mitigation, can also be provided as Virtual Network Functions (VNF) in the NFV system. By the efficient virtualization technology, these
VNFs might be automatically provisioned and dynamically migrated based on real-time security requirements. This document presents a YANG data model to implement security functions based on NFV.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC3444] RFC8174 [RFC8174].

3. Terminology

This document uses the terminology described in [i2nsf-terminology][client-facing-inf-req].

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

4. Information Model for Policy

A Policy object represents a mechanism to express a Security Policy by Security Administrator (i.e., I2NSF User) using Consumer-Facing Interface toward Security Controller; the policy would be enforced on an NSF. Figure 2 shows the YANG tree of the Policy object. The Policy object SHALL have the following information:

- **Name**: This field identifies the name of this object.
- **Date**: Date when this object was created or last modified.
- **Rules**: This field contains a list of rules. These rules are defined for 1) communication between two Endpoint Groups, 2) for preventing communication with externally or internally identified threats, and 3) for implementing business requirement such as controlling access to internal or external resources for meeting regulatory compliance or business objectives. An organization may restrict certain communication between a set of user and applications for example. The threats may be from threat feeds obtained from external sources or dynamically identified by using specialty devices in the network. Rule conflict analysis should be triggered by the monitoring service to perform an exhaustive detection of anomalies among the configuration rules installed into the security functions.
A policy is a container of Rules. In order to express a Rule, a Rule must have complete information such as where and when a policy needs to be applied. This is done by defining a set of managed objects and relationship among them. A Policy Rule may be related segmentation, threat mitigation or telemetry data collection from an NSF in the network, which will be specified as the sub-model of the policy model in the subsequent sections. Figure 3 shows the YANG tree of the Rule object. The rule object SHALL have the following information:

Name:  This field identifies the name of this object.

Event:  This field includes the information to determine whether the Rule Condition can be evaluated or not. See details in Section 3.1.

Condition:  This field contains all the checking conditions to apply to the objective traffic. See details in Section 4.2.

Action:  This field identifies the action taken when a rule is matched. There is always an implicit action to drop traffic if no rule is matched for a traffic type. See details in Section 4.3.

IPsec-Method:  This field contains the information about IPsec method type. There are two types such as IPsec-IKE and IPsec-IKEless [i2nsf-ipsec].

Owner:  This field contains the owner of the rule. For example, the person who created it, and eligible for modifying it.
4.1. Event Sub-model

The Event Object contains information related to scheduling a Rule. The Rule could be activated based on a set time or security event. Figure 4 shows the YANG tree of the Event object. Event object SHALL have following information:

- **Security-event**: This field identifies for which security event the policy is enforced. The examples of security events are: "DDOS", "spyware", "trojan", and "ransomware".

- **Enforce-type**: This field identifies whether the event of triggering policy enforcement is "Admin" or "Time".

- **Admin**: This represents the enforcement type based on admin’s decision.

- **Time**: This represents the security rule is enforced based on begin-time and end-time information.

- **Frequency**: This represents how frequent the rule should be enforced. There are four options: "only-once", "daily", "weekly" and "monthly".

```
+--rw event
  +--rw security-event identityref
  +--rw (enforce-type)?
    |  +--:(admin)
    |     +--rw admin? identityref
    |     +--:(time)
    |        +--rw time-information
    |        +--rw begin-time? yang:date-and-time
    |        +--rw end-time? yang:date-and-time
    +--rw frequency? enumeration
```

Figure 4: Event Sub-model YANG Data Tree
4.2. Condition Sub-model

This object represents Conditions that Security Administrator wants to apply the checking on the traffic in order to determine whether the set of actions in the Rule can be executed or not. The Condition Sub-model consists of three different types of containers each representing different cases, such as general firewall and DDoS-mitigation cases, and a case when the condition is based on the payload strings of packets. Each containers have source-target and destination-target to represent the source and destination for each case. Figure 5 shows the YANG tree of the Condition object. The Condition Sub-model SHALL have following information:

Case (Firewall-condition): This field represents the general firewall case, where a security admin can set up firewall conditions using the information present in this field. The source and destination is represented as firewall-source and firewall-destination, each referring to the IP-address-based groups defined in the endpoint-group.

DDoS-condition: This field represents the condition for DDoS mitigation, where a security admin can set up DDoS mitigation conditions using the information present in this field. The source and destination is represented as ddos-source and ddos-destination, each referring to the device-groups defined and registered in the endpoint-group.

Custom-condition: This field contains the payload string information. This information is useful when security rule condition is based on the string contents of incoming or outgoing packets. The source and destination is represented as custom-source and custom-destination, each referring to the payload-groups defined and registered in the endpoint-group.

Threat-feed-condition: This field contains the information obtained from threat-feeds (e.g., Palo-Alto, or RSA-netwitness). This information is useful when security rule condition is based on the existing threat reports gathered by other sources. The source and destination is represented as threat-feed-source and threat-feed-destination. For clarity, threat-feed-source/destination represent the source/destination of a target security threat, not the information source/destination of a threat-feed.
4.3. Action Sub-model

This object represents actions that Security Admin wants to perform based on certain traffic class. Figure 6 shows the YANG tree of the Action object. The Action object SHALL have following information:

Primary-action: This field identifies the action when a rule is matched by an NSF. The action could be one of "PASS", "DROP", "ALERT", "RATE-LIMIT", and "MIRROR".

Secondary-action: This field identifies the action when a rule is matched by an NSF. The action could be one of "log", "syslog", "session-log".

```
  +--rw action
      +--rw primary-action    identityref
      +--rw secondary-action?  identityref
```
5. Information Model for Multi-Tenancy

Multi-tenancy is an important aspect of any application that enables multiple administrative domains in order to manage application resources. An Enterprise organization may have multiple tenants or departments such as Human Resources (HR), Finance, and Legal, with each tenant having a need to manage their own Security Policies. In a Service Provider, a tenant could represent a Customer that wants to manage its own Security Policies. There are multiple managed objects that constitute multi-tenancy aspects as shown in Figure 7. This section lists these objects and the relationship among these objects. Below diagram shows an example of multi-tenancy in an Enterprise domain.

```
(Multi-Tenancy)       +-------------------+
                     |       Domain      |
                     |(e.g., Enterprise) |
                     +---------+---------+
                          |                    |
                          +--------------------+--------------------+
                          |                    |                    |
                          +--------+-------+  +---------+--------+  +--------+--------+
                          |  Department 1  |  |   Department 2   |  |  Department n   |
                          +--------+-------+  +---------+--------+  +--------+--------+
                                          |                    |
                                          +--------+--------+  +--------+--------+  +--------+--------+
                                          | Sub-domain 1..n |  | Sub-domain 1..n |  | Sub-domain 1..n |
                                          +--------+--------+  +--------+--------+  +--------+--------+
                                                            |                    |
                                                            +--------+--------+  +--------+--------+  +--------+--------+
                                                            |   Tenant 1..n   |  |   Tenant 1..n   |  |   Tenant 1..n   |
                                                            +-----------------+  +-----------------+  +-----------------+
```

Figure 7: Multi-tenancy Diagram

5.1. Policy Domain

This object defines a boundary for the purpose of policy management within a Security Controller. This may vary based on how the Security Controller is deployed and hosted. For example, if an Enterprise hosts a Security Controller in their network; the domain in this case could just be the one that represents that Enterprise. But if a Cloud Service Provider hosts managed services, then a domain could represent a single customer of that Provider. Figure 8 shows...
5.2. Policy Tenant

This object defines an entity within an organization. The entity could be a department or business unit within an Enterprise organization that would like to manage its own Policies due to regulatory compliance or business reasons. Figure 9 shows the YANG tree of the Policy-Tenant object. The Policy-Tenant object SHALL have the following information:

Tenant-type: This field represents the type of tenant within a domain. In an enterprise, the examples of tenants could be the departments or divisions, such as HR department and Finance department.

```
+--rw policy-tenant* [tenant-name]
    +--rw tenant-type identityref
```

Figure 9: Policy Tenant YANG Data Tree

5.3. Policy Role

This object defines a set of permissions assigned to a user in an organization that wants to manage its own Security Policies. It provides a convenient way to assign policy users to a job function or a set of permissions within the organization. Figure 10 shows the YANG tree of the Policy-Role object. The Policy-Role object SHALL have the following information:
Role-type: "This represent the roles within the tenants, in order to distinguish who may or may not have access to policies. The role types include "user", "group", "other", and "all". "user" represents an individual whereas "group" represents a group of users. "All" means both the individual and the group members, whereas "other" denotes anyone who is not a specific individual or a member of a specific group.

```
  +--rw policy-role*        [role-name]
  |   +--rw role-type       identityref
```

Figure 10: Policy Role YANG Data Tree

5.4. Policy User

This object represents a unique identity of a user within an organization. The identity authenticates with Security Controller using credentials such as a password or token in order to perform policy management. A user may be an individual, system, or application requiring access to Security Controller. Figure 11 shows the YANG tree of the Policy-User object. The Policy-User object SHALL have the following information:

Name: Name of a user.

Password: User password for basic authentication. The crypto-hash mechanism for this entry is ianach:crypt-hash.

Email: E-mail address of the user.

Access-profile: This represents the access profile for the user. The access-profile is based on the permission-type and the scope type defined. The permission-types include "no-permission", read", "write", "execute", "read-and-write", "read-and-execute", and "write-and-execute"

Scope-Type: This field identifies whether the user has domain-wide or tenant-wide privileges.
5.5. Policy Management Authentication Method

This object represents authentication schemes supported by Security Controller. Figure 12 shows the YANG tree of the Policy Management Authentication Method object. This Policy-Management-Authentication-Method object SHALL have the following information:

Policy-mgmt-auth-method-instance: This field represents the authentication instances. Each instance is based on either client authentication, server authentication or both (mutual) authentication.

Policy-mgmt-auth-method: This represents the choices of authentication methods. Each instance of authentication consists of authentication methods chosen by an entity, such as a security admin. There are "Password-based", "token-based", "certificate-based", and "IPsec" authentication methods.

Password-list: This list contains the passwords that are encrypted using crypto-has algorithm (ianach:crypto-hash).

Token-list: This list contains the information such as the access tokens and a token server.

Cert-server-list: This list contains the certification server information such as server address (IPv4 and IPv6) and certificate types.

IPsec: This list has IPsec method types based on the identities defined. There are two types such as IPsec-IKE and IPsec-IKEless.
Internet-Draft  Consumer-Facing Interface YANG Data Model  July 2019

Figure 12: Policy Management Authentication Method YANG Data Tree

6. Information Model for Policy Endpoint Groups

The Policy Endpoint Group is a very important part of building User-Construct based policies. A Security Administrator would create and use these objects to represent a logical entity in their business environment, where a Security Policy is to be applied. There are multiple managed objects that constitute a Policy’s Endpoint Group as shown in Figure 13. Figure 14 shows the YANG tree of the Endpoint-Group object. This section lists these objects and relationship among them.

Figure 13: Endpoint Group Diagram
6.1. User Group

This object represents a User-Group. Figure 15 shows the YANG tree of the User-Group object. The User-Group object SHALL have the following information:

Name: This field identifies the name of this object.

IP-address: This represents the IPv4 address of a user in the user group.

range-ipv4-address: This represents the IPv4 address of a user in the user group.

range-ipv6-address: This represents the IPv6 address of a user in the user group.

Figure 15: User Group YANG Data Tree
6.2. Device Group

This object represents a Device-Group. Figure 16 shows the YANG tree of the Device-group object. The Device-Group object SHALL have the following information:

- **Name**: This field identifies the name of this object.
- **IP-address**: This represents the IPv4 address of a device in the device group.
- **range-ipv4-address**: This represents the IPv4 address of a device in the device group.
- **range-ipv6-address**: This represents the IPv6 address of a device in the device group.
- **Protocol**: This represents the communication protocols used by the devices. The protocols are "SSH", "FTP", "SMTP", "HTTP", "HTTPS", and etc.

```yang
++-rw device-group* [name]
  ++-rw name                         string
  ++-rw (match-type)?
    ++-:(exact-match-ipv4)
      |  ++-rw ip-address*            inet:ipv4-address
    ++-:(exact-match-ipv6)
      |  ++-rw ip-address*            inet:ipv4-address
    ++-:(range-match-ipv4)
      |  ++-rw range-ipv4-address* [start-ipv4-address end-ipv4-address]
      |      ++-rw start-ipv4-address inet:ipv4-address
      |      ++-rw end-ipv4-address   inet:ipv4-address
    ++-:(range-match-ipv6)
      |      ++-rw range-ipv6-address* [start-ipv6-address end-ipv6-address]
      |      |      ++-rw start-ipv6-address inet:ipv6-address
      |      ++-rw end-ipv6-address   inet:ipv6-address
```

Figure 16: Device Group YANG Data Tree

6.3. Location Group

This object represents a location group based on either tag or other information. Figure 17 shows the YANG tree of the Location-Group object. The Location-Group object SHALL have the following information:

- **Name**: This field identifies the name of this object.
geo-ip-ipv4: This field represents the IPv4 Geo-ip of a location.
geo-ip-ipv6: This field represents the IPv6 Geo-ip of a location.
continent: This field represents the continent where the location group member is at.

```yaml
+--rw location-group* [name]
   +--rw name           string
   +--rw geo-ip-ipv4    inet:ipv4-address
   +--rw geo-ip-ipv6    inet:ipv6-address
   +--rw continent?     identityref
```

Figure 17: Location Group YANG Data Tree

7. Information Model for Threat Prevention

The threat prevention plays an important part in the overall security posture by reducing the attack surfaces. This information could come from various threat feeds (i.e., sources for obtaining the threat information), such as EmergingThreats.com or AlienVault.com. There are multiple managed objects that constitute this category. This section lists these objects and relationship among them. Figure 19 shows the YANG tree of a Threat-Prevention object.

```yaml
+-------------------+
| Threat Prevention |
+-------------------+

^ 1..n

+---------+---------+
| Threat-feed | payload-content |
+---------+---------+

Figure 18: Threat Prevention Diagram

```yaml
+--rw threat-prevention
   +--rw threat-feed-list* [name]
     ...
   +--rw payload-content* [name]
     ...
```

Figure 19: Threat Prevention YANG Data Tree
7.1. Threat Feed

This object represents a threat feed which provides signatures of malicious activities. Figure 20 shows the YANG tree of a Threat-feed-list. The Threat-Feed object SHALL have the following information:

- **Feed-name:** This field identifies the name of this object.
- **Feed-Server-ipv4:** This represents the IPv4 server address of the feed provider, it may be external or local servers.
- **Feed-Server-ipv6:** This represents the IPv6 server address of the feed provider, it may be external or local servers.
- **Feed-description:** This is the description of the threat feed. The descriptions should have clear indication of the security attack such as attack type (e.g., APT) and file types used (e.g., executable malware).
- **Threat-file-types:** This field identifies the information about the file types identified and reported by the threat-feed.
- **signatures:** This field contains the signatures of malicious programs or activities provided by the threat-feed. The examples of signature types are "YARA", "SURICATA", and "SNORT".

```
+--rw threat-prevention
   +--rw threat-feed-list* [feed-name]
      +--rw feed-name identityref
      +--rw feed-server-ipv4? inet:ipv4-address
      +--rw feed-server-ipv6? inet:ipv6-address
      +--rw feed-description? string
      +--rw threat-file-types* identityref
      +--rw signatures* identityref
```

Figure 20: Threat Feed YANG Data Tree

7.2. Payload Content

This object represents a custom list created for the purpose of defining exception to threat feeds. Figure 21 shows the YANG tree of a Payload-content list. The Payload-Content object SHALL have the following information:
Name: This field identifies the name of this object. For example, the name "backdoor" indicates the payload content is related to backdoor attack.

payload-description: This represents the description of how the payload content is related to a security attack.

Content: This contains the payload contents, which are involved in a security attack, as strings.

```yang
data-model
definitions {  
  payload-content*  
    name string  
    payload-description string  
    content* string
}
```

Figure 21: Payload Content in YANG Data Tree

8. Role-based Acess Control (RBAC)

Role-Based Access Control (RBAC) provides a powerful and centralized control within a network. It is a policy neutral access control mechanism defined around roles and privileges. The components of RBAC, such as role-permissions, user-role and role-role relationships, make it simple to perform user assignments.

```plaintext
                    +--------------+    +---------------+            +-------------+
                    |              |    |               |            |             |
                    |    User 1    + (has many)| (has many)|             |
                    |              |    +---------------+            +-------------+
                    |     \     |    |               |            |             |
                    |      .     |    +---------------+            +-------------+
                    |  ---> List of roles +----------->+ Permissions |
                    |              |    |               |            |             |
                    |    User n    +/  |               |            |             |
                    |              | (has many)    |               |             |
                    |              |    +---------------+            +-------------+

Figure 22: Role-based Acess Control Diagram
```

As shown in Figure 22, a role represents a collection of permissions (e.g., accessing a file server or other particular resources). A role may be assigned to one or multiple users. Both roles and permissions can be organized in a hierarchy. A role may consist of other roles and permissions.
Following are the steps required to build RBAC:

1. Defining roles and permissions.
2. Establishing relations among roles and permissions.
3. Defining users.
4. Associating rules with roles and permissions.
5. Assigning roles to users.

9. YANG Data Model for Security Policies for Consumer-Facing Interface

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

The semantics of the data model must be aligned with the information model of the Consumer-Facing Interface. The transformation of the information model was performed so that this YANG data model can facilitate the efficient delivery of the control or management messages.

This data model is designed to support the I2NSF framework that can be extended according to the security needs. In other words, the model design is independent of the content and meaning of specific policies as well as the implementation approach. This document suggests a VoIP/VoLTE security service as a use case for policy rule generation.

This section describes a YANG data model for Consumer-Facing Interface, based on the information model of Consumer-Facing Interface to Security Controller.

```<CODE BEGINS> file "ietf-cfi-policy.yang"
module ietf-i2nsf-cfi-policy {
  yang-version 1.1;
  namespace
  prefix
    cfi-policy;

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

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

import iana-crypt-hash {
    prefix ianach;
}

organization
    "IETF I2NSF (Interface to Network Security Functions) Working Group";

contact
    "WG Web: <http://tools.ietf.org/wg/i2nsf>
    WG List: <mailto:i2nsf@ietf.org>
    WG Chair: Adrian Farrel
    <mailto:Adrain@olddog.co.uk>
    WG Chair: Linda Dunbar
    <mailto:Linda.duhan@huawei.com>
    Editor: Jaehoon Paul Jeong
    <mailto:pauljeong@skku.edu>";

description
    "This module is a YANG module for Consumer-Facing Interface. Copyright (c) 2018 IETF Trust and the persons identified as authors of the code. All rights reserved. Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info). This version of this YANG module is part of RFC XXXX; see the RFC itself for full legal notices.";

revision "2019-07-21"{
    description "latest revision";
    reference
        "draft-ietf-consumer-facing-interface-dm-03";
identity permission-type {
    description
    "Base identity for the permission types.";
}

identity no-permission {
    base permission-type;
    description
    "Identity for no-permission.";
}

identity read {
    base permission-type;
    description
    "Identity for read permission.";
}

identity write {
    base permission-type;
    description
    "Identity for write permission.";
}

identity execute {
    base permission-type;
    description
    "Identity for execute permission.";
}

identity write-and-execute {
    base permission-type;
    description
    "Identity for write & execute permission.";
}

identity read-and-execute {
    base permission-type;
    description
    "Identity for read & execute permission.";
}

identity read-and-write {
    base permission-type;
    description
    "Identity for read & write permission.";
}

identity scope-type {
    description
    "Base Identity for scope-type.";
}

identity tenant-wide {
    base scope-type;
description
  "Base Identity for tenant-wide scope type."
}
identity domain-wide {
  base scope-type;
  description
  "Base Identity for domain-wide scope type."
}

identity malware-file-type {
  description
  "Base identity for malware file types."
}
identity executable-file {
  base malware-file-type;
  description
  "Identity for executable file types."
}
identity doc-file {
  base malware-file-type;
  description
  "Identity for Microsoft document file types."
}
identity html-app-file {
  base malware-file-type;
  description
  "Identity for html application file types."
}
identity javascript-file {
  base malware-file-type;
  description
  "Identity for Javascript file types."
}
identity pdf-file {
  base malware-file-type;
  description
  "Identity for pdf file types."
}
identity dll-file {
  base malware-file-type;
  description
  "Identity for dll file types."
}
identity msi-file {
  base malware-file-type;
  description
  "Identity for Microsoft installer file types."
}
identity security-event-type {
  description
    "Base identity for security event types.";
}

identity ddos {
  base malware-file-type;
  description
    "Identity for DDoS event types.";
}

identity spyware {
  base malware-file-type;
  description
    "Identity for spyware event types.";
}

identity trojan {
  base malware-file-type;
  description
    "Identity for Trojan infection event types.";
}

identity ransomware {
  base malware-file-type;
  description
    "Identity for ransomware infection event types.";
}

identity i2nsf-ipsec {
  description
    "Base identity for IPsec method types.";
}

identity ipsec-ike {
  base i2nsf-ipsec;
  description
    "Identity for ipsec-ike.";
}

identity ipsec-ikeless {
  base i2nsf-ipsec;
  description
    "Identity for ipsec-ikeless.";
}

identity continent {
  description
    "Base Identity for continent types.";
}

identity africa {
  base continent;
}
description "Identity for africa."
}

identity asia {
  base continent;
  description "Identity for asia."
}

identity europe {
  base continent;
  description "Identity for europe."
}

identity north-america {
  base continent;
  description "Identity for north-america."
}

identity south-america {
  base continent;
  description "Identity for south-america."
}

identity oceania {
  base continent;
  description "Identity for Oceania"
}

identity certificate-type {
  description
  "Base Identity for certificate-type. CRT certificate extension, which is used for certificates. The certificates may be encoded as binary DER or as ASCII PEM. The CER and CRT extensions are nearly synonymous. Most common among *nix systems. CER certificate extension, which is an alternate form of .crt (Microsoft Convention) You can use MS to convert .crt to .cer (.both DER encoded .cer, or base64[PEM] encoded .cer). The KEY extension is used both for public and private PKCS#8 keys. The keys may be encoded as binary DER or as ASCII PEM."
}

identity cer {
  base certificate-type;
  description "Identity for ".cer certificates."
}

identity crt {

base certificate-type;
description
"Identity for '.crt' certificates.";
}
identity key {
    base certificate-type;
description
"Identity for '.key' certificates.";
}

identity enforce-type {
    description
"This identity represents the event of
policy enforcement trigger type."
}
identity admin {
    base enforce-type;
description
"The identity for policy enforcement by admin."
}
identity time {
    base enforce-type;
description
"The identity for policy enforcement based on time."
}

identity protocol-type {
    description
"This identity represents the protocol types."
}
identity ftp {
    base protocol-type;
description
"The identity for ftp protocol."
}
identity ssh {
    base protocol-type;
description
"The identity for ssh protocol."
}
identity telnet {
    base protocol-type;
description
"The identity for telnet."
}
identity smtp {
    base protocol-type;
description
"The identity for smtp protocol."
}
"The identity for smtp."
}
identity sftp {
    base protocol-type;
    description "The identity for sftp."
}
identity http {
    base protocol-type;
    description "The identity for http."
}
identity https {
    base protocol-type;
    description "The identity for https."
}
identity pop3 {
    base protocol-type;
    description "The identity for pop3."
}
identity nat {
    base protocol-type;
    description "The identity for nat."
}

identity primary-action {
    description "This identity represents the primary actions, such as PASS, DROP, ALERT, RATE-LIMIT, and MIRROR."
}
identity pass {
    base primary-action;
    description "The identity for pass."
}
identity drop {
    base primary-action;
    description "The identity for drop."
}
identity alert {
    base primary-action;
    description "The identity for alert."
}
identity rate-limit {
    base primary-action;
    description
        "The identity for rate-limit.";
}

identity mirror {
    base primary-action;
    description
        "The identity for mirroring.";
}

identity secondary-action {
    description
        "This field identifies additional actions if a rule is
        matched. This could be one of 'LOG', 'SYSLOG', 'SESSION-LOG', etc.";
}

identity log {
    base secondary-action;
    description
        "The identity for logging.";
}

identity syslog {
    base secondary-action;
    description
        "The identity for system logging.";
}

identity session-log {
    base secondary-action;
    description
        "The identity for session logging.";
}

identity role-type {
    description
        "This is the base identity for the roles.";
}

identity user {
    base role-type;
    description
        "This represents the identity of the user role.";
}

identity group {
    base role-type;
    description
        "This represents the identity of any member of the
        security policy’s defined group.";
}
identity other {
    base role-type;
    description
    "This represents the identity of anyone else.”;
}

identity all {
    base role-type;
    description
    "This represents the identity of everyone
    (i.e., user, group, and other).”;
}

identity owner {
    description
    "This is the base identity for the owner";
}

identity dept-head {
    base owner;
    description
    "This represents the identity of the head of department.”;
}

identity manager {
    base owner;
    description
    "This represents the identity of the manager of the department.”;
}

identity employee {
    base owner;
    description
    "This represents the identity of department employees.”;
}

identity sec-head {
    base owner;
    description
    "This represents the identity of the head of security.”;
}

identity sec-admin {
    base owner;
    description
    "This represents the identity of security admin.”;
}

identity tenant-type {
    description
    "This is the base identity for the tenants
to represent the ownership of the security policies.”;
}

identity human-resources {
base tenant-type;
description
"This represents the identity of the marketing
department or division.";
}

identity customer-service {
  base tenant-type;
  description
  "This represents the identity of customer service
department or division.";
}

identity research {
  base tenant-type;
  description
  "This represents the identity of research
department or division.";
}

identity finance {
  base tenant-type;
  description
  "This represents the identity of finance
department or division.";
}

identity domain {
  description
  "This represents the base identity of different domains.";
}

identity enterprise {
  base domain;
  description
  "This represents the identity of an enterprise domain.";
}

identity signature-type {
  description
  "This represents the base identity for signature types.";
}

identity signature-yara {
  base signature-type;
  description
  "This represents the YARA signatures.";
}
identity signature-snort {
  base signature-type;
  description
    "This represents the SNORT signatures."
}

identity signature-suricata {
  base signature-type;
  description
    "This represents the SURICATA signatures."
}

identity threat-feed-type {
  description
    "This represents the base identity for threat-feed."
}

identity palo-alto {
  base threat-feed-type;
  description
    "This represents Palo-Alto threat-feed."
}

identity rsa-netwitness {
  base threat-feed-type;
  description
    "This represents RSA-netwitness threat-feed."
}

identity fireeye {
  base threat-feed-type;
  description
    "This represents FireEye threat-feed."
}

identity alienvault {
  base threat-feed-type;
  description
    "This represents Alienvault threat-feed."
}

identity auth-type {
  description
    "The base identity for authentication type."
}

identity auth-type-server {
  base auth-type;
  description
    "This represents the server authentication."
}

identity auth-type-client {
  base auth-type;
  description
    "This represents the client authentication."
}
"This represents the client authentication.";
}
identity auth-type-mutual {
    base auth-type;
    description
        "This represents the both server and client authentication.";
}

identity auth-method-type {
    description
        "Base identity for authentication-methods";
}

identity password-based {
    base auth-method-type;
    description
        "This is the identity for the password-based authentication type.";
}

identity token-based {
    base auth-method-type;
    description
        "This is the identity for the token-based authentication type.";
}

identity certificate-based {
    base auth-method-type;
    description
        "This is the identity for the certificate-based authentication type.";
}

/*
 * Groupings
 */

grouping ipv4-list {
    description
        "Grouping for ipv4 based ip-addresses.";
    leaf-list ipv4 {
        type inet:ipv4-address;
        description
            "This is the entry for the ipv4 ip-addresses.";
    }
}

grouping ipv6-list {
    description
        "Grouping for ipv6 based ip-addresses.";
    leaf-list ipv6 {

type inet:ipv6-address;
description
 "This is the entry for the ipv6 ip-addresses.";
}
}

grouping ipv4 {
description
 "Grouping for ipv4 based ip-address.";
leaf ipv4 {
 type inet:ipv4-address;
description
 "This is the entry for the ipv4 ip-address.";
}
}

grouping ipv6 {
description
 "Grouping for ipv6 based ip-address.";
leaf ipv6 {
 type inet:ipv6-address;
description
 "This is the entry for the ipv6 ip-address.";
}
}

grouping ip-address-info {
description
 "There are two types to configure a security policy for IPv4 address, such as exact match and range match.";
choice match-type {
 description
 "User can choose between 'exact match' and 'range match'.";
case exact-match-ipv4 {
 uses ipv4;
description
 "Exact ip-address match for ipv4 type addresses";
}
case exact-match-ipv6 {
 uses ipv6;
description
 "Exact ip-address match for ipv6 type addresses";
}
case range-match-ipv4 {
 list range-ipv4-address {
 key "start-ipv4-address end-ipv4-address";
 leaf start-ipv4-address {
 type inet:ipv4-address;
description
 "This is the entry for the ipv4 ip-address."
}
leaf end-ipv4-address {
 type inet:ipv4-address;
description
 "This is the entry for the ipv4 ip-address."
}
}
}
"Start IPv4 address for a range match.";
}
leaf end-ipv4-address {
  type inet:ipv4-address;
  description
    "End IPv4 address for a range match.";
}

description
  "Range match for an IP-address.";
}
}
}
case range-match-ipv6 {
  list range-ipv6-address {
    key "start-ipv6-address end-ipv6-address";
    leaf start-ipv6-address {
      type inet:ipv6-address;
      description
        "Start IPv6 address for a range match.";
    }
    leaf end-ipv6-address {
      type inet:ipv6-address;
      description
        "End IPv6 address for a range match.";
    }
    description
      "Range match for an IP-address.";
    }
  }
}
}
grouping password-based-method {
  list password-list {
    key "auth-method";
    leaf auth-method {
      type identityref {
        base auth-method-type;
      }
      description
        "This represents the authentication method is password-based.";
    }
    leaf password {
      type ianach:crypt-hash;
      description
        "The password for this entry.";
    }
    description
      "This represents the list of
encrypted passwords.;
}
}

grouping certificate-based-method {
  list cert-server-list {
    key "auth-method";
    description
    "This describes the certificate-based authentication list.";
    leaf auth-mthod {
      type identityref {
        base auth-method-type;
      }
      description
      "This represents the authentication method is certificate based method.";
    }
    leaf cert-server-name {
      type string;
      description
      "This field represents the name of the certificate-server name.";
    }
    leaf cert-server-ipv4 {
      type inet:ipv4-address;
      description
      "This represents ipv4 address of a certificate server.";
    }
    leaf cert-server-ipv6 {
      type inet:ipv6-address;
      description
      "This represents the ipv6 address of a certificate server.";
    }
  }
  list certificate {
    key "cert-type";
    description
    "This represents the certificate-types.";
    leaf cert-type {
      type identityref {
        base certificate-type;
      }
      description
      "This represents a certificate type.";
    }
  }
}
grouping token-based-method {
  list token-list {
    key "auth-method";
    description "This represents the list of tokens.";
    
    leaf auth-method {
      type identityref {
        base auth-method-type;
      }
      description "This represents the authentication type is token-based method.";
    }
    
    leaf token {
      type string;
      description "This object contains a string of a token.";
    }
    
    leaf token-server {
      type inet:ipv4-address;
      description "This represents the token-server information.";
    }
  }
}

grouping ipsec-based-method {
  list ipsec-method {
    key "method";
    description "This represents the list of IPsec method types.";
    
    leaf method {
      type identityref {
        base i2nsf-ipsec;
      }
      description "This represents IPsec IKE and IPsec IKEless cases.";
    }
  }
}

grouping user-group {
  description

"The grouping for user-group entities, and contains information such as name & ip-address."
leaf name {
    type string;
    description
        "This represents the name of a user.";
}
uses ip-address-info;
}


grouping device-group {
    description
        "This group represents device group information such as ip-address protocol."
leaf name {
    type string;
    description
        "This represents the name of a device.";
}
uses ip-address-info;
leaf-list protocol {
    type identityref {
        base protocol-type;
    }
    description
        "This represents the communication protocols of devices.";
}
}


grouping location-group {
    description
        "This group represents location-group information such as geo-ip and continent."
leaf name {
    type string;
    description
        "This represents the name of a location.";
}
leaf geo-ip-ipv4 {
    type inet:ipv4-address;
    description
        "This represents the IPv4 geo-ip of a location.";
}
leaf geo-ip-ipv6 {
    type inet:ipv6-address;
    description
        "This represents the IPv6 geo-ip of a location.";
}
leaf continent {
    type identityref {
        base continent;
    }
    description
        "location-group-based on geo-ip of respective continent."
}

grouping threat-feed-info {
    description
        "This is the grouping for the threat-feed-list";
    leaf feed-name {
        type identityref {
            base threat-feed-type;
        }
        description
            "This represents the name of the a threat-feed.";
    }
    leaf feed-server-ipv4 {
        type inet:ipv4-address;
        description
            "The IPv4 ip-address for the threat-feed server.";
    }
    leaf feed-server-ipv6 {
        type inet:ipv6-address;
        description
            "The IPv6 ip-address for the threat-feed server.";
    }
    leaf feed-description {
        type string;
        description
            "This represents the descriptions of a threat-feed. The description should include information, such as the type, related threat, method, and file type.";
    }
}

grouping payload-string {
    description
        "The grouping for payload-string content. It contains information such as name and string content.";
    leaf payload-description {
        type string;
        description
            "This represents the description of a payload.";
    }
}
leaf-list content {
  type string;
  description
  "This represents the payload string content.";
}

list i2nsf-cfi-policy {
  key "policy-name";
  description
  "This is the security policy list. Each policy in the list contains a list of security rules, and is a policy instance to have complete information such as where and when a policy needs to be applied.";
  leaf policy-name {
    type string;
    mandatory true;
    description
    "The name which identifies the policy.";
  }
  list rule {
    leaf rule-name {
      type string;
      mandatory true;
      description
      "This represents the name for rules.";
    }
    key "rule-name";
    description
    "There can be a single or multiple number of rules.";
    container event {
      description
      "This represents the event (e.g., a security event, which a security rule is made for.";
      leaf security-event {
        type identityref {
          base security-event-type;
        }
        mandatory true;
        description
        "This contains the description of security events.";
      }
      choice enforce-type {
        description
        "There are three different enforcement types; admin, and time.";
        case enforce-admin {
          
        }
      }
    }
  }
}
leaf admin {
    type identityref {
        base enforce-type;
    }
    description "This represents the enforcement type based on admin’s decision.";
}
}
case time {
    container time-information {
        description "The begin-time and end-time information when the security rule should be applied.";
        leaf enforce-time {
            type identityref {
                base enforce-type;
            }
            description "The enforcement type is time-enforced.";
        }
        leaf begin-time {
            type yang:date-and-time;
            description "This is start time for time zone";
        }
        leaf end-time {
            type yang:date-and-time;
            description "This is end time for time zone";
        }
    }
}
leaf frequency {
    type enumeration {
        enum only-once {
            description "This represents the rule is enforced only once.";
        }
        enum daily {
            description "This represents the rule is enforced on a daily basis.";
        }
        enum weekly {
            description "This represents the rule is enforced on a weekly basis.";
        }
        enum monthly {

description
  "This represents the rule is enforced on a monthly basis.";
}
}
default only-once;
description
  "This represents how frequent the rule should be enforced.";
}
}
container condition {
  choice condition {
    description
    "The conditions for general security policies.";
    case firewall-condition {
      description
      "The general firewall condition.";
      container firewall-source {
        description
        "This represents the source.";
        leaf src-target {
          type leafref {
            path "/i2nsf-cfi-policy/endpoint-group/user-group/name";
          }
          mandatory true;
          description
          "This describes the paths to the source reference.";
        }
      }

      container firewall-destination {
        description
        "This represents the destination.";
        leaf-list dest-target {
          type leafref {
            path "/i2nsf-cfi-policy/endpoint-group/user-group/name";
          }
          description
          "This describes the paths to the destination target reference.";
        }
      }
    }
    case ddos-condition {
      description
      "The condition for DDoS mitigation.";
      container ddos-source {
        description
        "This represents the source.";
      }
    }
  }
}
leaf-list src-target {
  type leafref {
    path "/i2nsf-cfi-policy/endpoint-group/device-group/name";
  }
  description
    "This describes the path to the source target references.";
}

container ddos-destination {
  description
    "This represents the target.";
  leaf-list dest-target {
    type leafref {
      path "/i2nsf-cfi-policy/endpoint-group/device-group/name";
    }
    description
      "This describes the path to the destination target references.";
  }
}

container rate-limit {
  description "This describes the rate-limit.";
  leaf packet-per-second {
    type uint16;
    description
      "The rate-limit limits the amount of incoming packets.";
  }
}

case custom-condition {
  description
    "The condition based on packet contents.";
  container custom-source {
    description
      "This represents the source.";
    leaf-list src-target {
      type leafref {
        path "/i2nsf-cfi-policy/threat-prevention/payload-content/name";
      }
      description
        "Describes the payload string content condition source.";
    }
  }
  container custom-destination {
    description
      "This represents the destination.";
  }
}
leaf dest-target {
  type leafref {
    path "/i2nsf-cfi-policy/threat-prevention/payload-content/name";
  }
  mandatory true;
  description  "Describes the payload string content condition destination.";
}
}
case threat-feed-condition {
  description  "The condition based on the threat-feed information.";
  container threat-feed-source {
    description  "This represents the source.";
    leaf-list src-target {
      type leafref {
        path "/i2nsf-cfi-policy/threat-prevention/threat-feed-list/feed-name";
      }
      description  "Describes the threat-feed condition source.";
    }
  }
  container threat-feed-destination {
    description  "This represents the destination.";
    leaf dest-target {
      type leafref {
        path "/i2nsf-cfi-policy/threat-prevention/threat-feed-list/feed-name";
      }
      mandatory true;
      description  "Describes the threat-feed condition destination.";
    }
  }
}
container action {
  description  "This is the action container.";
  leaf primary-action {
    type identityref {
      base primary-action;
    }
    mandatory true;
  }
}
container ipsec-method {
  description
  "This container represents the IPsec IKE and IKEless cases.";
  leaf method {
    type leafref {
      path "/i2nsf-cfi-policy/multi-tenancy/policy-mgmt-auth-method-instance/ipsec-method/method";
    }
    description
    "This references the IPsec method types, which includes IPsec IKE and IPsec IKEless cases.";
  }
  leaf owner {
    type identityref {
      base owner;
    }
    mandatory true;
    description
    "This field defines the owner of this rule. Only the owner is authorized to modify the contents of the rule.";
  }
}

container multi-tenancy {
  description
  "The multi-tenant environment information in which the policy is applied. The Rules in the Policy can refer to sub-objects (e.g., domain, tenant, role, and user) of it.";
  list policy-domain {
    key "domain-name";
    description
    "This represents the list of policy domains.";
  }
}
leaf domain-name {
    type identityref {
        base domain;
    }
    description "This represents the name of a domain.";
}
leaf address {
    type string;
    description "The address details of the organization or customer.";
}
leaf contact {
    type string;
    description "Contact information of the organization or customer.";
}
list policy-tenant {
    key "tenant-type";
    description "This field identifies the domain to which this tenant belongs. This should be reference to a 'Policy-Domain' object.";
    leaf tenant-type{
        type identityref {
            base tenant-type;
        }
        description "The name of the tenant, such as HR or Finance department.";
    }
}
list policy-role {
    key "role-type";
    description "This represent the roles within the tenants, in order to distinguish who may or may not have access to policies.";
    leaf role-type {
        type identityref {
            base role-type;
        }
        description "This represents the name of the role";
    }
}
list policy-user {
key "name";
description
"This represents the list of policy users.";

leaf name {
  type string;
  description
  "This represents the name of the user";
}
leaf password {
  type ianach:crypt-hash;
  description
  "User password for basic authentication";
}
leaf email {
  type string;
  description
  "The email account of a user";
}

list access-profile {
  key "permission-type scope-type";
  description
  "This field identifies the access profile for the role. The profile grants or denies access to policy objects.";
  leaf permission-type {
    type identityref {
      base permission-type;
    }
    description
    "This represents the permission types, such as read, write, execute, read-and-write, and etc.";
  }
  leaf scope-type {
    type identityref {
      base scope-type;
    }
    description
    "identifies whether a user has domain-wide or tenant-wide privileges";
  }
}

list policy-mgmt-auth-method-instance {
  key "auth-instance-type";
description
"This represents the list of instances for
policy management authentication methods.";

leaf auth-instance-type {
  type identityref {
    base auth-type;
  }
  description
  "This identifies whether the authentication type
  is server authentication, client authentication,
  or both.";
}

choice policy-mgmt-auth-method {
  description
  "This represents the choices for which
  authentication method is used.";
  case password-based {
    uses password-based-method;
  }
  case token-based {
    description
    "This represents the token-based method.";
    uses token-based-method;
  }
  case certificate-based {
    description
    "This represents the certificate-based-method.";
    uses certificate-based-method;
  }
  case ipsec {
    description
    "This represents authentication method based on IPSEC.";
    uses ipsec-based-method;
  }
  }
}

container endpoint-group {
  description
  "A logical entity in their business
  environment, where a security policy
  is to be applied.";
  list user-group {
    key "name";
    uses user-group;
    description
    "This represents the user group.";
  }
}
list device-group {
  key "name";
  uses device-group;
  description
      "This represents the device group.";
}
list location-group{
  key "name";
  uses location-group;
  description
      "This represents the location group.";
}

container threat-prevention {
  description
    "This describes the list of threat-prevention.";

  list threat-feed-list {
    key "feed-name";
    description
        "This represents the threat feed list.";
    uses threat-feed-info;

    leaf-list threat-file-types {
      type identityref {
        base malware-file-type;
      }
      default executable-file;
      description
        "This contains a list of file types needed to be scanned for the virus.";
    }

    leaf-list signatures {
      type identityref {
        base signature-type;
      }
      default signature-suricata;
      description
        "This contains a list of signatures or hash of the threats.";
    }
  }

  list payload-content {
    key "name";
    leaf name {
      type string;
      description
    }
  }
}
"This represents the name of payload-content".
It should give an idea of why specific payload content is marked as threat. For example, the name "backdoor" indicates the payload content is related to backdoor attack.;
}
description
"This represents the payload-string group.";
uses payload-string;
}
}
<CODE ENDS>

Figure 23: YANG for Consumer-Facing Interface

10. Example XML Output for Various Scenarios

This section describes the XML instances for different policies examples that are delivered through Consumer-Facing Interface. The considered use cases are: VoIP/VoLTE security service, DDoS-attack mitigation, time-based firewall as a web-filter.

10.1. DB Registration: Information of Positions and Devices (Endpoint Group)

If new endpoints are introduced to the network, it is necessary to first register their data to the database. For example, if new members are newly introduced in either of three different groups (i.e., user-group, device-group, and payload-group), each of them should be registered with information such as ip-addresses or protocols used by devices. Figure 24 shows an example XML representation of the registered information for the user-group and device-group.
<?xml version="1.0" encoding="UTF-8" ?>
<endpoint-group xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-cfi-policy">
  <user-group>
    <name>employees</name>
    <range-ip-address>
      <start-ip-address>221.159.112.1</start-ip-address>
      <end-ip-address>221.159.112.90</end-ip-address>
    </range-ip-address>
  </user-group>
  <device-group>
    <name>webservers</name>
    <range-ip-address>
      <start-ip-address>221.159.112.91</start-ip-address>
      <end-ip-address>221.159.112.97</end-ip-address>
    </range-ip-address>
    <protocol>http</protocol>
    <protocol>https</protocol>
  </device-group>
</endpoint-group xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-cfi-policy">

Figure 24: Registering User-group and Device-group Information

10.2. Scenario 1: Block SNS Access during Business Hours

The first example scenario is to "block SNS access during business hours" using a time-based firewall policy. In this scenario, all users registered as "employee" in the user-group list are unable to access Social Networking Services (SNS) during the office hours. The XML instance is described below:
<policy xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-cfi-policy">
  <policy-name>security_policy_for_blocking_sns</policy-name>
  <rule>
    <rule-name>block_access_to_sns_during_office_hours</rule-name>
    <event>
      <time-information>
        <begin-time>09:00</begin-time>
        <end-time>18:00</end-time>
      </time-information>
    </event>
    <condition>
      <firewall-condition>
        <source-target>
          <src-target>employees</src-target>
        </source-target>
        <destination-target>
          <dest-target>sns-websites</dest-target>
        </destination-target>
      </firewall-condition>
      <custom-condition>
        <primary-action>drop</primary-action>
      </custom-condition>
    </condition>
    <action>
      <ipsec-method>
        <method>ipsec-ike</method>
      </ipsec-method>
    </action>
  </rule>
</policy>

Figure 25: An XML Example for Time-based Firewall

Time-based-condition Firewall

1. The policy name is "security_policy_for_blocking_sns".
2. The rule name is "block_access_to_sns_during_office_hours".
3. The Source-target is "employees".
4. The destination target is "sns-websites". "sns-websites" is the key which represents the list containing the information, such as URL, about sns-websites.
5. The action required is to "drop" any attempt to connect to websites related to Social networking.
6. The IPsec method type used for nsf traffic steering is set to "ipsec-ike".

10.3. Scenario 2: Block Malicious VoIP/VoLTE Packets Coming to a Company

The second example scenario is to "block malicious VoIP/VoLTE packets coming to a company" using a VoIP policy. In this scenario, the calls coming from VoIP and/or VOLTE sources with VOLTE IDs that are classified as malicious are dropped. The IP addresses of the employees and malicious VoIP IDs should be blocked. They are stored in the database or datastore of the enterprise. Here and the rest of the cases assume that the security administrators or someone responsible for the existing and newly generated policies, are not aware of which and/or how many NSFs are needed to meet the security requirements. Figure 26 represents the XML document generated from YANG discussed in previous sections. Once a high-level security policy is created by a security admin, it is delivered by the Consumer-Facing Interface, through RESTCONF server, to the security controller. The XML instance is described below:

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<policy xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-cfi-policy">
  <policy-name>security_policy_for_blocking_malicious_voip_packets</policy-name>
  <rule>
    <rule-name>Block_malicious_voip_and_volte_packets</rule-name>
    <condition>
      <custom-condition>
        <source-target>malicious-id</source-target>
      </custom-condition>
      <firewall-condition>
        <destination-target>employees</destination-target>
      </firewall-condition>
    </condition>
    <action>
      <primary-action>drop</primary-action>
    </action>
    <ipsec-method>
      <method>ipsec-ikeless</method>
    </ipsec-method>
  </rule>
</policy xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-cfi-policy"
```

Figure 26: An XML Example for VoIP Security Service
Custom-condition Firewall

1. The policy name is "security_policy_for_blocking_malicious_voip_packets".

2. The rule name is "Block_malicious_voip_and_volte_packets".

3. The Source-target is "malicious-id". This can be a single ID or a list of IDs, depending on how the ID are stored in the database. The "malicious-id" is the key so that the security admin can read every stored malicious VOIP IDs that are named as "malicious-id".

4. The destination target is "employees". "employees" is the key which represents the list containing information about employees, such as IP addresses.

5. The action required is "drop" when any incoming packets are from "malicious-id".

6. The IPsec method used for nsf traffic steering is set to "ipsec-ikeless".

10.4. Scenario 3: Mitigate HTTP and HTTPS Flood Attacks on a Company Web Server

The third example scenario is to "Mitigate HTTP and HTTPS flood attacks on a company web server" using a DDoS-attack mitigation policy. Here, the time information is not set because the service provided by the network should be maintained at all times. If the packets sent by any sources are more than the set threshold, then the admin can set the percentage of the packets to be dropped to safely maintain the service. In this scenario, the source is set as "any" to block any sources which send abnormal amount of packets. The destination is set as "web_server01". Once the rule is set and delivered and enforced to the nsfs by the security controller, the NSF's will monitor the incoming packet amounts and the destination to act according to the rule set. The XML instance is described below:
<?xml version="1.0" encoding="UTF-8" ?>
<policy xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-cfi-policy">
  <policy-name>security_policy_for_ddos_attacks</policy-name>
  <rule>
    <rule-name>100_packets_per_second</rule-name>
    <condition>
      <ddos-condition>
        <destination-target>
          <dest-target>webservers</dest-target>
        </destination-target>
        <rate-limit>
          <packet-per-second>100</packet-per-second>
        </rate-limit>
      </ddos-condition>
    </condition>
    <action>
      <primary-action>drop</primary-action>
    </action>
    <ipsec-method>
      <method>ipsec-ikeless</method>
    </ipsec-method>
  </rule>
</policy>

Figure 27: An XML Example for DDoS-attack Mitigation

1. The policy name is "security_policy_for_ddos_attacks".
2. The rule name is "100_packets_per_second".
3. The destination target is "webservers". "webservers" is the key which represents the list containing information, such as IP addresses and ports, about web-servers.
4. The rate limit exists to limit the incoming amount of packets per second. In this case the rate limit is "100" packets per second. This amount depends on the packet receiving capacity of the server devices.
5. The Source-target is all sources which send abnormal amount of packets.
6. The action required is to "drop" packet reception is more than 100 packets per second.
7. The IPsec method used for nsf traffic steering is set to "ipsec-ike".

11. Security Considerations

The data model for the I2NSF Consumer-Facing Interface is based on the I2NSF framework [RFC8329], so the same security considerations with the I2NSF framework should be included in this document. The data model needs a secure communication channel to protect the Consumer-Facing Interface between the I2NSF User and Security Controller.

12. IANA Considerations

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

   Registrant Contact: The I2NSF.
   XML: N/A; the requested URI is an XML namespace.

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

   name: ietf-i2nsf-cfi-policy
   prefix: cfi-policy
   reference: RFC 7950

13. References

13.1. Normative References


13.2. Informative References

[client-facing-inf-req]

[i2nsf-capability-im]
[i2nsf-ipsec]

[i2nsf-terminology]
Appendix A. Changes from draft-ietf-i2nsf-consumer-facing-interface-dm-05

The following are major changes made from draft-ietf-i2nsf-consumer-facing-interface-dm-05:

- The container policy-mgmt-auth-method uses a list, and the policy-mgmt-auth-method consists of choice-cases.

- Policy-role is changed from container to list. The access-profile in the policy-role is not removed. Instead, it is placed inside policy-user.

- Container Condition consists of choice-cases to show that it is capable of configuring different triggering conditions.

- The enforce-type in Event container use a choice-case statement. This change shows the clarity that the enforce-type is relevant to each case (i.e., enforce-type == admin or time).

- The name for container "recursive" is changed to "frequency". This container represents how frequently the rule is enforced, so the name "frequency" is more appropriate.

- The certificate based authentication method is modified so that a certificate server can handle more than one (list) of certificate types.

The minor changes are as follows:

- Typos are corrected.

- IPv6 as well as IPv4 are included.

- Some misused types are corrected (e.g., enum -> identity)

- Some descriptions that are unclear, mistaken, or shortly explained are rewritten.

Appendix B. 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).
Appendix C. Contributors

This document is made by the group effort of I2NSF working group. Many people actively contributed to this document, such as Mahdi F. Dachmehchi and Daeyoung Hyun. The authors sincerely appreciate their contributions.

The following are co-authors of this document:

Hyoungshick Kim
Department of Computer Science and Engineering
Sungkyunkwan University
2066 Seo-ro Jangan-gu
Suwon, Gyeonggi-do 16419
Republic of Korea
EMail: hyoung@skku.edu

Seungjin Lee
Department of Electronic, Electrical and Computer Engineering
Sungkyunkwan University
2066 Seo-ro Jangan-gu
Suwon, Gyeonggi-do 16419
Republic of Korea
EMail: jine33@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

Anil Lohiya
Juniper Networks
1133 Innovation Way
Sunnyvale, CA 94089
US
EMail: alohiya@juniper.net
Susan Hares
Huawei
7453 Hickory Hill
Saline, MI 48176
USA

Phone: +1-734-604-0332
EMail: shares@ndzh.com
I2NSF Working Group                                               J. Kim
Internet-Draft                                                  J. Jeong
Intended status: Standards Track                 Sungkyunkwan University
Expires: January 26, 2020                                        J. Park
ETRI                                                    ETRI
S. Hares                                                      S. Hares
Q. Lin                                                       Q. Lin
Huawei                                                      Huawei
July 25, 2019

I2NSF Network Security Function-Facing Interface YANG Data Model
draft-ietf-i2nsf-nsf-facing-interface-dm-07

Abstract

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

Status of This Memo

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

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

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

This Internet-Draft will expire on January 26, 2020.

Copyright Notice

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

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

This document defines a YANG [RFC6020][RFC7950] data model for security policy rule configuration of Network Security Functions (NSF). The YANG data model corresponds to the information model [draft-ietf-i2nsf-capability] for NSF-Facing Interface in Interface to Network Security Functions (I2NSF). The YANG data model in this document focuses on security policy configuration for generic network security functions. Note that security policy configuration for
advanced network security functions are defined in [draft-dong-i2nsf-asf-config].

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

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

- Configuration of general security policy rule for generic network security functions.
- Configuration of event clause for generic network security functions.
- Configuration of condition clause for generic network security functions.
- Configuration of action clause for generic network security functions.

2. Requirements Language

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

3. Terminology

This document uses the terminology described in [draft-ietf-i2nsf-capability][RFC8431][draft-ietf-supageneric-policy-info-model]. Especially, the following terms are from [draft-ietf-supageneric-policy-info-model]:

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

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

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

4. YANG Tree Diagram

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

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


This section shows the YANG tree diagram for general I2NSF security policy rules.
module: ietf-i2nsf-policy-rule-for-nsf
    +++-rw i2nsf-security-policy
        +++-rw system-policy* [system-policy-name]
            ++-rw system-policy-name string
            +++-rw priority-usage? identityref
            +++-rw resolution-strategy? identityref
            +++-rw default-action? identityref
            +++-rw rules* [rule-name]
                ++-rw rule-name string
                +++-rw rule-description? string
                +++-rw rule-priority? uint8
                +++-rw rule-enable? boolean
                +++-rw rule-session-aging-time? uint16
                +++-rw rule-long-connection
                    ++-rw enable? boolean
                    +++-rw during? uint16
                +++-rw time-intervals
                    +++-rw absolute-time-interval
                        ++-rw start-time? start-time-type
                        ++-rw end-time? end-time-type
                    +++-rw periodic-time-interval
                        ++-rw day
                            ++-rw every-day? boolean
                            +++-rw specific-day* day-type
                        ++-rw month
                            ++-rw every-month? boolean
                            +++-rw specific-month* month-type
                    +++-rw event-clause-container
                        ...
                    +++-rw condition-clause-container
                        ...
                    +++-rw action-clause-container
                        ...
                +++-rw rule-group
                    +++-rw groups* [group-name]
                        ++-rw group-name string
                        +++-rw rule-range
                            ++-rw start-rule? string
                            ++-rw end-rule? string
                        +++-rw enable? boolean
                        +++-rw description? string
                    +++-rw i2nsf-ipsec? identityref

Figure 1: YANG Tree Diagram for Network Security Policy
This YANG tree diagram shows the general I2NSF security policy rule for generic network security functions.

The system policy provides for multiple system policies in one NSF, and each system policy is used by one virtual instance of the NSF/device. The system policy includes system policy name, priority usage, resolution strategy, default action, and rules.

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

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

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

4.2. Event Clause

This section shows the YANG tree diagram for an event clause for I2NSF security policy rules.
Figure 2: YANG Tree Diagram for an Event Clause

This YANG tree diagram shows an event clause of an I2NSF security policy rule for generic network security functions. An event clause is any important occurrence at a specific time of a change in the system being managed, and/or in the environment of the system being managed. An event clause is used to trigger the evaluation of the condition clause of the I2NSF Policy Rule. The event clause is defined as a system event and system alarm. The event clause can be extended according to specific vendor event features. The event clause is described in detail in [draft-ietf-i2nsf-capability].

4.3. Condition Clause

This section shows the YANG tree diagram for a condition clause of I2NSF security policy rules.
+--rw pkt-sec-ipv4-header-length
  +--rw (match-type)?
    | +--rw ipv4-header-length*  uint8
    | +--:(range-match)
    | +--rw range-ipv4-header-length*
  [start-ipv4-header-length end-ipv4-header-length]
  +--rw start-ipv4-header-length  uint8
  +--rw end-ipv4-header-length  uint8
+--rw pkt-sec-ipv4-tos*  identityref
+--rw pkt-sec-ipv4-total-length
  +--rw (match-type)?
    | +--rw ipv4-total-length*  uint16
    | +--:(range-match)
    | +--rw range-ipv4-total-length*
  [start-ipv4-total-length end-ipv4-total-length]
  +--rw start-ipv4-total-length  uint16
  +--rw end-ipv4-total-length  uint16
+--rw pkt-sec-ipv4-id*  uint16
+--rw pkt-sec-ipv4-fragment-flags*  identityref
+--rw pkt-sec-ipv4-fragment-offset
  +--rw (match-type)?
    | +--rw ipv4-fragment-offset*  uint16
    | +--:(range-match)
    | +--rw range-ipv4-fragment-offset*
  [start-ipv4-fragment-offset end-ipv4-fragment-offset]
  +--rw start-ipv4-fragment-offset  uint16
  +--rw end-ipv4-fragment-offset  uint16
+--rw pkt-sec-ipv4-ttl
  +--rw (match-type)?
    | +--rw ipv4-ttl*  uint8
    | +--:(range-match)
    | +--rw range-ipv4-ttl*
  [start-ipv4-ttl end-ipv4-ttl]
  +--rw start-ipv4-ttl  uint8
  +--rw end-ipv4-ttl  uint8
+--rw pkt-sec-ipv4-protocol*  identityref
+--rw pkt-sec-ipv4-src
  +--rw (match-type)?
    | +--rw ipv4* [ipv4]
    | +--rw ipv4  inet:ipv4-address
    | +--rw (subnet)?
      | +--:(prefix-length)
      | +--rw prefix-length?  uint8
Internet-Draft   NSF-Facing Interface YANG Data Model         July 2019

[81x742]Internet-Draft    NSF-Facing Interface YANG Data Model         July 2019

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

This YANG tree diagram shows a condition clause for an I2NSF security policy rule for generic network security functions. A condition clause is defined as a set of attributes, features, and/or values that are to be compared with a set of known attributes, features, and/or values in order to determine whether or not the set of actions in that (imperative) I2NSF policy rule can be executed or not. A condition clause is classified as a conditions of generic network security functions, advanced network security functions, or context. A condition clause of generic network security functions is defined as packet security IPv4 condition, packet security IPv6 condition, packet security tcp condition, and packet security icmp condition. A condition clause of advanced network security functions is defined as packet security url category condition, packet security voice condition, packet security DDoS condition, or packet security payload condition. A condition clause of context is defined as ACL number condition, application condition, target condition, user condition, and geography condition. Note that this document deals only with simple conditions of advanced network security functions. A condition clauses of advanced network security functions are described in detail in [draft-dong-i2nsf-asf-config]. A condition clause can be extended according to specific vendor condition.
features. A condition clause is described in detail in [draft-ietf-i2nsf-capability].

4.4. Action Clause

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

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

Figure 4: YANG Tree Diagram for an Action Clause

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

This section shows the YANG tree diagram for an I2NSF IPsec.

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

Figure 5: YANG Tree Diagram for I2NSF Internet Key Exchange

This YANG tree diagram shows an I2NSF IPsec specification for an Internet Key Exchange IKE). An I2NSF IPsec specification is used to define a method required to manage IPsec parameters for creating IPsec Security Associations (SAs) between two NSFs through either the IKEv2 protocol or the Security Controller [draft-ietf-i2nsf-sdn-ipsec-flow-protection]. I2NSF IPsec considers two cases, the IKE case (i.e., IPsec through IKE) and IKE-less case (i.e., IPsec not through IKE, but through a Security Controller). Refer to [draft-ietf-i2nsf-sdn-ipsec-flow-protection] for the detailed description of the I2NSF IPsec.

5. YANG Data Module

5.1. I2NSF NSF-Facing Interface YANG Data Module

This section contains a YANG data module for configuration of security policy rules on network security functions.

```yang
<CODE BEGINS> file "ietf-i2nsf-policy-rule-for-nsf@2019-07-25.yang"

module ietf-i2nsf-policy-rule-for-nsf {
  yang-version 1.1;
  namespace
  prefix
```

nsfintf;

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

organization
    "IETF I2NSF (Interface to Network Security Functions)
     Working Group";

contact
    "WG Web: <http://tools.ietf.org/wg/i2nsf>
     WG List: <mailto:i2nsf@ietf.org>
     WG Chair: Linda Dunbar
     <mailto:ldunbar@futurewei.com>
     WG Chair: Yoav Nir
     <mailto:ynir.ietf@gmail.com>
     Editor: Jingyong Tim Kim
     <mailto:timkim@skku.edu>
     Editor: Jaehoon Paul Jeong
     <mailto:pauljeong@skku.edu>
     Editor: Susan Hares
     <mailto:shares@ndzh.com>"

description
    "This module defines a YANG data module for the Network Security
     Functions (NSF) facing interface.

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

    Redistribution and use in source and binary forms, with or
     without modification, is permitted pursuant to, and subject
     to the license terms contained in, the Simplified BSD License
     set forth in Section 4.c of the IETF Trust’s Legal Provisions
     Relating to IETF Documents
     (http://trustee.ietf.org/license-info)."
revision "2019-07-25"{
description "Initial revision.";
reference
  "RFC XXXX: I2NSF Network Security Function-Facing Interface
   YANG Data Model";
}
/*
 * Identities
 */

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

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

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

identity event {
  description
    "Base identity for policy events";
  reference
    "draft-ietf-i2nsf-nsf-monitoring-data-model-01
     - Event";
}

identity system-event {
  base event;
  description
    "Identity for system events";
  reference
    "draft-ietf-i2nsf-nsf-monitoring-data-model-01
     - System event";
}
identity system-alarm {
    base event;
    description
      "Identity for system alarms";
    reference
      "draft-ietf-i2nsf-nsf-monitoring-data-model-01
          - System alarm";
}

identity access-violation {
    base system-event;
    description
      "Identity for access violation system events";
    reference
      "draft-ietf-i2nsf-nsf-monitoring-data-model-01
          - System event";
}

identity configuration-change {
    base system-event;
    description
      "Identity for configuration change system events";
    reference
      "draft-ietf-i2nsf-nsf-monitoring-data-model-01
          - System event";
}

identity memory-alarm {
    base system-alarm;
    description
      "Identity for memory alarm system alarms";
    reference
      "draft-ietf-i2nsf-nsf-monitoring-data-model-01
          - System alarm";
}

identity cpu-alarm {
    base system-alarm;
    description
      "Identity for CPU alarm system alarms";
    reference
      "draft-ietf-i2nsf-nsf-monitoring-data-model-01
          - System alarm";
}
identity disk-alarm {
  base system-alarm;
  description "Identity for disk alarm system alarms";
  reference "draft-ietf-i2nsf-nsf-monitoring-data-model-01 - System alarm";
}

identity hardware-alarm {
  base system-alarm;
  description "Identity for hardware alarm system alarms";
  reference "draft-ietf-i2nsf-nsf-monitoring-data-model-01 - System alarm";
}

identity interface-alarm {
  base system-alarm;
  description "Identity for interface alarm system alarms";
  reference "draft-ietf-i2nsf-nsf-monitoring-data-model-01 - System alarm";
}

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

identity traffic-class {
  description "Base identity for traffic-class of IPv6";
}

identity normal {
  base type-of-service;
  base traffic-class;
}
description
   "Identity for normal IPv4 TOS and IPv6 Traffic Class";
reference
   "RFC 791: Internet Protocol - Type of Service
RFC 8200: Internet Protocol, Version 6 (IPv6)
Specification - Traffic Class";
}

identity minimize-cost {
    base type-of-service;
    base traffic-class;
    description
       "Identity for ‘minimize monetary cost’ IPv4 TOS and
IPv6 Traffic Class";
    reference
       "RFC 791: Internet Protocol - Type of Service
RFC 8200: Internet Protocol, Version 6 (IPv6)
Specification - Traffic Class";
}

identity maximize-reliability {
    base type-of-service;
    base traffic-class;
    description
       "Identity for ‘maximize reliability’ IPv4 TOS and
IPv6 Traffic Class";
    reference
       "RFC 791: Internet Protocol - Type of Service
RFC 8200: Internet Protocol, Version 6 (IPv6)
Specification - Traffic Class";
}

identity maximize-throughput {
    base type-of-service;
    base traffic-class;
    description
       "Identity for ‘maximize throughput’ IPv4 TOS and
IPv6 Traffic Class";
    reference
       "RFC 791: Internet Protocol - Type of Service
RFC 8200: Internet Protocol, Version 6 (IPv6)
Specification - Traffic Class";
}

identity minimize-delay {
    base type-of-service;
    base traffic-class;
    description

"Identity for ‘minimize delay’ IPv4 TOS and IPv6 Traffic Class”;
reference
"RFC 791: Internet Protocol - Type of Service
}

identity maximize-security {
    base type-of-service;
    base traffic-class;
    description
    "Identity for ‘maximize security’ IPv4 TOS and IPv6 Traffic Class”;
    reference
    "RFC 791: Internet Protocol - Type of Service
}

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

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

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

identity reserved {
    base fragmentation-flags-type;
    description
    "Identity for reserved flags”;
    reference
    "RFC 791: Internet Protocol - Fragmentation Flags”;
}
identity protocol {
  description "Base identity for protocol of IPv4";
  reference "RFC 790: Assigned numbers - Assigned Internet Protocol Number
            RFC 791: Internet Protocol - Protocol";
}

identity next-header {
  description "Base identity for IPv6 next header";
}

identity icmp {
  base protocol;
  base next-header;
  description "Identity for ICMP IPv4 protocol and IPv6 nett header";
  reference "RFC 790: - Assigned numbers - Assigned Internet Protocol Number
            RFC 791: Internet Protocol - Protocol
}

identity igmp {
  base protocol;
  base next-header;
  description "Identity for IGMP IPv4 protocol and IPv6 next header";
  reference "RFC 790: - Assigned numbers - Assigned Internet Protocol Number
            RFC 791: Internet Protocol - Protocol
}
identity tcp {
    base protocol;
    base next-header;
    description
        "Identity for TCP protocol";
    reference
        "RFC 790: - Assigned numbers - Assigned Internet Protocol Number
RFC 791: Internet Protocol - Protocol
}

identity igrp {
    base protocol;
    base next-header;
    description
        "Identity for IGRP IPv4 protocol and IPv6 next header";
    reference
        "RFC 790: - Assigned numbers - Assigned Internet Protocol Number
RFC 791: Internet Protocol - Protocol
}

identity udp {
    base protocol;
    base next-header;
    description
        "Identity for UDP IPv4 protocol and IPv6 next header";
    reference
        "RFC 790: - Assigned numbers - Assigned Internet Protocol Number
RFC 791: Internet Protocol - Protocol
}

identity gre {
    base protocol;
    base next-header;
    description
        "Identity for GRE IPv4 protocol and IPv6 next header";
    reference

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

identity esp {
  base protocol;
  base next-header;
  description
    "Identity for ESP IPv4 protocol and IPv6 next header";
  reference
    "RFC 790: - Assigned numbers - Assigned Internet Protocol Number
RFC 791: Internet Protocol - Protocol
}

identity ah {
  base protocol;
  base next-header;
  description
    "Identity for AH IPv4 protocol and IPv6 next header";
  reference
    "RFC 790: - Assigned numbers - Assigned Internet Protocol Number
RFC 791: Internet Protocol - Protocol
}

identity mobile {
  base protocol;
  base next-header;
  description
    "Identity for mobile IPv4 protocol and IPv6 next header";
  reference
    "RFC 790: - Assigned numbers - Assigned Internet Protocol Number
RFC 791: Internet Protocol - Protocol
}
identity tlsp {
  base protocol;
  base next-header;
  description
    "Identity for TLSP IPv4 protocol
     and IPv6 next header";
  reference
    "RFC 790: - Assigned numbers - Assigned Internet
     Protocol Number
    RFC 791: Internet Protocol - Protocol
    RFC 8200: Internet Protocol, Version 6 (IPv6)
     Specification - Next Header";
}

identity skip {
  base protocol;
  base next-header;
  description
    "Identity for skip IPv4 protocol
     and IPv6 next header";
  reference
    "RFC 790: - Assigned numbers - Assigned Internet
     Protocol Number
    RFC 791: Internet Protocol - Protocol
    RFC 8200: Internet Protocol, Version 6 (IPv6)
     Specification - Next Header";
}

identity ipv6-icmp {
  base protocol;
  base next-header;
  description
    "Identity for IPv6 ICMP next header";
  reference
    "RFC 790: - Assigned numbers - Assigned Internet
     Protocol Number
    RFC 791: Internet Protocol - Protocol
    RFC 8200: Internet Protocol, Version 6 (IPv6)
     Specification - Next Header";
}

identity eigrp {
  base protocol;
  base next-header;
  description
    "Identity for EIGRP IPv4 protocol
     and IPv6 next header";
  reference
    "RFC 790: - Assigned numbers - Assigned Internet
     Protocol Number
    RFC 791: Internet Protocol - Protocol
    RFC 8200: Internet Protocol, Version 6 (IPv6)
     Specification - Next Header";
}
"RFC 790: - Assigned numbers - Assigned Internet Protocol Number
RFC 791: Internet Protocol - Protocol
}

identity ospf {
base protocol;
base next-header;
description
"Identity for OSPF IPv4 protocol and IPv6 next header";
reference
"RFC 790: - Assigned numbers - Assigned Internet Protocol Number
RFC 791: Internet Protocol - Protocol
}

identity l2tp {
base protocol;
base next-header;
description
"Identity for L2TP IPv4 protocol and IPv6 next header";
reference
"RFC 790: - Assigned numbers - Assigned Internet Protocol Number
RFC 791: Internet Protocol - Protocol
}

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

identity fin {
    base tcp-flags;
    description
        "Identity for 'Finish' TCP flag";
}
identity icmp-type {
  description
    "Base identity for ICMP Message types";
  reference
    "RFC 792: Internet Control Message Protocol";
}

identity echo-reply {
  base icmp-type;
  description
    "Identity for ‘Echo Reply’ ICMP message type";
  reference
    "RFC 792: Internet Control Message Protocol";
}

identity destination-unreachable {
  base icmp-type;
  description
    "Identity for ‘Destination Unreachable’ ICMP message type";
  reference
    "RFC 792: Internet Control Message Protocol";
}

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

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

identity router-advertisement {
  base icmp-type;
  description
    "Identity for ‘Router Advertisement’ ICMP message type";
  reference
    "RFC 792: Internet Control Message Protocol";
}
"Identity for ‘Router Advertisement’
ICMP message type";
reference
"RFC 792: Internet Control Message Protocol";
}

identity router-solicitation {
  base icmp-type;
  description
    "Identity for ‘Router Solicitation’
    ICMP message type";
  reference
    "RFC 792: Internet Control Message Protocol";
}

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

identity parameter-problem {
  base icmp-type;
  description
    "Identity for ‘Parameter Problem’
    ICMP message type";
  reference
    "RFC 792: Internet Control Message Protocol";
}

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

identity timestamp-reply {
  base icmp-type;
  description
    "Identity for ‘Timestamp Reply’
    ICMP message type";
  reference
    "RFC 792: Internet Control Message Protocol";
}
identity datagram-conversion-error {
    base icmp-type;
    description
        "Identity for ‘Datagram Conversion Error’ ICMP message type";
    reference
        "RFC 792: Internet Control Message Protocol";
}

identity experimental-mobility-protocols {
    base icmp-type;
    description
        "Identity for ‘Experimental Mobility Protocols’ ICMP message type";
    reference
        "RFC 792: Internet Control Message Protocol";
}

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

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

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

identity host-unreachable {

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

identity time-to-live-exceeded-in-transit {
  base icmp-type;
  description

"Identity for time to live exceeded in transit in time exceeded types";
reference
"RFC 792: Internet Control Message Protocol";
}

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

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

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

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

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

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

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

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

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

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

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

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

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

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

identity target-device {
    description
        "Base identity for target devices";
    reference
        "draft-ietf-i2nsf-capability-05: Information Model of NSFs Capabilities";
}

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

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

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

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

identity iot {
    base target-device;
    description
        "Identity for IoT";
}
}]

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

identity content-security-control {
    description
        "Base identity for content security control";
    reference
        "RFC 8329: Framework for Interface to
         Network Security Functions - Differences
         from ACL Data Models
         draft-ietf-i2nsf-capability-05: Information Model
         of NSFs Capabilities";
}

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

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

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

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

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

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

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

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

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

identity attack-mitigation-control {
    description
        "Base identity for attack mitigation control";
    reference
        "RFC 8329: Framework for Interface to Network Security Functions - Differences from ACL Data Models draft-ietf-i2nsf-capability-05: Information Model of NSFs Capabilities";
}

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

identity udp-flood {

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

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

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

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

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

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

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

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

identity ip-sweep {

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

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

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

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

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

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

identity ingress-action {
  description
  "Base identity for action";
  reference
  "draft-ietf-i2nsf-capability-05: Information Model of NSFs Capabilities - Ingress Action";
}

identity egress-action {
  description
  "Base identity for egress action";
  reference
  "draft-ietf-i2nsf-capability-05: Information Model
of NSFs Capabilities - Egress action";
}

identity default-action {
    description
        "Base identity for default action";
    reference
        "draft-ietf-i2nsf-capability-05: Information Model
        of NSFs Capabilities - Default action";
}

identity pass {
    base ingress-action;
    base egress-action;
    base default-action;
    description
        "Identity for pass";
    reference
        "draft-ietf-i2nsf-capability-05: Information Model
        of NSFs Capabilities - Actions and
        default action";
}

identity drop {
    base ingress-action;
    base egress-action;
    base default-action;
    description
        "Identity for drop";
    reference
        "draft-ietf-i2nsf-capability-05: Information Model
        of NSFs Capabilities - Actions and
        default action";
}

identity reject {
    base ingress-action;
    base egress-action;
    base default-action;
    description
        "Identity for reject";
    reference
        "draft-ietf-i2nsf-capability-05: Information Model
        of NSFs Capabilities - Actions and
        default action";
}

identity alert {

base ingress-action;
base egress-action;
base default-action;
description
  "Identity for alert";
reference
  "draft-ietf-i2nsf-capability-05: Information Model
  of NSFs Capabilities - Actions and
default action";
}

identity mirror {
  base ingress-action;
  base egress-action;
  base default-action;
  description
    "Identity for mirror";
  reference
    "draft-ietf-i2nsf-capability-05: Information Model
    of NSFs Capabilities - Actions and
default action";
}

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

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

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

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

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

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

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

identity resolution-strategy {
  description
  "Base identity for resolution strategy";
  reference
  "draft-ietf-i2nsf-capability-05: Information Model of NSFs Capabilities - Resolution Strategy";
}

identity fmr {
  base resolution-strategy;
  description
  "Identity for First Matching Rule (FMR)";
  reference
  "draft-ietf-i2nsf-capability-05: Information Model of NSFs Capabilities - Resolution Strategy";
}

identity lmr {
  base resolution-strategy;
  description
  "Identity for Last Matching Rule (LMR)";
  reference
  "draft-ietf-i2nsf-capability-05: Information Model of NSFs Capabilities - Resolution Strategy";
}

identity pmr {
  base resolution-strategy;
  description
  "Identity for Prioritized Matching Rule (PMR)";
  reference

"draft-ietf-i2nsf-capability-05: Information Model of NSFs Capabilities - Resolution Strategy";

identity pmre {
  base resolution-strategy;
  description "Identity for Prioritized Matching Rule with Errors (PMRE)";
  reference "draft-ietf-i2nsf-capability-05: Information Model of NSFs Capabilities - Resolution Strategy";
}

identity pmrn {
  base resolution-strategy;
  description "Identity for Prioritized Matching Rule with No Errors (PMRN)";
  reference "draft-ietf-i2nsf-capability-05: Information Model of NSFs Capabilities - Resolution Strategy";
}

identity i2nsf-ipsec {
  description "Internet Key Exchange for NSFs in the I2NSF framework";
  reference "draft-ietf-i2nsf-sdn-ipsec-flow-protection-04 - i2nsf-ipsec";
}

identity ike {
  base i2nsf-ipsec;
  description "IKE case: IPsec with IKE in the NSF";
  reference "draft-ietf-i2nsf-sdn-ipsec-flow-protection-04 - ike";
}

identity ikeless {
  base i2nsf-ipsec;
  description "IKEless case: IPsec without IKEv2 in the NSF";
  reference "draft-ietf-i2nsf-sdn-ipsec-flow-protection-04"
typedef start-time-type {
  type union {
    type yang:date-and-time;
    type enumeration {
      enum right-away {
        description "Immediate rule execution in the system.";
      }
    }
  }
}

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

typedef end-time-type {
  type union {
    type yang:date-and-time;
    type enumeration {
      enum infinitely {
        description "Infinite rule execution in the system.";
      }
    }
  }
}

description "End time when the rules are applied.";
}

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

description "Day of week when the rules are applied.";
}
enum tuesday {
    description
    "Tuesday for periodic day";
}
enum wednesday {
    description
    "Wednesday for periodic day";
}
enum thursday {
    description
    "Thursday for periodic day";
}
enum friday {
    description
    "Friday for periodic day";
}
enum saturday {
    description
    "Saturday for periodic day";
}

description
"This can be used for the rules to be applied according to periodic day";
}
typedef month-type {
    type enumeration {
        enum january {
            description
            "January for periodic month";
        }
        enum february {
            description
            "February for periodic month";
        }
        enum march {
            description
            "March for periodic month";
        }
        enum april {
            description
            "April for periodic month";
        }
        enum may {
            description

"May for periodic month";
}
enum june {
    description
    "June for periodic month";
}
enum july {
    description
    "July for periodic month";
}
enum august {
    description
    "August for periodic month";
}
enum september {
    description
    "September for periodic month";
}
enum october {
    description
    "October for periodic month";
}
enum november {
    description
    "November for periodic month";
}
enum december {
    description
    "December for periodic month";
}
}
description
"This can be used for the rules to be applied according to periodic month";
}
/*
 * Groupings
 */
grouping ipv4 {
    list ipv4-address {
        key "ipv4";
        description
        "The list of IPv4 addresses.";

        leaf ipv4 {
            type inet:ipv4-address;
        }
    }
}

description
"The value of IPv4 address."
}
choice subnet {
  description
  "The subnet can be specified as a prefix length or netmask.";
  leaf prefix-length {
    type uint8 {
      range "0..32";
    }
    description
    "The length of the subnet prefix.";
  }
  leaf netmask {
    type yang:dotted-quad;
    description
    "The subnet specified as a netmask.";
  }
}

description
"Grouping for an IPv4 address";

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

grouping ipv6 {
  list ipv6-address {
    key "ipv6";
    description
    "The list of IPv6 addresses.";
    leaf ipv6 {
      type inet:ipv6-address;
      description
      "The value of IPv6 address.";
    }
    leaf prefix-length {
      type uint8 {
        range "0..128";
      }
      description
      "The length of the subnet prefix.";
    }
  }
}
grouping pkt-sec-ipv4 {
  choice match-type {
    description "There are two types of security policy IPv4 address matching - exact match and range match.";
    case exact-match {
      uses ipv4;
      description "Exact match for an IPv4 address.";
    }
    case range-match {
      list range-ipv4-address {
        key "start-ipv4-address end-ipv4-address";
        leaf start-ipv4-address {
          type inet:ipv4-address;
          description "Starting IPv4 address for a range match.";
        }
        leaf end-ipv4-address {
          type inet:ipv4-address;
          description "Ending IPv4 address for a range match.";
        }
      }
      description "Range match for an IPv4 address.";
    }
  }
  description "Grouping for an IPv4 address.";
}

grouping pkt-sec-ipv6 {
  choice match-type {

description
  "There are two types of security policy IPv6 address
  matching - exact match and range match.";
case exact-match {
  uses ipv6;
description
  "Exact match for an IPv6 address.";
}
case range-match {
  list range-ipv6-address {
    key "start-ipv6-address end-ipv6-address";
    leaf start-ipv6-address {
      type inet:ipv6-address;
description
      "Starting IPv6 address for a range match.";
    }
leaf end-ipv6-address {
  type inet:ipv6-address;
description
  "Ending IPv6 address for a range match.";
}
description
  "Range match for an IPv6 address.";
}
}
description
  "Grouping for IPv6 address.";
reference
  "RFC 8200: Internet Protocol, Version 6 (IPv6)
  Specification - IPv6 address";
}
grouping pkt-sec-port-number {
  choice match-type {
    description
    "There are two types of security policy TCP/UDP port
    matching - exact match and range match.";
case exact-match {
  leaf-list port-num {
    type inet:port-number;
description
    "Exact match for a port number.";
  }
}
case range-match {

list range-port-num {
    key "start-port-num end-port-num";
    leaf start-port-num {
        type inet:port-number;
        description
            "Starting port number for a range match.";
    }
    leaf end-port-num {
        type inet:port-number;
        description
            "Ending port number for a range match.";
    }
    description
            "Range match for a port number.";
}

description
    "Grouping for port number.";

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

/*
 * Data nodes
 */

container i2nsf-security-policy {
    description
        "Container for security policy
        including a set of security rules according to certain logic,
        i.e., their similarity or mutual relations, etc. The network
        security policy can be applied to both the unidirectional
        and bidirectional traffic across the NSF.
        The I2NSF security policies use the Event-Condition-Action
        (ECA) policy model ";

    reference
        "RFC 8329: Framework for Interface to Network Security
        Functions - I2NSF Flow Security Policy Structure
        draft-ietf-i2nsf-capability-05: Information Model
        of NSFs Capabilities - Design Principles and ECA Policy Model
        Overview";

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

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

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

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

  reference
    "draft-ietf-i2nsf-capability-05: Information Model of NSFs Capabilities - Resolution strategy";
}

leaf default-action {
  type identityref {
    base default-action;
  }
  default alert;
  description
    "This default action can be used to specify a predefined action when no other alternative action was matched
by the currently executing I2NSF Policy Rule. An analogy is the use of a default statement in a C switch statement."

reference
"draft-ietf-i2nsf-capability-05: Information Model of NSFs Capabilities - Default action";

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

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

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

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

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

  leaf session-aging-time {
    type uint16;
    description
      "This is session aging time.";
  }
}
container long-connection {
    description
        "This is long-connection";

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

    leaf during {
        type uint16;
        description
            "This has long-connection during a time.";
    }
}

container time-intervals {
    description
        "Time zone when the rules are applied";
    container absolute-time-interval {
        description
            "Rule execution according to absolute time. The absolute time intervals mean the exact time to start or end.";

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

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

container periodic-time-interval {
    description
        "Rule execution according to periodic time. The periodic time intervals mean repeated time like day, week, or month.";

    container day {

description
"Rule execution according to day."
leaf every-day {
  type boolean;
  default true;
  description
  "Rule execution every day";
}
leaf-list specific-day {
  when "../every-day = 'false'";
  type day-type;
  description
  "Rule execution according to specific day";
}
container month {
  description
  "Rule execution according to month."
  leaf every-month {
    type boolean;
    default true;
    description
    "Rule execution every day";
  }
  leaf-list specific-month {
    when "../every-month = 'false'";
    type month-type;
    description
    "Rule execution according to month day";
  }
}
container event-clause-container {
  description
  "An event is defined as any important occurrence in time of a change in the system being managed, and/or in the environment of the system being managed. When used in the context of policy rules for a flow-based NSF, it is used to determine whether the Condition clause of the Policy Rule can be evaluated"
or not. Examples of an I2NSF event include time and user actions (e.g., logon, logoff, and actions that violate any ACL).

reference
draft-ietf-i2nsf-capability-05: Information Model of NSFs Capabilities - Design Principles and ECA Policy Model Overview

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

container event-clauses {
  description
    "System Event Clause - either a system event or system alarm";
  reference
draft-ietf-i2nsf-capability-05: Information Model of NSFs Capabilities - Design Principles and ECA Policy Model Overview

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

  leaf-list system-alarm {
    type identityref {
      base system-alarm;
    }
    description
container condition-clause-container {
  description
  "A condition is defined as a set of attributes, features, and/or values that are to be compared with a set of known attributes, features, and/or values in order to determine whether or not the set of Actions in that (imperative) I2NSF Policy Rule can be executed or not. Examples of I2NSF Conditions include matching attributes of a packet or flow, and comparing the internal state of an NSF to a desired state."
  reference

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

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

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

container pkt-sec-ipv4-header-length {
  choice match-type {
    description
    "The security policy rule according to system alarms.";
  }
}


"Security policy IPv4 Header length match -
exact match and range match."

```yang
case exact-match {
  leaf-list ipv4-header-length {
    type uint8 {
      range "5..15";
    }
    description
    "Exact match for an IPv4 header length."
  }
}

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

leaf-list pkt-sec-ipv4-tos {
  type identityref {
    base type-of-service;
  }
  description
  "The security policy rule according to
IPv4 type of service.
reference
"RFC 1394: Internet Protocol - Type of service";
}

container pkt-sec-ipv4-total-length {
choice match-type {
  description
  "Security policy IPv4 total length matching
  - exact match and range match.";
  case exact-match {
      leaf-list ipv4-total-length {
        type uint16;
        description
        "Exact match for an IPv4 total length.";
      }
    }
  case range-match {
      list range-ipv4-total-length {
        key "start-ipv4-total-length end-ipv4-total-length";
        leaf start-ipv4-total-length {
          type uint16;
          description
          "Starting IPv4 total length for a range match.";
        }
        leaf end-ipv4-total-length {
          type uint16;
          description
          "Ending IPv4 total length for a range match.";
        }
        description
        "Range match for an IPv4 total length.";
      }
    }
  }
}

leaf-list pkt-sec-ipv4-id {
  type uint16;
  description
  "The security policy rule according to
  IPv4 identification.";
  reference
"RFC 791: Internet Protocol - Total length";
}
leaf-list pkt-sec-ipv4-fragment-flags {
  type identityref {
    base fragmentation-flags-type;
  }
  description
  "The security policy rule according to IPv4 fragment flags."
  reference
  "RFC 791: Internet Protocol - Fragment flags"
}

container pkt-sec-ipv4-fragment-offset {
  choice match-type {
    description
    "There are two types to configure a security policy for IPv4 fragment offset, such as exact match and range match."
    case exact-match {
      leaf-list ipv4-fragment-offset {
        type uint16 {
          range "0..16383"
        }
        description
        "Exact match for an IPv4 fragment offset."
      }
    }
    case range-match {
      list range-ipv4-fragment-offset {
        key "start-ipv4-fragment-offset
          end-ipv4-fragment-offset";
        leaf start-ipv4-fragment-offset {
          type uint16 {
            range "0..16383"
          }
          description
          "Starting IPv4 fragment offset for a range match."
        }
        leaf end-ipv4-fragment-offset {
          type uint16 {
            range "0..16383"
          }
          description
          "Ending IPv4 fragment offset for a range match."
        }
      }
    }
  }
}
"Range match for an IPv4 fragment offset."
}  
}  

description  
"The security policy rule according to IPv4 fragment offset.";
reference  
"RFC 791: Internet Protocol - Fragment offset";
}  

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

case  
exact-match {  
leaf-list  
ipv4-ttl {  
type uint8;  
description  
"Exact match for an IPv4 TTL.";
  }
}

case  
range-match {  
list  
rang-ipv4-ttl {  
key "start-ipv4-ttl end-ipv4-ttl";  
leaf  
start-ipv4-ttl {  
type uint8;  
description  
"Starting IPv4 TTL for a range match.";
  }
leaf  
end-ipv4-ttl {  
type uint8;  
description  
"Ending IPv4 TTL for a range match.";
  }

description  
"Range match for an IPv4 TTL.";
  }
}

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

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

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

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

leaf pkt-sec-ipv4-same-ip {
  type boolean;
  description
    "Match on packets with the same IPv4 source and IPv4 destination address.";
}

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

container packet-security-ipv6-condition {
description
    "The purpose of this container is to represent
    IPv6 packet header information to determine
    if the set of policy actions in this ECA policy
    rule should be executed or not.";
reference
    "RFC 8200: Internet Protocol, Version 6 (IPv6)
    Specification";
leaf ipv6-description {
type string;
description
    "This is description for ipv6 condition.";
}

leaf-list pkt-sec-ipv6-traffic-class {
type identityref {
    base traffic-class;
}
description
    "The security policy rule according to
    IPv6 traffic class.";
reference
    "RFC 8200: Internet Protocol, Version 6 (IPv6)
    Specification - Traffic class";
}

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

case range-match {
    list range-ipv6-flow-label {
        key "start-ipv6-flow-label end-ipv6-flow-label";
        leaf start-ipv6-flow-label {
            type uint32 {
                range "0..1048575";
            }
            description "Starting IPv6 flow label for a range match.";
        }
        leaf end-ipv6-flow-label {
            type uint32 {
                range "0..1048575";
            }
            description "Ending IPv6 flow label for a range match.";
        }
        description "Range match for an IPv6 flow label.";
    }
    description "The security policy rule according to IPv6 flow label.";
}

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

leaf-list pkt-sec-ipv6-next-header {
    type identityref {
        base next-header;
    }
    description "The security policy rule according to IPv6 next header.";
}

container pkt-sec-ipv6-hop-limit {
    choice match-type {
        description "There are two types to configure a security policy for IPv6 hop limit, such as exact match and range match.";
        case exact-match {
            leaf-list ipv6-hop-limit {

case range-match {
    list range-ipv6-hop-limit {
        key "start-ipv6-hop-limit end-ipv6-hop-limit";
        leaf start-ipv6-hop-limit {
            type uint8;
            description
            "Start IPv6 hop limit for a range match.";
        }
        leaf end-ipv6-hop-limit {
            type uint8;
            description
            "End IPv6 hop limit for a range match.";
        }
        description
        "Range match for an IPv6 hop limit.";
    }
}
}

description
"The security policy rule according to IPv6 hop limit.";
reference
}

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

container pkt-sec-ipv6-dest {
    uses pkt-sec-ipv6;
    description
    "The security policy rule according to IPv6 destination address.";
    reference
    "RFC 8200: Internet Protocol, Version 6 (IPv6) Specification - IPv6 address";
}
container packet-security-tcp-condition {
  description
    "The purpose of this container is to represent
    TCP packet header information to determine
    if the set of policy actions in this ECA policy
    rule should be executed or not.";
  reference
    "RFC 793: Transmission Control Protocol";

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

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

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

container pkt-sec-tcp-seq-num {
  choice match-type {
    description
      "There are two types to configure a security
      policy for tcp sequence number,
      such as exact match and range match.";
    case exact-match {

leaf-list tcp-seq-num {
    type uint32;
    description
        "Exact match for an tcp sequence number.";
}

case range-match {
    list range-tcp-seq-num {
        key "start-tcp-seq-num end-tcp-seq-num";
        leaf start-tcp-seq-num {
            type uint32;
            description
                "Start tcp sequence number for a range match.";
        }
        leaf end-tcp-seq-num {
            type uint32;
            description
                "End tcp sequence number for a range match.";
        }
        description
            "Range match for a tcp sequence number.";
    }
}

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

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

leaf end-tcp-ack-num {
    type uint32;
    description
        "End tcp acknowledgement number for a range match.";
}

description
    "Range match for a tcp acknowledgement number.";
}

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

description
    "The security policy rule according to tcp acknowledgement number.";
reference
    "RFC 793: Transmission Control Protocol - Acknowledgement number";
description  "End tcp window size for a range match.";
}
description  "Range match for a tcp window size.";
}
}
}
description  "The security policy rule according to tcp window size.";
reference  "RFC 793: Transmission Control Protocol - Window size";
}

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

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

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

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

container pkt-sec-udp-dest-port-num {
  uses pkt-sec-port-number;
  description
  "The security policy rule according to udp destination port number.";
  reference
  "RFC 768: User Datagram Protocol - Total Length";
}

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

Kim, et al.  Expires January 26, 2020
container packet-security-icmp-condition {
  description
  "The purpose of this container is to represent ICMP packet header information to determine if the set of policy actions in this ECA policy rule should be executed or not.";
  reference
  "RFC 792: Internet Control Message Protocol
  RFC 8335: PROBE: A Utility for Probing Interfaces";
  leaf icmp-description {
    type string;
    description
    "This is description for icmp condition.";
  }
  leaf-list pkt-sec-icmp-type-and-code {
    type identityref {
      base icmp-type;
    }
    description
    "The security policy rule according to ICMP parameters.";
    reference
    "RFC 792: Internet Control Message Protocol
    RFC 8335: PROBE: A Utility for Probing Interfaces";
  }
}

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

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

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

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

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

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

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

container packet-security-ddos-condition {
    description
        "Condition for DDoS attack.";

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

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

container packet-security-payload-condition {
    description
        "Condition for packet payload";

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

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

container application-condition {
    description "Condition for application";
    leaf application-description {
        type string;
        description "This is description for application condition.";
    }
    leaf-list application-object {
        type string;
        description "This is application object.";
    }
    leaf-list application-group {
        type string;
        description "This is application group.";
    }
    leaf-list application-label {
        type string;
        description "This is application label.";
    }
    container category {
        description "This is application category";
        list application-category {
            key "name application-subcategory";
            description "This is application category list";
            leaf name {
                type string;
                description "This is name for application category.";
            }
            leaf application-subcategory {
                type string;
            }
        }
    }
}
container target-condition {
    description
    "Condition for target";
    leaf target-description {
        type string;
        description
        "This is description for target condition. Vendors can write instructions for target condition that vendor made";
    }
}

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

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

choice user-name {
    description
    "The name of the user.";
    case tenant {
        description
        "Tenant information.";
        leaf tenant {
            type uint8;
            description
            "User’s tenant information.";
        }
    }
    case vn-id {
        description
        "VN-ID information.";
        leaf vn-id {
            type uint8;
            description
            "User’s VN-ID information.";
        }
    }
}

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

choice group-name {
  description
  "The name of the user.";

  case tenant {
    description
    "Tenant information.";

    leaf tenant {
      type uint8;
      description
      "User’s tenant information.";
    }
  }

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

    leaf vn-id {
      type uint8;
      description
      "User’s VN-ID information.";
    }
  }

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

container gen-context-condition {
  description
  "Condition for generic context";

  leaf gen-context-description {
    type string;
    description
    "This is description for generic context condition. Vendors can write instructions for generic context condition that vendor made";
  }
}
container geographic-location {
    description "The location where network traffic is associated with. The region can be the geographic location such as country, province, and city, as well as the logical network location such as IP address, network section, and network domain.";

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

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

container action-clause-container {
    description "An action is used to control and monitor aspects of flow-based NSFs when the event and condition clauses are satisfied. NSFs provide security functions by executing various Actions. Examples of I2NSF Actions include providing intrusion detection and/or protection, web and flow filtering, and deep packet inspection for packets and flows.";


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

} container packet-action {
  description
  "Action for packets";
  reference
  "RFC 8329: Framework for Interface to Network Security
  Functions - I2NSF Flow Security Policy Structure
  draft-ietf-i2nsf-capability-05: Information Model
  of NSFs Capabilities - Design Principles and ECA
  Policy Model Overview";

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

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

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

} container advanced-action {
  description
  "If the packet need be additionally inspected,
  the packet are passed to advanced network
  security functions according to the profile.";
  reference
  "RFC 8329: Framework for Interface to Network Security
Functions - Differences from ACL Data Models;

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

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

container rule-group {
  description
      "This is rule group"

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

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

container rule-range {
  description

"This is a rule range."

leaf start-rule {
    type string;
    description
        "This is a start rule";
}
leaf end-rule {
    type string;
    description
        "This is a end rule";
}
leaf enable {
    type boolean;
    description
        "This is enable
        False is not enable.";
}
leaf description {
    type string;
    description
        "This is a desription for rule-group";
}
leaf i2nsf-ipsec {
    type identityref {
        base i2nsf-ipsec;
    }
    description
        "Internet Key Exchnage for NSFs
        in the I2NSF framework";
    reference
        "draft-ietf-i2nsf-sdn-ipsec-flow-protection-04
        - i2nsf-ipsec";
}

<CODE ENDS>

Figure 6: YANG Data Module of I2NSF NSF-Facing-Interface
6. IANA Considerations

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


Registrant Contact: The IESG.

XML: N/A; the requested URI is an XML namespace.

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

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


prefix: nsfintf

reference: RFC XXXX

7. Security Considerations

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

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

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

- ietf-i2nsf-policy-rule-for-nsf: The attacker may provide incorrect policy information of any target NSF by illegally modifying this.
Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

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

8. References

8.1. Normative References


8.2. Informative References

[draft-dong-i2nsf-asf-config]

[draft-ietf-i2nsf-capability]

[draft-ietf-i2nsf-capability-data-model]

[draft-ietf-i2nsf-sdn-ipsec-flow-protection]

[draft-ietf-supaf-generic-policy-info-model]
Appendix A. Configuration Examples

This section shows configuration examples of "ietf-i2nsf-policy-rule-for-nsf" module for security policy rules of network security devices. For security requirements, we assume that the NSFs (i.e., General firewall, Time based firewall, URL filter, VoIP/VoLTE filter, and http and https flood mitigation) described in Appendix A. Configuration Examples of [draft-ietf-i2nsf-capability-data-model] are registered in I2NSF framework. With the registered NSFs, we show configuration examples for security policy rules of network security functions according to the following three security requirements: (i) Block SNS access during business hours, (ii) Block malicious VoIP/VoLTE packets coming to the company, and (iii) Mitigate http and https flood attacks on company web server.

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

This section shows a configuration example for blocking SNS access during business hours.
Figure 7: Configuration XML for Time based Firewall to Block SNS Access during Business Hours
<i2nsf-security-policy
xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-policy-rule-for-nsf">
<system-policy>
  <system-policy-name>sns_access</system-policy-name>
  <rules>
    <rule-name>block_sns_access_during_operation_time</rule-name>
    <condition-clause-container>
      <packet-security-url-category-condition>
        <user-defined-category>facebook</user-defined-category>
        <user-defined-category>instagram</user-defined-category>
      </packet-security-url-category-condition>
    </condition-clause-container>
    <action-clause-container>
      <packet-action>
        <egress-action>drop</egress-action>
      </packet-action>
    </action-clause-container>
  </rules>
</system-policy>
</i2nsf-security-policy>

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

Figure 7 and Figure 8 show the configuration XML documents for time based firewall and web filter to block SNS access during business hours. For the security requirement, two NSFs (i.e., a time based firewall and a web filter) were used because one NSF can not meet the security requirement. The instances of XML documents for the time based firewall and the web filter are as follows: Note that a detailed data model for the configuration of the advanced network security function (i.e., web filter) is described in [draft-dong-i2nsf-asf-config].

Time based Firewall

1. The name of the system policy is sns_access.

2. The name of the rule is block_sns_access_during_operation_time.

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

4. The rule inspects a source IPv4 address (i.e., from 221.159.112.1 to 221.159.112.90) to inspect the outgoing packets of employees.
5. If the outgoing packets match the rules above, the time based firewall sends the packets to url filtering for additional inspection because the time based firewall can not inspect contents of the packets for the SNS URL.

Web Filter

1. The name of the system policy is sns_access.
2. The name of the rule is block_facebook_and_instagram.
3. The rule inspects URL address to block the access packets to the facebook or the instagram.
4. If the outgoing packets match the rules above, the packets are blocked.

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

This section shows a configuration example for blocking malicious VoIP/VoLTE packets coming to the company.
Figure 9: Configuration XML for General Firewall to Block Malicious VoIP/VoLTE Packets Coming to the Company
<i2nsf-security-policy xmlns="urn:ietf:params:xml:ns:yang:i2nsf-policy-rule-for-nsf">
  <system-policy>
    <system-policy-name>voip_volte_inspection</system-policy-name>
    <rules>
      <rule-name>block_malicious_voice_id</rule-name>
      <condition-clause-container>
        <packet-security-voice-condition>
          <pkt-sec-src-voice-id>11111@voip.black.com</pkt-sec-src-voice-id>
          <pkt-sec-src-voice-id>22222@voip.black.com</pkt-sec-src-voice-id>
        </packet-security-voice-condition>
      </condition-clause-container>
      <action-clause-container>
        <packet-action>
          <ingress-action>drop</ingress-action>
        </packet-action>
      </action-clause-container>
    </rules>
  </system-policy>
</i2nsf-security-policy>

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

Figure 9 and Figure 10 show the configuration XML documents for general firewall and VoIP/VoLTE filter to block malicious VoIP/VoLTE packets coming to the company. For the security requirement, two NSF's (i.e., a general firewall and a VoIP/VoLTE filter) were used because one NSF can not meet the security requirement. The instances of XML documents for the general firewall and the VoIP/VoLTE filter are as follows: Note that a detailed data model for the configuration of the advanced network security function (i.e., VoIP/VoLTE filter) is described in [draft-dong-i2nsf-asf-config].

General Firewall

1. The name of the system policy is voip_volte_inspection.

2. The name of the rule is block_malicious_voice_id.

3. The rule inspects a destination IPv4 address (i.e., from 221.159.112.1 to 221.159.112.90) to inspect the packets coming into the company.

4. The rule inspects a port number (i.e., 5060 and 5061) to inspect VoIP/VoLTE packet.
5. If the incoming packets match the rules above, the general firewall sends the packets to VoIP/VoLTE filter for additional inspection because the general firewall can not inspect contents of the VoIP/VoLTE packets.

VoIP/VoLTE Filter

1. The name of the system policy is malicious_voice_id.

2. The name of the rule is block_malicious_voice_id.

3. The rule inspects the voice id of the VoIP/VoLTE packets to block the malicious VoIP/VoLTE packets (i.e., 11111@voip.black.com and 22222@voip.black.com).

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


This section shows a configuration example for mitigating http and https flood attacks on a company web server.
<i2nsf-security-policy
xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-policy-rule-for-nsf">
<system-policy>
<system-policy-name>flood_attack_mitigation</system-policy-name>
<rules>
<rule-name>mitigate_http_and_https_flood_attack</rule-name>
<condition-clause-container>
<packet-security-ipv4-condition>
<pkt-sec-ipv4-dest>
<ipv4-address>
<ipv4>221.159.112.95</ipv4>
</ipv4-address>
</pkt-sec-ipv4-dest>
</packet-security-ipv4-condition>
<packet-security-tcp-condition>
<pkt-sec-tcp-dest-port-num>
<port-num>80</port-num>
<port-num>443</port-num>
</pkt-sec-tcp-dest-port-num>
</packet-security-tcp-condition>
</condition-clause-container>
<action-clause-container>
<advanced-action>
<attack-mitigation-control>http-and-https-flood</attack-mitigation-control>
</advanced-action>
</action-clause-container>
</rules>
</system-policy>
</i2nsf-security-policy>

Figure 11: Configuration XML for General Firewall to Mitigate HTTP and HTTPS Flood Attacks on a Company Web Server
<i2nsf-security-policy
xmlns="urn:ietf:params:xml:ns:yang:i2nsf-policy-rule-for-nsf">
<system-policy>
  <system-policy-name>flood_attack_mitigation</system-policy-name>
  <rules>
    <rule-name>mitigate_http_and_https_flood_attack</rule-name>
    <condition-clause-container>
      <packet-security-ddos-condition>
        <pkt-sec-alert-rate>100</pkt-sec-alert-rate>
      </packet-security-ddos-condition>
    </condition-clause-container>
    <action-clause-container>
      <packet-action>
        <ingress-action>drop</ingress-action>
      </packet-action>
    </action-clause-container>
  </rules>
</system-policy>
</i2nsf-security-policy>

Figure 12: Configuration XML for HTTP and HTTPS Flood Attack Mitigation to Mitigate HTTP and HTTPS Flood Attacks on a Company Web Server

Figure 11 and Figure 12 show the configuration XML documents for general firewall and http and https flood attack mitigation to mitigate http and https flood attacks on a company web server. For the security requirement, two NSFs (i.e., a general firewall and a http and https flood attack mitigation) were used because one NSF can not meet the security requirement. The instances of XML documents for the general firewall and http and https flood attack mitigation are as follows: Note that a detailed data model for the configuration of the advanced network security function (i.e., http and https flood attack mitigation) is described in [draft-dong-i2nsf-asf-config].

General Firewall

1. The name of the system policy is flood_attack_mitigation.

2. The name of the rule is mitigate_http_and_https_flood_attack.

3. The rule inspects a destination IPv4 address (i.e., 221.159.112.95) to inspect the access packets coming into the company web server.

4. The rule inspects a port number (i.e., 80 and 443) to inspect http and https packet.
5. If the packets match the rules above, the general firewall sends the packets to http and https flood attack mitigation for additional inspection because the general firewall cannot control the amount of packets for http and https packets.

HTTP and HTTPS Flood Attack Mitigation

1. The name of the system policy is http_and_https_flood_attack_mitigation.

2. The name of the rule is 100_per_second.

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

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

Appendix B. Changes from draft-ietf-i2nsf-nsf-facing-interface-dm-06

The following changes are made from draft-ietf-i2nsf-nsf-facing-interface-dm-06:

- The version is revised according to the comments from Acee Lindem who is a YANG doctor for review.

Appendix C. 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).

Appendix D. Contributors

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

- Hyoungshick Kim (Sungkyunkwan University)
- Daeyoung Hyun (Sungkyunkwan University)
- Dongjin Hong (Sungkyunkwan University)
- Liang Xia (Huawei)
Authors’ Addresses

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

Phone: +82 10 8273 0930
EMail: timkim@skku.edu

Jaehoon Paul Jeong
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

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

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

Susan Hares
Huawei
7453 Hickory Hill
Saline, MI  48176
USA

Phone: +1-734-604-0332
EMail: shares@ndzh.com
Qiushi Lin
Huawei
Huawei Industrial Base
Shenzhen, Guangdong 518129
China

EMail: linqiushi@huawei.com
Abstract

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

Status of This Memo

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

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

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

This Internet-Draft will expire on January 25, 2020.
Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust’s Legal
Provisions Relating to IETF Documents
(https://trustee.ietf.org/license-info) in effect on the date of
publication of this document. Please review these documents
carefully, as they describe your rights and restrictions with respect
to this document. Code Components extracted from this document must
include Simplified BSD License text as described in Section 4.e of
the Trust Legal Provisions and are provided without warranty as
described in the Simplified BSD License.

Table of Contents

1.  Introduction .................................................. 3
2.  Terminology .................................................. 4
  2.1. Requirements Notation .................................. 4
  2.2. Definitions ............................................... 4
  2.3. YANG ....................................................... 4
3.  Use Cases for NSF Monitoring Data ............................. 4
4.  Classification of NSF Monitoring Data .......................... 5
  4.1. Retention and Emission .................................. 6
  4.2. Notifications and Events ................................ 7
  4.3. Unsolicited Poll and Solicited Push ...................... 8
  4.4. I2NSF Monitoring Terminology for Retained Information .. 8
5.  Conveyance of NSF Monitoring Information .................... 9
  5.1. Information Types and Acquisition Methods ............... 10
6.  Basic Information Model for All Monitoring Data ............... 10
7.  Extended Information Model for All Monitoring Data .......... 11
  7.1. System Alarm ............................................... 11
    7.1.1. Memory Alarm ...................................... 11
    7.1.2. CPU Alarm ......................................... 12
    7.1.3. Disk Alarm ......................................... 12
    7.1.4. Hardware Alarm .................................... 12
    7.1.5. Interface Alarm .................................... 13
  7.2. System Events ............................................. 13
    7.2.1. Access Violation .................................. 13
    7.2.2. Configuration Change ............................... 14
  7.3. System Log ................................................ 14
    7.3.1. Access Logs ....................................... 14
    7.3.2. Resource Utilization Logs .......................... 15
    7.3.3. User Activity Logs ................................ 15
  7.4. System Counters .......................................... 16
    7.4.1. Interface counters ................................. 16
  7.5. NSF Events ................................................ 17
1. Introduction

According to [I-D.ietf-i2nsf-terminology], the interface provided by Network Security Functions (NSFs) (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 NSF-Facing Interface [I-D.ietf-i2nsf-nsf-facing-interface-dm]. 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 NSF monitoring information model that provides visibility for an NSF for Security Controller. 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-Facing Interface. The information model for monitoring 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-capability].

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

2. Terminology

2.1. Requirements Notation

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

2.2. Definitions

The terms, which are used in this document, are defined in the I2NSF terminology document [I-D.ietf-i2nsf-terminology] [RFC8329].

2.3. YANG

This document follows the guidelines of [RFC6087], 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 the security controller 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 a security controller 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 security controller 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 which may be 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
[I-D.ietf-i2nsf-terminology] 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 [I-D.ietf-i2nsf-terminology] 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-Facing Interface
[I-D.ietf-i2nsf-terminology]. 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-Facing Interfaces.

Information emitted via standardized interfaces can be consumed by an
I2NSF User [I-D.ietf-i2nsf-terminology] that includes the capability
to consume information not only via an I2NSF Interface(e.g.,
[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
- an I2NSF Component that creates the information

4.2. Notifications and Events

A specific task of I2NSF User is to process I2NSF Policy Rules [I-D.ietf-i2nsf-terminology]. 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 in [I-D.ietf-i2nsf-terminology], 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 which may be 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 a monitor entity (e.g., Security Controller and I2NSF User) NSF timely. 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 mechanism 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 not 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 sets 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_version: It indicates the version of the data format and is a two-digit decimal numeral starting from 01.
- message_type: Event, Alert, Alarm, Log, Counter, etc.
- time_stamp: It indicates the time when the message is generated.
- vendor_name: The name of the NSF vendor.
- NSF_name: The name (or IP) of the NSF generating the message.
- Module_name: The module name outputting the message.
- Severity: It indicates the level of the logs. There are total eight levels, from 0 to 7. 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 Alarm

Characteristics:

- acquisition_method: subscription
- emission_type: on-change
- dampening_type: no-dampening

7.1.1. Memory Alarm

The following information should be included in a Memory Alarm:

- event_name: MEM_USAGE_ALARM
- module_name: It indicates the NSF module responsible for generating this alarm.
- usage: specifies the amount of memory used.
7.1.2. CPU Alarm

The following information should be included in a CPU Alarm:

- event_name: CPU_USAGE_ALARM
- usage: Specifies the amount of CPU used.
- threshold: The threshold triggering the event
- severity: The severity of the alarm such as critical, high, medium, low
- 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 amount 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.
- threshold: The threshold triggering the alarm
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_mode: User authentication mode. e.g., Local Authentication, Third-Party Server Authentication, Authentication Exemption, Single Sign-On (SSO) Authentication
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_mode: User authentication mode. e.g., Local Authentication, Third-Party Server Authentication, Authentication Exemption, SSO Authentication
- message: Configuration is modified.

7.3. System Log

Characteristics:

- acquisition_method: subscription
- emission_type: on-change
- dampening_type: on-repetition

7.3.1. Access Logs

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
7.3.2. Resource Utilization Logs

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

7.3.3. User Activity Logs

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_address`: Login IP address of a user
- `authentication_mode`: User authentication mode. e.g., Local Authentication, Third-Party Server Authentication, Authentication Exemption, SSO Authentication
7.4. System Counters

Characteristics:

- acquisition_method: subscription or query
- emission_type: periodical
- dampening_type: none

7.4.1. Interface counters

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_ave_rate: Inbound traffic average rate in pps
- in_traffic_peak_rate: Inbound traffic peak rate in pps
7.5. NSF Events

Characteristics:

- acquisition_method: subscription
- emission_type: on-change
- dampening_type: none

7.5.1. DDoS Event

The following information should be included in a DDoS Event:

- event_name: SEC_EVENT_DDoS
- sub_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_id: The ID of the rule being triggered
7.5.2. Session Table Event

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

- `event_name`: SESSION_USAGE_HIGH
- `current`: The number of concurrent sessions
- `max`: 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.

7.5.3. Virus Event

The following information should be included in a Virus Event:

- `event_name`: SEC_EVENT_VIRUS
- `virus_type`: 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
o  file_type: The type of the file where the virus is hided within
o  file_name: The name of the file where the virus is hided within
o  virus_info: The brief introduction of the virus
o  raw_info: The information describing the packet triggering the event.
    o  rule_id: The ID of the rule being triggered
    o  rule_name: The name of the rule being triggered
    o  profile: Security profile that traffic matches.

7.5.4. Intrusion Event

The following information should be included in an Intrusion Event:

o  event_name: The name of event. e.g., SEC_EVENT_Intrusion
o  sub_attack_type: Attack type, e.g., brutal force and buffer overflow
o  src_ip: The source IP address of the packet
o  dst_ip: The destination IP address of the packet
o  src_port: The source port number of the packet
o  dst_port: The destination port number of the packet
o  src_zone: The source security zone of the packet
o  dst_zone: The destination security zone of the packet
o  protocol: The employed transport layer protocol. e.g., TCP and UDP
o  app: The employed application layer protocol. e.g., HTTP and FTP
o  rule_id: The ID of the rule being triggered
o  rule_name: The name of the rule being triggered
o  profile: Security profile that traffic matches
o  intrusion_info: Simple description of intrusion
raw_info: The information describing the packet triggering the event

7.5.5. Botnet Event

The following information should be included in a Botnet Event:

- **event_name**: The name of event. e.g., SEC_EVENT_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
- **app**: The employed application layer protocol. e.g., HTTP and FTP
- **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
- **botnet_info**: Simple description of Botnet
- **rule_id**: The ID of the rule being triggered
- **rule_name**: The name of the rule being triggered
7.5.6. Web Attack Event

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

- event_name: The name of event. e.g., SEC_EVENT_WebAttack
- sub_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
- req_method: The method of requirement. For instance, "PUT" and "GET" in HTTP
- req_url: Requested URL
- url_category: Matched URL category
- filtering_type: URL filtering type. e.g., Blacklist, Whitelist, User-Defined, Predefined, Malicious Category, and Unknown
- rule_id: The ID of the rule being triggered
- rule_name: The name of the rule being triggered
- profile: Security profile that traffic matches

7.6. NSF Logs

Characteristics:

- acquisition_method: subscription
7.6.1. DDoS Logs

Besides the fields in a DDoS Alarm, the following information should be included in a DDoS Logs:

- **attack_type**: DDoS
- **attack_ave_rate**: The average pps of the attack traffic within the recorded time
- **attack_ave_speed**: The average bps of the attack traffic within the recorded time
- **attack_pkt_num**: The number of attack packets within the recorded time
- **attack_src_ip**: The source IP addresses of attack traffics. If there are a large number of IP addresses, then pick a certain number of resources according to different rules.
- **action**: Actions against DDoS attacks. e.g., Allow, Alert, Block, Discard, Declare, Block-ip, and Block-service.

7.6.2. Virus Logs

Besides the fields in a Virus Alarm, the following information should be included in a Virus Logs:

- **attack_type**: Virus
- **protocol**: The transport layer protocol
- **app**: The name of the application layer protocol
- **times**: The time of detecting the virus
- **action**: The actions dealing with the virus. e.g., alert and block
- **os**: The OS that the virus will affect. e.g., all, android, ios, unix, and windows
7.6.3. Intrusion Logs

Besides the fields in an Intrusion Alarm, the following information should be included in an Intrusion Logs:

- **attack_type**: Intrusion
- **times**: The times of intrusions happened in the recorded time
- **os**: The OS that the intrusion will affect. e.g., all, android, ios, unix, and windows
- **action**: The actions dealing with the intrusions. e.g., Allow, Alert, Block, Discard, Declare, Block-ip, and Block-service
- **attack_rate**: NUM the pps of attack traffic
- **attack_speed**: NUM the bps of attack traffic

7.6.4. Botnet Logs

Besides the fields in a Botnet Alarm, the following information should be included in a Botnet Logs:

- **attack_type**: Botnet
- **botnet_pkt_num**: The number of the packets sent to or from the detected botnet
- **action**: The actions dealing with the detected packets. e.g., Allow, Alert, Block, Discard, Declare, Block-ip, and Block-service.
- **os**: The OS that the attack aims at. e.g., all, android, ios, unix, and windows.

7.6.5. DPI Logs

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.

- **type**: DPI action types. e.g., File Blocking, Data Filtering, and Application Behavior Control
- **file_name**: The file name
7.6.6. Vulnerability Scanning Logs

Vulnerability scanning logs record the victim host and its related vulnerability information that should 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
- vulnerability_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
7.6.7. Web Attack Logs

Besides the fields in a Web Attack Alarm, the following information should be included in a Web Attack Report:

- **attack_type**: Web Attack
- **rsp_code**: Response code
- **req_clientapp**: The client application
- **req_cookies**: Cookies
- **req_host**: The domain name of the requested host
- **raw_info**: The information describing the packet triggering the event.

7.7. NSF Counters

Characteristics:

- **acquisition_method**: subscription or query
- **emission_type**: periodical
- **dampening_type**: none

7.7.1. Firewall counters

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
7.7.2. Policy Hit Counters

Policy Hit Counters record the security policy that traffic matches and its hit count. It can check if policy configurations are correct.
Internet-Draft    I2NSF NSF Monitoring YANG Data Model         July 2019

- 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 the security controller. The security controller 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 Security Controller to the DMS via Registration Interface. These data are delivered from the DMS to the VNF Manager in the Management and Orchestration (MANO) in the NFV system [I-D.yang-i2nsf-nfv-architecture].

- I2NSF NSF-Facing Interface
  [I-D.ietf-i2nsf-nsf-facing-interface-dm]: 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 specifies the list of events that can trigger Event-Condition-Action (ECA) policies via NSF-Facing Interface.

9. Tree Structure

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

```
module: ietf-i2nsf-monitor
   +--rw counters
     +--rw system-interface
      |   +--rw acquisition-method? identityref
      |   +--rw emission-type? identityref
      |   +--rw dampening-type? identityref
      |   +--rw interface-name? string
      |   +--rw in-total-traffic-pkts? uint32
      |   +--rw out-total-traffic-pkts? uint32
      |   +--rw in-total-traffic-bytes? uint32
      |   +--rw out-total-traffic-bytes? uint32
      |   +--rw in-drop-traffic-pkts? uint32
      |   +--rw out-drop-traffic-pkts? uint32
```
Internet-Draft    I2NSF NSF Monitoring YANG Data Model         July 2019

| +--rw in-drop-traffic-bytes?     uint32
| +--rw out-drop-traffic-bytes?    uint32
| +--rw total-traffic?             uint32
| +--rw in-traffic-ave-rate?       uint32
| +--rw in-traffic-peak-rate?      uint32
| +--rw in-traffic-ave-speed?      uint32
| +--rw in-traffic-peak-speed?     uint32
| +--rw out-traffic-ave-rate?      uint32
| +--rw out-traffic-peak-rate?     uint32
| +--rw out-traffic-ave-speed?     uint32
| +--rw out-traffic-peak-speed?    uint32
| +--rw message?                   string
| +--rw time-stamp?                yang:date-and-time
| +--rw vendor-name?               string
| +--rw nsf-name?                  string
| +--rw module-name?               string
| +--rw severity?                  severity

+--rw nsf-firewall
    +--rw acquisition-method?       identityref
    +--rw emission-type?            identityref
    +--rw dampening-type?           identityref
    +--rw src-ip?                   inet:ipv4-address
    +--rw dst-ip?                   inet:ipv4-address
    +--rw src-port?                 inet:port-number
    +--rw dst-port?                 inet:port-number
    +--rw src-zone?                 string
    +--rw dst-zone?                 string
    +--rw src-region?               string
    +--rw dst-region?               string
    +--rw policy-id?                uint8
    +--rw policy-name?              string
    +--rw src-user?                 string
    +--rw protocol?                 identityref
    +--rw app?                      string
    +--rw total-traffic?            uint32
    +--rw in-traffic-ave-rate?      uint32
    +--rw in-traffic-peak-rate?     uint32
    +--rw in-traffic-ave-speed?     uint32
    +--rw in-traffic-peak-speed?    uint32
    +--rw out-traffic-ave-rate?     uint32
    +--rw out-traffic-peak-rate?    uint32
    +--rw out-traffic-ave-speed?    uint32
    +--rw out-traffic-peak-speed?   uint32

+--rw nsf-policy-hits
    +--rw acquisition-method?       identityref
    +--rw emission-type?            identityref
    +--rw dampening-type?           identityref
    +--rw src-ip?                   inet:ipv4-address
**Internet-Draft**    I2NSF NSF Monitoring YANG Data Model        July 2019

```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+--rw dst-ip?</td>
<td>inet:ipv4-address</td>
</tr>
<tr>
<td></td>
<td>+--rw src-port?</td>
<td>inet:port-number</td>
</tr>
<tr>
<td></td>
<td>+--rw dst-port?</td>
<td>inet:port-number</td>
</tr>
<tr>
<td></td>
<td>+--rw src-zone?</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>+--rw dst-zone?</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>+--rw src-region?</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>+--rw dst-region?</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>+--rw policy-id?</td>
<td>uint8</td>
</tr>
<tr>
<td></td>
<td>+--rw policy-name?</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>+--rw src-user?</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>+--rw dst-user?</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>+--rw protocol?</td>
<td>identityref</td>
</tr>
<tr>
<td></td>
<td>+--rw app?</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>+--rw message?</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>+--rw time-stamp?</td>
<td>yang:date-and-time</td>
</tr>
<tr>
<td></td>
<td>+--rw vendor-name?</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>+--rw nsf-name?</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>+--rw module-name?</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>+--rw severity?</td>
<td>severity</td>
</tr>
<tr>
<td></td>
<td>+--rw hit-times?</td>
<td>uint32</td>
</tr>
</tbody>
</table>
```

**notifications:**

```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+--n system-detection-alarm</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro alarm-category?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro acquisition-method?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro emission-type?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro dampening-type?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro usage?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro threshold?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro message?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro time-stamp?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro vendor-name?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro nsf-name?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro module-name?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro severity?</td>
</tr>
<tr>
<td></td>
<td>+--n system-detection-event</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro event-category?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro acquisition-method?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro emission-type?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro dampening-type?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro user</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro group</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro login-ip-addr</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro authentication?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro message?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro time-stamp?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro vendor-name?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro nsf-name?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+--ro module-name?</td>
</tr>
</tbody>
</table>
```

 Internet-Draft    I2NSF NSF Monitoring YANG Data Model         July 2019

|  +--ro severity?             severity
|  +--n nsf-detection-flood
|   |  +--ro event-name?        identityref
|   |  +--ro dst-ip?            inet:ipv4-address
|   |  +--ro dst-port?          inet:port-number
|   |  +--ro rule-id            uint8
|   |  +--ro rule-name          string
|   |  +--ro profile?           string
|   |  +--ro raw-info?          string
|   |  +--ro sub-attack-type?   identityref
|   |  +--ro start-time         yang:date-and-time
|   |  +--ro end-time           yang:date-and-time
|   |  +--ro attack-rate?       uint32
|   |  +--ro attack-speed?      uint32
|   |  +--ro message?           string
|   |  +--ro time-stamp?        yang:date-and-time
|   |  +--ro vendor-name?       string
|   |  +--ro nsf-name?          string
|   |  +--ro module-name?       string
|   |  +--ro severity?          severity
|  +--n nsf-detection-session-table
|   |  +--ro current-session?   uint8
|   |  +--ro maximum-session?   uint8
|   |  +--ro threshold?         uint8
|   |  +--ro message?           string
|   |  +--ro time-stamp?        yang:date-and-time
|   |  +--ro vendor-name?       string
|   |  +--ro nsf-name?          string
|   |  +--ro module-name?       string
|   |  +--ro severity?          severity
|  +--n nsf-detection-virus
|   |  +--ro src-ip?            inet:ipv4-address
|   |  +--ro dst-ip?            inet:ipv4-address
|   |  +--ro src-port?          inet:port-number
|   |  +--ro dst-port?          inet:port-number
|   |  +--ro src-zone?          string
|   |  +--ro dst-zone?          string
|   |  +--ro rule-id            uint8
|   |  +--ro rule-name          string
|   |  +--ro profile?           string
|   |  +--ro raw-info?          string
|   |  +--ro virus?             identityref
|   |  +--ro virus-name?        string
|   |  +--ro file-type?         string
|   |  +--ro file-name?         string
|   |  +--ro message?           string
|   |  +--ro time-stamp?        yang:date-and-time
|   |  +--ro vendor-name?       string

| +--ro nsf-name?      string |
| +--ro module-name?   string |
| +--ro severity?      severity |
| +---n nsf-detection-intrusion |
|   +--ro src-ip?            inet:ipv4-address |
|   +--ro dst-ip?            inet:ipv4-address |
|   +--ro src-port?          inet:port-number |
|   +--ro dst-port?          inet:port-number |
|   +--ro src-zone?          string |
|   +--ro dst-zone?          string |
|   +--ro rule-id           uint8 |
|   +--ro rule-name          string |
|   +--ro profile?          string |
|   +--ro raw-info?         string |
|   +--ro protocol?         identityref |
|   +--ro app?              string |
|   +--ro sub-attack-type?   identityref |
|   +--ro message?          string |
|   +--ro time-stamp?       yang:date-and-time |
|   +--ro vendor-name?      string |
|   +--ro nsf-name?         string |
|   +--ro module-name?      string |
|   +--ro severity?         severity |
| +---n nsf-detection-botnet |
|   +--ro src-ip?        inet:ipv4-address |
|   +--ro dst-ip?        inet:ipv4-address |
|   +--ro src-port?      inet:port-number |
|   +--ro dst-port?      inet:port-number |
|   +--ro src-zone?      string |
|   +--ro dst-zone?      string |
|   +--ro rule-id        uint8 |
|   +--ro rule-name      string |
|   +--ro profile?       string |
|   +--ro raw-info?      string |
|   +--ro attack-type?   identityref |
|   +--ro protocol?      identityref |
|   +--ro botnet-name?   string |
|   +--ro role?          string |
|   +--ro message?      string |
|   +--ro time-stamp?   yang:date-and-time |
|   +--ro vendor-name?  string |
|   +--ro nsf-name?     string |
|   +--ro module-name?  string |
|   +--ro severity?     severity |
| +---n nsf-detection-web-attack |
|   +--ro src-ip?             inet:ipv4-address |
|   +--ro dst-ip?             inet:ipv4-address |
|   +--ro src-port?           inet:port-number |
+++ ro dst-port?           inet:port-number
+++ ro src-zone?          string
+++ ro dst-zone?          string
+++ ro rule-id           uint8
+++ ro rule-name         string
+++ ro profile?          string
+++ ro raw-info?         string
+++ ro sub-attack-type?  identityref
+++ ro request-method?   identityref
+++ ro req-uri?          string
+++ ro uri-category?     string
+++ ro filtering-type*   identityref
+++ ro message?          string
+++ ro time-stamp?       yang:date-and-time
+++ ro vendor-name?      string
+++ ro nsf-name?         string
+++ ro module-name?      string
+++ ro severity?         severity
--- n system-access-log
+++ ro login-ip           inet:ipv4-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
--- n 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
--- n system-user-activity-log
+++ ro acquisition-method? identityref
+++ ro emission-type?    identityref
+++ ro dampening-type?   identityref
+++ ro user               string
+++ro group
  +--ro login-ip-addr 
  +--ro authentication? 
  +--ro access? 
  +--ro online-duration? 
  +--ro logout-duration? 
  +--ro additional-info? 
+++n nsf-log-ddos
  +--ro attack-type? 
  +--ro attack-ave-rate? 
  +--ro attack-ave-speed? 
  +--ro attack-pkt-num? 
  +--ro attack-src-ip? 
  +--ro action? 
  +--ro acquisition-method? 
  +--ro emission-type? 
  +--ro dampening-type? 
  +--ro message? 
  +--ro time-stamp? 
  +--ro vendor-name? 
  +--ro module-name? 
  +--ro severity? 
+++n nsf-log-virus
  +--ro attack-type? 
  +--ro action? 
  +--ro os? 
  +--ro time 
  +--ro acquisition-method? 
  +--ro emission-type? 
  +--ro dampening-type? 
  +--ro message? 
  +--ro time-stamp? 
  +--ro vendor-name? 
  +--ro module-name? 
  +--ro severity? 
+++n nsf-log-intrusion
  +--ro attack-type? 
  +--ro action? 
  +--ro time 
  +--ro acquisition-rate? 
  +--ro attack-speed? 
  +--ro acquisition-method? 
  +--ro emission-type? 
  +--ro dampening-type? 
  +--ro message? 
  +--ro time-stamp?
---ro vendor-name?          string
  +++ro nsf-name?             string
  +++ro module-name?          string
  +++ro severity?             severity

  +++-n nsf-log-botnet
  +++-ro attack-type?         identityref
  +++-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 time-stamp?          yang:date-and-time
  +++-ro vendor-name?         string
  +++-ro nsf-name?            string
  +++-ro module-name?         string
  +++-ro severity?            severity

  +++-n nsf-log-dpi
  +++-ro attack-type?         dpi-type
  +++-ro acquisition-method?  identityref
  +++-ro emission-type?       identityref
  +++-ro dampening-type?      identityref
  +++-ro src-ip?              inet:ipv4-address
  +++-ro dst-ip?              inet:ipv4-address
  +++-ro src-port?            inet:port-number
  +++-ro dst-port?            inet:port-number
  +++-ro src-zone?            string
  +++-ro dst-zone?            string
  +++-ro src-region?          string
  +++-ro dst-region?          string
  +++-ro policy-id?           uint8
  +++-ro policy-name?         string
  +++-ro src-user?            string
  +++-ro protocol?            identityref
  +++-ro app?                 string
  +++-ro message?             string
  +++-ro time-stamp?          yang:date-and-time
  +++-ro vendor-name?         string
  +++-ro nsf-name?            string
  +++-ro module-name?         string
  +++-ro severity?            severity

  +++-n nsf-log-vuln-scan
  +++-ro vulnerability-id?    uint8
  +++-ro victim-ip?           inet:ipv4-address
  +++-ro protocol?            identityref
  +++-ro port-num?            inet:port-number
  +++-ro level?               severity
Figure 1: Information Model for NSF Monitoring

10. YANG Data Model

This section introduces a YANG data model for the information model of the NSF monitoring information model.

<CODE BEGINS> file "ietf-i2nsf-monitor@2019-07-23.yang"
module ietf-i2nsf-monitor {
  yang-version 1.1;
  namespace
  prefix iim;
  import ietf-inet-types{
    prefix inet;
    reference
      "Section 4 of RFC 6991";
  }

| +++-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 time-stamp? yang:date-and-time |
| +++-ro vendor-name? string |
| +++-ro nsf-name? string |
| +++-ro module-name? string |
| +++-ro severity? severity |

---n nsf-log-web-attack
| +++-ro attack-type? identityref |
| +++-ro rsp-code? string |
| +++-ro req-clientapp? string |
| +++-ro req-cookies? string |
| +++-ro req-host? string |
| +++-ro raw-info? string |
| +++-ro acquisition-method? identityref |
| +++-ro emission-type? identityref |
| +++-ro dampening-type? identityref |
| +++-ro message? string |
| +++-ro time-stamp? yang:date-and-time |
| +++-ro vendor-name? string |
| +++-ro nsf-name? string |
| +++-ro module-name? string |
| +++-ro severity? severity |

Figure 1: Information Model for NSF Monitoring
import ietf-yang-types {
    prefix yang;
    reference
        "Section 3 of RFC 6991";
}

organization
    "IETF I2NSF (Interface to Network Security Functions)
       Working Group";
contact
    "WG Web: <http://tools.ietf.org/wg/i2nsf>
    WG List: <mailto:i2nsf@ietf.org>
    WG Chair: Linda Dunbar
       <mailto:Linda.dunbar@huawei.com>
    Editor: Jaehoon Paul Jeong
       <mailto:pauljeong@skku.edu>
    Editor: Chaehong Chung
       <mailto:darkhong@skku.edu>";

description
    "This module is a YANG module for monitoring NSFs.
    
    Copyright (c) 2018 IETF Trust and the persons identified as
    authors of the code. All rights reserved.
    
    Redistribution and use in source and binary forms, with or
    without modification, is permitted pursuant to, and subject
    to the license terms contained in, the Simplified BSD License
    set forth in Section 4.c of the IETF Trust’s Legal Provisions
    
    This version of this YANG module is part of RFC 6087; see
    the RFC itself for full legal notices.";

revision "2019-07-23" {
    description "First revision";
    reference
        "RFC XXXX: I2NSF NSF Monitoring YANG Data Model";
}

typedef severity {
    type enumeration {
        enum high {
            description
            }
        }
}

"high-level";
}
enum middle {
  description
  "middle-level";
}
enum low {
  description
  "low-level";
}

description
  "An indicator representing severity";
}
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
  "This is used for protocol";
}
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
  "This is used for dpi type";
}
typedef operation-type{
  type enumeration {
    enum login{
      description
      "Login operation";
    }
    enum logout{
      description
      "Logout operation";
    }
    enum configuration{
      description
      "Configuration operation";
    }
  }
  description
  "An indicator representing operation-type";
}
typedef login-mode{
  type enumeration {
    enum root{
      description
      "Root login-mode";
    }
    enum user{
      description
      "User login-mode";
    }
    enum guest{
      description
      "Guest login-mode";
    }
  }
  description
  "This is used for login-mode";
}
"Guest login-mode";
    }
}  

description
    "An indicator representing login-mode";
}

identity characteristics {
    description
        "Base identity for monitoring information characteristics";
}

identity acquisition-method {
    base characteristics;
    description
        "The type of acquisition-method. 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-part-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 flood-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    
    "An 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    
    "An ICMP flood is detected";  
}
identity https-flood {  
  base flood-type;  
  description    
    "A HTTPS flood is detected";  
}
identity http-flood {  
  base flood-type;  
  description    
    "A HTTP flood is detected";  
}
identity dns-reply-flood {  
  base flood-type;  
  description
"A DNS reply flood is detected";
)
identity dns-query-flood {
  base flood-type;
  description
    "A DNS query flood is detected";
}
identity sip-flood {
  base flood-type;
  description
    "A SIP flood is detected";
}

identity nsf-event-name {
  description
    "Base identity for detectable nsf event types";
}
identity SEC-EVENT-DDOS {
  base nsf-event-name;
  description
    "The nsf event is sec-event-ddos.";
}
identity SESSION-USAGE-HIGH {
  base nsf-event-name;
  description
    "The nsf event is session-usage-high";
}
identity SEC-EVENT-VIRUS {
  base nsf-event-name;
  description
    "The nsf event is sec-event-virus";
}
identity SEC-EVENT-INTRUSION {
  base nsf-event-name;
  description
    "The nsf event is sec-event-intrusion";
}
identity SEC-EVENT-BOTNET {
  base nsf-event-name;
  description
    "The nsf event is sec-event-botnet";
}
identity SEC-EVENT-WEBATTACK {
  base nsf-event-name;
  description
    "The nsf event is sec-event-webattack";
}
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 is an ID stub limited to indicating
    that this attack type is botnet.
    The usual semantic and taxonomy is missing
    and name is used.";
}
identity virus-type {
  base nsf-attack-type;
  description
   "The type of virus. Can 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 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 ddos-attack-type {
  base nsf-attack-type;
  description
    "The attack type is associated with a detected nsf-log event";
}

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, for example, a web-attack. Can be applicable to more than web-attacks. 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 wrt 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";
}
identity icmpv4 {
  base ipv4;
  description
    "ICMPv4 protocol type.";
}
identity icmpv6 {
  base ipv6;
  description
    "ICMPv6 protocol type.";
}
identity ip {
  base protocol-type;
  description
    "General IP protocol type.";
  reference
    "RFC 791: Internet Protocol
RFC 2460: 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 2460: Internet Protocol, Version 6 (IPv6)";
}
identity http {
  base tcp;
  description
    "HTTP protocol type.";
  reference
    "RFC 2616: Hypertext Transfer Protocol";
}
identity ftp {
  base tcp;
  description
    "FTP protocol type.";
  reference
    "RFC 959: File Transfer Protocol";
}
grouping common-monitoring-data {
  description
    "The data set of common monitoring";
  leaf message {
    type string;
    description
      "This is a freetext annotation of
monitoring notification content";
}
leaf time-stamp {
  type yang:date-and-time;
  description
    "Indicates the time of message generation";
}
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 module-name {
  type string;
  description
    "The module name outputting the message";
}
leaf severity {
  type severity;
  description
    "The severity of the alarm such
as critical, high, middle, low.";
}
}

grouping characteristics{
  description
    "A set of monitoring information characteristics";
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";
}
}


description
    "A set of system alarm type contents";
leaf usage {
    type uint8;
    description
        "specifies the amount of usage";
}
leaf threshold {
    type uint8;
    description
        "The threshold triggering the alarm or the event";
}


description
    "System event metadata associated with system events caused by user activity.";
leaf user {
    type string;
    mandatory true;
    description
        "Name of a user";
}
leaf group {
    type string;
    mandatory true;
    description
        "Group to which a user belongs.";
}
leaf login-ip-addr {
    type inet:ipv4-address;
    mandatory true;
    description
        "Login IP address of a user.";
}
leaf authentication {
    type identityref {
        base authentication-mode;
    }  
    description

grouping i2nsf-nsf-event-type-content-extend {
  description "A set of common IPv4-related NSF event content elements";
  leaf src-ip {
    type inet:ipv4-address;
    description "The source IP address of the packet";
  }
  leaf dst-ip {
    type inet:ipv4-address;
    description "The destination IP address of the packet";
  }
  leaf src-port {
    type inet:port-number;
    description "The source port of the packet";
  }
  leaf dst-port {
    type inet:port-number;
    description "The destination port of the packet";
  }
  leaf src-zone {
    type string;
    description "The source security zone of the packet";
  }
  leaf dst-zone {
    type string;
    description "The destination security zone of the packet";
  }
  leaf rule-id {
    type uint8;
    mandatory true;
    description "The ID of the rule being triggered";
  }
  leaf rule-name {
    type string;
    mandatory true;
    description "The name of the rule being triggered";
}
leaf profile {
    type string;
    description
        "Security profile that traffic matches.";
}
leaf raw-info {
    type string;
    description
        "The information describing the packet
         triggering the event.";
}
}
grouping i2nsf-nsf-event-type-content {
    description
        "A set of common IPv4-related NSF event
         content elements";
    leaf dst-ip {
        type inet:ipv4-address;
        description
            "The destination IP address of the packet";
    }
    leaf dst-port {
        type inet:port-number;
        description
            "The destination port of the packet";
    }
    leaf rule-id {
        type uint8;
        mandatory true;
        description
            "The ID of the rule being triggered";
    }
    leaf rule-name {
        type string;
        mandatory true;
        description
            "The name of the rule being triggered";
    }
    leaf profile {
        type string;
        description
            "Security profile that traffic matches.";
    }
    leaf raw-info {
        type string;
        description
            "The information describing the packet
triggering the event."
}
}
grouping traffic-rates {
  description
    "A set of traffic rates for statistics data";
  leaf total-traffic {
    type uint32;
    description
      "Total traffic";
  }
  leaf in-traffic-ave-rate {
    type uint32;
    description
      "Inbound traffic average rate in pps";
  }
  leaf in-traffic-peak-rate {
    type uint32;
    description
      "Inbound traffic peak rate in pps";
  }
  leaf in-traffic-ave-speed {
    type uint32;
    description
      "Inbound traffic average speed in bps";
  }
  leaf in-traffic-peak-speed {
    type uint32;
    description
      "Inbound traffic peak speed in bps";
  }
  leaf out-traffic-ave-rate {
    type uint32;
    description
      "Outbound traffic average rate in pps";
  }
  leaf out-traffic-peak-rate {
    type uint32;
    description
      "Outbound traffic peak rate in pps";
  }
  leaf out-traffic-ave-speed {
    type uint32;
    description
      "Outbound traffic average speed in bps";
  }
  leaf out-traffic-peak-speed {

type uint32;
description
"Outbound traffic peak speed in bps";
}
)
}
grouping i2nsf-system-counter-type-content{
description
"A set of system counter type contents";
leaf interface-name {

type string;
description
"Network interface name configured in NSF";
}
leaf in-total-traffic-pkts {

type uint32;
description
"Total inbound packets";
}
leaf out-total-traffic-pkts {

type uint32;
description
"Total outbound packets";
}
leaf in-total-traffic-bytes {

type uint32;
description
"Total inbound bytes";
}
leaf out-total-traffic-bytes {

type uint32;
description
"Total outbound bytes";
}
leaf in-drop-traffic-pkts {

type uint32;
description
"Total inbound drop packets";
}
leaf out-drop-traffic-pkts {

type uint32;
description
"Total outbound drop packets";
}
leaf in-drop-traffic-bytes {

type uint32;
description
"Total inbound drop bytes";
}
leaf out-drop-traffic-bytes {
    type uint32;
    description
    "Total outbound drop bytes";
}
uses traffic-rates;
}
grouping i2nsf-nsf-counters-type-content {
    description
    "A set of nsf counters type contents";
    leaf src-ip {
        type inet:ipv4-address;
        description
        "The source IP address of the packet";
    }
    leaf dst-ip {
        type inet:ipv4-address;
        description
        "The destination IP address of the packet";
    }
    leaf src-port {
        type inet:port-number;
        description
        "The source port of the packet";
    }
    leaf dst-port {
        type inet:port-number;
        description
        "The destination port of the packet";
    }
    leaf src-zone {
        type string;
        description
        "The source security zone of the packet";
    }
    leaf dst-zone {
        type string;
        description
        "The destination security zone of the packet";
    }
    leaf src-region {
        type string;
        description
        "Source region of the traffic";
    }
    leaf dst-region {
        type string;
        description
        "Destination region of the traffic";
    }
}

"Destination region of the traffic";
}
leaf policy-id {
    type uint8;
    description "The ID of the policy being triggered";
}
leaf policy-name {
    type string;
    description "The name of the policy being triggered";
}
leaf src-user {
    type string;
    description "User who generates traffic";
}
leaf protocol {
    type identityref {
        base protocol-type;
    }
    description "Protocol type of traffic";
}
leaf app {
    type string;
    description "Application type of traffic";
}
}

notification 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";
    }
    uses characteristics;
    uses i2nsf-system-alarm-type-content;
    uses common-monitoring-data;
}
notification 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;
}
notification nsf-detection-flood {
    description
        "This notification is sent, when a specific flood type is detected";
    leaf event-name {
        type identityref {
            base SEC-EVENT-DDOS;
        }
        description
            "The event name for nsf-detection-flood";
    }
    uses i2nsf-nsf-event-type-content;
    leaf sub-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 flood, HTTPS flood, HTTP 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-rate {
    type uint32;
    description
        "The PPS rate of attack traffic";
}
leaf attack-speed {
    type uint32;
    description
        "The BPS speed of attack traffic";
}
uses common-monitoring-data;
}
notification nsf-detection-session-table {
    description
        "This notification is sent, when a session table
        event is detected";
    leaf current-session {
        type uint8;
        description
            "The number of concurrent sessions";
    }
    leaf maximum-session {
        type uint8;
        description
            "The maximum number of sessions that the session
            table can support";
    }
    leaf threshold {
        type uint8;
        description
            "The threshold triggering the event";
    }
    uses common-monitoring-data;
}
notification 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
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).";
}

notification 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 sub-attack-type {
        type identityref {
            base intrusion-attack-type;
        } 
        description
            "The sub attack type for intrusion attack";
    } 
    uses common-monitoring-data;
} 

notification 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 common-monitoring-data;
}
notification nsf-detection-web-attack {
  description
  "This notification is sent, when an attack event is detected";
  uses i2nsf-nsf-event-type-content-extend;
  leaf sub-attack-type {
    type identityref {
      base web-attack-type;
    }
    description
    "Concrete web attack type, e.g., sql injection, command injection, XSS, 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, Unknown";
}
uses common-monitoring-data;
}
notification 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:ipv4-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;
}
notification 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 amount of
        cpu usage wrt platform resources";
    }
    leaf memory-usage {
        type uint8;
        description
        "Specifies the amount of memory usage";
    }
    leaf disk-usage {
        type uint8;
        description
        "Specifies the amount of disk usage";
    }
    leaf disk-left {
        type uint8;
        description
        "Specifies the amount of disk left";
    }
    leaf session-num {
        type uint8;
        description
        "The current systems
        running status";
    }
}
leaf process-num {
    type uint8;
    description
        "The total number of process";
}
leaf in-traffic-rate {
    type uint32;
    description
        "The total inbound traffic rate in pps";
}
leaf out-traffic-rate {
    type uint32;
    description
        "The total outbound traffic rate in pps";
}
leaf in-traffic-speed {
    type uint32;
    description
        "The total inbound traffic speed in bps";
}
leaf out-traffic-speed {
    type uint32;
    description
        "The total outbound traffic speed in bps";
}
uses characteristics;
}
notification 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;
    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, Unknown";
}

notification nsf-log-ddos {
  description
  "This notification is sent, if there is
  a new DDoS event log entry in the nsf log";

  leaf attack-type {
    type identityref {
      base ddos-attack-type;
    }
    description
    "The ddos attack type for
    nsf-log-ddos notification";
  }

  leaf attack-ave-rate {
    type uint32;
    description
    "The ave PPS of attack traffic";
  }

  leaf attack-ave-speed {
    type uint32;
    description
    "the ave bps of attack traffic";
  }

  leaf attack-pkt-num {
    type uint32;
    description
    "the number of attack packets";
  }

  leaf attack-src-ip {
    type inet:ipv4-address;
    description
    "The source IP addresses of attack
    traffics. If there are a large
amount of IP addresses, then
pick a certain number of resources
according to different rules."
}
leaf action {
  type log-action;
  description
    "Action type: allow, alert,
      block, discard, declare,
      block-ip, block-service";
}
uses characteristics;
uses common-monitoring-data;
}
notification nsf-log-virus {
  description
    "This notification is sent, if there is
    a new virus event log entry in the nsf log";
  leaf attack-type {
    type identityref {
      base virus-type;
    }
    description
      "The virus type for nsf-log-virus notification";
  }
  leaf action {
    type log-action;
    description
      "Action type: allow, alert,
        block, discard, declare,
        block-ip, block-service";
  }
  leaf os{
    type string;
    description
      "simple os information";
  }
  leaf time {
    type yang:date-and-time;
    mandatory true;
    description
      "Indicate the time when the message
       is generated";
  }
  uses characteristics;
  uses common-monitoring-data;
}
notification nsf-log-intrusion {
description
"This notification is sent, if there is a new intrusion event log entry in the nsf log";
leaf attack-type {
  type identityref {
    base intrusion-attack-type;
  }
  description
  "The intrusion attack type for nsf-log-intrusion notification";
}
leaf action {
  type log-action;
  description
  "Action type: allow, alert, block, discard, declare, block-ip, block-service";
}
leaf time {
  type yang:date-and-time;
  mandatory true;
  description
  "Indicate the time when the message is generated";
}
leaf attack-rate {
  type uint32;
  description
  "The PPS of attack traffic";
}
leaf attack-speed {
  type uint32;
  description
  "The bps of attack traffic";
}
uses characteristics;
uses common-monitoring-data;
}
notification nsf-log-botnet {
  description
  "This notification is sent, if there is a new botnet event log in the nsf log";
  leaf attack-type {
    type identityref {
      base botnet-attack-type;
    }
    description
    "The botnet attack type for
nsf-log-botnet notification;
}
leaf action {
    type log-action;
    description
        "Action type: allow, alert,
           block, discard, declare,
           block-ip, block-service";
}
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;
}
notification 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;
}
notification 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:ipv4-address;
    description
        "The victim's IP";
}


"IP 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;
}
notification nsf-log-web-attack {
  description
    "This notification is sent, if there is a new web-attack event in the nsf log";
leaf attack-type {
  type identityref {
    base web-attack-type;
  }
  description
    "The web attack type for
    nsf-log-web-attack notification";
}
leaf rsp-code {
  type string;
  description
    "Response code";
}
leaf req-clientapp {
  type string;
  description
    "The client application";
}
leaf req-cookies {
  type string;
  description
    "Cookies";
}
leaf req-host {
  type string;
  description
    "The domain name of the requested host";
}
leaf raw-info {
  type string;
  description
    "The information describing
    the packet triggering the event.";
}
uses characteristics;
uses common-monitoring-data;
}
container counters {
  description
    "This is probably better covered by an import
    as this will not be notifications.
    Counter are not very suitable as telemetry, maybe
    via periodic subscriptions, which would still
    violate principle of least surprise.";
  container system-interface {
    description
      "The system counter type is interface counter";
    uses characteristics;
  }
}
uses i2nsf-system-counter-type-content;
uses common-monitoring-data;
}
container nsf-firewall {
  description
    "The nsf counter type is firewall counter";
  uses characteristics;
  uses i2nsf-nsf-counters-type-content;
  uses traffic-rates;
}
container nsf-policy-hits {
  description
    "The counters of policy hit";
  uses characteristics;
  uses i2nsf-nsf-counters-type-content;
  uses common-monitoring-data;
  leaf hit-times {
    type uint32;
    description
    "The hit times for policy";
  }
}

Figure 2: Data Model of Monitoring

11. IANA Considerations

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

    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 [RFC6020][RFC7950].

    name: ietf-i2nsf-monitor
    prefix: iim
    reference: RFC XXXX
12. 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 security controller 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 security controller). 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 NSF’s sending massive notifications to the security controller, the rate limiting or similar mechanisms should be considered in an NSF and security controller, whether in advance or just in the process of DDoS attack.

13. References

13.1. Normative References

13.2.  Informative References

[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., Kim, E., Ahn, T., Kumar, R., and S. Hares,
[I-D.ietf-i2nsf-nsf-facing-interface-dm]
Kim, J., Jeong, J., J., P., Hares, S., and Q.
Lin, "I2NSF Network Security Function-Facing Interface
YANG Data Model", draft-ietf-i2nsf-nsf-facing-interface-
dm-06 (work in progress), June 2019.

[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-04 (work in
progress), June 2019.

[I-D.ietf-i2nsf-terminology]
Hares, S., Strassner, J., Lopez, D., Xia, L., and H.
Birkholz, "Interface to Network Security Functions (I2NSF)
Terminology", draft-ietf-i2nsf-terminology-08 (work in
progress), July 2019.

[I-D.yang-i2nsf-nfv-architecture]
Yang, H., Kim, Y., Jeong, J., and J. Kim, "I2NSF on the
NFV Reference Architecture", draft-yang-i2nsf-nfv-
arquitect-05 (work in progress), July 2019.

[I-D.yang-i2nsf-security-policy-translation]
Yang, J., Jeong, J., and J. Kim, "Security Policy
Translation in Interface to Network Security
Functions", draft-yang-i2nsf-security-policy-translation-03 (work in
progress), March 2019.


[RFC6087] Bierman, A., "Guidelines for Authors and Reviewers of YANG
Data Model Documents", RFC 6087, DOI 10.17487/rfc6087,

Kumar, "Framework for Interface to Network Security
Functions", RFC 8329, DOI 10.17487/rfc8329, February 2018,

BCP 215, RFC 8340, DOI 10.17487/rfc8340, March 2018,
Appendix A. Changes from draft-ietf-i2nsf-nsf-monitoring-data-model-00

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

- In Section 2.1, Requirements Notation is updated.
- In Section 2.2, the reference [RFC8329] is added.
- In Section 2.3, the reference [RFC8342] is added.
- In Section 11, the reference [RFC6020] is added.
- Many editorial errors have been corrected.

Appendix B. 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 MSIT, Korea, under the ITRC (Information Technology Research Center) support program (IITP-2019-2017-0-01633) supervised by the IITP.

Appendix C. Contributors

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

- Jinyong Tim Kim (Sungkyunkwan University)
- Dongjin Hong (Sungkyunkwan University)
- Dacheng Zhang (Huawei)
- Yi Wu (Aliababa Group)
- Rakesh Kumar (Juniper Networks)
- Anil Lohiya (Juniper Networks)
Authors’ Addresses

Jaehoon Paul Jeong  
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

Chaehong Chung  
Department of Electronic, Electrical and Computer Engineering  
Sungkyunkwan University  
2066 Seobu-Ro, Jangan-Gu  
Suwon, Gyeonggi-Do 16419  
Republic of Korea  
Phone: +82 10 8541 7158  
EMail: darkhong@skku.edu

Susan Hares  
Huawei  
7453 Hickory Hill  
Saline, MI 48176  
USA  
Phone: +1-734-604-0332  
EMail: shares@ndzh.com

Liang Xia (Frank)  
Huawei  
101 Software Avenue, Yuhuatai District  
Nanjing, Jiangsu  
China  
EMail: Frank.xialiang@huawei.com
Abstract

This document defines an information model and a YANG data model for Registration Interface between Security Controller and Developer’s Management System (DMS) in the Interface to Network Security Functions (I2NSF) framework to register Network Security Functions (NSF) of the DMS into the Security Controller. The objective of these information and data models is to support NSF capability registration and query via I2NSF Registration Interface.

Status of This Memo

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

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

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

This Internet-Draft will expire on January 25, 2020.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents
Table of Contents

1.  Introduction ................................................. 3
2.  Requirements Language ..................................... 3
3.  Terminology .................................................. 3
4.  Objectives .................................................... 4
5.  Information Model ............................................ 4
   5.1.  NSF Capability Registration ............................. 5
      5.1.1.  NSF Capability Information .......................... 6
      5.1.2.  NSF Access Information ............................... 8
   5.2.  NSF Capability Query ..................................... 8
6.  Data Model ..................................................... 8
   6.1.  YANG Tree Diagram ....................................... 8
      6.1.1.  Definition of Symbols in Tree Diagrams ............ 9
      6.1.2.  I2NSF Registration Interface ....................... 9
      6.1.3.  NSF Capability Information .......................... 11
      6.1.4.  NSF Access Information ............................... 12
   6.2.  YANG Data Modules ....................................... 12
7.  IANA Considerations .......................................... 17
8.  Security Considerations ...................................... 17
9.  References .................................................... 19
   9.1.  Normative References .................................... 19
   9.2.  Informative References ................................... 20
Appendix A.  XML Example of Registration Interface Data Model ..... 22
   A.1.  Example 1: Registration for Capabilities of General
        Firewall ................................................... 22
   A.2.  Example 2: Registration for Capabilities of Time based
        Firewall ................................................... 23
   A.3.  Example 3: Registration for Capabilities of Web Filter . 25
   A.4.  Example 4: Registration for Capabilities of VoIP/VoLTE
        Filter ...................................................... 27
   A.5.  Example 5: Registration for Capabilities of HTTP and
        HTTPS Flood Mitigation ..................................... 28
   A.6.  Example 6: Query for Capabilities of Time based Firewall 30
Appendix B.  NSF Lifecycle Management in NFV Environments .......... 32
Appendix C.  Changes from draft-ietf-i2nsf-registration-
            interface-dm-04 ........................................ 32
Appendix D.  Acknowledgments ...................................... 32
Appendix E.  Contributors ......................................... 32
Authors’ Addresses ............................................. 33
1. Introduction

A number of Network Security Functions (NSF) may exist in the Interface to Network Security Functions (I2NSF) framework [RFC8329]. Since each of these NSFs likely has different security capabilities from each other, it is important to register the security capabilities of the NSF into the security controller. In addition, it is required to search NSFs of some required security capabilities on demand. As an example, if additional security capabilities are required to serve some security service request(s) from an I2NSF user, the security controller should be able to request the DMS for NSFs that have the required security capabilities.

This document describes an information model (see Section 5) and a YANG [RFC7950] data model (see Section 6) for the I2NSF Registration Interface (RFC8329) between the security controller and the developer’s management system (DMS) to support NSF capability registration and query via the registration interface. It also describes the operations which should be performed by the security controller and the DMS via the Registration Interface using the defined model.

2. Requirements Language

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

3. Terminology

This document uses the following terms defined in [i2nsf-terminology], [capability-dm], [RFC8329], [supa-policy-data-model], and [supa-policy-info-model]

- Network Security Function (NSF): A function that is responsible for specific treatment of received packets. A Network Security Function can act at various layers of a protocol stack (e.g., at the network layer or other OSI layers). Sample Network Security Service Functions are as follows: Firewall, Intrusion Prevention/Detection System (IPS/IDS), Deep Packet Inspection (DPI), Application Visibility and Control (AVC), network virus and malware scanning, sandbox, Data Loss Prevention (DLP), Distributed Denial of Service (DDoS) mitigation and TLS proxy.

- Data Model: A data model is a representation of concepts of interest to an environment in a form that is dependent on data
repository, data definition language, query language, implementation language, and protocol. [supa-policy-info-model]

- Information Model: An information model is a representation of concepts of interest to an environment in a form that is independent of data repository, data definition language, query language, implementation language, and protocol. [supa-policy-info-model]

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

4. Objectives

- Registering NSFs to I2NSF framework: Developer’s Management System (DMS) in I2NSF framework is typically run by an NSF vendor, and uses Registration Interface to provide NSFs developed by the NSF vendor to Security Controller. DMS registers NSFs and their capabilities to I2NSF framework through Registration Interface. For the registered NSFs, Security Controller maintains a catalog of the capabilities of those NSFs.

- Updating the capabilities of registered NSFs: After an NSF is registered into Security Controller, some modifications on the capability of the NSF may be required later. In this case, DMS uses Registration Interface to update the capability of the NSF, and this update should be reflected on the catalog of NSFs.

- Querying DMS about some required capabilities: In cases that some security capabilities are required to serve the security service request from an I2NSF user, Security Controller searches through the registered NSFs to find ones that can provide the required capabilities. But Security Controller might fail to find any NSFs having the required capabilities among the registered NSFs. In this case, Security Controller need to request DMS for additional NSF(s) that can provide the required security capabilities via Registration Interface.

5. Information Model

The I2NSF registration interface is used by Security Controller and Developer’s Management System (DMS) in I2NSF framework. The following summarizes the operations done through the registration interface:
1) DMS registers NSFs and their capabilities to Security Controller via the registration interface. DMS also uses the registration interface to update the capabilities of the NSFs registered previously.

2) In case that Security Controller fails to find any registered NSF that can provide some required capabilities, Security Controller queries DMS about NSF(s) having the required capabilities via the registration interface.

Figure 1 shows the information model of the I2NSF registration interface, which consists of two submodels: NSF capability registration and NSF capability query. Each submodel is used for the operations listed above. The remainder of this section will provide in-depth explanations of each submodel.

5.1. NSF Capability Registration

This submodel is used by DMS to register an NSF to Security Controller. Figure 2 shows how this submodel is constructed. The most important part in Figure 2 is the NSF capability, and this specifies the set of capabilities that the NSF to be registered can offer. The NSF Name contains a unique name of this NSF with the specified set of capabilities. When registering the NSF, DMS additionally includes the network access information of the NSF which is required to enable network communications with the NSF.

The following will further explain the NSF capability information and the NSF access information in more detail.
5.1.1. NSF Capability Information

NSF Capability Information basically describes the security capabilities of an NSF. In Figure 3, we show capability objects of an NSF. Following the information model of NSF capabilities defined in [capability-dm], we share the same I2NSF security capabilities: Time Capabilities, Event Capabilities, Condition Capabilities, Action Capabilities, Resolution Strategy Capabilities, Default Action Capabilities, and IPsec Method [i2nsf-ipsec]. Also, NSF Capability Information additionally contains the performance capabilities of an NSF as shown in Figure 3.
Figure 3: NSF Capability Information

5.1.1.1. Performance Capabilities

This information represents the processing capability of an NSF. This information can be used to determine whether the NSF is in congestion by comparing this with the workload that the NSF currently undergoes. Moreover, this information can specify an available amount of each type of resources such as processing power which are available on the NSF. (The registration interface can control the usages and limitations of the created instance and make the appropriate request according to the status.) As illustrated in Figure 4, this information consists of two items: Processing and Bandwidth. Processing information describes the NSF’s available processing power. Bandwidth describes the information about available network amount in two cases, outbound, inbound. This two information can be used for the NSF’s instance request.
5.1.2. NSF Access Information

NSF Access Information contains the followings that are required to communicate with an NSF: IPv4 address, IPv6 address, port number, and supported transport protocol(s) (e.g., Virtual Extensible LAN (VXLAN) [RFC 7348], Generic Protocol Extension for VXLAN (VXLAN-GPE) [nvo3-vxlan-gpe], Generic Route Encapsulation (GRE), Ethernet etc.). In this document, NSF Access Information is used to identify a specific NSF instance (i.e., NSF Access Information is the signature (unique identifier) of an NSF instance in the overall system).

5.2. NSF Capability Query

Security Controller may require some additional capabilities to serve the security service request from an I2NSF user, but none of the registered NSFs has the required capabilities. In this case, Security Controller makes a description of the required capabilities by using the NSF capability information sub-model in Section 5.1.1, and sends DMS a query about which NSF(s) can provide these capabilities.

6. Data Model

6.1. YANG Tree Diagram

This section provides the YANG Tree diagram of the I2NSF registration interface.
6.1.1. Definition of Symbols in Tree Diagrams

A simplified graphical representation of the data model is used in this section. The meaning of the symbols used in the following diagrams [RFC8431] is as follows:

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

Abbreviations before data node names: "rw" means configuration (read-write) and "ro" state data (read-only).

Symbols after data node names: "?" means an optional node and "*" denotes a "list" and "leaf-list".

Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").

Ellipsis ("...") stands for contents of subtrees that are not shown.

6.1.2. I2NSF Registration Interface

```
module : ietf-i2nsf-reg-interface
  +--rw nsf-capability-registration
    |  uses i2nsf-nsf-registrations

rpcs :
  +---x nsf-capability-query
    |  uses i2nsf-nsf-capability-query
```

Figure 5: YANG Tree of I2NSF Registration Interface

The I2NSF registration interface is used for the following purposes. Developer’s Management System (DMS) registers NSFs and their capabilities into Security Controller via the registration interface. In case that Security Controller fails to find any NSF among the registered NSFs which can provide some required capabilities, Security Controller uses the registration interface to query DMS about NSF(s) having the required capabilities. The following sections describe the YANG data models to support these operations.

6.1.2.1. NSF Capability Registration

This section expands the i2nsf-nsf-registrations in Figure 5.
When registering an NSF to Security Controller, DMS uses this module to describe what capabilities the NSF can offer. DMS includes the network access information of the NSF which is required to make a network connection with the NSF as well as the capability description of the NSF.

6.1.2.2. NSF Capability Query

This section expands the i2nsf-nsf-capability-query in Figure 5.

Security Controller may require some additional capabilities to provide the security service requested by an I2NSF user, but none of the registered NSFs has the required capabilities. In this case,
Security Controller makes a description of the required capabilities using this module and then queries DMS about which NSF(s) can provide these capabilities. Use NETCONF RPCs to send a NSF capability query. Input data is query-i2nsf-capability-info and output data is nsf-access-info. In Figure 7, the ietf-i2nsf-capability refers to the module defined in [capability-dm].

6.1.3. NSF Capability Information

This section expands the i2nsf-nsf-capability-info in Figure 6 and Figure 7.

NSF Capability Information

---rw i2nsf-nsf-capability-info
  +---rw i2nsf-capability
    | uses ietf-i2nsf-capability
  +---rw nsf-performance-capability
    | uses i2nsf-nsf-performance-capability

Figure 8: YANG Tree of I2NSF NSF Capability Information

In Figure 8, the ietf-i2nsf-capability refers to the module defined in [capability-dm]. The i2nsf-nsf-performance-capability is used to specify the performance capability of an NSF.

6.1.3.1. NSF Performance Capability

This section expands the i2nsf-nsf-performance-capability in Figure 8.

NSF Performance Capability

---rw i2nsf-nsf-performance-capability
  +---rw processing
    | +---rw processing-average  uint16
    | +---rw processing-peak     uint16
  +---rw bandwidth
    | +---rw outbound
    |   | +---rw outbound-average  uint16
    |   | +---rw outbound-peak     uint16
    | +---rw inbound
    |   | +---rw inbound-average  uint16
    |   | +---rw inbound-peak      uint16

Figure 9: YANG Tree of I2NSF NSF Performance Capability
This module is used to specify the performance capabilities of an NSF when registering or initiating the NSF.

6.1.4. NSF Access Information

This section expands the i2nsf-nsf-access-info in Figure 6.

```
+--rw i2nsf-nsf-access-info
   +--rw nsf-instance-name      string
   +--rw i2nsf-nsf-address      nsf-address
   +--rw nsf-port-number        inet:port-number
```

Figure 10: YANG Tree of I2NSF NSF Access Information

This module contains the network access information of an NSF that is required to enable network communications with the NSF.

6.2. YANG Data Modules

This section provides YANG modules of the data model for the registration interface between Security Controller and Developer’s Management System, as defined in Section 5.

<CODE BEGINS> file "ietf-i2nsf-reg-interface@2019-07-24.yang"

```
module ietf-i2nsf-reg-interface {
  yang-version 1.1;
  prefix nsfreg;

  import ietf-inet-types {
    prefix inet;
    reference "RFC 6991";
  }
  import ietf-i2nsf-capability {
    prefix capa;
    reference "draft-ietf-i2nsf-capability-data-model-05";
  }

  organization
  "IETF I2NSF (Interface to Network Security Functions) Working Group";
```

<CODE ENDS>
This module defines a YANG data model for I2NSF registration interface.

Copyright (c) <2019> IETF Trust and the persons identified as authors of the code. All rights reserved.

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

This version of this YANG module is part of draft-ietf-i2nsf-registration-interface-dm-05; see the draft itself for full legal notices."

revision 2019-07-24 {
  description "The fifth revision";
  reference
    "draft-ietf-i2nsf-registration-interface-dm-05";
}

typedef nsf-address {
  type union {
    type inet:ipv4-address;
    type inet:ipv6-address;
  }
  description "IPv4/IPv6 address of this NSF";
}
rpc i2nsf-nsf-capability-query {
  description
  "Description of the capabilities that the Security Controller requests to the DMS";
  input {
    container query-i2nsf-capability-info {
      description
      "Description of the capabilities to request";
      uses "capa:nsf-capabilities";
      reference
      "draft-ietf-i2nsf-capability-data-model-05";
    }
  }
  output {
    container nsf-access-info {
      description
      "Network access information of an NSF with the requested capabilities";
      uses i2nsf-nsf-access-info;
    }
  }
}

container i2nsf-nsf-registrations {
  description
  "Information of an NSF that DMS registers to Security Controller";
  list i2nsf-nsf-capability-registration {
    key "nsf-name";
    description
    "Required information for registration";
    leaf nsf-name {
      type string;
      mandatory true;
      description
      "Unique name of this registered NSF";
    }
    container nsf-capability-info {
      description
      "Capability description of this NSF";
      uses i2nsf-nsf-capability-info;
    }
    container nsf-access-info {
      description
      "Network access information of this NSF";
      uses i2nsf-nsf-access-info;
    }
  }
}
grouping i2nsf-nsf-performance-capability {
  description
  "Description of the performance capabilities of an NSF";
  container processing {
    description
    "Processing power of an NSF in the unit of GHz (gigahertz)";
    leaf processing-average {
      type uint16;
      description
      "Average processing power";
    }
    leaf processing-peak {
      type uint16;
      description
      "Peak processing power";
    }
  }
  container bandwidth {
    description
    "Network bandwidth available on an NSF in the unit of Gbps (gigabits per second)";
    container outbound {
      description
      "Outbound network bandwidth";
      leaf outbound-average {
        type uint16;
        description
        "Average outbound bandwidth";
      }
      leaf outbound-peak {
        type uint16;
        description
        "Peak outbound bandwidth";
      }
    }
    container inbound {
      description
      "Inbound network bandwidth";
      leaf inbound-average {
        type uint16;
        description
        "Average inbound bandwidth";
      }
    }
  }
}
"Average inbound bandwidth";
}
leaf inbound-peak {
  type uint16;
  description
  "Peak inbound bandwidth";
}
}
}

grouping i2nsf-nsf-capability-info {
  description
  "Capability description of an NSF";
  container i2nsf-capability {
    description
    "Description of the security capabilities of an NSF";
    uses "capa:nsf-capabilities";
    reference "draft-ietf-i2nsf-capability-data-model-05";
  }
  container nsf-performance-capability {
    description
    "Description of the performance capabilities of an NSF";
    uses i2nsf-nsf-performance-capability;
  }
}

grouping i2nsf-nsf-access-info {
  description
  "Information required to access an NSF";
  leaf nsf-instance-name {
    type string;
    description
    "Unique name of this NSF instance";
  }
  leaf i2nsf-nsf-address {
    type nsf-address;
    description
    "IPv4/IPv6 address of this NSF";
  }
  leaf nsf-port {
    type inet:port-number;
    description
    "Port available on this NSF";
  }
}
7. IANA Considerations

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

Registrant Contact: The IESG.
XML: N/A; the requested URI is an XML namespace.

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

Name:      ietf-i2nsf-reg-interface
Prefix:    nsfreg
Reference: RFC XXXX

8. Security Considerations

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

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

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:
o i2nsf-nsf-registrations: The attacker may exploit this to register a compromised or malicious NSF instead of a legitimate NSF to the Security Controller.

o i2nsf-nsf-performance-capability: The attacker may provide incorrect information of the performance capability of any target NSF by illegally modifying this.

o i2nsf-nsf-capability-info: The attacker may provide incorrect information of the security capability of any target NSF by illegally modifying this.

o i2nsf-nsf-access-info: The attacker may provide incorrect network access information of any target NSF by illegally modifying this.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

o i2nsf-nsf-registrations: The attacker may try to gather some sensitive information of a registered NSF by sniffing this.

o i2nsf-nsf-performance-capability: The attacker may gather the performance capability information of any target NSF and misuse the information for subsequent attacks.

o i2nsf-nsf-capability-info: The attacker may gather the security capability information of any target NSF and misuse the information for subsequent attacks.

o i2nsf-nsf-access-info: The attacker may gather the network access information of any target NSF and misuse the information for subsequent attacks.

The RPC operation in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control access to this operation. The following is the operation and its sensitivity/vulnerability:

o i2nsf-nsf-capability-query: The attacker may exploit this RPC operation to deteriorate the availability of the DMS and/or gather the information of some interested NSFs from the DMS.
9. References

9.1. Normative References


9.2. Informative References


[supa-policy-data-model]

[supa-policy-info-model]
Appendix A. XML Example of Registration Interface Data Model

This section describes XML examples of the I2NSF Registration Interface data model under the assumption of registering several types of NSFs and querying NSF capability.

A.1. Example 1: Registration for Capabilities of General Firewall

This section shows an XML example for registering the capabilities of general firewall.

```xml
<i2nsf-nsf-registrations
  <i2nsf-nsf-capability-registration>
    <nsf-name>general_firewall_capability</nsf-name>
    <nsf-capability-info>
      <i2nsf-capability>
        <condition-capabilities>
          <generic-nsf-capabilities>
            <ipv4-capability>capa:ipv4-protocol</ipv4-capability>
            <ipv4-capability>capa:exact-ipv4-address</ipv4-capability>
            <ipv4-capability>capa:range-ipv4-address</ipv4-capability>
            <tcp-capability>capa:exact-tcp-port-num</tcp-capability>
            <tcp-capability>capa:range-tcp-port-num</tcp-capability>
          </generic-nsf-capabilities>
        </condition-capabilities>
        <action-capabilities>
          <ingress-action-capability>capa:pass</ingress-action-capability>
          <ingress-action-capability>capa:drop</ingress-action-capability>
          <ingress-action-capability>capa:alert</ingress-action-capability>
          <egress-action-capability>capa:pass</egress-action-capability>
          <egress-action-capability>capa:drop</egress-action-capability>
          <egress-action-capability>capa:alert</egress-action-capability>
        </action-capabilities>
        <ipsec-method>capa:ikeless</ipsec-method>
      </i2nsf-capability>
    </nsf-capability-info>
  </i2nsf-nsf-capability-registration>
</i2nsf-nsf-registrations>
```

Figure 12: Configuration XML for Registration of General Firewall

Figure 12 shows the configuration XML for registering the general firewall and its capabilities as follows.

1. The instance name of the NSF is general_firewall.
2. The NSF can inspect protocol, exact IPv4 address, and range IPv4 address for IPv4 packets.
3. The NSF can inspect exact port number and range port number for tcp packets.
4. The NSF can determine whether the packets are allowed to pass, drop, or alert.
5. The NSF can support IPsec not through IKEv2, but through a Security Controller [i2nsf-ipsec].
6. The NSF can have processing power and bandwidth.
7. The location of the NSF is 2001:DB8:8:4::2.
8. The port of the NSF is 3000.

A.2. Example 2: Registration for Capabilities of Time based Firewall

This section shows an XML example for registering the capabilities of time-based firewall.

```xml
<i2nsf-nsf-registrations
 xmlns="urn:ietf:params:xml:ns:yang:i2nsf-reg-interface"
 xmlns:capa="urn:ietf:params:xml:ns:yang:i2nsf-capability">
```

<i2nsf-nsf-capability-registration>
  <nsf-name>time_based_firewall_capability</nsf-name>
  <nsf-capability-info>
    <i2nsf-capability>
      <time-capabilities>absolute-time</time-capabilities>
      <time-capabilities>periodic-time</time-capabilities>
      <condition-capabilities>
        <generic-nsf-capabilities>
          <ipv4-capa>capa:ipv4-protocol</ipv4-capa>
          <ipv4-capa>capa:exact-ipv4-address</ipv4-capa>
          <ipv4-capa>capa:range-ipv4-address</ipv4-capa>
        </generic-nsf-capabilities>
      </condition-capabilities>
      <action-capabilities>
        <ingress-action-capa>capa:pass</ingress-action-capa>
        <ingress-action-capa>capa:drop</ingress-action-capa>
        <ingress-action-capa>capa:alert</ingress-action-capa>
        <egress-action-capa>capa:pass</egress-action-capa>
        <egress-action-capa>capa:drop</egress-action-capa>
        <egress-action-capa>capa:alert</egress-action-capa>
      </action-capabilities>
      <ipsec-method>capa:ikey</ipsec-method>
    </i2nsf-capability>
  </nsf-capability-info>
  <nsf-performance-capability>
    <processing>
      <processing-average>1000</processing-average>
      <processing-peak>5000</processing-peak>
    </processing>
    <bandwidth>
      <outbound>
        <outbound-average>1000</outbound-average>
        <outbound-peak>5000</outbound-peak>
      </outbound>
      <inbound>
        <inbound-average>1000</inbound-average>
        <inbound-peak>5000</inbound-peak>
      </inbound>
    </bandwidth>
  </nsf-performance-capability>
</i2nsf-nsf-capability-registration>
Figure 13 shows the configuration XML for registering the time-based firewall and its capabilities as follows.

1. The instance name of the NSF is time_based_firewall.
2. The NSF can enforce the security policy rule according to absolute time and periodic time.
3. The NSF can inspect protocol, exact IPv4 address, and range IPv4 address for IPv4 packets.
4. The NSF can determine whether the packets are allowed to pass, drop, or alert.
5. The NSF can support IPsec through IKEv2 [i2nsf-ipsec].
6. The NSF can have processing power and bandwidth.
7. The location of the NSF is 2001:DB8:8:4::3.
8. The port of the NSF is 3000.

A.3. Example 3: Registration for Capabilities of Web Filter

This section shows an XML example for registering the capabilities of web filter.

```xml
<i2nsf-nsf-registrations
 <i2nsf-nsf-capability-registration>
   <nsf-name>web_filter_capability</nsf-name>
   <nsf-capability-info>
     <i2nsf-capability>
       <condition-capabilities>
         <advanced-nsf-capabilities>
           <url-capa>capa:user-defined</url-capa>
         </advanced-nsf-capabilities>
       </condition-capabilities>
       <action-capabilities>
         <ingress-action-capa>capa:pass</ingress-action-capa>
         <ingress-action-capa>capa:drop</ingress-action-capa>
         <ingress-action-capa>capa:alert</ingress-action-capa>
         <egress-action-capa>capa:pass</egress-action-capa>
         <egress-action-capa>capa:drop</egress-action-capa>
         <egress-action-capa>capa:alert</egress-action-capa>
       </action-capabilities>
     </i2nsf-capability>
   </nsf-capability-info>
 </i2nsf-nsf-capability-registration>
</i2nsf-nsf-registrations>
```
Figure 14 shows the configuration XML for registering the web filter, and its capabilities are as follows.

1. The instance name of the NSF is web_filter.
2. The NSF can inspect url for http and https packets.
3. The NSF can determine whether the packets are allowed to pass, drop, or alert.
4. The NSF can support IPsec not through IKEv2, but through a Security Controller [i2nsf-ipsec].
5. The NSF can have processing power and bandwidth.
6. The location of the NSF is 2001:DB8:8:4::4.
7.  The port of the NSF is 3000.

A.4.  Example 4: Registration for Capabilities of VoIP/VoLTE Filter

This section shows an XML example for registering the capabilities of VoIP/VoLTE filter.

```xml
<i2nsf-nsf-registrations
  xmlns="urn:ietf:params:xml:ns:yang:i2nsf-reg-interface"
  xmlns:capa="urn:ietf:params:xml:ns:yang:i2nsf-capability">
  <i2nsf-nsf-capability-registration>
    <nsf-name>voip_volte_filterCapability</nsf-name>
    <nsf-capability-info>
      <i2nsf-capability>
        <condition-capabilities>
          <advanced-nsf-capabilities>
            <voip-volte-capa>capa:voice-id</voip-volte-capa>
          </advanced-nsf-capabilities>
        </condition-capabilities>
        <action-capabilities>
          <ingress-action-capa>capa:pass</ingress-action-capa>
          <ingress-action-capa>capa:drop</ingress-action-capa>
          <ingress-action-capa>capa:alert</ingress-action-capa>
          <egress-action-capa>capa:pass</egress-action-capa>
          <egress-action-capa>capa:drop</egress-action-capa>
          <egress-action-capa>capa:alert</egress-action-capa>
        </action-capabilities>
        <ipsec-method>capa:ikeless</ipsec-method>
      </i2nsf-capability>
      <nsf-performance-capability>
        <processing>
          <processing-average>1000</processing-average>
          <processing-peak>5000</processing-peak>
        </processing>
        <bandwidth>
          <outbound>
            <outbound-average>1000</outbound-average>
            <outbound-peak>5000</outbound-peak>
          </outbound>
          <inbound>
            <inbound-average>1000</inbound-average>
            <inbound-peak>5000</inbound-peak>
          </inbound>
        </bandwidth>
      </nsf-performance-capability>
    </nsf-capability-info>
    <nsf-access-info>
      <nsf-instance-name>voip_volte_filter</nsf-instance-name>
    </nsf-access-info>
  </i2nsf-nsf-capability-registration>
</i2nsf-nsf-registrations>
```
Figure 15: Configuration XML for Registration of VoIP/VoLTE Filter

Figure 15 shows the configuration XML for registering VoIP/VoLTE filter, and its capabilities are as follows.

1. The instance name of the NSF is voip volte_filter.
2. The NSF can inspect voice id for VoIP/VoLTE packets.
3. The NSF can determine whether the packets are allowed to pass, drop, or alert.
4. The NSF can support IPsec not through IKEv2, but through a Security Controller [i2nsf-ipsec].
5. The NSF can have processing power and bandwidth.
6. The location of the NSF is 2001:DB8:8::5.
7. The port of the NSF is 3000.

A.5. Example 5: Registration for Capabilities of HTTP and HTTPS Flood Mitigation

This section shows an XML example for registering the capabilities of http and https flood mitigation.

```xml
<i2nsf-nsf-registrations
 <i2nsf-nsf-capability-registration>
   <nsf-name>
     http_and_https_flood_mitigation_capability
   </nsf-name>
   <nsf-capability-info>
     <i2nsf-capability>
       <condition-capabilities>
         <advanced-nsf-capabilities>
           <antiddos-capa>capa:http-flood-action</antiddos-capa>
           <antiddos-capa>capa:https-flood-action</antiddos-capa>
         </advanced-nsf-capabilities>
       </condition-capabilities>
     </i2nsf-capability>
   </nsf-capability-info>
 </i2nsf-nsf-capability-registration>
</i2nsf-nsf-registrations>
```
<condition-capabilities>
</condition-capabilities>
<action-capabilities>
    <ingress-action-capa>capa:pass</ingress-action-capa>
    <ingress-action-capa>capa:drop</ingress-action-capa>
    <ingress-action-capa>capa:alert</ingress-action-capa>
    <egress-action-capa>capa:pass</egress-action-capa>
    <egress-action-capa>capa:drop</egress-action-capa>
    <egress-action-capa>capa:alert</egress-action-capa>
</action-capabilities>
<ipsec-method>capa:ike</ipsec-method>

<nsf-performance-capability>
    <processing>
        <processing-average>1000</processing-average>
        <processing-peak>5000</processing-peak>
    </processing>
    <bandwidth>
        <outbound>
            <outbound-average>1000</outbound-average>
            <outbound-peak>5000</outbound-peak>
        </outbound>
        <inbound>
            <inbound-average>1000</inbound-average>
            <inbound-peak>5000</inbound-peak>
        </inbound>
    </bandwidth>
</nsf-performance-capability>

<nsf-access-info>
    <nsf-instance-name>
        http_and_https_flood_mitigation
    </nsf-instance-name>
    <i2nsf-nsf-address>2001:DB8:8:4::6</i2nsf-nsf-address>
    <nsf-port-address>3000</nsf-port-address>
</nsf-access-info>
</i2nsf-nsf-capability-registration>
</i2nsf-nsf-registrations>

Figure 16: Configuration XML for Registration of HTTP and HTTPS Flood Mitigation

Figure 16 shows the configuration XML for registering the http and https flood mitigator, and its capabilities are as follows.

1. The instance name of the NSF is http_and_https_flood_mitigation.
2. The NSF can control the amount of packets for http and https packets.

3. The NSF can determine whether the packets are allowed to pass, drop, or alert.

4. The NSF can support IPsec through IKEv2 [i2nsf-ipsec].

5. The NSF can have processing power and bandwidth.

6. The location of the NSF is 2001:DB8:8:4::6.

7. The port of the NSF is 3000.

A.6. Example 6: Query for Capabilities of Time based Firewall

This section shows an XML example for querying the capabilities of time-based firewall.
<rpc message-id="101"
     xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
   <i2nsf-nsf-capability-query
     xmlns="urn:ietf:params:xml:ns:yang:i2nsf-reg-interface"
     xmlns:capa="urn:ietf:params:xml:ns:yang:i2nsf-capability">
     <query-i2nsf-capability-info>
       <time-capabilities>absolute-time</time-capabilities>
       <time-capabilities>periodic-time</time-capabilities>
       <condition-capabilities>
         <generic-nsf-capabilities>
           <ipv4-capa>capa:ipv4-protocol</ipv4-capa>
           <ipv4-capa>capa:exact-ipv4-address</ipv4-capa>
           <ipv4-capa>capa:range-ipv4-address</ipv4-capa>
         </generic-nsf-capabilities>
       </condition-capabilities>
       <action-capabilities>
         <ingress-action-capa>capa:pass</ingress-action-capa>
         <ingress-action-capa>capa:drop</ingress-action-capa>
         <ingress-action-capa>capa:alert</ingress-action-capa>
         <egress-action-capa>capa:pass</egress-action-capa>
         <egress-action-capa>capa:drop</egress-action-capa>
         <egress-action-capa>capa:alert</egress-action-capa>
       </action-capabilities>
       <ipsec-method>capa:ikeless</ipsec-method>
     </query-i2nsf-capability-info>
   </i2nsf-nsf-capability-query>
</rpc>

<rpc-reply message-id="101"
     xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
   <nsf-access-info
     xmlns="urn:ietf:params:xml:ns:yang:i2nsf-reg-interface">
     <nsf-instance-name>time-based-firewall</nsf-instance-name>
     <i2nsf-nsf-address>2001:DB8:8:4::7</i2nsf-nsf-address>
     <nsf-port-address>8080</nsf-port-address>
   </nsf-access-info>
</rpc-reply>

Figure 17: Configuration XML for Query of Time-based Firewall

Figure 17 shows the XML configuration for querying the capabilities of the time-based firewall.
Appendix B. NSF Lifecycle Management in NFV Environments

Network Functions Virtualization (NFV) can be used to implement I2NSF framework. In NFV environments, NSFs are deployed as virtual network functions (VNFs). Security Controller can be implemented as an Element Management (EM) of the NFV architecture, and is connected with the VNF Manager (VNFM) via the Ve-Vnfm interface [nfv-framework]. Security Controller can use this interface for the purpose of the lifecycle management of NSFs. If some NSFs need to be instantiated to enforce security policies in the I2NSF framework, Security Controller could request the VNFM to instantiate them through the Ve-Vnfm interface. Or if an NSF, running as a VNF, is not used by any traffic flows for a time period, Security Controller may request deinstantiating it through the interface for efficient resource utilization.

Appendix C. Changes from draft-ietf-i2nsf-registration-interface-dm-04

The following changes have been made from draft-ietf-i2nsf-registration-interface-dm-04:

- This version is revised according to the comments from Reshad Rahman who reviewed this document as a YANG doctor.

Appendix D. 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).

Appendix E. Contributors

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

- Jinyong Tim Kim (Sungkyunkwan University)
- Chaehong Chung (Sungkyunkwan University)
- Susan Hares (Huawei)
- Diego R. Lopez (Telefonica)
Authors’ Addresses

Sangwon Hyun
Department of Computer Engineering
Chosun University
309, Pilmun-daero, Dong-gu
Gwangju, Jeollanam-do 61452
Republic of Korea
EMail: shyun@chosun.ac.kr

Jaehoon Paul Jeong
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

Taekyun Roh
Department of Electronic, Electrical and Computer Engineering
Sungkyunkwan University
2066 Seobu-Ro, Jangan-Gu
Suwon, Gyeonggi-Do 16419
Republic of Korea
Phone: +82 31 290 7222
Fax: +82 31 299 6673
EMail: tkroh0198@skku.edu

Sarang Wi
Department of Electronic, Electrical and Computer Engineering
Sungkyunkwan University
2066 Seobu-Ro, Jangan-Gu
Suwon, Gyeonggi-Do 16419
Republic of Korea
Phone: +82 31 290 7222
Fax: +82 31 299 6673
EMail: dnl9795@skku.edu
Security Policy Translation in Interface to Network Security Functions
draft-yang-i2nsf-security-policy-translation-04

Abstract

This document proposes a scheme of security policy translation (i.e., Security Policy Translator) in Interface to Network Security Functions (I2NSF) Framework. When I2NSF User delivers a high-level security policy for a security service, Security Policy Translator in Security Controller translates it into a low-level security policy for Network Security Functions (NSFs). For this security policy translation, this document specifies the mapping between a high-level security policy based the Consumer-Facing Interface YANG data model and a low-level security policy based on the NSF-Facing Interface YANG data model. Also, it describes an architecture of a security policy translator along with an NSF database, and the process of security policy translation with the NSF database.

Status of This Memo

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

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

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

This Internet-Draft will expire on January 25, 2020.

Copyright Notice

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

This document defines a scheme of a security policy translation in Interface to Network Security Functions (I2NSF) Framework [RFC8329]. First of all, this document explains the necessity of a security
policy translator (shortly called policy translator) in the I2NSF framework.

The policy translator resides in Security Controller in the I2NSF framework and translates a high-level security policy to a low-level security policy for Network Security Functions (NSFs). A high-level policy is specified by I2NSF User in the I2NSF framework and is delivered to Security Controller via Consumer-Facing Interface [consumer-facing-inf-dm]. It is translated into a low-level policy by Policy Translator in Security Controller and is delivered to NSFs to execute the rules corresponding to the low-level policy via NSF-Facing Interface [nsf-facing-inf-dm].

2. Terminology

This document uses the terminology specified in [i2nsf-terminology] [RFC8329].

3. Necessity for Policy Translator

Security Controller acts as a coordinator between I2NSF User and NSFs. Also, Security Controller has capability information of NSFs that are registered via Registration Interface [registration-inf-dm] by Developer’s Management System [RFC8329]. As a coordinator, Security Controller needs to generate a low-level policy in the form of security rules intended by the high-level policy, which can be understood by the corresponding NSFs.

A high-level security policy is specified by RESTCONF/YANG [RFC8040][RFC6020], and a low-level security policy is specified by NETCONF/YANG [RFC6241][RFC6020]. The translation from a high-level security policy to the corresponding low-level security policy will be able to rapidly elevate I2NSF in real-world deployment. A rule in a high-level policy can include a broad target object, such as employees in a company for a security service (e.g., firewall and web filter). Such employees may be from human resource (HR) department, software engineering department, and advertisement department. A keyword of employee needs to be mapped to these employees from various departments. This mapping needs to be handled by a policy translator in a flexible way while understanding the intention of a policy specification. Let us consider the following two policies:

- Block my son’s computers from malicious websites.
- Drop packets from the IP address 10.0.0.1 and 10.0.0.3 to harm.com and illegal.com
The above two sentences are examples of policies for blocking malicious websites. Both policies are for the same operation. However, NSF cannot understand the first policy, because the policy does not have any specified information for NSF. To set up the policy at an NSF, the NSF MUST receive at least the source IP address and website address for an operation. It means that the first sentence is NOT compatible for an NSF policy. Conversely, when I2NSF Users request a security policy to the system, they never make a security policy like the second example. For generating a security policy like the second sentence, the user MUST know that the NSF needs to receive the specified information, source IP address and website address. It means that the user understands the NSF professionally, but there are not many professional users in a small size of company or at a residential area. In conclusion, the I2NSF User prefers to issue a security policy in the first sentence, but an NSF will require the same policy as the second sentence with specific information. Therefore, an advanced translation scheme of security policy is REQUIRED in I2NSF.

This document proposes an approach using Automata theory [Automata] for the policy translation, such as Deterministic Finite Automaton (DFA) and Context Free Grammar (CFG). Note that Automata theory is the foundation of programming language and compiler. Thus, with this approach, I2NSF User can easily specify a high-level security policy that will be enforced into the corresponding NSFs with a compatibly low-level security policy with the help of Policy Translator. Also, for easy management, a modularized translator structure is proposed.

4. Design of Policy Translator

Commonly used security policies are created as XML(Extensible Markup Language) [XML] files. A popular way to change the format of an XML file is to use an XSLT (Extensible Stylesheet Language Transformation) [XSLT] document. XSLT is an XML-based language to transform an input XML file into another output XML file. However, the use of XSLT makes it difficult to manage the policy translator and to handle the registration of new capabilities of NSFs. With the necessity for a policy translator, this document describes a policy translator based on Automata theory.

4.1. Overall Structure of Policy Translator
Figure 1 shows the overall design for Policy Translator in Security Controller. There are three main components for Policy Translator: Data Extractor, Data Converter, and Policy Generator.

Extractor is a DFA-based module for extracting data from a high-level policy which I2NSF User delivered via Consumer-Facing Interface. Data Converter converts the extracted data to the capabilities of target NSFs for a low-level policy. It refers to an NSF Database (DB) in order to convert an abstract subject or object into the corresponding concrete subject or object (e.g., IP address and website URL). Policy Generator generates a low-level policy which will execute the NSF capabilities from Converter.
4.2. DFA-based Data Extractor

4.2.1. Design of DFA-based Data Extractor

Figure 2: DFA Architecture of Data Extractor

Figure 2 shows a design for Data Extractor in the policy translator. If a high-level policy contains data along the hierarchical structure of the standard Consumer-Facing Interface YANG data model [consumer-facing-inf-dm], data can be easily extracted using the state transition machine, such as DFA. The extracted data can be processed and used by an NSF to understand it. Extractor can be constructed by designing a DFA with the same hierarchical structure as a YANG data model.

After constructing a DFA, Data Extractor can extract all of data in the entered high-level policy by using state transitions. Also, the DFA can easily detect the grammar errors of the high-level policy. The extracting algorithm of Data Extractor is as follows:

1. Start from the ‘accepter’ state.
2. Read the next tag from the high-level policy.
3. Transit to the corresponding state.
4. If the current state is in ‘extractor’, extract the corresponding data, and then go back to step 2.
5. If the current state is in ‘middle’, go back to step 2.

6. If there is no possible transition and arrived at ‘accepter’ state, the policy has no grammar error. Otherwise, there is a grammar error, so stop the process with failure.

4.2.2. Example Scenario for Data Extractor

```
<I2NSF>
  <name>block_web</name>
  <cond>
    <src>Son’s_PC</src>
    <dest>malicious_websites</dest>
  </cond>
  <action>block</action>
</I2NSF>
```

Figure 3: The Example of High-level Policy

```
+----------+
| accepter |
+----------+

+---------------------------------------------+
|                                       ^ |
|                                      <I2NSF>| |
|                                      v |
+---------------------------------------------+

v | middle 1 |
+---------------------------------------------+

v | extractor 1 |
+---------------------------------------------+

block_web | ^ | middle 2 | ^ |
||<src>v | ^ |
||<dest>v |
+---------------------------------------------+

v | extractor 2 |
+---------------------------------------------+

Son’s_PC | ^ |
+---------------------------------------------+

v | extractor 3 |
+---------------------------------------------+

block | ^ |
+---------------------------------------------+

v | malicious_websites |
+---------------------------------------------+

Figure 4: The Example of Data Extractor

To explain the Data Extractor process by referring to an example scenario, assume that Security Controller received a high-level
policy for a web-filtering as shown in Figure 3. Then we can construct DFA-based Data Extractor by using the design as shown in Figure 2. Figure 4 shows the architecture of Data Extractor that is based on the architecture in Figure 2 along with the input high-level policy in Figure 3. Data Extractor can automatically extract all of data in the high-level policy according to the following process:

1. Start from the ‘accepter’ state.
2. Read the first opening tag called ‘<I2NSF>’, and transit to the ‘middle 1’ state.
3. Read the second opening tag called ‘<name>’, and transit to the ‘extractor 1’ state.
4. The current state is an ‘extractor’ state. Extract the data of ‘name’ field called ‘block_web’.
5. Read the second closing tag called ‘</name>’, and go back to the ‘middle 1’ state.
6. Read the third opening tag called ‘<cond>’, and transit to the ‘middle 2’ state.
7. Read the fourth opening tag called ‘<src>’, and transit to the ‘extractor 2’ state.
8. The current state is an ‘extractor’ state. Extract the data of ‘src’ field called ‘Son’s_PC’.
9. Read the fourth closing tag called ‘</src>’, and go back to the ‘middle 2’ state.
10. Read the fifth opening tag called ‘<dest>’, and transit to the ‘extractor 3’ state.
11. The current state is an ‘extractor’ state. Extract the data of ‘dest’ field called ‘malicious_websites’.
12. Read the fifth closing tag called ‘</dest>’, and go back to the ‘middle 2’ state.
13. Read the third closing tag called ‘</cond>’, and go back to the ‘middle 1’ state.
14. Read the sixth opening tag called ‘<action>’, and transit to the ‘extractor 4’ state.
15. The current state is an ‘extractor’ state. Extract the data of ‘action’ field called ‘block’.

16. Read the sixth closing tag called ‘</action>’, and go back to the ‘middle 1’ state.

17. Read the first closing tag called ‘</I2NSF>’, and go back to the ‘accepter’ state.

18. There is no further possible transition, and the state is finally on ‘accepter’ state. There is no grammar error in Figure 3 so the scanning for data extraction is finished.

The above process is constructed by an extracting algorithm. After finishing all the steps of the above process, Data Extractor can extract all of data in Figure 3, ‘block_web’, ‘Son’s_PC’, ‘malicious’, and ‘block’.

Since the translator is modularized into a DFA structure, a visual understanding is feasible. Also, the performance of Data Extractor is excellent compared to one-to-one searching of data for a particular field. In addition, the management is efficient because the DFA completely follows the hierarchy of Consumer-Facing Interface. If I2NSF User wants to modify the data model of a high-level policy, it only needs to change the connection of the relevant DFA node.

4.3. Data Converter

4.3.1. Role of Data Converter

Every NSF has its own unique capabilities. The capabilities of an NSF are registered into Security Controller by a Developer’s Management System, which manages the NSF, via Registration Interface. Therefore, Security Controller already has all information about the capabilities of NSFs. This means that Security Controller can find target NSFs with only the data (e.g., subject and object for a security policy) of the high-level policy by comparing the extracted data with all capabilities of each NSF. This search process for appropriate NSFs is called by policy provisioning, and it eliminates the need for I2NSF User to specify the target NSFs explicitly in a high-level security policy.

Data Converter selects target NSFs and converts the extracted data into the capabilities of selected NSFs. If Security Controller uses this data converter, it can provide the policy provisioning function to I2NSF User automatically. Thus, the translator design provides big benefits to the I2NSF Framework.
4.3.2. NSF Database

The NSF Database contains all the information needed to convert high-level policy data to low-level policy data. The contents of NSF Database are classified as the following two: "endpoint information" and "NSF capability information".

The first is "endpoint information". Endpoint information is necessary to convert an abstract high-level policy data such as Son's_PC, malicious to a specific low-level policy data such as 10.0.0.1, illegal.com. In the high-level policy, the range of endpoints for applying security policy MUST be provided abstractly. Thus, endpoint information is needed to specify the abstracted high-level policy data. Endpoint information is provided by I2NSF User as the high-level policy through Consumer-Facing Interface, and Security Controller builds NSF Database based on received information.

The second is "NSF capability information". Since capability is information that allows NSF to know what features it can support, NSF capability information is used in policy provisioning process to search the appropriate NSFs through the security policy. NSF capability information is provided by Developer’s Management System (DMS) through Registration Interface, and Security Controller builds NSF Database based on received information. In addition, if the NSF sends monitoring information such as initiating information to Security Controller through NSF-Facing Interface, Security Controller can modify NSF Database accordingly.
Figure 5 shows an Entity-Relationship Diagram (ERD) of NSF Database designed to include both endpoint information received from I2NSF User and NSF capability information received from DMS. By designing the NSF database based on the ERD, all the information necessary for security policy translation can be stored, and the network system administrator can manage the NSF database efficiently.

ERD was expressed by using Crow’s Foot notation. Crow’s Foot notation represents a relationship between entities as a line and represents the cardinality of the relationship as a symbol at both ends of the line. Attributes prefixed with * are key values of each entity. A link with two vertical lines represents one-to-one mapping, and a bird-shaped link represents one-to-many mapping. An NSF entity stores the NSF name (nsf_name), NSF specification (inbound, outbound, bandwidth), and NSF activation (activated). A Capability entity stores the capability name (capa_name) and the
index of the capability field in a Registration Interface Data Model (capa_index). An Endpoint entity stores the keyword of abstract data conversion from I2NSF User (keyword). A Field entity stores the field name (field_name), the index of the field index in an NSF-Facing Interface Data Model, and converted data by referring to the Endpoint entity and a 'convert' relationship.

4.3.3. Data Conversion in Data Converter

Figure 6: Example of Data Conversion

Figure 6 shows an example for describing a data conversion in Data Converter. High-level policy data MUST be converted into low-level policy data which are compatible with NSFs. If a system administrator attaches a database to Data Converter, it can convert contents by referring to the database with SQL queries. Data conversion in Figure 6 is based on the following list:

- ‘Rule Name’ field does NOT need the conversion.
- ‘Source’ field SHOULD be converted into a list of target IPv4 addresses.
Internet-Draft      I2NSF Security Policy Translation          July 2019

- ‘Destination’ field SHOULD be converted into a URL category list of malicious websites.
- ‘Action’ field SHOULD be converted into the corresponding action(s) in NSF capabilities.

Figure 7 shows a mapping list of data fields between Consumer-Facing Interface Data Model and NSF-Facing Interface Data Model. Figure 7 describes the process of passing the data value to the appropriate data field of the Data Model in detail after the data conversion.

/consumer-facing/policy/policy-name
  -> mapping: /nsf-facing/i2nsf-security-policy/system-policy
    /system-policy-name

/consumer-facing/policy/rule/rule-name
  -> mapping: /nsf-facing/i2nsf-security-policy/system-policy
    /rules/rule-name

/consumer-facing/policy/rule/event/time-information/time/begin-time
  -> mapping: /nsf-facing/i2nsf-security-policy/system-policy
    /rules/time-zone/absolute-time-zone/start-time

/consumer-facing/policy/rule/event/time-information/time/end-time
  -> mapping: /nsf-facing/i2nsf-security-policy/system-policy
    /rules/time-zone/absolute-time-zone/end-time

/consumer-facing/policy/rule/condition
/firewall-condition/source-target/src-target
  -> reference: /consumer-facing/policy
    /endpoint-group/user-group/name
  -> extract: /consumer-facing/policy
    /endpoint-group/user-group/date
  -> mapping: /nsf-facing/i2nsf-security-policy
    /rule/date

/firewall-condition/source-target/src-target
  -> mapping: /nsf-facing/i2nsf-security-policy
    /system-policy/rules
    /condition-clause-container
    /packet-security-ipv4-condition
    /pkt-sec-ipv4-src/ipv4-address/ipv4
  -> extract: /consumer-facing/policy
    /endpoint-group/user-group/range-ip-address
    /start-ip-address
  -> mapping: /nsf-facing/i2nsf-security-policy
Internet-Draft      I2NSF Security Policy Translation          July 2019

/system-policy/rules
/condition-clause-container
/packet-security-ipv4-condition
/pkt-sec-ipv4-src/range-ipv4-address
/start-ipv4-address

-> extract: /consumer-facing/policy
/endpoint-group/user-group/range-ip-address
/end-ip-address

-> mapping: /nsf-facing/i2nsf-security-policy
/system-policy/rules
/condition-clause-container
/packet-security-ipv4-condition
/pkt-sec-ipv4-src/range-ipv4-address
/end-ipv4-address

/consumer-facing/policy/rule/condition
/firewall-condition/destination-target/dest-target

-> reference: /consumer-facing/policy
/endpoint-group/user-group/name

-> extract: /consumer-facing/policy
/endpoint-group/user-group/date

-> extract: /consumer-facing/policy
/endpoint-group/user-group/ip-address

-> mapping: /nsf-facing/i2nsf-security-policy
/system-policy/rules
/condition-clause-container
/packet-security-ipv4-condition
/pkt-sec-ipv4-dest/ipv4-address/ipv4

-> extract: /consumer-facing/policy
/endpoint-group/user-group
/range-ip-address/start-ip-address

-> mapping: /nsf-facing/i2nsf-security-policy
/system-policy/rules
/condition-clause-container
/packet-security-ipv4-condition
/pkt-sec-ipv4-dest/range-ipv4-address/start-ipv4-address

-> extract: /consumer-facing/policy
/endpoint-group/user-group
/range-ip-address/end-ip-address

-> mapping: /nsf-facing/i2nsf-security-policy
/system-policy/rules
/condition-clause-container
/packet-security-ipv4-condition
/pkt-sec-ipv4-dest/range-ipv4-address/end-ipv4-address
/consumer-facing/policy/rule/condition
/ddos-condition/source-target/src-target
  -> reference: /consumer-facing/policy
/endpoint-group/device-group/name
  -> extract: /consumer-facing/policy
/endpoint-group/device-group/date
  -> mapping: /nsf-facing/i2nsf-security-policy/rule/date
  -> extract: /consumer-facing/policy
/endpoint-group/device-group/ip-address
  -> mapping: /nsf-facing/i2nsf-security-policy
/system-policy/rules
/condition-clause-container
/packet-security-ipv4-condition
/pkt-sec-ipv4-src/ipv4-address/ipv4
  -> extract: /consumer-facing/policy
/endpoint-group/device-group
/range-ip-address/start-ip-address
  -> mapping: /nsf-facing/i2nsf-security-policy
/system-policy/rules
/condition-clause-container
/packet-security-ipv4-condition
/pkt-sec-ipv4-src/range-ipv4-address
/start-ipv4-address
  -> extract: /consumer-facing/policy
/endpoint-group/device-group
/range-ip-address/end-ip-address
  -> mapping: /nsf-facing/i2nsf-security-policy
/system-policy/rules
/condition-clause-container
/packet-security-ipv4-condition
/pkt-sec-ipv4-src/range-ipv4-address
/end-ipv4-address
  /consumer-facing/policy/rule/condition
/ddos-condition/destination-target/dest-target
  -> reference: /consumer-facing/policy
/endpoint-group/device-group/name
  -> extract: /consumer-facing/policy
/endpoint-group/device-group/date
  -> mapping: /nsf-facing/i2nsf-security-policy/rule/date
  -> extract: /consumer-facing/policy
/endpoint-group/device-group/ip-address
  -> mapping: /nsf-facing/i2nsf-security-policy
/system-policy/rules
/condition-clause-container
/packet-security-ipv4-condition
/pkt-sec-ipv4-dest/ipv4-address/ipv4
  -> extract: /consumer-facing/policy
/endpoint-group/device-group
/range-ip-address/start-ip-address
-> mapping: /nsf-facing/i2nsf-security-policy
/system-policy/rules
/condition-clause-container
/packet-security-ipv4-condition
/pkt-sec-ipv4-dest/range-ipv4-address
/start-ipv4-address

-> extract: /consumer-facing/policy
/endpoint-group/device-group
/range-ip-address/end-ip-address
-> mapping: /nsf-facing/i2nsf-security-policy
/system-policy/rules
/condition-clause-container
/packet-security-ipv4-condition
/pkt-sec-ipv4-dest/range-ipv4-address
/end-ipv4-address

/consumer-facing/policy/rule/condition
ddos-condition/rate-limit/packet-per-second
-> mapping: /nsf-facing/i2nsf-security-policy
/system-policy/rules/condition-clause-container
/packet-security-ddos-condition/pkt-sec-alert-rate

/consumer-facing/policy/rule/condition
custom-condition/source-target/src-target
-> reference: /consumer-facing/policy
/threat-prevention/payload-content/name
-> extract: /consumer-facing/policy
/threat-prevention/payload-content/date
-> mapping: /nsf-facing/i2nsf-security-policy
/system-policy/rules/date
-> extract: /consumer-facing/policy
/threat-prevention/payload-content/content
-> mapping: /nsf-facing/i2nsf-security-policy
/system-policy/rules
/condition-clause-container
/packet-security-payload-condition
/pkt-payload-content

/consumer-facing/policy/rule/condition
custom-condition/destination-target/dest-target
-> reference: /consumer-facing/policy
/threat-prevention/payload-content/name
-> extract: /consumer-facing/policy
/threat-prevention/payload-content/date
-> mapping: /nsf-facing/i2nsf-security-policy
/system-policy/rules/date
extract: /consumer-facing/policy
         /threat-prevention/payload-content/content
mapping: /nsf-facing/i2nsf-security-policy
         /system-policy/rules
         /condition-clause-container
         /packet-security-payload-condition
         /pkt-payload-content

/consumer-facing/policy/rule/condition
/custom-condition/threat-feed-condition
/source-target/src-target

reference: /consumer-facing/policy

extract: /consumer-facing/policy
         /threat-prevention/threat-feed-list/name
mapping: /nsf-facing/i2nsf-security-policy
         /system-policy/rules/date

extract: /consumer-facing/policy
         /threat-prevention/threat-feed-list
         /threat-feed-server/ip-address
mapping: /nsf-facing/i2nsf-security-policy
         /system-policy/rules
         /condition-clause-container
         /packet-security-ipv4-condition
         /pkt-sec-ipv4-src/ipv4-address/ipv4

extract: /consumer-facing/policy
         /threat-prevention/threat-feed-list
         /threat-feed-server/range-ip-address
         /start-ip-address
mapping: /nsf-facing/i2nsf-security-policy
         /system-policy/rules
         /condition-clause-container
         /packet-security-ipv4-condition
         /pkt-sec-ipv4-src/range-ipv4-address
         /start-ipv4-address

extract: /consumer-facing/policy
         /threat-prevention/threat-feed-list
         /threat-feed-server/range-ip-address
         /end-ip-address
mapping: /nsf-facing/i2nsf-security-policy
         /system-policy/rules
         /condition-clause-container
         /packet-security-ipv4-condition
         /pkt-sec-ipv4-src/range-ipv4-address
         /end-ipv4-address

extract: /consumer-facing/policy
         /threat-prevention/threat-feed-list
         /threat-feed-server
Internet-Draft    I2NSF Security Policy Translation    July 2019

/threat-feed-description
-> mapping: /nsf-facing/i2nsf-security-policy
   /system-policy/rules
   /condition-clause-container
   /packet-security-ipv4-condition
   /ipv4-description

/consumer-facing/policy/rule/condition
/custom-condition/threat-feed-condition
/destination-target/dest-target
  -> reference: /consumer-facing/policy
  /threat-prevention/threat-feed-list/name
  -> extract: /consumer-facing/policy
  /threat-prevention/threat-feed-list/date
  -> mapping: /nsf-facing/i2nsf-security-policy
  /system-policy/rules/date
  -> extract: /consumer-facing/policy
  /threat-prevention/threat-feed-list
  /threat-feed-server/ip-address
  -> mapping: /nsf-facing/i2nsf-security-policy
  /system-policy/rules
  /condition-clause-container
  /packet-security-ipv4-condition
  /pkt-sec-ipv4-dest/ipv4-address/ipv4
  -> extract: /consumer-facing/policy
  /threat-prevention/threat-feed-list
  /threat-feed-server/range-ip-address
  /start-ip-address
  -> mapping: /nsf-facing/i2nsf-security-policy
  /system-policy/rules
  /condition-clause-container
  /packet-security-ipv4-condition
  /pkt-sec-ipv4-dest/range-ipv4-address
  /start-ipv4-address
  -> extract: /consumer-facing/policy
  /threat-prevention/threat-feed-list
  /threat-feed-server/range-ip-address
  /end-ip-address
  -> mapping: /nsf-facing/i2nsf-security-policy
  /system-policy/rules
  /condition-clause-container
  /packet-security-ipv4-condition
  /pkt-sec-ipv4-dest/range-ipv4-address
  /end-ipv4-address
  -> extract: /consumer-facing/policy
  /threat-prevention/threat-feed-list
  /threat-feed-server
  /threat-feed-description
-> mapping: /nsf-facing/i2nsf-security-policy
   /system-policy/rules
   /condition-clause-container
   /packet-security-ipv4-condition
   /ipv4-description

/consumer-facing/policy/rule/action/name
   -> mapping: /nsf-facing/i2nsf-security-policy
      /system-policy/rules/action-clause-container
      /packet-action/ingress-action
   -> mapping: /nsf-facing/i2nsf-security-policy
      /system-policy/rules/action-clause-container
      /packet-action/egress-action
   -> mapping: /nsf-facing/i2nsf-security-policy
      /system-policy/rules/action-clause-container
      /packet-action/log-action
   -> mapping: /nsf-facing/i2nsf-security-policy
      /system-policy/rules/action-clause-container
      /advanced-action/content-security-control
   -> mapping: /nsf-facing/i2nsf-security-policy
      /system-policy/rules/action-clause-container
      /advanced-action/attack-mitigation-control

/consumer-facing/policy-rule/ipsec-method
   -> mapping: /nsf-facing/i2nsf-ipsec

/consumer-facing/policy-domain/name
   -> mapping: /nsf-facing/i2nsf-security-policy
      /system-policy/rule-group/groups/group-name

Figure 7: Mapping Information for Data Conversion

4.3.4. Policy Provisioning
Generator searches for proper NSFs which can cover all of capabilities in the high-level policy. Generator searches for target NSFs by comparing only NSF capabilities which is registered by Vendor Management System. This process is called by "policy provisioning" because Generator finds proper NSFs by using only the policy. If target NSFs are found by using other data which is not included in a user's policy, it means that the user already knows the specific knowledge of an NSF in the I2NSF Framework. Figure 8 shows an example of policy provisioning. In this example, log-keeper NSF and web-filter NSF are selected for covering capabilities in the security policy. All of capabilities can be covered by two selected NSFs.
4.4. CFG-based Policy Generator

Generator makes low-level security policies for each target NSF with the extracted data. We constructed Generator by using Context Free Grammar (CFG). CFG is a set of production rules which can describe all possible strings in a given formal language (e.g., programming language). The low-level policy also has its own language based on a YANG data model of NSF-Facing Interface. Thus, we can construct the productions based on the YANG data model. The productions that make up the low-level security policy are categorized into two types, 'Content Production' and 'Structure Production'.

4.4.1. Content Production

Content Production is for injecting data into low-level policies to be generated. A security manager (i.e., a person (or software) to make productions for security policies) can construct Content Productions in the form of an expression as the following productions:

- \[ \text{cont_prod} \rightarrow \text{cont_prod}\text{cont_prod} \] (Where duplication is allowed.)
- \[ \text{cont_prod} \rightarrow <\text{cont_tag}>\text{cont_data}</\text{cont_tag}> \]
- \[ \text{cont_data} \rightarrow \text{data}_1 | \text{data}_2 | \ldots | \text{data}_n \]

Square brackets mean non-terminal state. If there are no non-terminal states, it means that the string is completely generated. When the duplication of content tag is allowed, the security manager adds the first production for a rule. If there is no need to allow duplication, the first production can be skipped because it is an optional production.

The second production is the main production for Content Production because it generates the tag which contains data for low-level policy. Last, the third production is for injecting data into a tag which is generated by the second production. If data is changed for an NSF, the security manager needs to change "only the third production" for data mapping in each NSF.

For example, if the security manager wants to express a low-level policy for source IP address, Content Production can be constructed in the following productions:

- \[ \text{cont_ipv4} \rightarrow \text{cont_ipv4}\text{cont_ipv4} \] (Allow duplication.)
- \[ \text{cont_ipv4} \rightarrow <\text{ipv4}>\text{cont_ipv4_data}</\text{ipv4}> \]
4.4.2. Structure Production

Structure Production is for grouping other tags into a hierarchy. The security manager can construct Structure Production in the form of an expression as the following production:

\[\text{struct_prod} \rightarrow <\text{struct_tag}>[\text{prod}_1]...[\text{prod}_n]</\text{struct_tag}>\]

Structure Production can be expressed as a single production. The above production means to group other tags by the name of a tag which is called by ‘struct_tag’. \([\text{prod}_x]\) is a state for generating a tag which wants to be grouped by Structure Production. \([\text{prod}_x]\) can be both Content Production and Structure Production. For example, if the security manager wants to express the low-level policy for the I2NSF tag, which is grouping ‘name’ and ‘rules’, Structure Production can be constructed as the following production where \([\text{cont}_\text{name}]\) is the state for Content Production and \([\text{struct_rule}]\) is the state for Structure Production.

\[\text{struct_i2nsf} \rightarrow <\text{I2NSF}>[\text{cont}_\text{name}][\text{struct_rule}]/\text{I2NSF}>\]

4.4.3. Generator Construction

The security manager can build a generator by combining the two productions which are described in Section 4.4.1 and Section 4.4.2. Figure 9 shows the CFG-based Generator construction of the web-filter NSF. It is constructed based on the NSF-Facing Interface Data Model in [nsf-facing-inf-dm]. According to Figure 9, the security manager can express productions for each clause as in following CFG:

1. \([\text{cont}_\text{name}] \rightarrow <\text{rule-name}>[\text{cont}_\text{name}_\text{data}]</\text{rule-name}>\)
2. \([\text{cont}_\text{name}_\text{data}] \rightarrow \text{block}_\text{web}\)
3. \([\text{cont}_\text{ipv4}] \rightarrow [\text{cont}_\text{ipv4}][\text{cont}_\text{ipv4}] \text{(Allow duplication)}\)
4. \([\text{cont}_\text{ipv4}] \rightarrow <\text{ipv4}>[\text{cont}_\text{ipv4}_\text{data}]</\text{ipv4}>\)
5. \([\text{cont}_\text{ipv4}_\text{data}] \rightarrow 10.0.0.1 | 10.0.0.3\)
6. \([\text{cont}_\text{url}] \rightarrow [\text{cont}_\text{url}][\text{cont}_\text{url}] \text{(Allow duplication)}\)
7. \([\text{cont}_\text{url}] \rightarrow <\text{url}>[\text{cont}_\text{url}_\text{data}]</\text{url}>\)
8. \([\text{cont}_\text{url}_\text{data}] \rightarrow \text{harm.com} | \text{illegal.com}\)
9. [cont_action] -> <action>[cont_action_data]</action>
10. [cont_action_data] -> drop
12. [struct_payload] -> <payload>[cont_url]</payload>
15. [struct_i2nsf] -> <I2NSF>[cont_name][struct_rules]</I2NSF>

Then, Generator generates a low-level policy by using the above CFG. The low-level policy is generated by the following process:

1. Start: [struct_i2nsf]
2. Production 15: <I2NSF>[cont_name][struct_rules]</I2NSF>


12. Production 6: `<I2NSF><rule-name>block_web</rule-name><rules><condition><packet><ipv4>10.0.0.1</ipv4><ipv4>10.0.0.3</ipv4></packet><[payload]<[cont_url]<[cont_url]]></condition><[cont_action]]></rules></I2NSF>


The last production has no non-terminal state, and the low-level policy is completely generated. Figure 10 shows the generated low-level policy where tab characters and newline characters are added.
Figure 9: Generator Construction for Web-Filter NSF
5. Implementation Considerations

The implementation considerations in this document include the following three: "data model auto-adaptation", "data conversion", and "policy provisioning".

5.1. Data Model Auto-adaptation

Security Controller which acts as the intermediary MUST process the data according to the data model of the connected interfaces. However, the data model can be changed flexibly depending on the situation, and Security Controller may adapt to the change. Therefore, Security Controller can be implemented for convenience so that the security policy translator can easily adapt to the change of the data model.

The translator constructs and uses the DFA to adapt to Consumer-Facing Interface Data Model. In addition, the CFG is constructed and used to adapt to NSF-Facing Interface Data Model. Both the DFA and the CFG follow the same tree structure of YANG Data Model.

The DFA starts at the node and expands operations by changing the state according to the input. Based on the YANG Data Model, a container node is defined as a middle state and a leaf node is defined as an extractor node. After that, if the nodes are connected in the same way as the hierarchical structure of the data model, Security Controller can automatically construct the DFA. The DFA can be conveniently built by investigating the link structure using the stack, starting with the root node.
The CFG starts at the leaf nodes and is grouped into clauses until all the nodes are merged into one node. A leaf node is defined as the content production, and a container node is defined as the structure production. After that, if the nodes are connected in the same way as the hierarchy of the data model, Security Controller can automatically construct the CFG. The CFG can be conveniently constructed by investigating the link structure using the priority queue data, starting with the leaf nodes.

5.2. Data Conversion

Security Controller requires the ability to materialize the abstract data in the high-level security policy and forward it to NSFs. Security Controller can receive endpoint information as keywords through the high-level security policy. At this time, if the endpoint information corresponding to the keyword is mapped and the query is transmitted to the NSF Database, the NSF Database can be conveniently registered with necessary information for data conversion. When a policy tries to establish a policy through the keyword, Security Controller searches the details corresponding to the keyword registered in the NSF Database and converts the keywords into the appropriate and specified data.

5.3. Policy Provisioning

This document stated that policy provisioning function is necessary to enable users without expert security knowledge to create policies. Policy provisioning is determined by the capability of the NSF. If the NSF has information about the capability in the policy, the probability of selection increases.

Most importantly, selected NSFs may be able to perform all capabilities in the security policy. This document recommends a study of policy provisioning algorithms that are highly efficient and can satisfy all capabilities in the security policy.

6. Features of Policy Translator Design

First, by showing a visualized translator structure, the security manager can handle various policy changes. Translator can be shown by visualizing DFA and Context-free Grammar so that the manager can easily understand the structure of Policy Translator.

Second, if I2NSF User only keeps the hierarchy of the data model, I2NSF User can freely create high-level policies. In the case of DFA, data extraction can be performed in the same way even if the order of input is changed. The design of the policy translator is
more flexible than the existing method that works by keeping the tag’s position and order exactly.

Third, the structure of Policy Translator can be updated even while Policy Translator is operating. Because Policy Translator is modularized, the translator can adapt to changes in the NSF capability while the I2NSF framework is running. The function of changing the translator’s structure can be provided through Registration Interface.

7. Security Considerations

There is no security concern in the proposed security policy translator as long as the I2NSF interfaces (i.e., Consumer-Facing Interface, NSF-Facing Interface, and Registration Interface) are protected by secure communication channels.

8. Acknowledgments

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea MSIT (Ministry of Science and ICT) (R-20160222-002755, Cloud based Security Intelligence Technology Development for the Customized Security Service Provisioning).

This work was supported in part by the MSIT under the IITP (Information Technology Research Center) support program (IITP-2018-0-01633) supervised by the IITP.

9. References

9.1. Normative References


9.2. Informative References

[Automata]

[consumer-facing-inf-dm]

[i2nsf-terminology]

[nsf-facing-inf-dm]

[registration-inf-dm]

[XML]

[XSLT]
Appendix A. Changes from draft-yang-i2nsf-security-policy-translation-03

The following changes are made from draft-yang-i2nsf-security-policy-translation-03:

- In Section 4.3.2, an Entity-Relationship Diagram (ERD) is added for describing the architecture of NSF Database. It describes the design of the NSF Database.

- In Section 4.3.3, a mapping list between Consumer-Facing Interface Data Model and NSF-Facing Interface Data Model is added for describing the data mapping process in detail after the data conversion. This section provides guidelines of data converter for a security policy translator developer. Also, this mapping helps this document to be useful as a standard document.

Authors' Addresses

Jaehoon Paul Jeong
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

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

Phone: +82 10 8520 8039
EMail: jin.hyuk@skku.edu