Workgroup: I2NSF Working Group
Internet-Draft:
draft-lingga-i2nsf-application-interface-dm-00
Published: 27 August 2021
Intended Status: Standards Track
Expires: 28 February 2022
Authors: P. Lingga, Ed.
Sungkyunkwan University
J. Jeong, Ed.
Sungkyunkwan University
Y. Choi
ETRI

I2NSF Application Interface YANG Data Model

Abstract

This document describes an information model and a YANG data model for the Application Interface between an Interface to Network Security Functions (I2NSF) Analyzer and Security Controller in an I2NSF system in a Network Functions Virtualization (NFV) environment. The information model and YANG data model is based on the I2NSF Consumer-Facing Interface for enabling feedback delivery based on the information received from the Network Security Function (NSF).

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 28 February 2022.

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

In Interface to Network Security Functions (I2NSF) [RFC8329], the Monitoring Interface [I-D.ietf-i2nsf-nsf-monitoring-data-model] is defined as an interface to collect information (e.g., network statistics, resources) from NSF(s). The information can be received by a query or a report. In a query-based, the information is obtained by a request from a client (I2NSF Analyzer). But in a report-based, the information is provided by a server (NSFs) when the notification or alarm is triggered by an event. In this model, the report-based collection information is used for realizing the Security Management Automation (SMA) in cloud-based security services [I-D.jeong-i2nsf-security-management-automation]. as the information is sent automatically by the NSFs. Figure 1 shows the I2NSF Framework for Security Management Automation.

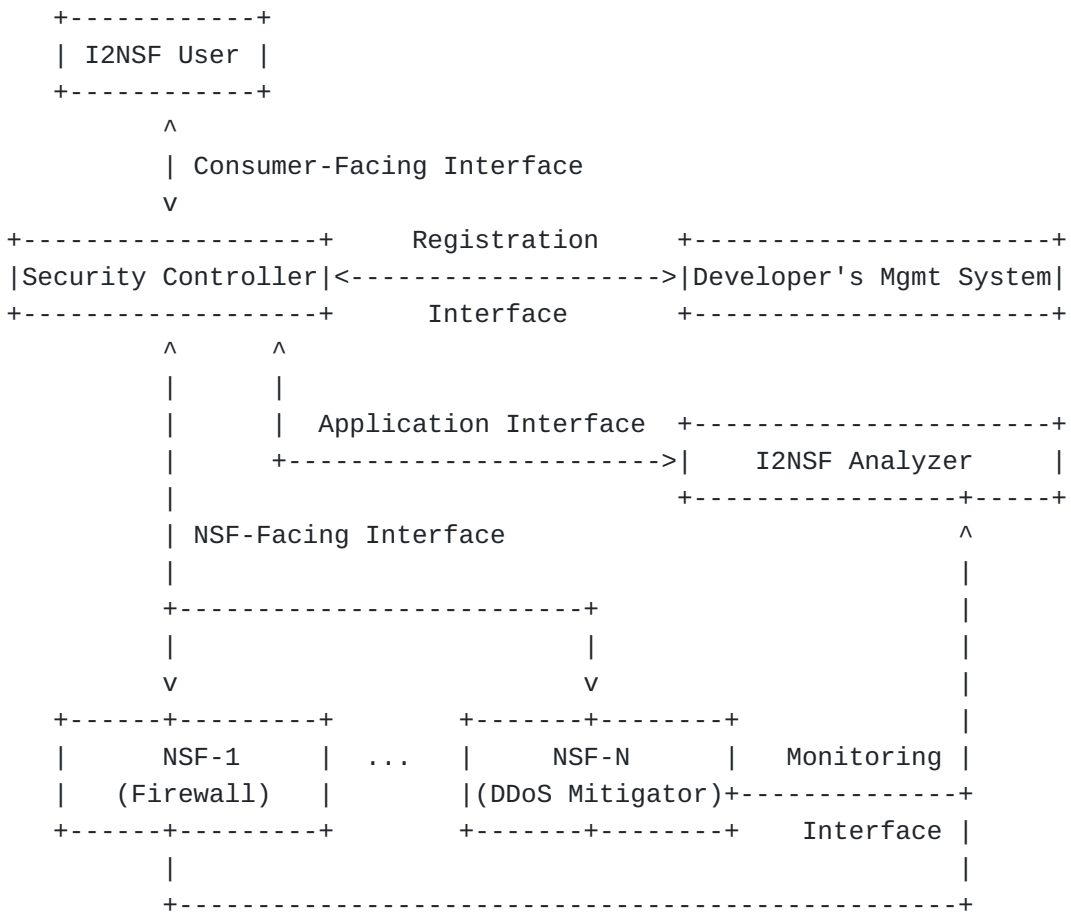

Figure 1: I2NSF Framework for Security Management Automation

The automatic reports by the NSFs are collected in a single instance (i.e., I2NSF Analyzer) to be analyzed. By analyzing the information, a new security policy can be produced to further enhance the security of the network. To create the automated system, the analyzer should be done automatically with the help of machine learning. The automated analyzer is not in the scope of this document.

The new security policy needs to be delivered from the I2NSF Analyzer to the Security Controller so the new policy can be listed and monitored properly. For that purpose, this document introduces the Application Interface as the intermediary between the I2NSF Analyzer and the Security Controller. Then the policy should be delivered directly to the NSFs by the Security Controller via the NSF-Facing Interface [I-D.ietf-i2nsf-nsf-facing-interface-dm].

The purpose of this document is to provide a standard for a feedback interface in an I2NSF Framework called Application Interface. With

the provided Application Interface, the realization of Security Management Automation (SMA) should be possible.

2. Terminology

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

This document uses the terminology described in [RFC8329].

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

3. Information Model for Application Interface

This document introduces Application Interface as an interface to deliver a report of the augmentation or generation of security policy rules created by I2NSF Analyzer to Security Controller [I-D.jeong-i2nsf-security-management-automation]. This allows Security Controller to actively reinforce the network with its security policy management. Figure 2 shows the high-level concept of Application Interface such as Policy Reconfiguration and Feedback Information.

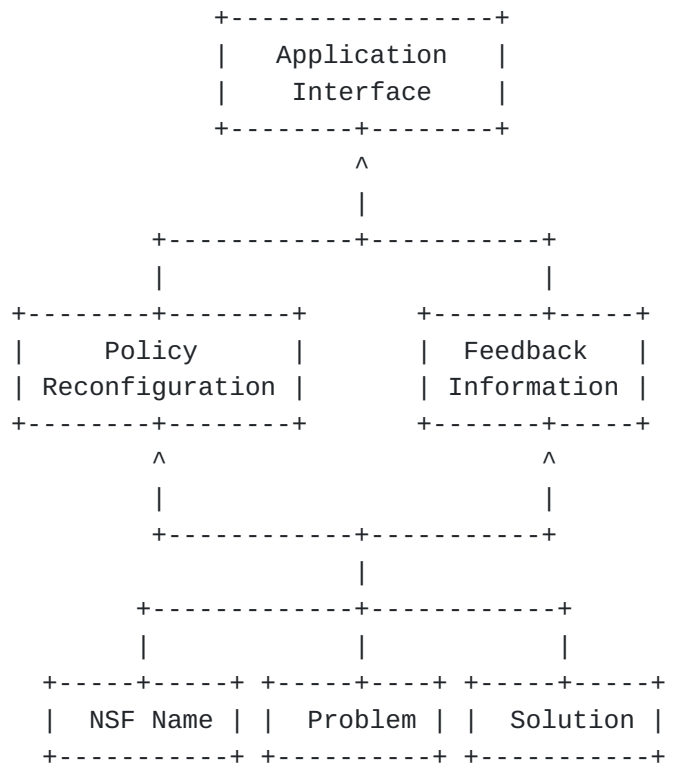

Figure 2: Diagram for Application Interface

Both policy reconfiguration and feedback information provide the following high-level abstraction:

*NSF Name: It is the name or IP address of the NSF for identifying the NSF with problem. The name is a unique string to identity an NSF, including a Fully Qualified Domain Name (FQDN).

*Problem: It describes the issue(s) in the NSF that needs to be handled.

*Solution: It specifies the possible solution(s) for the problem.

3.1. Information Model for Policy Reconfiguration

Policy reconfiguration is the rearrangement of a security policy in a different form or combination of the existing security policy to enhance the security service in the network. A policy reconfiguration is generated by the I2NSF Analyzer after receiving and analyzing monitoring information of NSF Events from an NSF [I-D.ietf-i2nsf-nsf-monitoring-data-model].

Policy reconfiguration works together with the three I2NSF interfaces defined for the I2NSF Framework, i.e., NSF-Facing

Interface [I-D.ietf-i2nsf-nsf-facing-interface-dm], NSF Monitoring Interface [I-D.ietf-i2nsf-nsf-monitoring-data-model], and Application Interface, to create a closed-loop system for reinforcing the network security. Figure 3 shows an illustration of the closed-loop system for the I2NSF Framework.

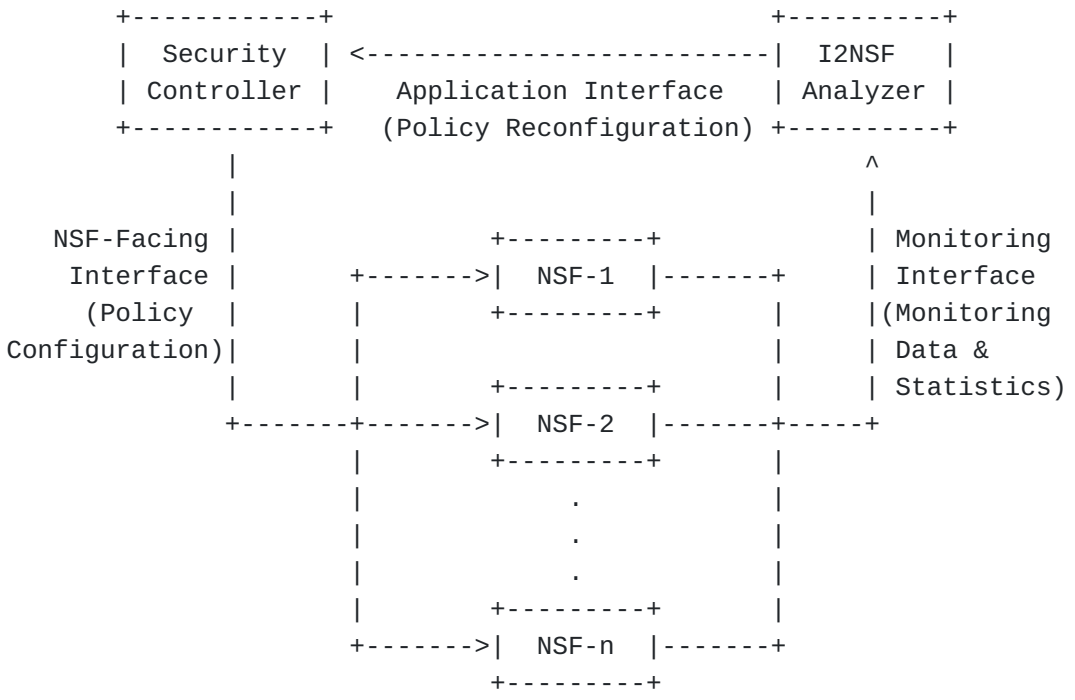

Figure 3: A Close Loop Architecture for Security Management Automation (SMA)

Figure 3 shows a close-loop system between Security Controller, NSF, and I2NSF Analyzer. The Security Controller delivers a security policy to an appropriate NSF via the NSF-Facing Interface [I-D.ietf-i2nsf-nsf-facing-interface-dm]. The NSF will prepare for a security service according to the given configuration and provide a security service for the network. The NSF SHOULD also provide monitoring information (e.g., NSF Events and System Alarms) to be analyzed. This monitoring information can be delivered from the NSF to an I2NSF Analyzer via the Monitoring Interface [I-D.ietf-i2nsf-nsf-monitoring-data-model]. Then the I2NSF Analyzer analyzes the monitoring information for the reconfiguration of an existing security policy, the generation of a new security policy, and the feedback for security system management (e.g., the scaling-up or scaling-down of resources related to NSFs). To fully automate the close-loop system, the I2NSF Analyzer should analyze the monitoring information automatically using machine learning techniques (e.g., Deep Learning [Deep-Learning]). The results of the analysis may

trigger the reconfiguration of an existing security policy or the generation of a new security policy to strengthen the network security. The reconfiguration or configuration request will be delivered from the I2NSF Analyzer to the Security Controller via the Application Interface.

To realize the close loop system, the Application Interface needs to properly follow the similar guidelines for the I2NSF Framework [RFC8329]. The Application Interface follows [I-D.ietf-i2nsf-nsf-facing-interface-dm] to create a security policy to reconfigure an existing security policy of NSF(s) or to generate a new security policy.

Application Interface holds a list of security policies so that the (re)configuration of a security policy and the feedback information can be provided to the Security Controller. Each policy consists of a list of rule to be enhanced on the NSF. Note that the synchronization of the list of security policies should be done between the Security Controller and the I2NSF Analyzer and the specific mechanism is out of the scope of this document. A (re)configured security policy rule should be able to cope with attacks or failures that can happen to the network in near future. Such a rule is reconfigured or generated by the I2NSF Analyzer to tackle a detected problem in the network. It uses the Event-Condition-Action (ECA) model as the basis for the design of I2NSF Policy (Re)configuration as described in [RFC8329] and [I-D.ietf-i2nsf-capability-data-model].

An example of Policy (Re)configuration is a DDoS Attack that is detected by a DDoS Mitigator. The DDoS Mitigator creates monitoring information and delivers it to the I2NSF Analyzer. The I2NSF Analyzer analyzes the information and generates a new policy to handle the DDoS Attack, such as a firewall rule to drop all packets from the source of the DDoS Attack.

3.2. YANG Tree Structure for Policy Reconfiguration

The YANG tree structure for policy reconfiguration is provided through the augmentation of the NSF-Facing Interface YANG Module [I-D.ietf-i2nsf-nsf-facing-interface-dm] as follows:

```
augment /nsfintf:i2nsf-security-policy:
  +--rw nsf-name?   union
  +--rw problem
     +--rw (attack-detection)?
        +--:(ddos-detected)
        |  +--rw ddos-detected
        |     +--rw attack-src-ip*     inet:ip-address
        |     +--rw attack-dst-ip*     inet:ip-prefix
        |     +--rw attack-src-port*   inet:port-number
        |     +--rw attack-dst-port*   inet:port-number
        +--:(virus-detected)
        |  +--rw virus-detected
        |     +--rw virus-name?   string
        |     +--rw file-type?    string
        |     +--rw file-name?    string
        +--:(intrusion-detected)
        |  +--rw intrusion-detected
        |     +--rw protocol?      identityref
        |     +--rw app?           identityref
        |     +--rw attack-type?   identityref
        +--:(web-attack-detected)
        |  +--rw web-attack-detected
        |     +--rw attack-type?      identityref
        |     +--rw request-method?   identityref
        |     +--rw req-uri?          string
        |     +--rw req-user-agent?   string
        |     +--rw req-cookie?       string
        |     +--rw req-host?         string
        |     +--rw response-code?    string
        +--:(voip-volte-detected)
           +--rw voip-volte-detected
              +--rw source-voice-id*        string
              +--rw destination-voice-id*   string
              +--rw user-agent*             string
```

Figure 4: YANG Tree Structure of Policy Reconfiguration

The policy reconfiguration must include the following information:

NSF Name: The name or IP address (IPv4 or IPv6) of the NSF to be configured. If the given nsf-name is not IP address, the name can be an arbitrary string including FQDN (Fully Qualified Domain Name).

Problem: The issue that is emitted by an NSF via the I2NSF Monitoring Interface. The problem for policy configuration includes the NSF Events described in NSF Monitoring Interface YANG Data Model [I-D.ietf-i2nsf-nsf-monitoring-data-model], such

as DDoS detection, Virus detection, Intrusion detection, Web-attack detection, and VoIP/VoLTE violation detection.

Solution: The solution for policy (re)configuration is the security policy that is reconfigured or generated to solve a detected attack. The security policy can be configured using the NSF-Facing Interface YANG data model [I-D.ietf-i2nsf-nsf-facing-interface-dm].

3.3. Information Model for Feedback Information

Feedback information is information about problem(s) of an NSF for a security service such as system resource over-usage or malfunction. This problem cannot be handled by creating a new policy. In the similar way with policy reconfiguration, the feedback information should be delivered from the I2NSF Analyzer to the Security Controller that will be able to handle the reported problem(s).

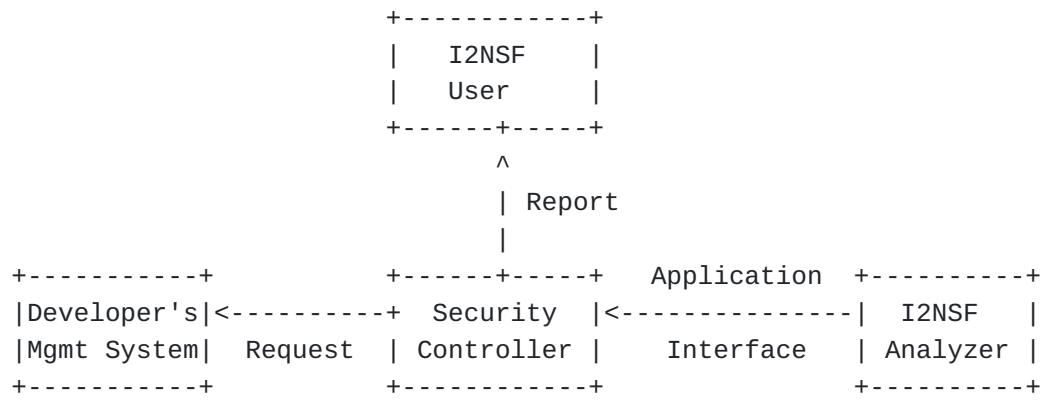

Figure 5: Handling of Feedback Information

Figure 5 shows the handling of feedback information. For feedback information, the given feedback is not a security policy, hence the Security Controller needs to take an action to handle the reported problem(s). The action includes the reporting to the I2NSF User and the requesting of the system resource management of the relevant NSF(s) to the Developer's Management System (DMS). DMS will communicate with the Management and Orchestration (MANO) Unit in the Network Functions Virtualization (NFV) Framework to deal with the system management issue(s) of the relevant NSFs [I-D.ietf-i2nsf-applicability]. The details of the handling process are out of the scope of this document.

3.4. YANG Tree Structure for Feedback Information

The YANG tree structure for feedback information is provided with the use of the NSF Monitoring Interface YANG Module [I-D.ietf-i2nsf-nsf-monitoring-data-model] as follows:

```
module: ietf-i2nsf-feedback-policy
  +--rw i2nsf-feedback-information* [nsf-name time]
     +--rw nsf-name    union
     +--rw time        yang:date-and-time
     +--rw problem
     |  +--rw (alarm-type)?
     |     +--:(memory-alarm)
     |     |  +--rw memory-alarm
     |     |     +--rw usage?      uint8
     |     |     +--rw message?    string
     |     |     +--rw duration?   uint32
     |     +--:(cpu-alarm)
     |     |  +--rw cpu-alarm
     |     |     +--rw usage?      uint8
     |     |     +--rw message?    string
     |     |     +--rw duration?   uint32
     |     +--:(disk-alarm)
     |     |  +--rw disk-alarm
     |     |     +--rw disk-id?    string
     |     |     +--rw usage?      uint8
     |     |     +--rw message?    string
     |     |     +--rw duration?   uint32
     |     +--:(hardware-alarm)
     |     |  +--rw hardware-alarm
     |     |     +--rw component-name?   string
     |     |     +--rw message?          string
     |     |     +--rw duration?         uint32
     |     +--:(interface-alarm)
     |        +--rw interface-alarm
     |           +--rw interface-id?      string
     |           +--rw interface-state?   enumeration
     |           +--rw message?           string
     |           +--rw duration?          uint32
     +--rw solution*   string
```

Figure 6: YANG Tree Structure of Feedback Information

Figure 6 shows the high-level abstraction of Feedback Information. The feedback information should include:

*NSF Name: The name or IP address (IPv4 or IPv6) of the NSF that detected the problem. If the given nsf-name is not IP address, the name can be an arbitrary string including FQDN.

*Time: The time of the delivery of the feedback information.

*Problem: The issue that is emitted by an NSF via the I2NSF Monitoring Interface. The problem for feedback information includes the system alarms described in NSF Monitoring Interface YANG Data Model [I-D.ietf-i2nsf-nsf-monitoring-data-model], such as Memory alarm, CPU alarm, Disk alarm, Hardware alarm, and Interface alarm.

*Solution: A possible solution given as feedback is in the form of a free-form string (as a high-level instruction).

4. YANG Data Model of Application Interface

This section shows the YANG module of Application Interface. The YANG module in this document is referencing to [RFC6991] [I-D.ietf-i2nsf-nsf-facing-interface-dm]

```
<CODE BEGINS> file "ietf-i2nsf-feedback-policy@2021-08-27.yang"

module ietf-i2nsf-feedback-policy {
  yang-version 1.1;
  namespace
    "urn:ietf:params:xml:ns:yang:ietf-i2nsf-feedback-policy";
  prefix nsffbck;

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

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

  import ietf-i2nsf-policy-rule-for-nsf {
    prefix nsfintf;
    reference
      "Section 4.1 of draft-ietf-i2nsf-nsf-facing-interface-dm-13";
  }

  import ietf-i2nsf-nsf-monitoring {
    prefix nsfmi;
    reference
      "Section 7 of draft-ietf-i2nsf-nsf-monitoring-data-model-09";
  }

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

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

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

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

  description
    "This module is a YANG module for Consumer-Facing Interface.

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

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

   This version of this YANG module is part of RFC XXXX
   (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
   for full legal notices.";

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

revision "2021-08-27" {
  description "Initial revision.";
  reference
    "RFC XXXX: I2NSF Application Interface YANG Data Model";

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

augment "/nsfintf:i2nsf-security-policy" {
  description
  "Augment the NSF-Facing Interface Data Model for the policy
   reconfiguration";
  leaf nsf-name {
    type union {
      type string;
      type inet:ip-address;
    }
    description
      "The name or IP address (IPv4 or IPv6) of the NSF to be
       configured. If the given nsf-name is not IP address, the
       name can be an arbitrary string including FQDN (Fully
       Qualified Domain Name).";
  }

  container problem {
    description
     "Problem: The issue that is emitted by an NSF via the
      I2NSF Monitoring Interface such as DDoS detection, Virus
      detection, Intrusion detection, Web-attack detection, and
      VoIP/VoLTE violation detection.";
    choice attack-detection {
      description
        "The detected attack type";
      case ddos-detected {
```

```
  container ddos-detected {
    leaf-list attack-src-ip {
      type inet:ip-address;
      description
        "The source IPv4 (or IPv6) addresses of attack
         traffic. It can hold multiple IPv4 (or IPv6)
         addresses.";
    }
    leaf-list attack-dst-ip {
      type inet:ip-prefix;
      description
        "The destination IPv4 (or IPv6) addresses of attack
         traffic. It can hold multiple IPv4 (or IPv6)
         addresses.";
    }
    leaf-list attack-src-port {
      type inet:port-number;
      description
        "The source ports of the DDoS attack";
    }
    leaf-list attack-dst-port {
      type inet:port-number;
      description
        "The destination ports of the DDoS attack";
    }
    description
      "A container for DDoS Attack";
  }
  description
    "A DDoS Attack is detected";
}
case virus-detected {
  container virus-detected {
    leaf virus-name {
      type string;
      description
        "The name of the detected virus";
    }
    leaf file-type {
      type string;
      description
        "The type of file virus code is found in (if
         applicable).";
      reference
        "IANA Website: Media Types";
    }
    leaf file-name {
      type string;
      description
```

```
        "The name of file virus code is found in (if
         applicable).";
    }
    description
      "A Virus Attack is detected";
  }
  description
    "A virus is detected";
}
case intrusion-detected {
  container intrusion-detected {
    leaf protocol {
      type identityref {
        base nsfmi:transport-protocol;
      }
      description
        "The transport protocol type for
         nsf-detection-intrusion notification";
    }
    leaf app {
      type identityref {
        base nsfmi:application-protocol;
      }
      description
        "The employed application layer protocol";
    }
    leaf attack-type {
      type identityref {
        base nsfmi:intrusion-attack-type;
      }
      description
        "The sub attack type for intrusion attack";
    }
    description
      "An intrusion is detected";
  }
}
case web-attack-detected {
  container web-attack-detected {
    leaf attack-type {
      type identityref {
        base nsfmi:web-attack-type;
      }
      description
        "Concrete web attack type, e.g., SQL injection,
         command injection, XSS, and CSRF.";
    }
    leaf request-method {
      type identityref {
```

```
        base nsfmi:request-method;
      }
      description
        "The HTTP request method, e.g., PUT or GET.";
      reference
        "RFC 7231: Hypertext Transfer Protocol (HTTP/1.1):
         Semantics and Content - Request Methods";
    }
    leaf req-uri {
      type string;
      description
        "The Requested URI";
    }
    leaf req-user-agent {
      type string;
      description
        "The request user agent";
    }
    leaf req-cookie {
      type string;
      description
        "The HTTP Cookie previously sent by the server with
         Set-Cookie";
    }
    leaf req-host {
      type string;
      description
        "The domain name of the requested host";
    }
    leaf response-code {
      type string;
      description
        "The HTTP Response code";
      reference
        "IANA Website: Hypertext Transfer Protocol (HTTP)
         Status Code Registry";
    }
    description
      "A web attack is detected";
  }
  description
    "A web attack is detected";
}
case voip-volte-detected {
  container voip-volte-detected {
    leaf-list source-voice-id {
      type string;
      description
        "The detected source voice ID for VoIP and VoLTE that
```

```
              violates the security policy.";
          }
          leaf-list destination-voice-id {
            type string;
            description
              "The detected destination voice ID for VoIP and VoLTE
               that violates the security policy.";
          }
          leaf-list user-agent {
            type string;
            description
              "The detected user-agent for VoIP and VoLTE that
               violates the security policy.";
          }
          description
            "A violation of VoIP/VoLTE is detected";
        }
        description
          "A violation of VoIP/VoLTE is detected";
      }
    }
  }
}

list i2nsf-feedback-information {
  key "nsf-name time";

  description
    "Feedback information is information about problem(s) of an
     NSF for a security service such as system resource over-usage
     or malfunction. ";

  leaf nsf-name {
    type union {
      type string;
      type inet:ip-address;
    }
    description
      "The name or IP address (IPv4 or IPv6) of the NSF to be
       configured. If the given nsf-name is not IP address, the
       name can be an arbitrary string including FQDN (Fully
       Qualified Domain Name).";
  }

  leaf time {
    type yang:date-and-time;
    description
      "The time of the feedback information delivered";
  }
```

```
container problem {
  description
    "The issue that is emitted by an NSF via the I2NSF Monitoring
     Interface. The problem for feedback information includes the
     system alarms, such as Memory alarm, CPU alarm, Disk alarm,
     Hardware alarm, and Interface alarm.";
  choice alarm-type {
    description
      "The detected alarm type";
    case memory-alarm {
      container memory-alarm {
        leaf usage {
          type uint8 {
            range "0..100";
          }
          units "percent";
          description
            "The average usage for the duration of the alarm.";
        }
        leaf message {
          type string;
          description
            "A message explaining the problem.";
        }
        leaf duration {
          type uint32;
          description
            "Specify the duration of the first alarm triggered
             until the feedback information is created.";
        }
        description
          "The container for memory-alarm";
      }
      description
        "The detected alarm type is memory-alarm";
    }
    case cpu-alarm {
      container cpu-alarm {
        leaf usage {
          type uint8 {
            range "0..100";
          }
          units "percent";
          description
            "The average usage for the duration of the alarm.";
        }
        leaf message {
          type string;
```

```
      description
        "A message explaining the problem.";
    }
    leaf duration {
      type uint32;
      description
        "Specify the duration of the first alarm triggered
         until the feedback information is created.";
    }
    description
      "The container for cpu-alarm";
  }
  description
    "The detected alarm type is cpu-alarm";
}
case disk-alarm {
  container disk-alarm {
    leaf disk-id {
      type string;
      description
        "The ID of the storage disk. It is a free form
         identifier to identify the storage disk.";
    }
    leaf usage {
      type uint8 {
        range "0..100";
      }
      units "percent";
      description
        "The average usage for the duration of the alarm.";
    }
    leaf message {
      type string;
      description
        "A message explaining the problem.";
    }
    leaf duration {
      type uint32;
      description
        "Specify the duration of the first alarm triggered
         until the feedback information is created.";
    }
    description
      "The container for disk-alarm";
  }
  description
    "The detected alarm type is disk-alarm";
}
case hardware-alarm {
```

```
  container hardware-alarm {
    leaf component-name {
      type string;
      description
        "The hardware component responsible for generating
         the message. Applicable for Hardware Failure
         Alarm.";
    }
    leaf message {
      type string;
      description
        "A message explaining the problem.";
    }
    leaf duration {
      type uint32;
      description
        "Specify the duration of the first alarm triggered
         until the feedback information is created.";
    }
    description
      "The container for hardware-alarm";
  }
  description
    "The detected alarm type is hardware-alarm";
}
case interface-alarm {
  container interface-alarm {
    leaf interface-id {
      type string;
      description
        "The interface ID responsible for generating
         the message.";
    }
    leaf interface-state {
      type enumeration {
        enum down {
          description
            "The interface state is down.";
        }
        enum up {
          description
            "The interface state is up and not congested.";
        }
        enum congested {
          description
            "The interface state is up but congested.";
        }
      }
      description
```

```
                "The state of the interface (i.e., up, down,
                 congested). Applicable for Network Interface Failure
                 Alarm.";
              }
              leaf message {
                type string;
                description
                  "A message explaining the problem.";
              }
              leaf duration {
                type uint32;
                description
                  "Specify the duration of the first alarm triggered
                   until the feedback information is created.";
              }
              description
                "The container for interface-alarm";
            }
            description
              "The detected alarm type is interface-alarm";
          }
        }
      }

      leaf-list solution {
        type string;
        description
           "A possible solution given as feedback is in the form of
            a free-form string (as a high-level instruction).";
      }
    }
}

<CODE ENDS>
```

Figure 7: YANG for Application Interface

5. IANA Considerations

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

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

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

```
name: ietf-i2nsf-feedback-policy
namespace: urn:ietf:params:xml:ns:yang:ietf-i2nsf-feedback-policy
prefix: nsffb
reference: RFC XXXX

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

6. XML Configuration Examples of Feedback Policy

This section shows XML configuration examples of feedback policy rules that are delivered from the I2NSF Analyzer to the Security Controller over the Application Interface after the I2NSF Analyzer analyzes the Monitoring Information.

6.1. Feedback Policy for DDoS Detection

In this example, the scenario can be seen in Figure 8.

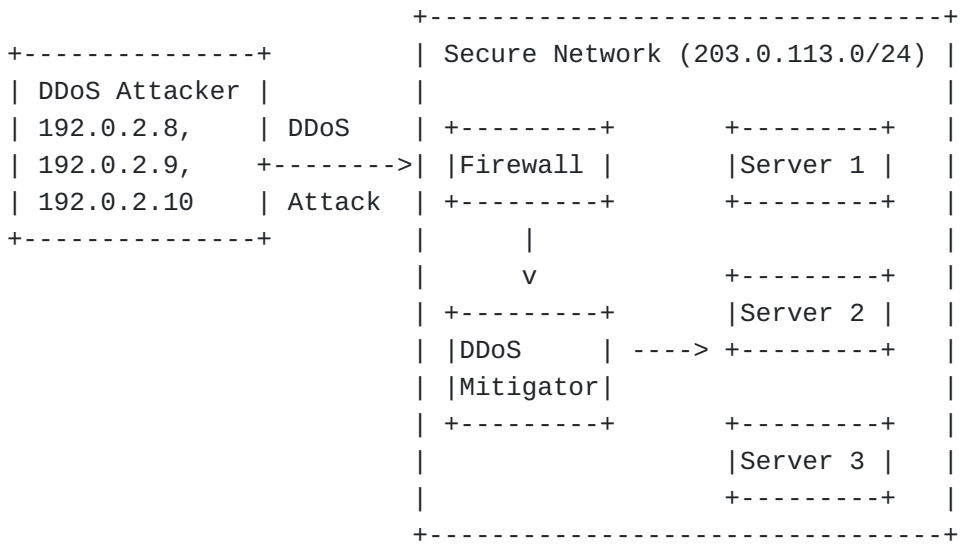

Figure 8: A Scenario Example of DDoS Attack

In this scenario, a DDoS Mitigator detects a DDoS Attack and sends a notification to the I2NSF Analyzer as shown in Figure 9.

```
<?xml version="1.0" encoding="UTF-8"?>
<notification
 xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2021-08-27T09:00:01.00Z</eventTime>
  <i2nsf-nsf-event
    xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-nsf-monitoring">
    <i2nsf-nsf-detection-ddos>
      <attack-type
        xmlns:nsfmi="urn:ietf:params:xml:ns:yang:\
                    ietf-i2nsf-nsf-monitoring">
        nsfmi:syn-flood
      </attack-type>
      <acquisition-method
        xmlns:nsfmi="urn:ietf:params:xml:ns:yang:\
                     ietf-i2nsf-nsf-monitoring">
        nsfmi:subscription
      </acquisition-method>
      <emission-type
        xmlns:nsfmi="urn:ietf:params:xml:ns:yang:\
                     ietf-i2nsf-nsf-monitoring">
        nsfmi:on-change
      </emission-type>
      <dampening-type
        xmlns:nsfmi="urn:ietf:params:xml:ns:yang:\
                     ietf-i2nsf-nsf-monitoring">
        nsfmi:on-repetition
      </dampening-type>
      <start-time>2021-08-27T09:00:00.00Z</start-time>
      <attack-src-ip>192.0.2.8</attack-src-ip>
      <attack-src-ip>192.0.2.9</attack-src-ip>
      <attack-src-ip>192.0.2.10</attack-src-ip>
      <attack-dst-ip>203.0.113.0/24</attack-dst-ip>
      <attack-rate>100</attack-rate>
      <message>A DDoS Attack is detected</message>
      <nsf-name>DDoS_mitigator</nsf-name>
    </i2nsf-system-detection-alarm>
  </i2nsf-event>
</notification>
```

Figure 9: A Detected DDoS Attack by DDoS Mitigator

In the scenario shown in Figure 9, the description of the XML example is as follows:

1. The DDoS attack is detected at 9 am on August 27 in 2021.

2. The sources of the attack are 192.0.2.8, 192.0.2.9, and 192.0.2.10.

3. The destination of the attack is 203.0.113.0/24.

After receiving the information, the I2NSF Analyzer analyzes the data and creates a new feedback policy to enforce the security of the network. The I2NSF Analyzer delivers a feedback policy to the Security Controller as shown in Figure 10.

```
<?xml version="1.0" encoding="UTF-8" ?>
<i2nsf-security-policy
 xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-feedback-policy">
  <system-policy-name>
    feedback_policy_for_ddos_attack
  </system-policy-name>
  <rules>
    <rule-name>deny_ddos_attack</rule-name>
    <event>
      <start-date-time>2021-08-27T09:00:01.00Z</start-date-time>
    </event>
    <condition>
      <ipv4>
        <ipv4-range>
          <start>192.0.2.8</start>
          <end>192.0.2.10</end>
        <ipv4-range>
      </ipv4>
    </condition>
    <actions>
      <packet-action>
        <ingress-action>drop</ingress-action>
      </packet-action>
    </actions>
  </rules>
  <nsf-name>Firewall</nsf-name>
  <problem>
    <ddos-detected>
      <attack-src-ip>192.0.2.8</attack-src-ip>
      <attack-src-ip>192.0.2.9</attack-src-ip>
      <attack-src-ip>192.0.2.10</attack-src-ip>
      <attack-dst-ip>203.0.113.0/24</attack-dst-ip>
    </ddos-detected>
  </problem>
</i2nsf-security-policy>
```

Figure 10: Policy Reconfiguration for a Detected DDoS Attack

The policy reconfiguration in Figure 10 means the following:

1. The feedback policy is named as "feedback_policy_for_ddos_attack".

2. The rule is named as "deny_ddos_attack".

3. The rule starts from 09:00 am on August 24 in 2021. The condition of the rule is from the sources of the IP addresses 192.0.2.8, 192.0.2.9, and 192.0.2.10.

4. The action required is to "drop" any access from the the IP addresses have been identified as malicious.

5. The NSF to be configured is named "Firewall".

6. The problem that triggered the generation of the feedback is a DDoS attack from the sources of the IP addresses 192.0.2.8, 192.0.2.9, and 192.0.2.10 to the protected network of 203.0.113.0/24.

6.2. Feedback Information for Overloaded NSF

In this scenario, an NSF is overloaded and sends a notification to the I2NSF Analyzer as shown in Figure 11.

```
<?xml version="1.0" encoding="UTF-8"?>
<notification
 xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2021-08-27T07:43:52.181088+00:00</eventTime>
  <i2nsf-event
    xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-nsf-monitoring">
    <i2nsf-system-detection-alarm>
      <alarm-category
        xmlns:nsfmi="urn:ietf:params:xml:ns:yang:\
                   ietf-i2nsf-nsf-monitoring">
        nsfmi:memory-alarm
      </alarm-category>
      <acquisition-method
        xmlns:nsfmi="urn:ietf:params:xml:ns:yang:\
                     ietf-i2nsf-nsf-monitoring">
        nsfmi:subscription
      </acquisition-method>
      <emission-type
        xmlns:nsfmi="urn:ietf:params:xml:ns:yang:\
                     ietf-i2nsf-nsf-monitoring">
        nsfmi:on-change
      </emission-type>
      <dampening-type
        xmlns:nsfmi="urn:ietf:params:xml:ns:yang:\
                     ietf-i2nsf-nsf-monitoring">
        nsfmi:on-repetition
      </dampening-type>
      <usage>98</usage>
      <threshold>80</threshold>
      <message>Memory Usage Exceeded the Threshold</message>
      <nsf-name>firewall</nsf-name>
      <severity>high</severity>
    </i2nsf-system-detection-alarm>
  </i2nsf-event>
</notification>
```

Figure 11: The Monitoring of an Overloaded NSF

In the scenario shown in Figure 11, the description of the XML example is as follows:

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

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

3. The memory usage of the NSF is 98 percent.

4. The memory threshold to trigger the alarm is 80 percent.

5. The event is delivered at 2021-08-27T07:43:52.181088+00:00.

After receiving the information, the I2NSF Analyzer analyzes the data and creates a new feedback policy to solve the problem that is detected in the NSF. The I2NSF Analyzer delivers a feedback information to the Security Controller as shown in Figure 12.

```
<?xml version="1.0" encoding="UTF-8" ?>
<i2nsf-feedback-information
 xmlns="urn:ietf:params:xml:ns:yang:ietf-i2nsf-feedback-policy">
  <time>2021-08-27T08:43:52.000000+00:00</time>
  <nsf-name>Firewall</nsf-name>
  <problem>
    <memory-alarm>
      <usage>95</usage>
      <message>Memory Usage Exceeded the Threshold</message>
      <duration>3600</duration>
    </memory-alarm>
  </problem>
  <solution>
    Add more memory capacity to the NSF
  </solution>
  <solution>
    Create a new NSF with the same security service
  </solution>
</i2nsf-feedback-information>
```

Figure 12: Feedback Information for the Overloaded NSF

The feedback information in Figure 12 means the following:

1. The name of the NSF that needs to be handled is called "Firewall".

2. The feedback information is delivered at 2021-08-27T08:43:52.000000+00:00.

3. The problem is that the Memory Usage Exceeded the Threshold with the average usage of memory as 95.

4. The problem persists for 3,600 seconds (1 hour) without any fix.

5. The proposed solution to the problem is to add more memory capacity in hardware to the NSF or to create a new NSF with the same security service.

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. And the data model in this document uses the data model from NSF-Facing Interface data model, it MUST follow the Security Considerations mentioned in the [I-D.ietf-i2nsf-nsf-facing-interface-dm].

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. This document also MUST follow the Security Considerations about the readable data nodes mentioned in the [I-D.ietf-i2nsf-nsf-facing-interface-dm].

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) (2020-0-00395, Standard Development of Blockchain based Network Management Automation Technology). This work was supported in part by the IITP (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 Information Technology Research Center (ITRC) support program (IITP-2021-2017-0-01633) supervised by the IITP.

9. Contributors

This document is made by the group effort of I2NSF working group. Many people actively contributed to this document, such as Linda Dunbar, Yoav Nir, Susan Hares, and Diego Lopez. The authors sincerely appreciate their contributions.

The following are co-authors of this document:

Jeonghyeon Kim Department of Computer Science and Engineering Sungkyunkwan University 2066 Seo-ro Jangan-gu Suwon, Gyeonggi-do 16419 Republic of Korea EMail: jeonghyeon12@skku.edu

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

Jung-Soo Park Electronics and Telecommunications Research Institute 218 Gajeong-Ro, Yuseong-Gu Daejeon, 34129 Republic of Korea EMail: pjs@etri.re.kr

Younghan Kim School of Electronic Engineering Soongsil University 369, Sangdo-ro, Dongjak-gu Seoul 06978 Republic of Korea EMail: younghak@ssu.ac.kr

10. References

10.1. Normative References

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

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

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol

(NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

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

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

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

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

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

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

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

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

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

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

10.2. Informative References

[RFC8329] Lopez, D., Lopez, E., Dunbar, L., Strassner, J., and R. Kumar, "Framework for Interface to Network Security

Functions", RFC 8329, DOI 10.17487/RFC8329, February 2018, <https://www.rfc-editor.org/info/rfc8329>.

[I-D.ietf-i2nsf-nsf-monitoring-data-model] Jeong, J. (., Lingga, P., Hares, S., Xia, L. (., and H. Birkholz, "I2NSF NSF Monitoring Interface YANG Data Model", Work in Progress, Internet-Draft, draft-ietf-i2nsf-nsf-monitoring-data-model-08, 29 April 2021, <https://www.ietf.org/archive/id/draft-ietf-i2nsf-nsf-monitoring-data-model-08.txt>.

[I-D.jeong-i2nsf-security-management-automation] Jeong, J. (., Lingga, P., and J. Park, "An Extension of I2NSF Framework for Security Management Automation in Cloud-Based Security Services", Work in Progress, Internet-Draft, draft-jeong-i2nsf-security-management-automation-02, 21 August 2021, <https://www.ietf.org/archive/id/draft-jeong-i2nsf-security-management-automation-02.txt>.

[I-D.ietf-i2nsf-nsf-facing-interface-dm] Kim, J. (., Jeong, J. (., Park, J., Hares, S., and Q. Lin, "I2NSF Network Security Function-Facing Interface YANG Data Model", Work in Progress, Internet-Draft, draft-ietf-i2nsf-nsf-facing-interface-dm-12, 8 March 2021, <https://www.ietf.org/archive/id/draft-ietf-i2nsf-nsf-facing-interface-dm-12.txt>.

[I-D.ietf-i2nsf-capability-data-model] Hares, S., Jeong, J. (., Kim, J. (., Moskowitz, R., and Q. Lin, "I2NSF Capability YANG Data Model", Work in Progress, Internet-Draft, draft-ietf-i2nsf-capability-data-model-17, 14 August 2021, <https://www.ietf.org/archive/id/draft-ietf-i2nsf-capability-data-model-17.txt>.

[I-D.ietf-i2nsf-applicability] Jeong, J. P., Hyun, S., Ahn, T., Hares, S., and D. R. Lopez, "Applicability of Interfaces to Network Security Functions to Network-Based Security Services", Work in Progress, Internet-Draft, draft-ietf-i2nsf-applicability-18, 16 September 2019, <https://www.ietf.org/archive/id/draft-ietf-i2nsf-applicability-18.txt>.

[Deep-Learning] Goodfellow, I., Bengio, Y., and A. Courville, "Deep Learning", Publisher: The MIT Press, URL: https://www.deeplearningbook.org/, November 2016.

Authors' Addresses

Patrick Lingga (editor)

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

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

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

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

Yunchul Choi
Electronics and Telecommunications Research Institute
218 Gajeong-Ro, Yuseong-Gu
Daejeon
305-700
Republic of Korea

Phone: +82 42 860 5978
Email: cyc79@etri.re.kr