

Consumer-Facing Interface YANG Data Model for Interface to Network Security Functions

(draft-jeong-i2nsf-consumer-facing-interface-dm-01)



IETF 98, Chicago, US

Mar. 27, 2017

Jaehoon Paul Jeong*, Mahdi Daghmehchi,
Tae-Jin Ahn, Rakesh Kumar, and Susan Hares

Contents

I Introduction

II Architecture of Security Management

III Use Case-VoLTE security service

IV Update of Version

V Next Step



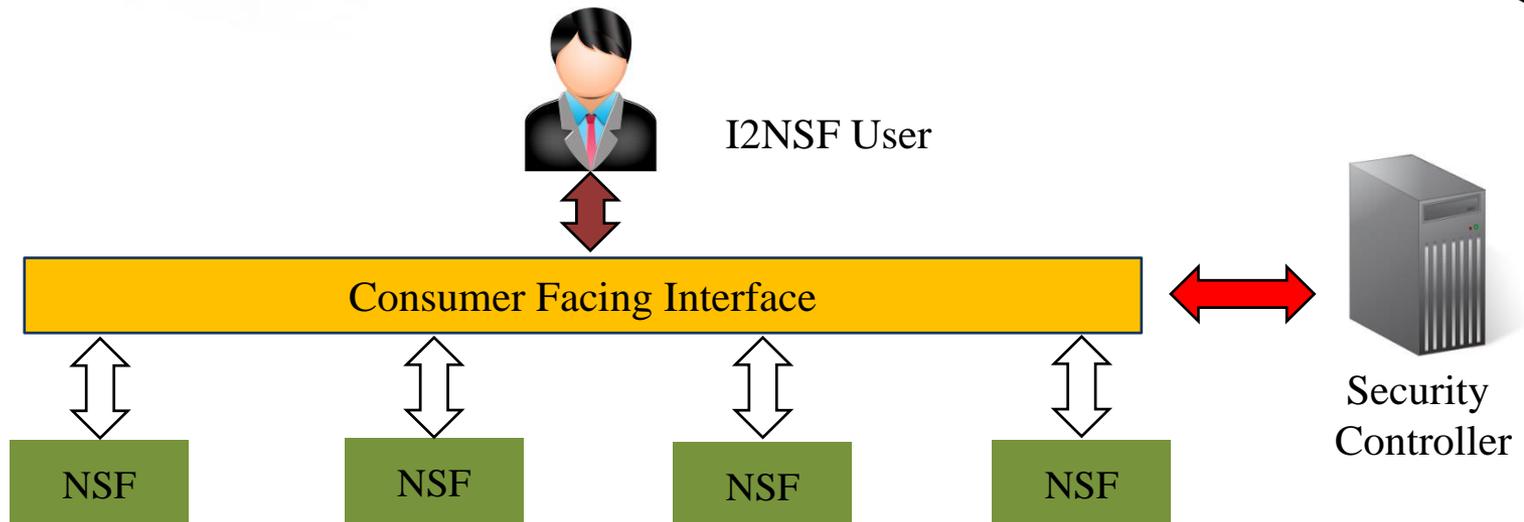
Introduction (1/2)

- This document describes a data model for security management based on I2NSF framework by using NFV
- A data model to perform VoIP-VoLTE security service

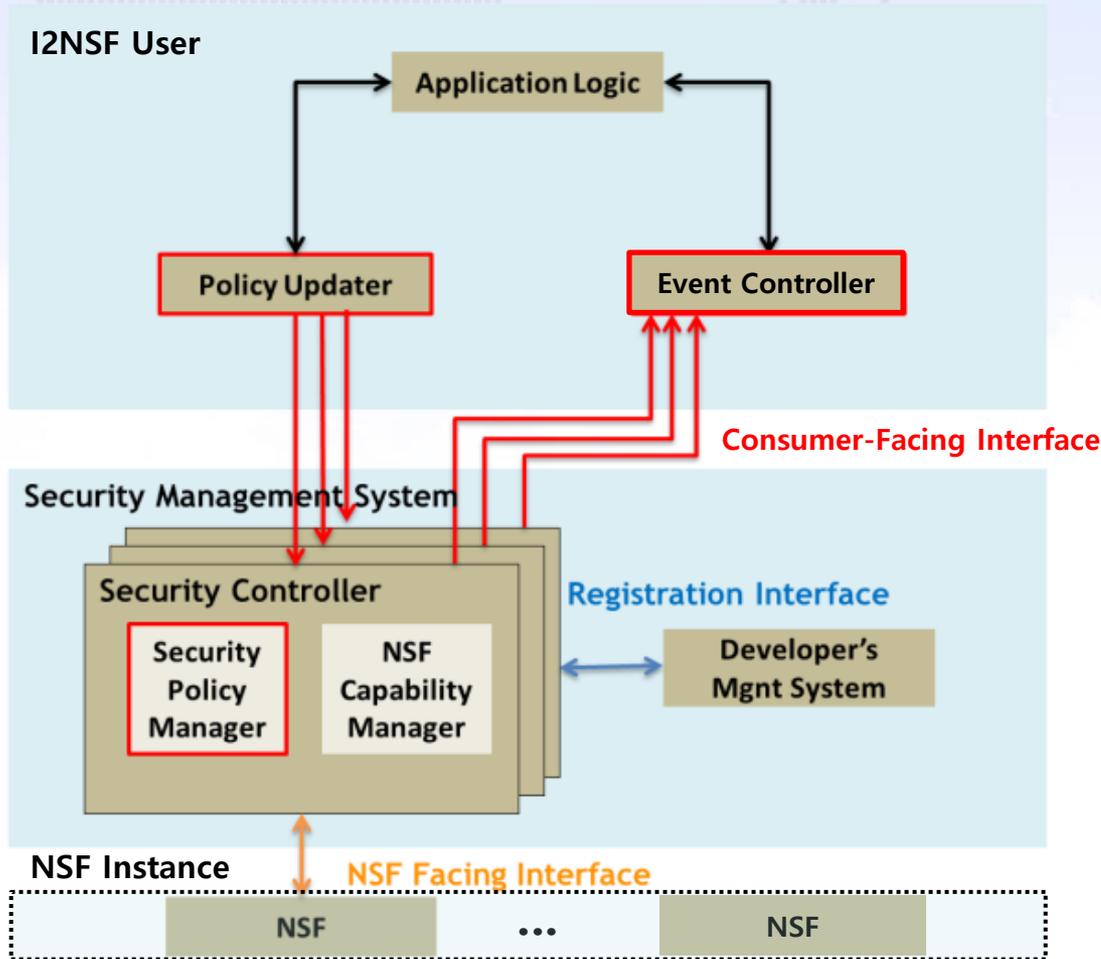


Introduction (2/2)

- Defining high-level policies and translate them to several low-level policies
- Updating low-level policies based on NSF capabilities
- Monitoring network's events and implementing security functions based on NFV

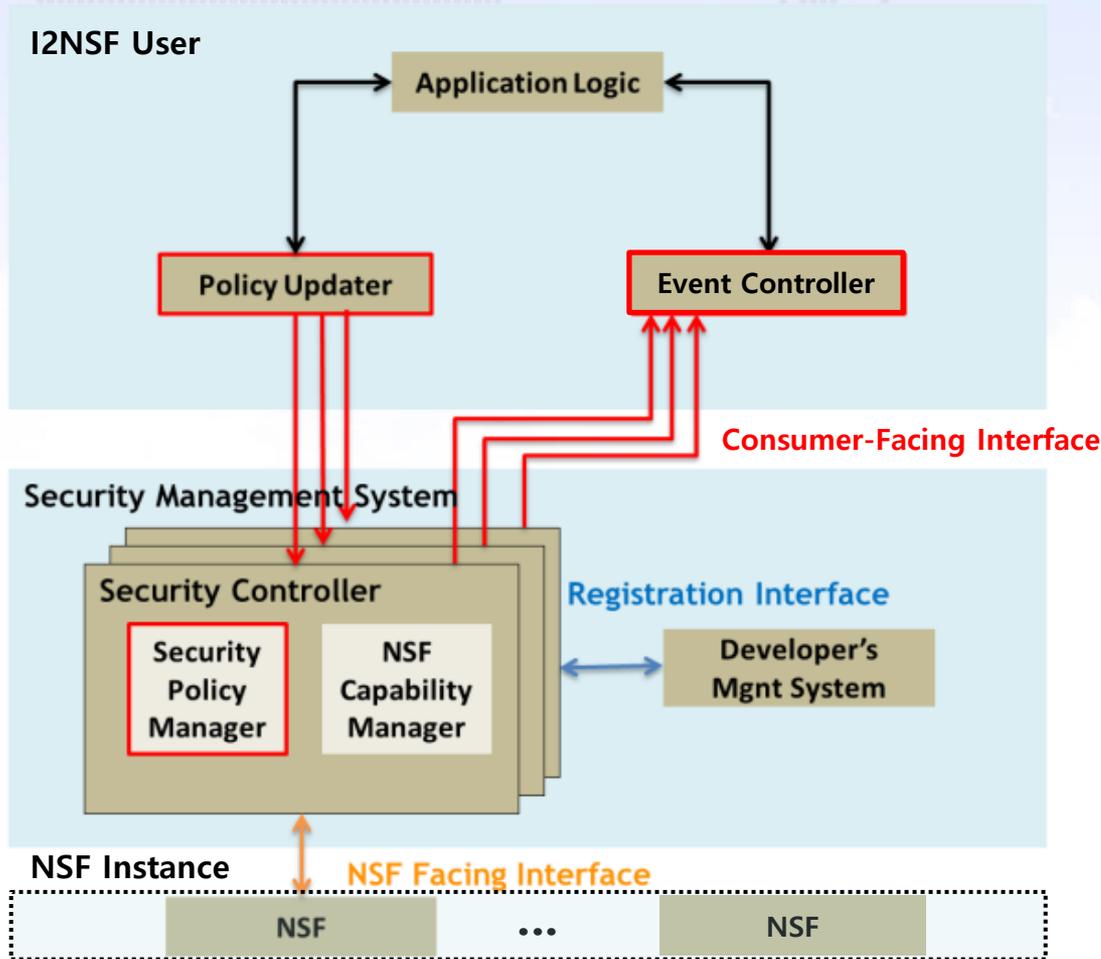


Security Management Architecture (1/3)



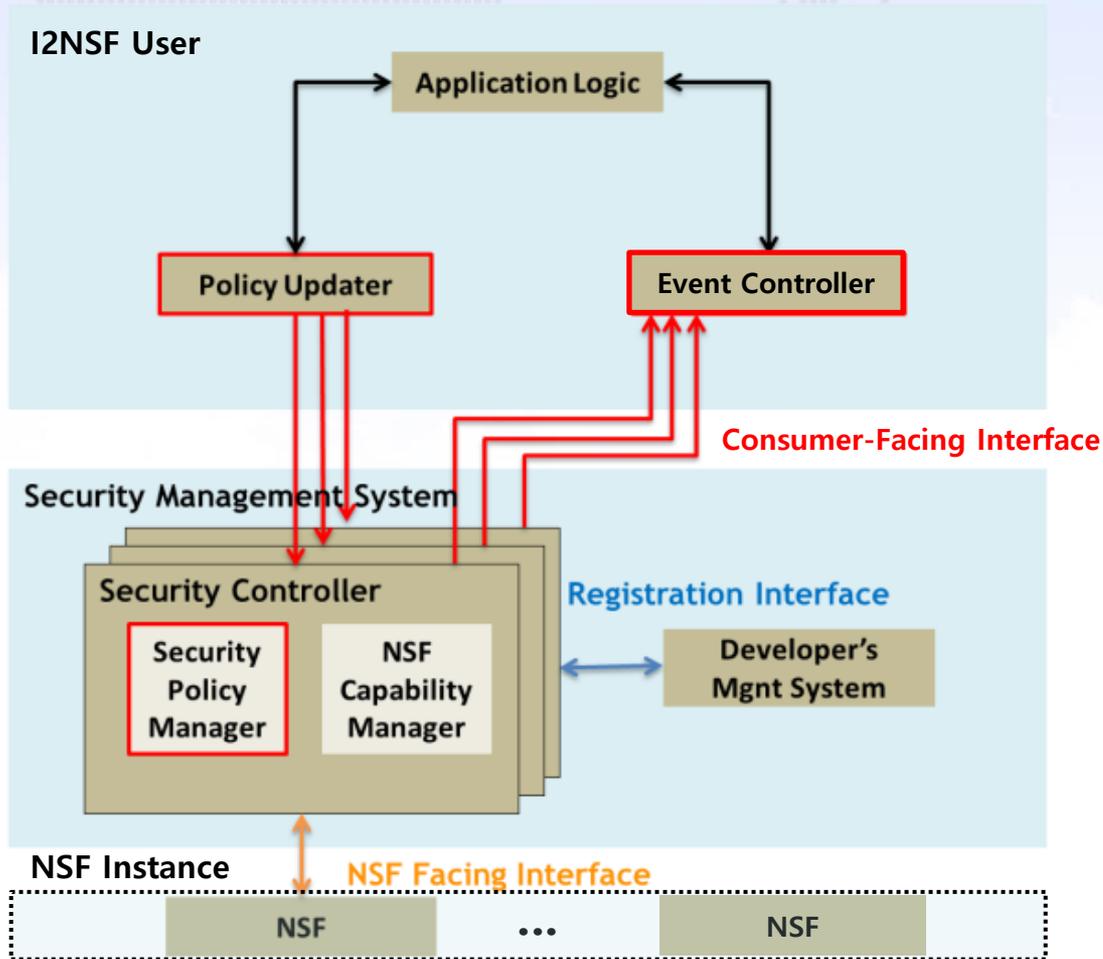
- **Application Logic**
Generating high-level security policies
- **Event Controller**
Event monitoring and sending to Application logic
- **Policy Updater**
Distributing high-level policies to the Security Controller

Security Management Architecture (2/3)



- **Security Policy Manager**
 - Mapping high-level policies into several low-level policies
 - Delivering low level policies to NSF(s)
- **NSF capability manager**
Storing the NSF's capability and sharing it with Security policy manager
- **Developer's Mgmt system**
Registering new NSF's capabilities into NSF capability manager

Security Management Architecture (3/3)



- **NSF Instance**
Exploiting low-level policies delivered by the Security policy manager

Security Management for VoLTE

VoLTE/VoIP security management: Application Logic

- Defining security conditions (e.g., blacklists of IP addresses & source ports, expire time, user agents)
- Updating the illegal devices information (manually/automatically)
- Generating new high-level security policies
- Updating the VoIP-VoLTE database based on the NSF's anomalous detection

Information model for Consumer-Facing Interface *

Information Model for:

- Threat Prevention
To reduce the attack surface (e.g., Botnet)
- Policy endpoint groups
Where a security policy is to be applied
- Policy Instance
A complete information for any policy instance (e.g., where/when a policy need to be applied)

The image features a stylized city skyline with various skyscrapers in shades of gray, positioned along a horizon line. Below the skyline is a large, white grid pattern that recedes into the distance, creating a sense of depth. In the upper left portion of the sky, a small jet airplane is shown in flight, leaving a long, thin white contrail that extends towards the center of the frame. The sky is a light, clear blue with a few small, white clouds scattered across it. The overall composition is clean and modern, with a focus on geometric shapes and perspective.

Update of Version

Update of Version (1/3)

- The changes from draft-jeong-i2nsf-consumer-facing-interface-dm-00:
 - Addition of a new component (Update for NSF's feedback) and its description in data model.
 - Implementation of the corrected data model based on YANG model.
- draft-jeong-i2nsf-consumer-facing-interface-dm-01 defines an overall structure of consumer-facing interface and its YANG data model.

Update of Version (2/3)

Data Model for VoLTE Security Service

High-level policies basements:

- Blacklisting countries
- Time interval specification
- Caller's priority levels

The data model consists of:

- Policy life cycle management
- Policy rule
- Action
- Update (NSF's Feedback or Unexpected Event)

```
+---: (ieft-i2nsf-policy)
+--rw policy-lifecycle-list
+--rw policy-lifecycle-container *(policy-lifecycle-id)
|
| +--rw expiration-event
| | +--rw enabled                boolean
| | +--rw event-id              uint 16
| | +--rw event-date            date-and-time
+--rw expiration-time
+--rw enabled                    boolean
+--rw time                       date-and-time
+--rw policy-rule-list
+--rw policy-rule-container *[policy-rule-id]
|
| +--rw policy-rule-id          uint 16
+--rw policy-name               string
+--rw policy-date               date-and-time
+--rw service
| +--voip-handling              boolean
| +--volte-handling             boolean
+--rw condition *[condition-id]
+--rw caller
| +--rw caller-id              uint 16
| +--rw caller-location
| | +--rw country              string
| | +--rw city                 string
+--rw callee
| +--rw callee-id              uint 16
| +--rw callee-location
| | +--rw country              string
| | +--rw city                 string
+--rw valid-time-interval
+--rw start-time                data-and-time
+--rw end-time                  data-and-time
+--rw action-list
+--rw action-container
|
| +--rw action-date            date-and-time
+--rw action-name               string
+---: (action-name-ingress)
| +--rw permit?               boolean
| +--rw mirror?               boolean
| +--rw log?                   boolean
+---: (action-name-engress)
+--rw redirection?              boolean
+--rw update-list
+--rw update-container          *(update-id)
+--rw update-event
+--rw update-event-id          uint 16
+--rw update-enabled            boolean
+--rw update-event-date        date-and-time
+--rw update-log                string
```

Update of Version (3/3)

Data Model for VoLTE Security Service

Policy life cycle management

Specifies an expiration time and/or event to determine the life-time of the policy itself

Policy rule

Represents the specific information about a high-level policy
e.g., service types, conditions and valid time interval

Action

Specifies the actions which should be performed when a policy rule is matched by NSF

Update

Update a policy to reflect upon the event triggered by NSFs.

```
+---: (ieft-i2nsf-policy)
+--rw policy-lifecycle-list
+--rw policy-lifecycle-container *(policy-lifecycle-id)
|
| +--rw expiration-event
| | +--rw enabled                boolean
| | +--rw event-id              uint 16
| | +--rw event-date            date-and-time
+--rw expiration-time
+--rw enabled                    boolean
+--rw time                       date-and-time
+--rw policy-rule-list
+--rw policy-rule-container *[policy-rule-id]
+--rw policy-rule-id             uint 16
+--rw policy-name                string
+--rw policy-date                date-and-time
+--rw service
| +--voip-handling              boolean
| +--volte-handling             boolean
+--rw condition *[condition-id]
+--rw caller
| +--rw caller-id              uint 16
| +--rw caller-location
| | +--rw country              string
| | +--rw city                 string
+--rw callee
| +--rw callee-id              uint 16
| +--rw callee-location
| | +--rw country              string
| | +--rw city                 string
+--rw valid-time-interval
+--rw start-time                 data-and-time
+--rw end-time                   data-and-time
+--rw action-list
+--rw action-container
+--rw action-date                date-and-time
+--rw action-name                string
+---: (action-name-ingress)
| +--rw permit?                 boolean
| +--rw mirror?                 boolean
| +--rw log?                    boolean
+---: (action-name-engress)
+--rw redirection?              boolean
+--rw update-list
+--rw update-container           *(update-id)
+--rw update-event
+--rw update-event-id            uint 16
+--rw update-enabled             boolean
+--rw update-event-date          date-and-time
+--rw update-log                 string
```

Next Step

- **Generic YANG Data Model**
Modify current data model to be a Generic model
- **Implementation of more use cases**
e.g., Untrusted domain (malware distributor) detecting, and access control function (time/location depended)

