Registration Interface Information Model and Data Model

draft-hyun-i2nsf-registration-interface-im-02,
draft-hyun-i2nsf-registration-interface-dm-01

Sangwon Hyun*, Jaehoon Paul Jeong, Taekyun Roh,
Sarang Wi, and Jung-Soo Park

IETF 99, Prague, Czech
July 18, 2017
The Registration interface in the I2NSF framework can be utilized:
- to register NSF and its capabilities into Security Controller
- to request dynamic instantiation/deinstantiation of NSFs.

Information model & data model for the Registration interface are required for the following functions:
- Request Developer's Management System (DMS) to dynamically instantiate/deinstantiate an NSF
- Register new NSF instances created by DMS

Secure registration of distributed NSFs via the Registration interface in a centralized manner
Updates from Previous Version

- In draft-hyun-i2nsf-registration-interface-im-02, we further developed the portion of the information model of performance capability.

- In draft-hyun-i2nsf-registration-interface-dm-01, we updated the YANG data model accordingly in order to align with the updates in draft-hyun-i2nsf-registration-interface-im-02.
Registration of a New NSF Instance

- DMS creates a new NSF instance that has the security capability(s) requested by Security Controller, and registers the created NSF instance to Security Controller via Registration Interface.

The existing information model (draft-xibassnez-i2nsf-capability-02) & data model (draft-hares-i2nsf-capability-data-model-03) are used to describe the security capability(s) of an NSF.
Dynamic Instantiation/Deinstantiation of NSFs

- **Motivations:**
  - When an NSF calls another NSF for an **advanced security action** of the suspicious packet but **no instance of the callee is available in the system**
  - When an NSF instance is currently under congestion
  - When an NSF instance is in **idle**

The existing information model (draft-xibassnez-i2nsf-capability-02) & data model (draft-hares-i2nsf-capability-data-model-03) are used to describe the security capability(s) of an NSF.
• We refined the performance capability portion of the IM and DM to describe the attributes of each type of resource available for an NSF.

  – e.g., processing power, memory, and network bandwidth, etc.

```c
NSF Performance Capability
+-rw i2nsf-nsf-performance-caps
  +-rw vcpus
    |  +--rw cpu-num uint16
    |  +--rw cpu-topology
    |    |  +-- rw flavor-cores uint16
    |    |  +-- rw flavor-socket uint16
    |    |  +-- rw flavor-threads uint16
    |  +--rw (cpu-limit uint16)?
    |  +--rw (cpu-reservation uint16)?
  +-rw disk
    |  +--rw disk-size uint16
    |  +--rw (disk-limit uint16)?
    |  +--rw (disk-reservation uint16)?
  +-rw memory
    |  +--rw memory-size uint16
    |  +--rw (memory-limit uint16)?
    |  +--rw (memory-reservation uint16)?
  +-rw bandwidth
    |  +--rw outbound
    |     |  +--rw outbound-average uint16
    |     |  +--rw outbound-peak uint16
    |  +--rw inbound
    |     |  +--rw inbound-average uint16
    |     |  +--rw inbound-peak uint16
```
Next Steps

- We will implement the following functions in the next IETF hackathon:
  1) Registration of NSF and its capabilities into Security Controller
  2) Dynamic instantiation/deinstantiation of NSF instance(s) via I2NSF Registration Interface.