I2NSF Framework

July, 2015

Edward Lopez (elopez@fortinet.com)
Diego R. Lopez (diego.r.lopez@telefonica.com)
XiaoJun Zhuang (zhuangxiaojun@chinamobile.com)
Linda Dunbar (linda.dunbar@huawei.com)
John Strassner (John.sc.Strassner@huawei.com)
Joe Parrott (joe.parrott@bt.com)
Ramki Krishnan (ramki_krishnan@dell.com)
Seetharama Rao Durbha (S.Durbha@cablelabs.com)
Major Component of I2NSF

**Security Service Layer**
For clients to express and monitor security policies for their specific flows

**Capability Layer**
For controller to specify and monitor the capabilities and functional profiles that are supported by the NSFs

**NSF Registration**
For NSF developers to register their available security functions and set of functional profiles that can be dynamically set by 3rd parties.

**Developer management system**

**I2NSF Client**
App controller, Users

**Service-Oriented Security Policy**

**Controller**
- Service-oriented I2NSF Agent
- Service2Functional Translator
- Functional-oriented I2NSF Client

**Service Flow**
- RESTCONF
- NETCONF
- Functional-Oriented Security Policy

**Security Service Layer For clients to express and monitor security policies for their specific flows**

**Capability Layer**
For controller to specify and monitor the capabilities and functional profiles that are supported by the NSFs
Figure in the draft: Major Components of I2NSF

+------------------------------------------+
|       Client or App Controller           | |       (e.g. Video conference Ctrl        | | Admin, OSS/BSS, or Service Orchestration)| +-------+----------------------------------+
|  Client Facing (service layer) Interface +-----+---------------+        |Service Provider mgmt|              +------------- +
|  Security Controller | < -------- > | Vendor      |
+---------------------+ Vendor Facing| Sys |
| NSF Facing (capability Layer) Interface +------------------------------------------------+|                                                | |                                                |
+------+         +------+             +------+       +------+
| NSF-1+ ------- + NSF-n+             +NSF-1 + ----- +NSF-m + . . .
+--------+       +-------+          +--------+       +--------+
Vendor A                             Vendor B
Capability Layer Interface
Problems

- Unlike traditional networking device, network-based security functions (NSFs) do not operate relative to standards
  - Many evaluative bodies exist, which review the efficacy of network security product
  - Many regulatory/compliance directives call for the use of loosely defined classes of network security
- How do we define interfaces to devices that have no standardized implementations?
Potential for Imposed Constraints

• Narrowly defined NSF categories, or their roles when implemented within a network
• Attempts to impose functional requirements or constraints, either directly or indirectly, upon NSF developers
• Result in a limited lowest-common denominator approach, where interfaces can only support a limited set standardized functions, without allowing for specific functional profiles
• Results in endorsing a best-common-practice for the implementation of NSFs
Packet-Based Paradigm for FlowBased NSF

• Rather than attempting to create a standard based on NSF classes, a solution may exist in provisioning packet processing

• All NSFs, regardless of function, process:
  – Packet headers
  – Packet payloads
  – Contextual and state information associated with packets
Three Sub-Interface Types

• Configuration
  – Device configuration
  – Network configuration

• Signaling
  – Status
  – Counters
  – Queries
  – Alerts

• Rules Provisioning
  – Capabilities
  – Policy
  – Object Configuration
Suggested Rules Provisioning Structure

Subject

- Match values based on packet data
  Packet header - Can be standardized
  Packet payload - Provided by NSF capabilities

Object

- Match values based on context
  Ex.: State, Direction, time, geo-location,
  Many can (and should) be standardized,
  but many also from NSF capabilities

Action

- Egress processing
  Invoke signaling
  Packet forwarding and/or transformation
  Possibility for SDN/NFV integration

Perform the referenced profiles

- Vendor Unique innovation, Vendor specific
  e.g. IPS:<Profile>
  Profile: signature, Anti-virus, URL filtering, etc.
  Integrated and one-pass checks on the content of packets.
Controller Hierarchy

Characteristics:

- Single NSF can have multiple instantiations that are distributed across the network.
- Different rules/policies could be imposed to different instantiations.
- Each NSF may have its own sub-controller for any cluster of its instantiations.
- Policies to one cluster can be moved/copied to another NSF cluster.
- Multiple NSFs collectively together to enforce the rules for large flows.
Service Layer Interface
**Simple service layer rule structure**

- **Composite Groups or Roles (I2NSF-Role):**
  - This is a group of users, applications, virtual networks, or traffic patterns to which a service layer policy can be applied. An I2NSF-Role may be mapped to a client virtual Subnet (i.e. with private address prefix), a subnet with public address families, specific applications, destinations, or any combination of them with logical operators (Logical AND, OR, or NOT). An I2NSF-Role can have one or more Policy Rule Sets.

- **Target.**
  - This is used by the application client to establish communications over the network. A Target can be mapped to a physical/logical ingress port, a set of destinations, or a physical/logical egress port.

- **Policy Rule Set.**
  - A Policy Rule Set is used to determine how the traffic between a pair of I2NSF-Role and Target is to be treated. A Policy Rule Set consists of one or more Policy Rules.

- **Policy Rule.**
  - A Policy Rule consists of a Policy Conditions and a set of Actions to be applied to the traffic.

- **Policy Condition.**
  - Describes when a Policy Rule set is to be applied. It can be expressed as a direction, a list of L4 ports, time range, or a protocol, etc.

- **Policy Action:**
  - This is the action applied to the traffic that matches the Conditions. An action may be a simple ACL action (i.e. allow, deny, mirroring), applying a well known statistics functions (e.g. X minutes count, Y hours court), applying client specified functions (with URL provided), or may refer to an ordered sequence of functions.
Service Layer Policy Structure

```
+---------+     +--------+      +-------+
| Logical  Port
| Roles   |---->| Policy |<----+Target +-|- Ingress  Port
+----+----+     +----+---+      +-------+
| *<-------+      +---------------+
+--+------+
| Simple   ||| Compound Operators:   +--+ Policy |
| Group    | | Logical AND: &&          | AND/OR/NOT |
+----------+---------------+-- +---+--+  +--+------- +
| App      |     |virtual  |     | Subnet  |     |        |-Direction |
| Group    |     | Subnet  |     | host list|     |        |-timer     |
+----------+       |              |          |        |-L4 port  |
+-------------+--+------+-------+--- |
| Client Grp|       |              |          |
+----------+     +--------+      +-------+
+---------+     +--------+
| IPv4    |     | IPv6  |
| Header  |     | Header|
+---------+     +---------+
```
Service Layer extension from PCIM (RFC3060) or ITU-T X.1036

Compound operators:
- Logical AND: `&&`
- Logical OR: `||`
- Logical NOT: `!`

Simple Action
Function/Profile
I2NSF Service Layer model extension

Target Group
- IPv4 header
- IPv6 header
- MAC-header
- App ID
- Client ID
- Traffic type

Policy Context
- Time
- State
- Directions