Information Model of NSFs Capabilities
draft-xibassnez-i2nsf-capability-00

Liang Xia, John Strassner, DaCheng Zhang
Kepeng Li
Cataldo Basile, Antonio Lioy
Diego R. Lopez
Edward Lopez
Nicolas BOUTHORS
Luyuan Fang

Huawei
Alibaba
Politecnico di Torino
Telefonica I+D
Fortinet
Qosmos
Microsoft

November 2016  Seoul
Monitoring Part of I2NSF Architecture

**Service Interface**
For clients or App Gateway to express and monitor security policies for their specific flows.

**NSF Registration**
For NSF vendors to register their available security functions and set of policies (or Service Profiles) that can be dynamically set by 3rd parties.

**Capability Interface**
For controller to define explicit rules for individual NSFs to treat packets, as well as methods to monitor the execution status of those functions.

**Vendor management system**
What Happened

• A big step forward: 2 complementary drafts are converged:
  – Draft-xia-i2nsf-capability-interface-IM-06: “ECA” model, basic framework and detailed class design of capability information model, ...;
  – draft-baspez-i2nsf-capabilities-00: accurate definition of I2NSF capability, geometric model complementing “ECA” with resolution strategies, external data, default action, more specific condition types, ...

• Many thanks to Aldo and Diego (joining as draft co-authors) who brought good inputs for this work based on their experiences gained from EU FP7 SECURED project, which really helps a lot!
What is I2NSF Capability?

- Terminology update:
  - capability interface -> NSF-facing interface;
  - Capability: “Defines a set of features that are available from a managed entity”. There should be NSF Capabilities and Controller Capabilities, which are announced through the Registration Interface;

- Capability Model:
  - based on actions and traffic classification features, used to define
    - generic security functions (GNSF) = known classes of security functions (like: packet filter, URL filter, HTTP filter, VPN, gateway, anti-virus, ...)
    - and extensions: modules, extensions, additional features
    - composed with an Algebra of Capabilities. Example:
      - iptables = generic packet filter + set of stateful TCP conditions
      - iptables with time module installed = iptables + conditions of the time module
  - 3 NSFs Categories already analyzed:
    - network security
    - content security
    - attack mitigation

A_{pf} = \{Allow, Deny, some GNSFs\}
C_{pf} = \{IPsrc, IPdst, Ps欐rst, Pdst, protType\}
C_{time} = \{timestart, days, datestart, datestop\}
C_{CPstate} = \{}
cap_{pf} = (A_{pf}; C_{pf}; \{FMR\}; F)
iptables = cap_{pf} + C_{CPstate}
iptables_{time} = iptables + C_{time}
More Fine-grained I2NSF NSF-facing Interface Policy Information Model

• Geometric Model:
  – \((R, RS, E, d)\): the rule set \(R\) \(\{r = (\text{condition, action})\}\), the resolution function \(RS\), the set \(E\) of mappings to the external attributes, and the default action \(d\)
  – Resolution function (RS): FMR (First Matching Rule), LMR, Priority-based, ad hoc RS, ...
  – External attributes (E): priority, identity of the creator, and creation time
  – Condition types: exact-match, range-based, regex-based, and custom-match
The Overall I2NSF IM Design

```
+------------------------------------------+ 0..n  0..n +------------------------------------------+
|                                         |     |     |
|                         | / \   \   \   \     | External     |
|  External ECA Info Model + A ---------------+ Metadata   |
|                                         |     |     |
|                                         | / \  /  /  /    | Info Model  |
|                                         |     |     |
|                                         |     |     |
|                                         |     |     |
|                                         |     |     |
|                                         |     |     |
|                                         |     |     |
|                                         |     |     |
|                                         |     |     |
```

```
+------------------------------------------+ Metadata +------------------------------------------+
<p>| | | |
|                                         |     |     |
|                                         | / \   \   \   \     |
|                                         |     |     |
|                                         |     |     |
|                                         |     |     |
| Capability                             |</p>
<table>
<thead>
<tr>
<th>Sub-Model</th>
</tr>
</thead>
</table>
```
Network Security Info Sub-Model

ECAPolicyRule Extensions

Figure 3. Network Security Info Sub-Model ECAPolicyRule Extensions

Figure 5. Modeling Authorization Mechanisms
Event sub-class for Network Security

Example:

**UserSecurityEvent** has the attributes as below:

- **usrSecEventContent**: string;
- **usrSecEventFormat**
  - 0: unknown
  - 1: GUID (Generic Unique IDentifier)
  - 2: UUID (Universal Unique IDentifier)
  - 3: URI (Uniform Resource Identifier)
  - 4: FQDN (Fully Qualified Domain Name)
  - 5: FQPN (Fully Qualified Path Name)
- **usrSecEventType**
  - 0: unknown
  - 1: new user created
  - 2: new user group created
  - 3: user deleted
  - 4: user group deleted
  - 5: user logon
  - 6: user logoff
  - 7: user access request
  - 8: user access granted
  - 9: user access violation

Figure 10. Network Security Info Sub-Model Event Class Extensions
Figure 11. Network Security Info Sub-Model Condition Class Extensions

Figure 12. Network Security Info Sub-Model PacketSecurityCondition Class Extensions
Action sub-class for Network Security

- **IngressAction**: The purpose of this Class is to represent actions performed on packets that enter an NSF. Examples include pass, drop, mirror traffic.
- **EgressAction**: The purpose of this Class is to represent actions performed on packets that exit an NSF. Examples include pass, drop, mirror traffic, signal, encapsulate.
- **ApplyProfileAction**: The purpose of this Class is to represent applying a profile to packets to perform content security and/or attack mitigation control.
- **ApplySignatureAction**: The purpose of this Class is to represent applying a signature file to packets to perform content security and/or attack mitigation control.
Next Step

• Comments are welcome!
• Keep on being aligned with I2NSF framework and terminology drafts
• 3 information models
  – **General I2NSF capability information model** for Register interface: be included or another individual draft?
  – **NSF-facing interface policy information model**: be complemented with resolution strategies, external data, default action in next version
  – **Customer-facing interface policy information model**: this draft will not cover
• More polishing work
• Call for WG adoption
Thanks!

Liang Xia (Frank)