Security Policy Translation in I2NSF

draft-yang-i2nsf-security-policy-translation-03

IETF 104, Prague
March 26, 2019

Jinhyuk Yang, Jaehoon (Paul) Jeong, and Jinyong (Tim) Kim
Necessity for Policy Translator

• Policy Representation according to Users
  – The first policy is for I2NSF Users, and the second policy is for NSFs.

  - Block my son's computers from malicious websites.
  - Drop packets from the IP address 10.0.0.1 and 10.0.0.3 to harm.com and illegal.com

  – Even if I2NSF User gives the first high-level policy, I2NSF System needs to automatically translate it into the second low-level policy.
Previous Translation

• **XSLT-based Policy Translation**
  – Proposed by W3C at 1999.

• **Limitation**

  1. **Difficulty** of Security Policy Construction
     - The manager **MUST select** the proper NSF directly.

  2. **Inefficient** Maintenance
     - **Cannot adapt** automatically to a Data Model’s changes.
Proposed Translation

• Automata-based Policy Translation

• Approach
  1. Ease of Security Policy Construction
     - The manager doesn’t need to select the proper NSF.
  2. Efficient Maintenance
     - Can adapt automatically to a Data Model’s changes.
Translation Architecture

High-level policy

Low-level policy

Policy Translator

Data Extractor

Data Converter

Policy Generator

NSF Database

Consumer-Facing Interface

Target NSF

I2NSF User

High-level policy data

Low-level policy data

Translation
Step 1: Extractor (DFA)

High-level policy

Extraction

High-level policy data

<table>
<thead>
<tr>
<th>Rule Name</th>
<th>block_web</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>Son’s_PC</td>
</tr>
<tr>
<td>Destination</td>
<td>malicious</td>
</tr>
<tr>
<td>Action</td>
<td>block</td>
</tr>
</tbody>
</table>
Step 2: Data Converter (1/3)

High-level policy data

<table>
<thead>
<tr>
<th>Rule Name</th>
<th>block_web</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>Son’s_PC</td>
</tr>
<tr>
<td>Destination</td>
<td>malicious</td>
</tr>
<tr>
<td>Action</td>
<td>block</td>
</tr>
</tbody>
</table>

Low-level policy data

<table>
<thead>
<tr>
<th>Rule Name</th>
<th>block_web</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source IPv4</td>
<td>[10.0.0.1, 10.0.0.3]</td>
</tr>
<tr>
<td>URL Category</td>
<td>[harm.com, illegal.com]</td>
</tr>
<tr>
<td>Log Action</td>
<td>True</td>
</tr>
<tr>
<td>Drop Action</td>
<td>True</td>
</tr>
</tbody>
</table>
Step 2: Data Converter (2/3)

- **Log-keeper**
  - Rule Name
  - Source IPv4
  - Log Action

- **Web-filter**
  - Rule Name
    - Source IPv4
    - URL Category
    - Drop Action

- **DDoS Mitigation**
  - Rule Name
    - Source IPv4
    - Delay Time
    - Drop Action

- Eliminate Common Capability
- Eliminate Common Capability
- Delay Time
Step 2: Data Converter (3/3)

Low-level policy data

<table>
<thead>
<tr>
<th>Rule Name</th>
<th>block_web</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source IPv4</td>
<td>[10.0.0.1, 10.0.0.3]</td>
</tr>
<tr>
<td>URL Category</td>
<td>[harm.com, illegal.com]</td>
</tr>
<tr>
<td>Log Action</td>
<td>True</td>
</tr>
<tr>
<td>Drop Action</td>
<td>True</td>
</tr>
</tbody>
</table>

Policy Provisioning

Log-keeper

<table>
<thead>
<tr>
<th>Rule Name</th>
<th>block_web</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source IPv4</td>
<td>[10.0.0.1, 10.0.0.3]</td>
</tr>
<tr>
<td>Log Action</td>
<td>True</td>
</tr>
</tbody>
</table>

Web-filter

<table>
<thead>
<tr>
<th>Rule Name</th>
<th>block_web</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source IPv4</td>
<td>[10.0.0.1, 10.0.0.3]</td>
</tr>
<tr>
<td>URL Category</td>
<td>[harm.com, illegal.com]</td>
</tr>
<tr>
<td>Drop Action</td>
<td>True</td>
</tr>
</tbody>
</table>
Step 3: Generator (CFG)

Content Production

<table>
<thead>
<tr>
<th>Rule Name</th>
<th>Source IPv4</th>
<th>URL Category</th>
<th>Drop Action</th>
</tr>
</thead>
</table>

Structure Production

Low-level policy data

- Rule Name: block_web
- Source IPv4: [10.0.0.1, 10.0.0.3]
- URL Category: [harm.com, illegal.com]
- Drop Action: True

Generation

Low-level policy

```
<rule-name>block_web</rule-name>
<rules>
  <condition>
    <packet>
      <ipv4>10.0.0.1</ipv4>
      <ipv4>10.0.0.3</ipv4>
    </packet>
    <payload>
      <url>harm.com</url>
      <url>illegal.com</url>
    </payload>
  </condition>
  <action>drop</action>
</rules>
</I2NSF>
```

Figure 7: Generator Construction for Web-Filter NSF
Updates from the Previous Versions

• The Previous Draft:
  – draft-yang-i2nsf-security-policy-translastion-02

• Changes from the previous versions
  – Explanations have been added for explaining NSF Database component.
  – The section “Implementation Consideration” is added for guidelines.
  – Other changes are described in detail in Appendix section.
Next Steps

- **WG Adoption Call at IETF 104**
  - Key Functionality for I2NSF’s Implementation & Deployment in the real world.
  - This draft can provide the I2NSF developers with the guidelines to implement Security Policy Translator.
  - This draft aims at an Informational RFC.
  - The security policy translator is proved through IETF-104 Hackathon.