Security Policy Translation in I2NSF

(draft-jeong-i2nsf-security-policy-translation-04)

IETF 105, Montreal

July 25, 2019

Jaehoon (Paul) Jeong [Presenter], Jinhyuk Yang, Chaehong Chung, and Jinyong (Tim) Kim

Necessity for Policy Translator

- Policy Specification in Different Levels
  - Both policies are equivalent. The first policy is for I2NSF Users, and the second policy is for NSFs.

```
o  Block my son's computers from malicious websites.

o  Drop packets from the IP address 10.0.0.1 and 10.0.0.3 to harm.com
   and illegal.com
```

  - I2NSF requires a translator that automatically converts the first policy into the second policy for the security policy enforcement.

Proposed Translation

- Automata-based Policy Translation
  - New method for XML-based policy translation.

- Approach
  1. Ease of Security Policy Construction
     - A security administrator doesn't need to select proper NSFs manually.
  2. Efficient Maintenance
     - The translator can adapt automatically to the changes of data models.

Updates from the Previous Version

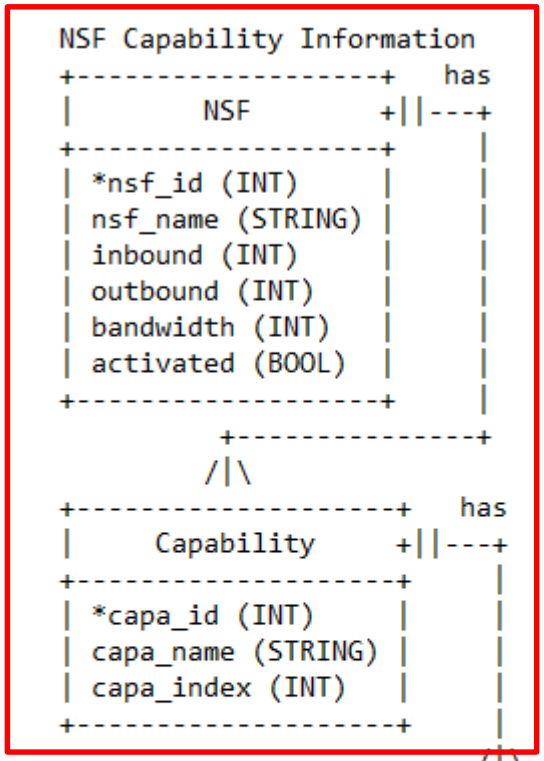
- The Previous Draft:
  - draft-yang-i2nsf-security-policy-translastion-03

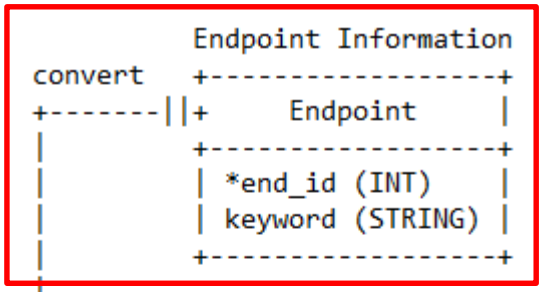
- Changes from the previous versions
  - In Section “NSF Database”, Entity-Relationship Diagram (ERD) of NSF Database is added.

  - In Section “Data Conversion in Data Converter”, a mapping list of data model fields between Consumer-Facing Interface and NSF-Facing Interface is added for describing the process of data conversion in detail.

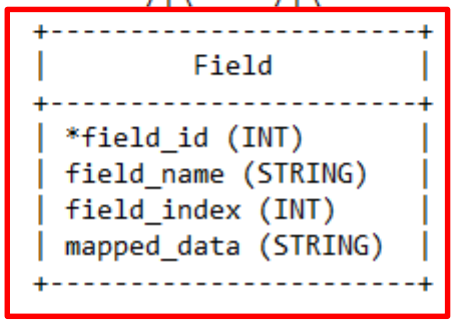
Change 1: ERD of NSF Database

Registration Interface DM





Consumer-Facing Interface DM



NSF-Facing Interface DM

Change 2: Mapping List

- Mapping List of Attributes
  - List all the mapped attributes between Consumer-Facing Interface Data Model and NSF-Facing Interface Data Model.
  - Describe the process of mapping attributes in detail.

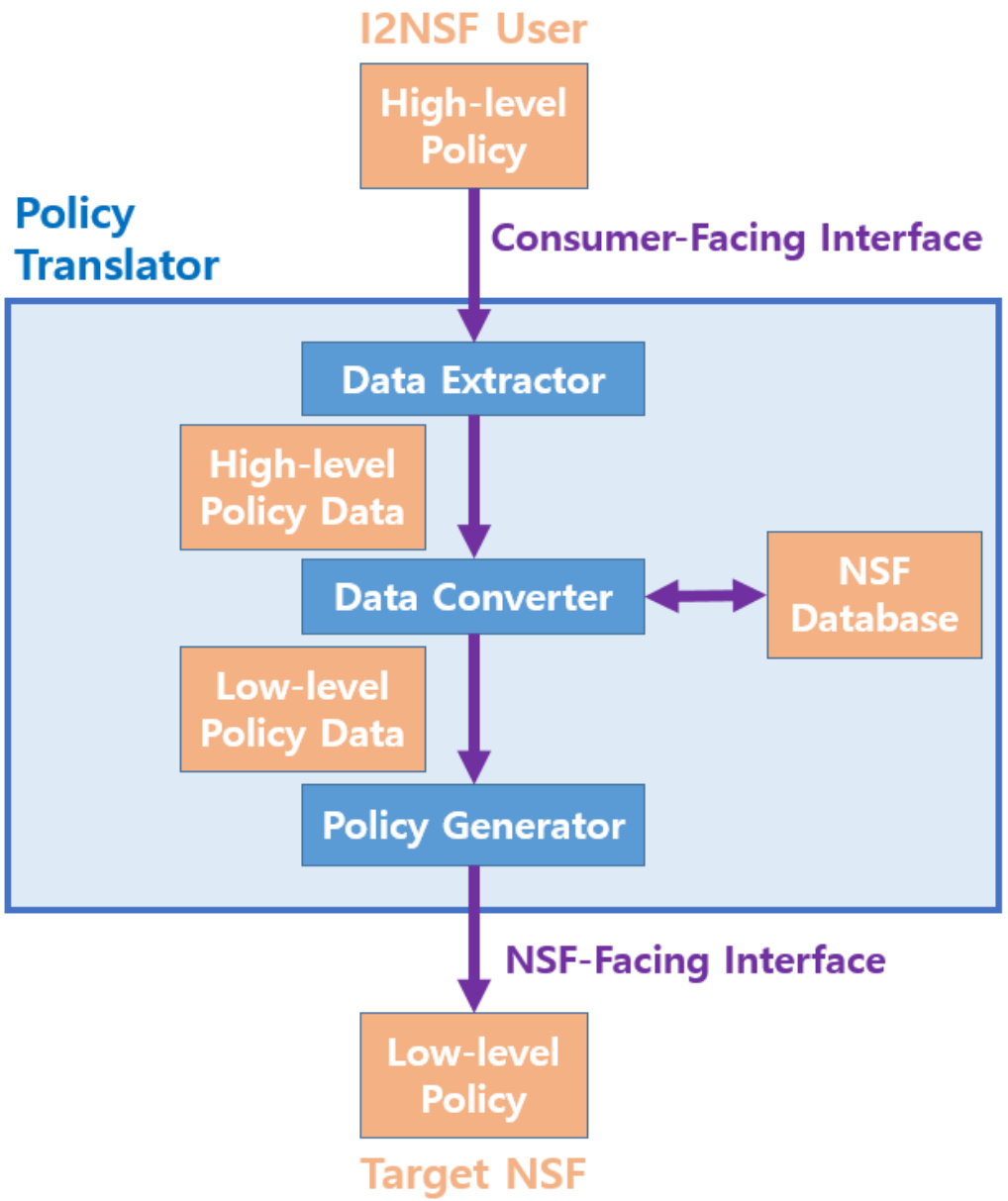
```
policy/rule/condition/ddos-condition/source-target/src-target
    -> reference: policy/endpoint-group/device-group/name
        -> extract: policy/endpoint-group/device-group/date
            -> mapping: i2nsf-security-policy/rule/date
        -> extract: policy/endpoint-group/device-group/ip-address
            -> mapping: i2nsf-security-policy/system-policy/rules/condition-clause-
container/packet-security-ipv4-condition/pkt-sec-ipv4-src/ipv4-address/ipv4
        -> extract: policy/endpoint-group/device-group/range-ip-address/start-ip-address
            -> mapping: i2nsf-security-policy/system-policy/rules/condition-clause-
container/packet-security-ipv4-condition/pkt-sec-ipv4-src/range-ipv4-address/start-ipv4-address
        -> extract: policy/endpoint-group/device-group/range-ip-address/end-ip-address
            -> mapping: i2nsf-security-policy/system-policy/rules/condition-clause-
container/packet-security-ipv4-condition/pkt-sec-ipv4-src/range-ipv4-address/end-ipv4-address
```

Next Steps

- **Request for WG Adoption Call**
  - Security Policy Translation is a key part of Security Controller.
  - This translation is required for the implementation and deployment of I2NSF:
    - It provides the mapping of the high-level YANG data model and low-level YANG data model.
    - It gives the developers a good example of security policy translator in terms of the architecture and process of the translator.
  - This draft aims at an Informational RFC.

Appendix:
Architecture and Process of Security Policy Translator

Architecture of Security Policy Translator



High-level policy

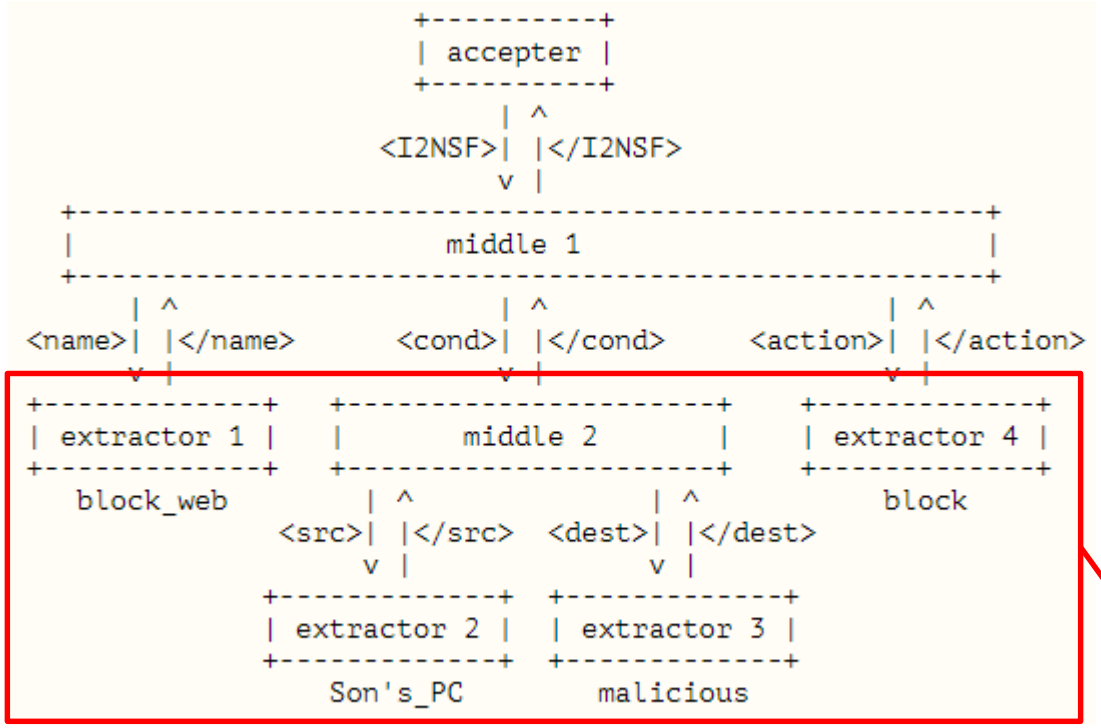
```
<I2NSF>
    <name>block_web</name>
    <cond>
        <src>Son's_PC</src>
        <dest>malicious</dest>
    </cond>
    <action>block<action>
</I2NSF>
```

Translation

Low-level policy

```
<I2NSF>
    <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>
```

Step 1: Extractor (DFA)



High-level policy

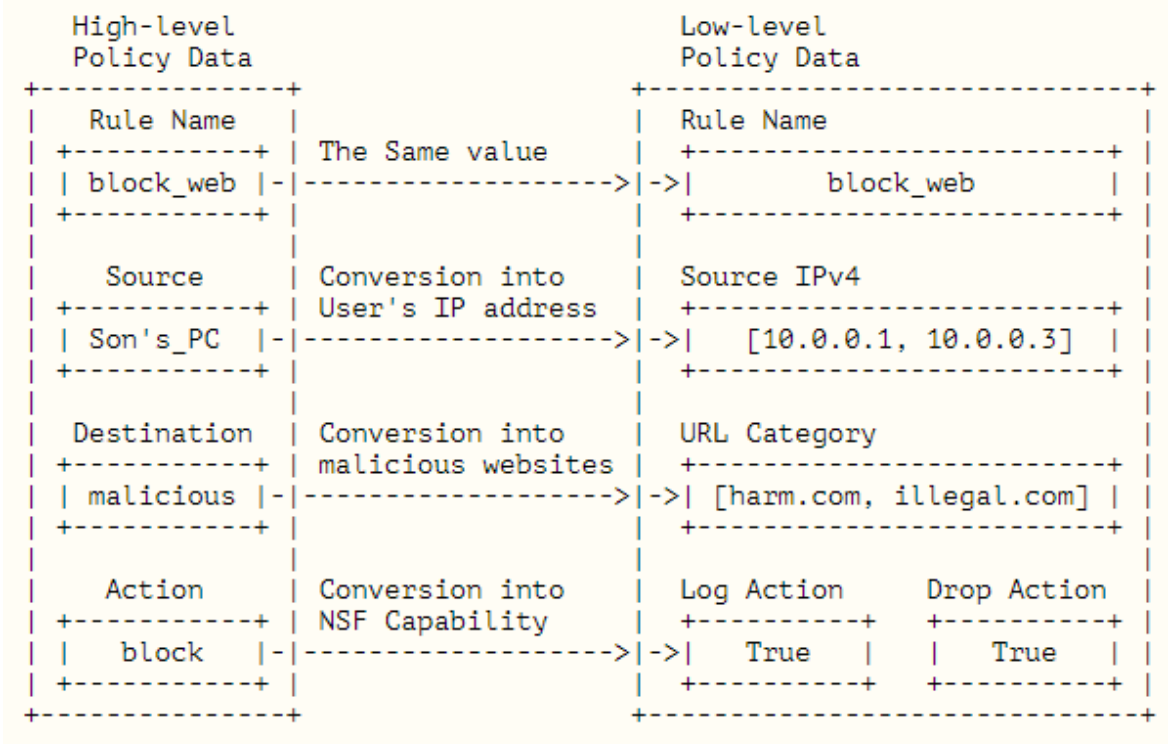
```
<I2NSF>
    <name>block_web</name>
    <cond>
        <src>Son's_PC</src>
        <dest>malicious</dest>
    </cond>
    <action>block<action>
</I2NSF>
```

Extraction

High-level policy data

Rule Name	block_web
Source	Son's_PC
Destination	malicious
Action	block

Step 2: Data Converter (1/3)



High-level policy data

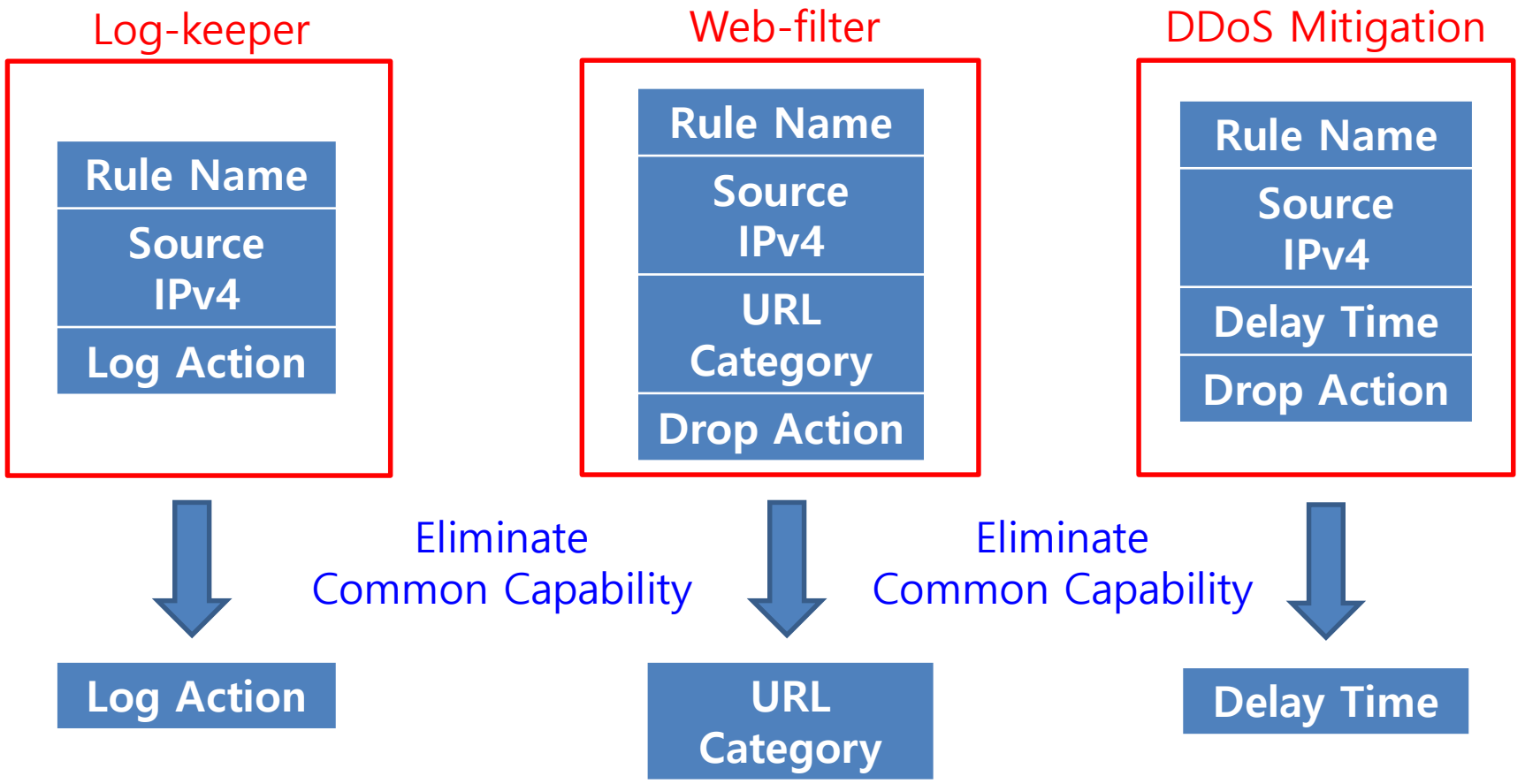
Rule Name	block_web
Source	Son's_PC
Destination	malicious
Action	block

Data Conversion

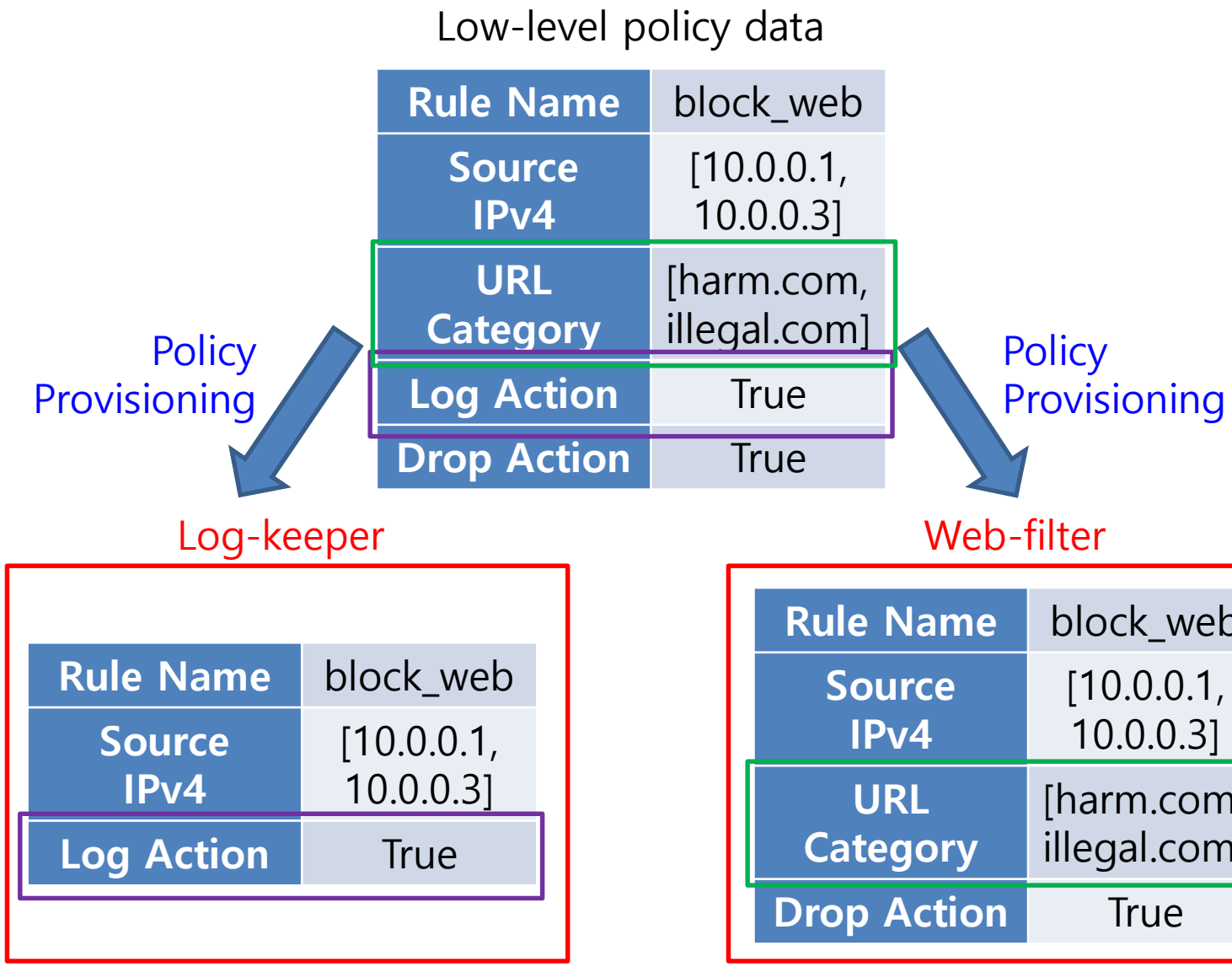
Low-level policy data

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

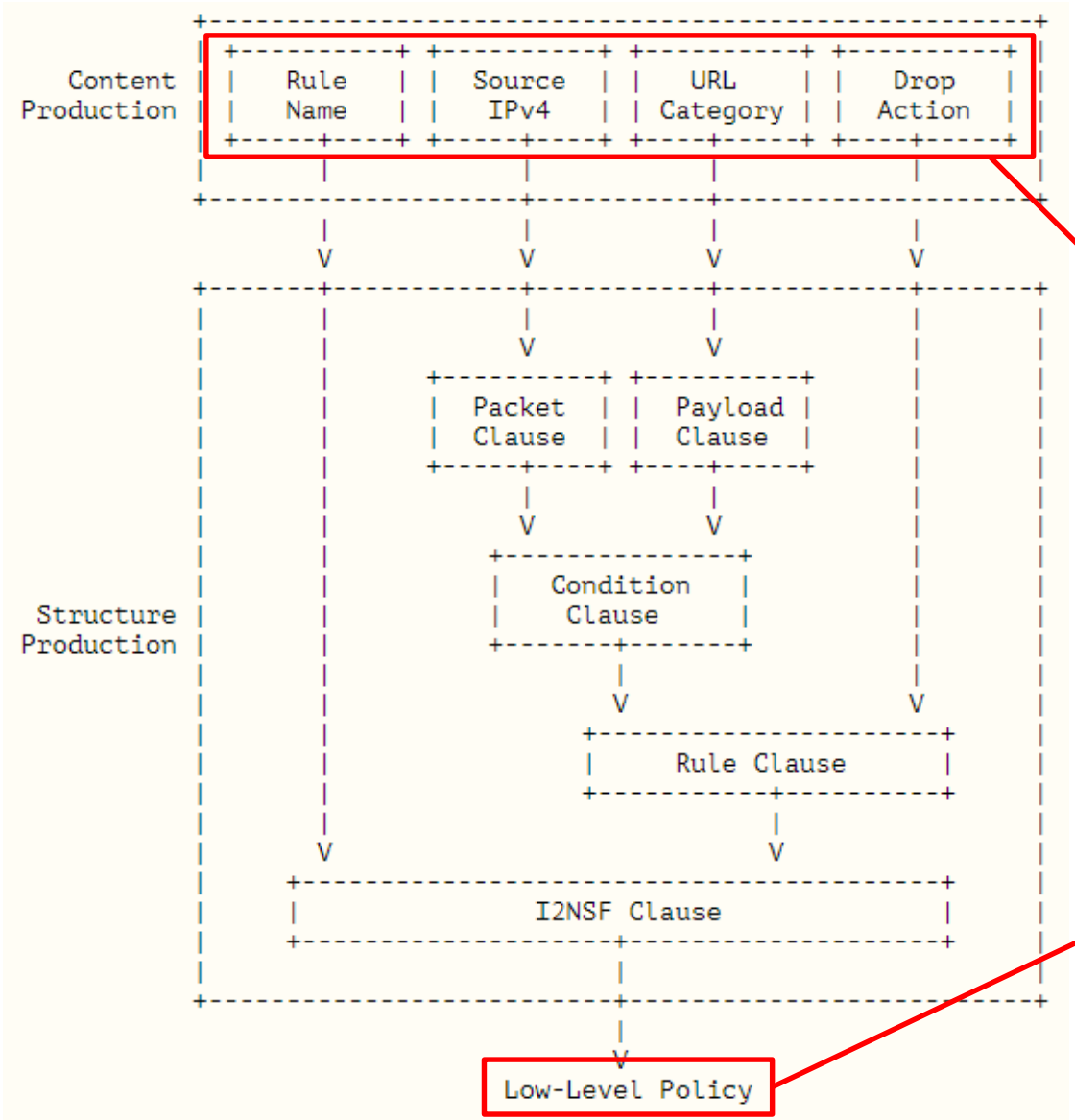
Step 2: Data Converter (2/3)



Step 2: Data Converter (3/3)



Step 3: Generator (CFG)



Generator Construction for Web-Filter NSF

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

```
<I2NSF>
    <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>
```