SDN-based Security Services using I2NSF
(draft-jeong-i2nsf-sdn-security-services-03)


Jaehoon (Paul) Jeong
pauljeong@skku.edu
Contents

1 Architecture for SDN-based Security Services
2 Program Execution for Firewall Filtering
3 Next Steps
Architecture for SDN-based Security Services
High-level Architecture for SDN-based Security Services

- An administrator enforces security policies for the security services.
- Access control rules are applied to network by SDN controller.
- Network resources (e.g., switches) act to mitigate network attacks. e.g., dropping packets for security policies or suspicious patterns.
1. **Client/AppGW** asks for security services with high-level security policies to Security Controller via **Service Layer Interface**.

2. **Security Controller** calls function-level security services via **Capability Layer Interface**.

3. **Security Function** tells Switch Controller its required security services via **Northbound Interface**.

4. **Switch Controller** sets up forwarding rules for the security services on Switches via **Southbound Interface**.
Procedure of SDN-based Security Services

1. Service Layer Interface (e.g., RESTCONF, SUPA)
2. Capability Layer Interface (e.g., NETCONF/YANG)
3. Northbound Interface (e.g., NETCONF/YANG)
4. Southbound Interface (e.g., NETCONF/YANG, I2RS)
5. Traffic Generation
6. Packet Filtering
7. Filtered Packets

App Gateway → Security Controller → Firewall → Web Filter → Switch Controller

Clients

Install new rules (e.g., drop packets with security policy)

Switch 1 → Switch 2 → Switch 3

Valid packets
Invalid packets

Incoming packets

Outgoing packets
Use Cases of SDN-based Security Services

Security Functions

- Firewall
- DDoS-Attack Mitigator
- Web Filter, DPI, etc.

Switch Controller

Switch 1 → Switch Controller
Switch 2
Switch 3

Incoming packets ➔ Filtered packets

Use Cases:
1. Firewall
2. DDoS-Attack Mitigator
3. Web Filter, DPI, etc.
Program Execution for Firewall Filtering
1. Client and Server make a session by using NETCONF/YANG.

2. Client configures the **firewall table** of Server to block specific IP addresses.

3. Server (as Security Function in virtual machine) asks firewall filtering to be set up in Switches through Switch Controller.

4. After the configuration of the firewall table, packets from Attacker are dropped.
module filter {
    namespace "http://skku.com/cps/example/filter";
    prefix filter;

    import ietf-inet-types {
        prefix inet;
    }

    import tailf-common {
        prefix tailf;
    }

    /* A set of filtering structures */
    container filters {
        tailf:callpoint hcp;

        list filter {
            key identification;
            max-elements 64;
            leaf identification {
                type string;
            }
            leaf where {
                type string;
                mandatory true;
            }
            leaf ip {
                type inet:ip-address;
                mandatory true;
            }
        }
    }
}
NETCONF Command for IP Address Filtering (1/4)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.0</capability>
  </capabilities>
</hello>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <edit-config>
    <target>
      <running/>
    </target>
    <config>
        <filter nc:operation="create">
          <identification>Malicious_Node_1</identification>
          <where>Source</where>
          <ip>115.145.178.166</ip>
        </filter>
      </filter>
    </config>
  </edit-config>
</rpc>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="2">
  <close-session/>
</rpc>
```
NETCONF Command for IP Address Filtering (2/4)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <capabilities>
        <capability>urn:ietf:params:netconf:base:1.0</capability>
    </capabilities>
</hello>

<?xml version="1.0" encoding="UTF-8"?>
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
    <edit-config>
        <target>
            <running/>
        </target>
        <config>
            <filters xmlns="http://skku.com/cps/example/filter"
                     xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
                <filter nc:operation="create">
                    <identification>Malicious_Node_2</identification>
                    <where>Source</where>
                    <ip>115.145.178.167</ip>
                </filter>
            </filters>
        </config>
    </edit-config>
</rpc>

<?xml version="1.0" encoding="UTF-8"?>
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="2">
    <close-session/>
</rpc>
```
NETCONF Command for IP Address Filtering (3/4)

NETCONF Command

```xml
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <ok/>
</rpc-reply>
```

IP Addresses for Filtering

```
> show
  IP Malicious_Node_1          deny 115.145.178.166
  IP Malicious_Node_2          deny 115.145.178.167
```
### NETCONF Command for IP Address Filtering (4/4)

**Chain INPUT (policy ACCEPT 10 packets, 1280 bytes)**

<table>
<thead>
<tr>
<th>pkts</th>
<th>bytes</th>
<th>target</th>
<th>prot</th>
<th>opt in</th>
<th>out</th>
<th>source</th>
<th>destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>all</td>
<td>--</td>
<td>*</td>
<td>127.0.0.1</td>
<td>0.0.0.0/0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>tcp</td>
<td>--</td>
<td>*</td>
<td>0.0.0.0/0</td>
<td>0.0.0.0/0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>tcp</td>
<td>--</td>
<td>*</td>
<td>0.0.0.0/0</td>
<td>0.0.0.0/0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>tcp</td>
<td>--</td>
<td>*</td>
<td>0.0.0.0/0</td>
<td>0.0.0.0/0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>tcp</td>
<td>--</td>
<td>*</td>
<td>0.0.0.0/0</td>
<td>0.0.0.0/0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>tcp</td>
<td>--</td>
<td>*</td>
<td>0.0.0.0/0</td>
<td>0.0.0.0/0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>tcp</td>
<td>--</td>
<td>*</td>
<td>0.0.0.0/0</td>
<td>0.0.0.0/0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>tcp</td>
<td>--</td>
<td>*</td>
<td>0.0.0.0/0</td>
<td>0.0.0.0/0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>tcp</td>
<td>--</td>
<td>*</td>
<td>0.0.0.0/0</td>
<td>0.0.0.0/0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>tcp</td>
<td>--</td>
<td>*</td>
<td>0.0.0.0/0</td>
<td>0.0.0.0/0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>udp</td>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>127.0.0.1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>udp</td>
<td>0</td>
<td>0</td>
<td>DROP</td>
<td>115.145.178.166</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>udp</td>
<td>0</td>
<td>0</td>
<td>DROP</td>
<td>115.145.178.167</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>udp</td>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>115.145.178.166</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>tcp</td>
<td>--</td>
<td>*</td>
<td>0.0.0.0/0</td>
<td>0.0.0.0/0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>tcp</td>
<td>--</td>
<td>*</td>
<td>0.0.0.0/0</td>
<td>0.0.0.0/0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>tcp</td>
<td>--</td>
<td>*</td>
<td>0.0.0.0/0</td>
<td>0.0.0.0/0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>tcp</td>
<td>--</td>
<td>*</td>
<td>0.0.0.0/0</td>
<td>0.0.0.0/0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>udp</td>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>115.145.178.166</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>udp</td>
<td>0</td>
<td>0</td>
<td>DROP</td>
<td>115.145.178.167</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>udp</td>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>115.145.178.166</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>all</td>
<td>0</td>
<td>0</td>
<td>ACCEPT</td>
<td>115.145.178.166</td>
</tr>
</tbody>
</table>

---

Drop Rules for Firewall

---

14
Next Steps

• We will design and implement our **Framework of SDN-based Security Services using I2NSF**:
  
  – Service Layer Interface
    • Use SUPA WG’s Policy Abstraction and RESTCONF
  
  – Capability Layer Interface
    • Use **draft-xia-i2nsf-capability-interface-im-04**
  
  – Northbound Interface
    • Use **NETCONF/YANG** and **OpenDayLight**
  
  – Southbound Interface
    • Use **NETCONF/YANG** and SFC WG’s Service Chaining
    • Construct SDN Network using **Mininet**