SDN-based Security Services using I2NSF

(draft-jeong-i2nsf-sdn-security-services-03)

http://datatracker.ietf.org/doc/draft-jeong-i2nsf-sdn-security-services/

성균관대학교 SUNG KYUN KWAN UNIVERSITY

Jaehoon (Paul) Jeong
pauljeong@skku.edu

ETRI 한국전자통신연구원 Electronics and Telecommunications Research Institute

Contents

Ⅰ Architecture for SDN-based Security Services

Ⅱ Program Execution for Firewall Filtering

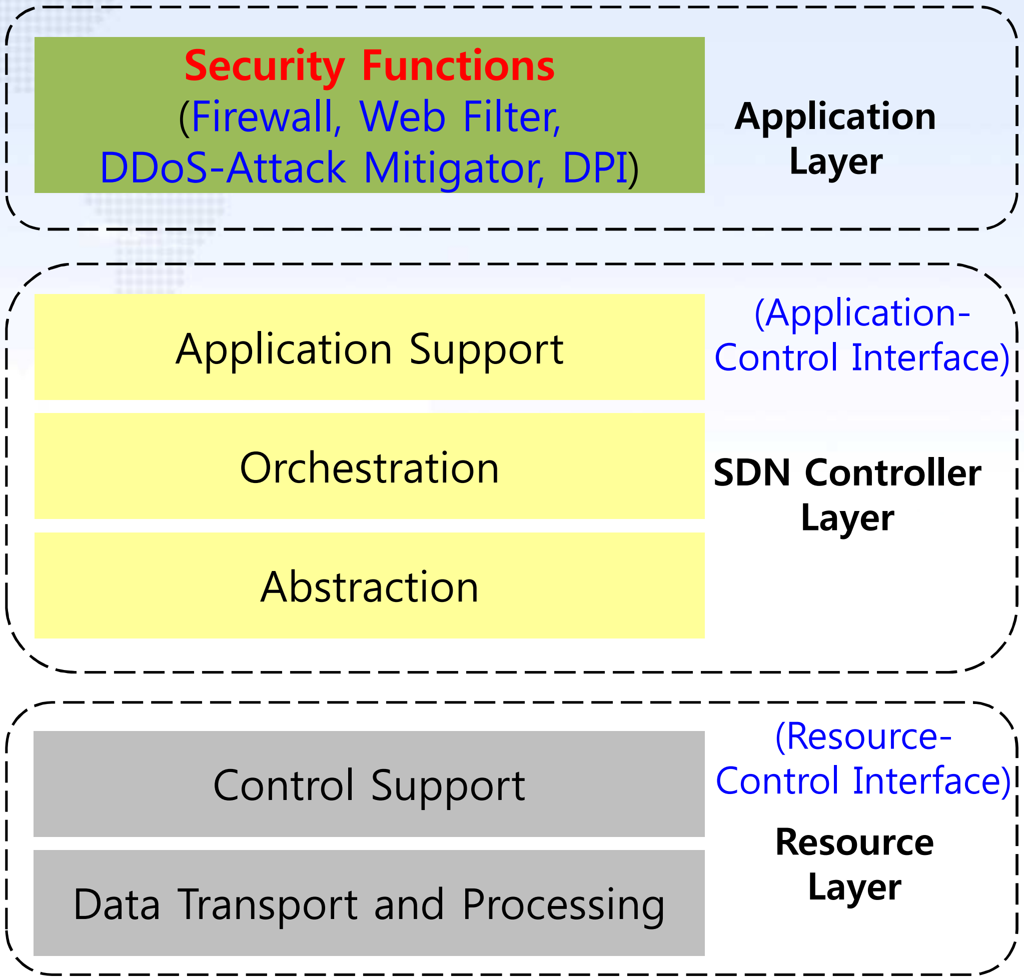
Ⅲ Next Steps

Architecture for SDN-based Security Services

Architecture (1/2)

❖ 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


Security Functions
(Firewall, Web Filter, DDoS-Attack Mitigator, DPI)
Application Layer
Application Support
(Application-Control Interface)
Orchestration
SDN Controller Layer
Abstraction
Control Support
(Resource-Control Interface)
Resource Layer
Data Transport and Processing

Architecture (2/2)

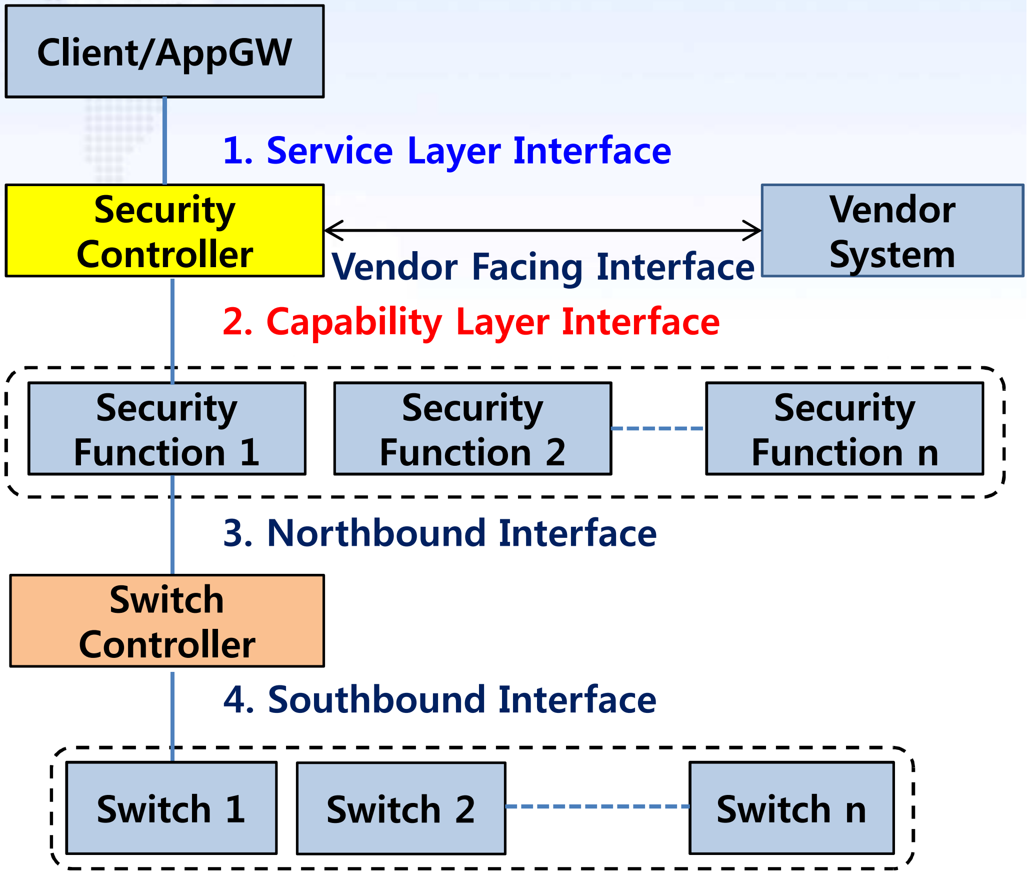
A framework to support SDN-based security services using I2NSF

1. **Client/AppGW** asks for security services with high-level security policies to Security Controller via **Service Layer Interface.**

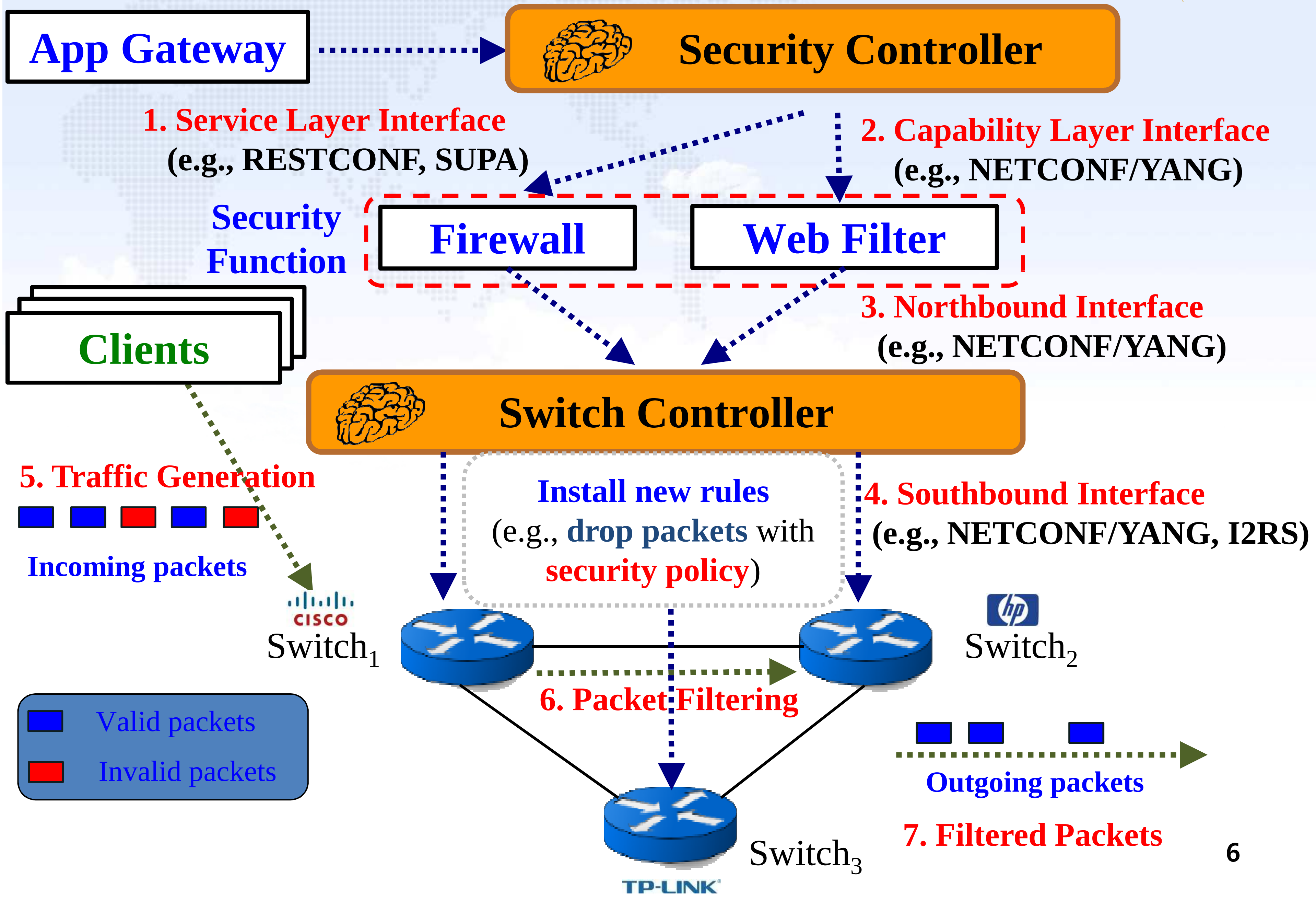
2. **Security Controller** calls function-level security services via **Capability Layer Interface.**

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

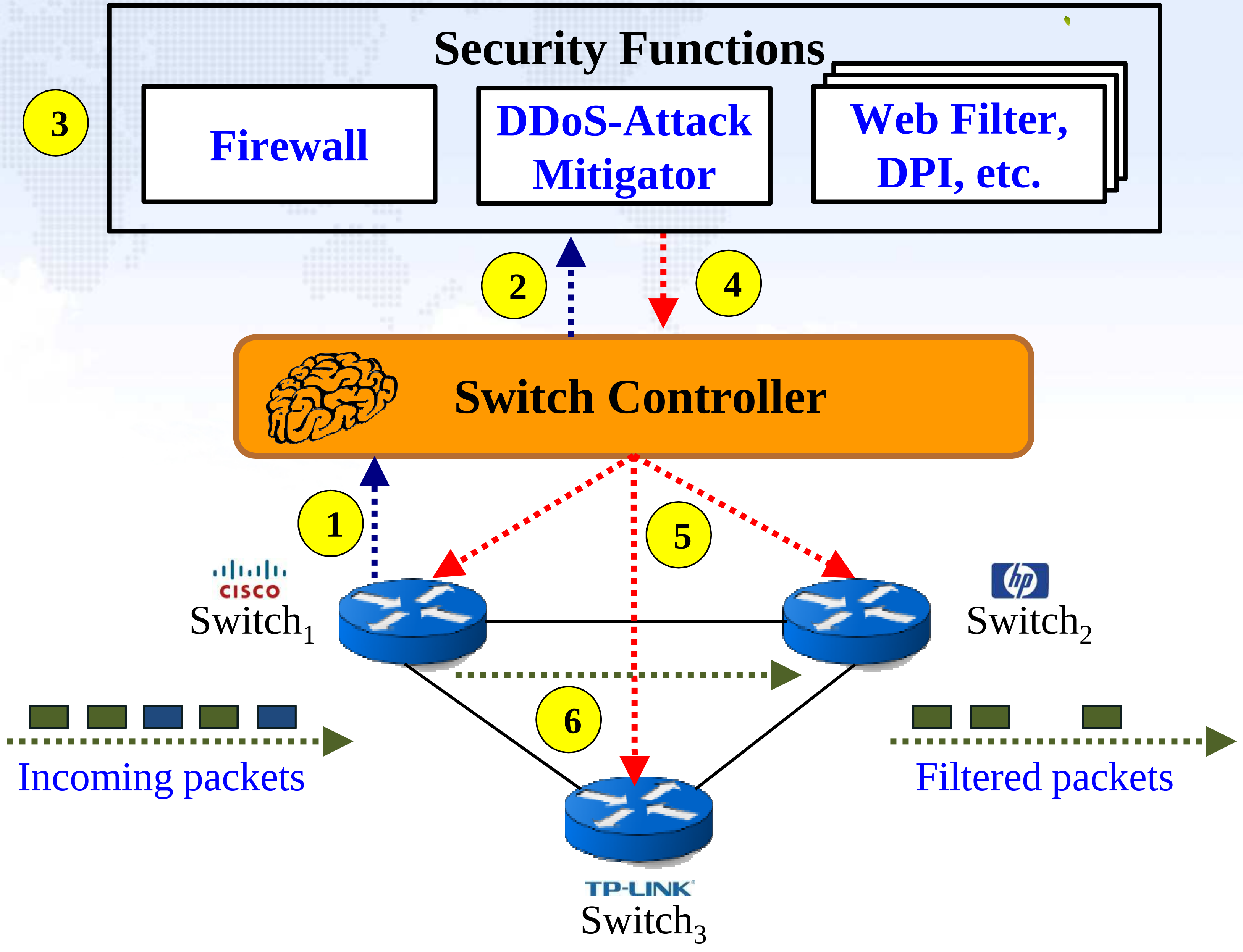
4. **Switch Controller** sets up forwarding rules for the security services on Switches via **Southbound Interface.**



Procedure of SDN-based Security Services

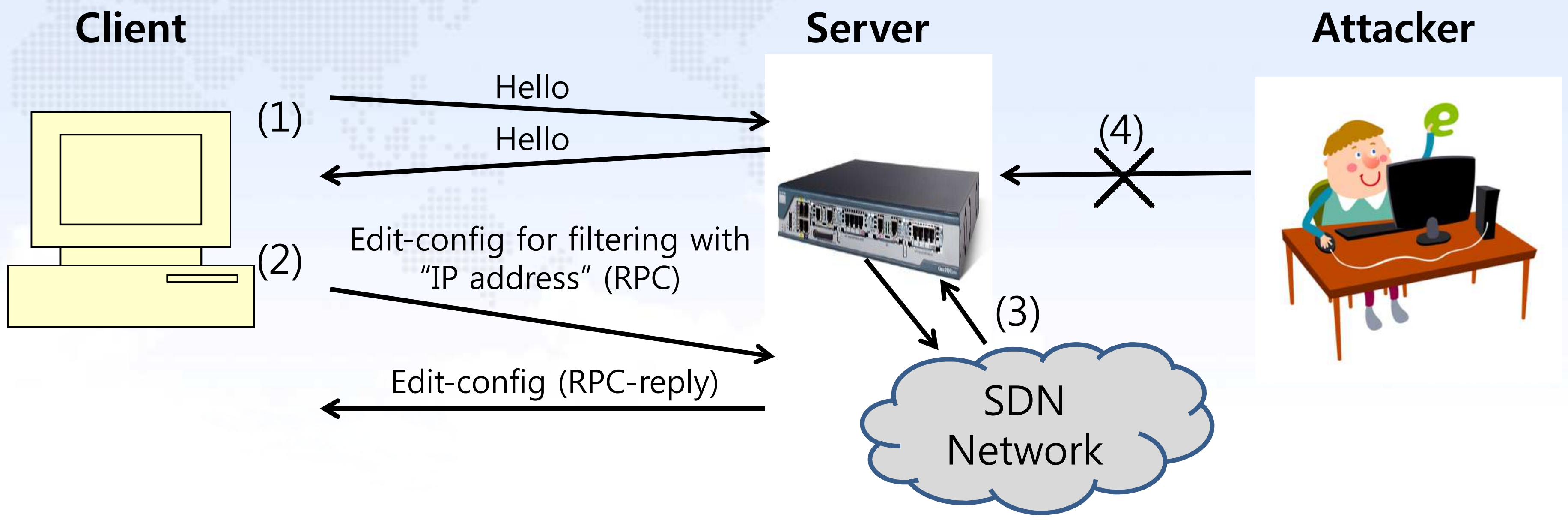


Use Cases of SDN-based Security Services



Program Execution for Firewall Filtering

Procedure for SDN-based 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.

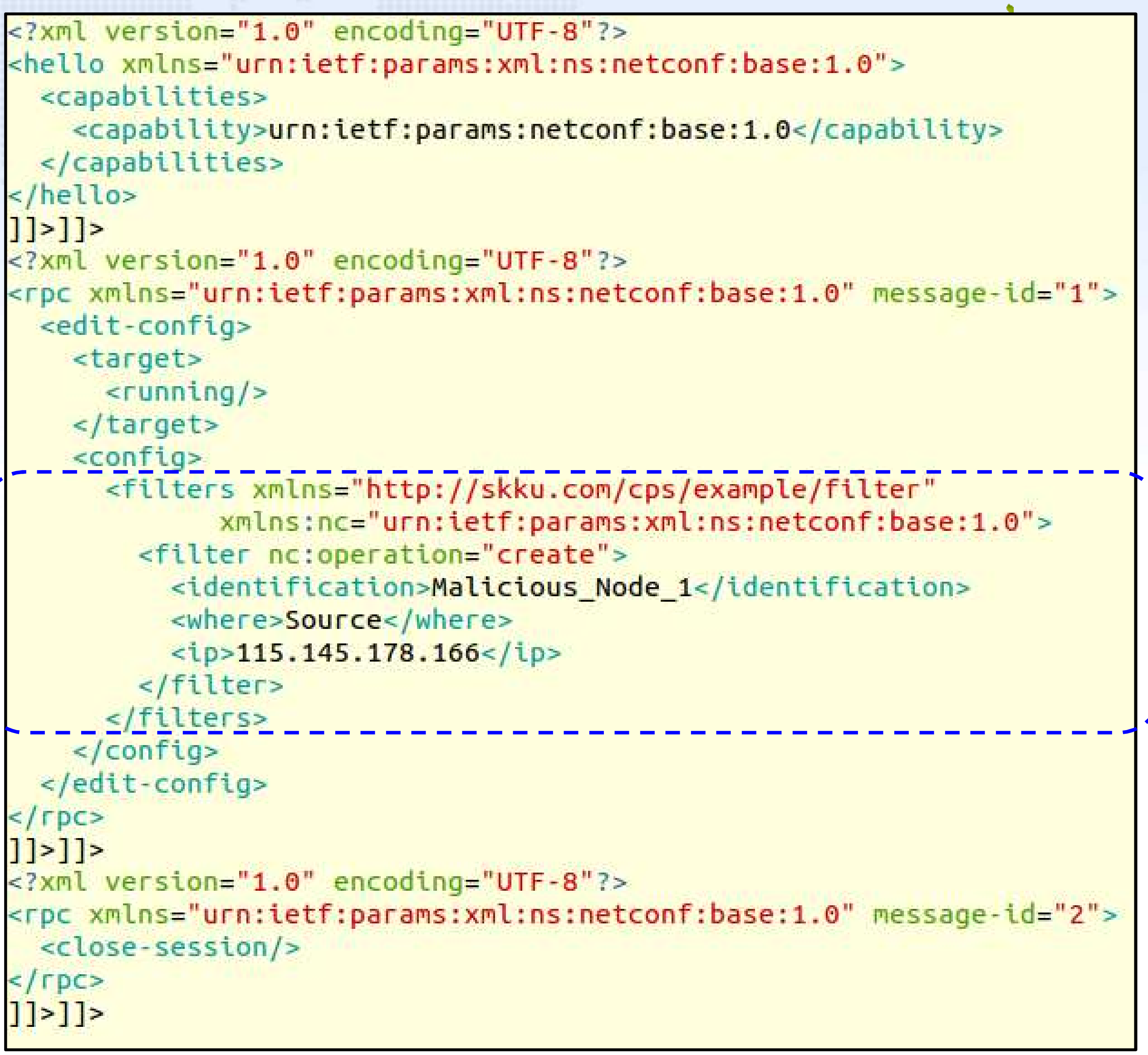
YANG Data Modeling for IP Address Filtering

```
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;
      }
    }
  }
}
```

IP Address Filtering

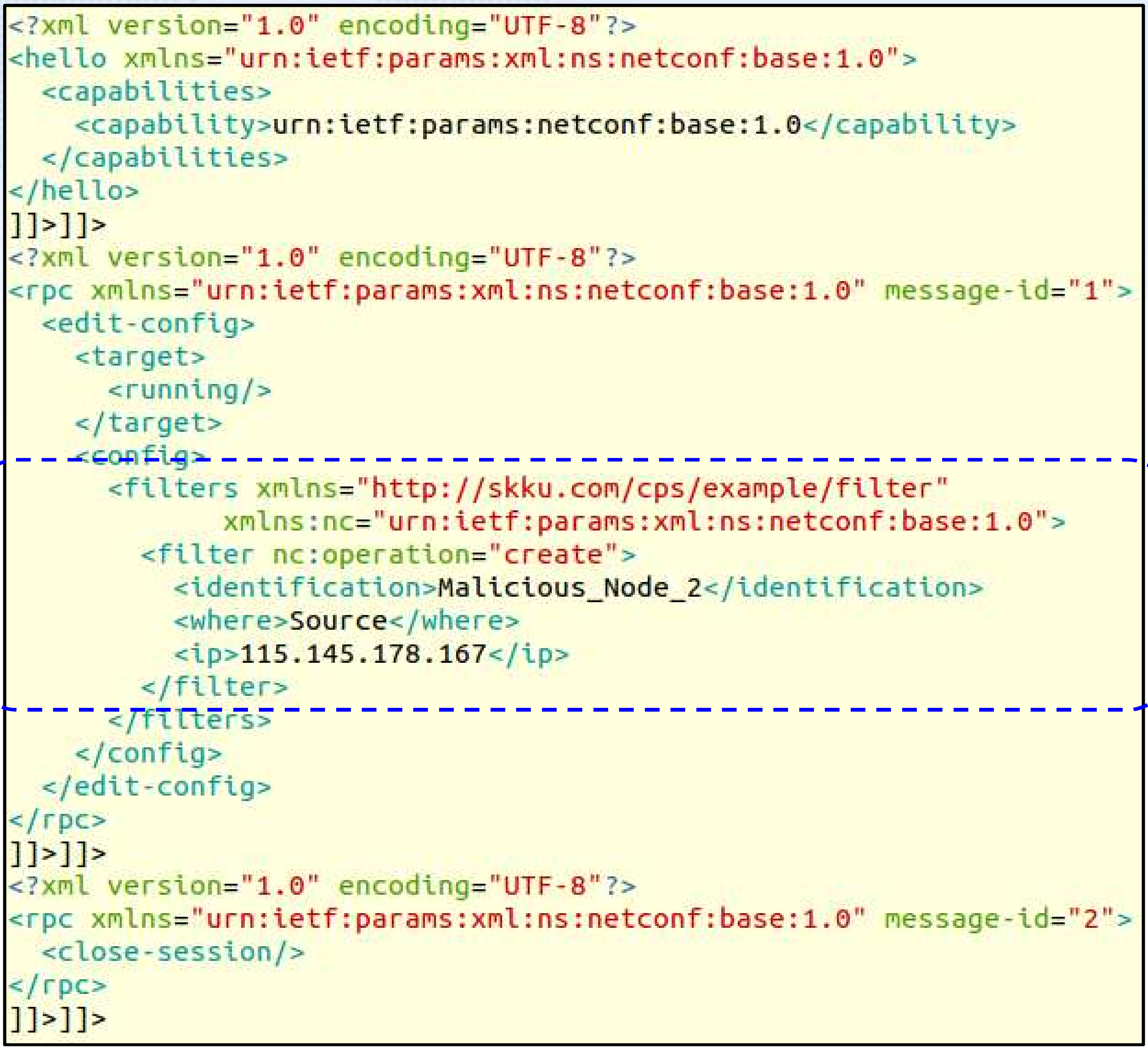
NETCONF Command for IP Address Filtering (1/4)

IP Address Filtering For Malicious Node 1



```
<?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_1</identification>
          <where>Source</where>
          <ip>115.145.178.166</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 (2/4)

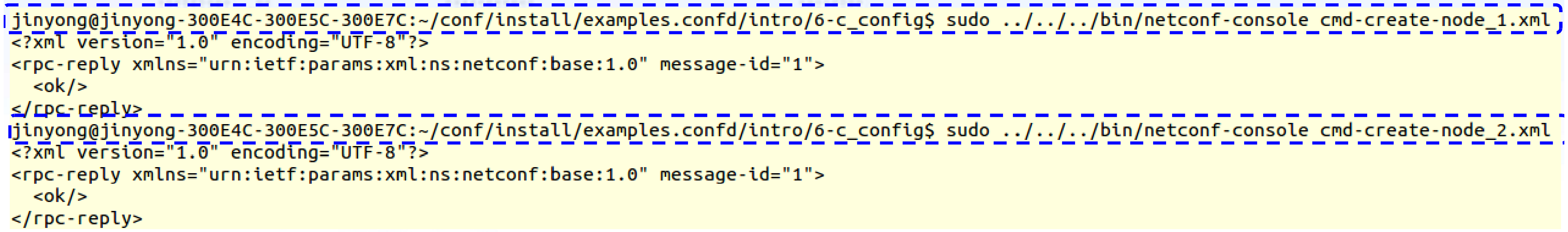


```
<?xml version="1.0" encoding="UTF-8"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.0</capability>
  </capabilities>
</hello>
]]>]]>
<?xml version="1.0" encoding="UTF-8"?>
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"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

IP Address Filtering For Malicious Node 2

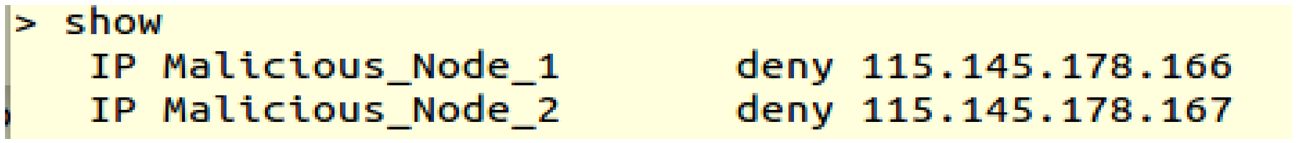
NETCONF Command for IP Address Filtering (3/4)

NETCONF Command



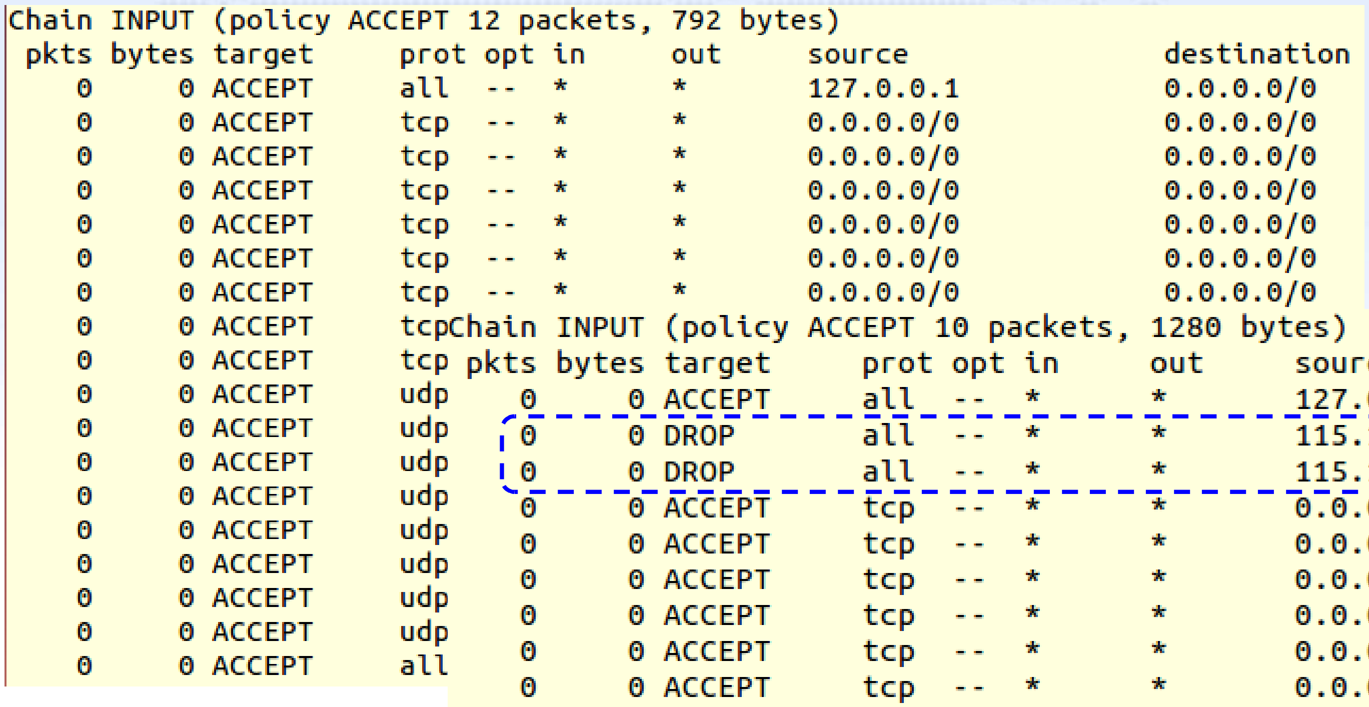
```
jinyong@jinyong-300E4C-300E5C-300E7C:~/conf/install/examples.confd/intro/6-c_config$ sudo ../../../bin/netconf-console cmd-create-node_1.xml
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <ok/>
</rpc-reply>
jinyong@jinyong-300E4C-300E5C-300E7C:~/conf/install/examples.confd/intro/6-c_config$ sudo ../../../bin/netconf-console cmd-create-node_2.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

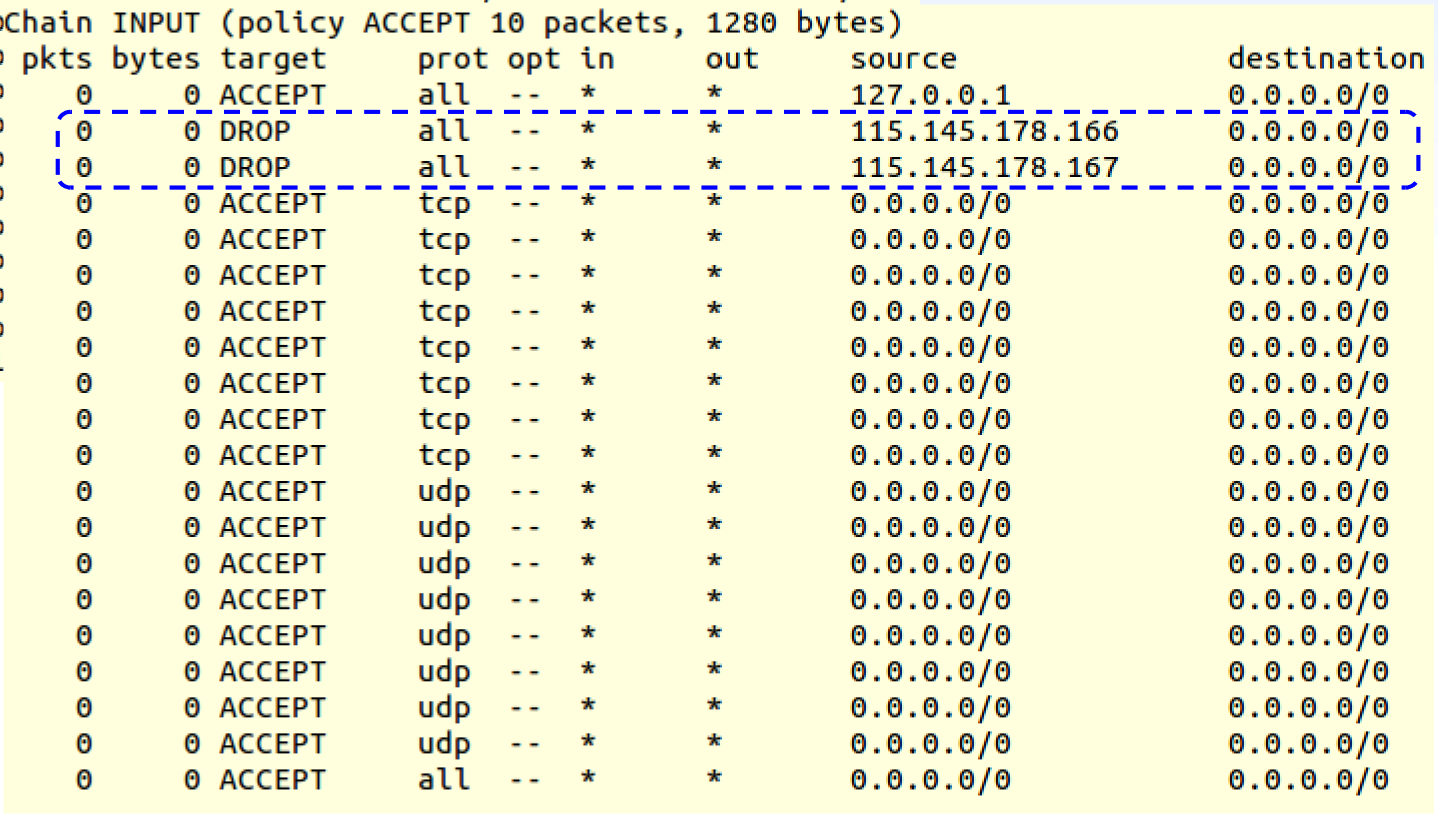


```
> show
  IP Malicious_Node_1        deny 115.145.178.166
  IP Malicious_Node_2        deny 115.145.178.167
```

NETCONF Command for IP Address Filtering (4/4)



```
Chain INPUT (policy ACCEPT 12 packets, 792 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  *      *       127.0.0.1            0.0.0.0/0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     tcp
    0     0 ACCEPT     tcp
    0     0 ACCEPT     udp
    0     0 ACCEPT     udp
    0     0 ACCEPT     udp
    0     0 ACCEPT     udp
    0     0 ACCEPT     udp
    0     0 ACCEPT     udp
    0     0 ACCEPT     udp
    0     0 ACCEPT     udp
    0     0 ACCEPT     all
```



```
Chain INPUT (policy ACCEPT 10 packets, 1280 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  *      *       127.0.0.1            0.0.0.0/0
    0     0 DROP       all  --  *      *       115.145.178.166      0.0.0.0/0
    0     0 DROP       all  --  *      *       115.145.178.167      0.0.0.0/0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     all  --  *      *       0.0.0.0/0            0.0.0.0/0
```

Drop Rules for Firewall

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**