# Laboratory of Network Programmability

Assignment 1

Andrea Serafini, Gustavo Mazzanti, Piero Sanchi

Maggio 2023

# Indice

# Capitolo 1

# Requisiti

## 1.1 Requisitazione e Topologia di rete

L'assignment consiste nell'implementazione di una topologia formata da uno switch, tre host ed un firewall con le seguenti regole:

- default "Deny";
- H1 and H2 can talk for any IP traffic;
- H1 can talk to port 80 of H3 only.

La suddetta topologia su cui il firewall dovrà operare è rappresentata nella seguente figura:

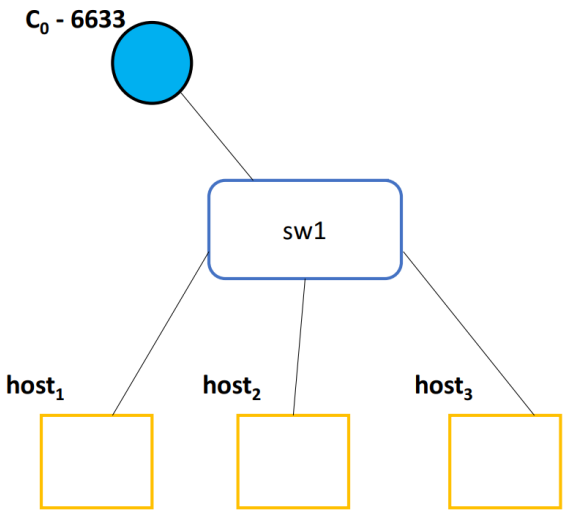


Figura 1.1: Topologia di rete formata da un firewall, uno switch e tre host.

La soluzione proposta è composta da uno script per creare la topologia sdn ed un controller ryu custom. Lo script è una versione di quello fornito a lezione, modificata a dovere per adattarlo a questa topologia. Il vero e proprio controller nasce dall'implementazione di ryu "simple_switch_13", modificato affinché soddisfi appieno i requisiti sopra riportati. In aggiunta a questo abbiamo ritenuto comodo adottare degli script utili per eseguire comandi in successione più agilmente.

# Capitolo 2

# Soluzione proposta

## 2.1 Struttura del firewall

Di seguito illustreremo brevemente la struttura del firewall oggetto dell'elaborato:

Listing 2.1: Firewall Structure

```
if out_port != ofproto.OFPP_FLOOD and eth.ethertype != ether_types.ETH_TYPE_ARP:

            isRule2 = False
            isTCP = False

            # Extract packet details
            if eth.ethertype == ether_types.ETH_TYPE_IP:
                ip = pkt.get_protocol(ipv4.ipv4)
                srcip = ip.src
                dstip = ip.dst
                protocol = ip.proto
                # If TCP Protocol
                if protocol == in_proto.IPPROTO_TCP:
                    isTCP = True
                    tu = pkt.get_protocol(tcp.tcp)

            #Rule 1: H1-H2
            if (src == H1 and out_port == H2_PORT) or (src == H2 and out_port == H1_PORT) :

                #self.logger.info("H1 - H2")
                actions = [parser.OFPActionOutput(out_port)]

            #Rule 2: H1-H3:80
            elif isTCP and ((src == H1 and out_port == H3_PORT and tu.dst_port==H3_TCP_PORT)
                or (src == H3 and out_port == H1_PORT and tu.src_port==H3_TCP_PORT)) :

                #self.logger.info("H1 - H3:80")
                isRule2 = True
                actions = [parser.OFPActionOutput(out_port)]

            else: #Rule 0: default Deny
                self.logger.info("DENIED")
                return
        else:
            actions = [parser.OFPActionOutput(out_port)]
```

Listing 2.2: Comunicazione tra H1 ed H3

```
match = parser.OFPMatch(eth_type=ether_types.ETH_TYPE_IP, ipv4_src=srcip,
    ipv4_dst=dstip, ip_proto=protocol, tcp_dst=t.dst_port)
```

# Capitolo 3

# Deployment ed esecuzione

## 3.1 Step per l'esecuzione

Di seguito illustreremo gli step per eseguire l'ambiente:

1. Clonare **il Repository** nella directory "ryu/ryu/app"
2. Aprire due terminali all'interno del repository clonato

   ```
   cd ryu/ryu/app/sdn_ryu_firewall
   ```

3. Far partire il controller nel primo dei due terminali con il comando:

   ```
   ryu-manager firewall_simple_switch_13.py
   ```

4. Occorre ora controllare l'ip del Firewall lanciando Wireshark con privilegi amministrativi ed aggiornare il file "renew.sh" con l'ip appena visualizzato
5. Nel secondo terminale lanciare ora lo script per pulire eventuali precedenti sdn e lanciarne una nuova:

   ```
   sudo ./renew.sh
   ```

   L'output atteso dovrebbe assomigliare al seguente:

```
1  ovs-vsctl: no bridge named LAN1
2  net.ipv6.conf.all.disable_ipv6 = 1
3  net.ipv6.conf.default.disable_ipv6 = 1
4  net.ipv6.conf.lo.disable_ipv6 = 1
5  Host H1 done
6  net.ipv6.conf.all.disable_ipv6 = 1
7  net.ipv6.conf.default.disable_ipv6 = 1
8  net.ipv6.conf.lo.disable_ipv6 = 1
9  Host H2 done
10 net.ipv6.conf.all.disable_ipv6 = 1
11 net.ipv6.conf.default.disable_ipv6 = 1
12 net.ipv6.conf.lo.disable_ipv6 = 1
13 Host H3 done
14 eth-H1
15 eth-H2
16 eth-H3
17 cookie=0x0, duration=0.029s, table=0, n_packets=0, n_bytes=0, priority=0 actions=CONTROLLER:65535
```

6. Ora per verificare lo stato dello switch possiamo utilizzare il seguente comando:

   ```
   sudo ovs-vsctl show
   ```

l'output atteso dovrebbe essere:

```
eef462c4-a4bf-4893-9724-d27d18c02cbf
    Bridge LAN1
        Controller "tcp:10.201.107.109:6633"
            is_connected: true
        Port LAN1
            Interface LAN1
                type: internal
        Port eth-H1
            Interface eth-H1
        Port eth-H2
            Interface eth-H2
        Port eth-H3
            Interface eth-H3
    ovs_version: "2.17.3"
```

# Capitolo 4

# Testing

## 4.1 Procedura per il testing

Per testare il comportamento della rete si possono utilizzare i seguenti comandi:

```
1 sudo ip netns exec H1 ping -c 3 192.168.1.2
2 sudo ip netns exec H2 ping -c 3 192.168.1.1
3
4 sudo ip netns exec H1 nping --tcp -p 80 192.168.1.3
5 sudo ip netns exec H3 nping --tcp -g 80 192.168.1.1
```

Di conseguenza con il comando:

```
sudo ovs-ofctl dump-flows LAN1
```

possiamo impostare le regole nello switch come da assignment:

```
cookie=0x0, duration=16.106s, table=0, n_packets=6, n_bytes=532, priority=1,in_port=
    "eth-H1",dl_src=00:00:00:11:11:11,dl_dst=00:00:00:12:12:12 actions=output:"eth-
    H2"
cookie=0x0, duration=16.104s, table=0, n_packets=6, n_bytes=532, priority=1,in_port=
    "eth-H2",dl_src=00:00:00:12:12:12,dl_dst=00:00:00:11:11:11 actions=output:"eth-
    H1"
cookie=0x0, duration=11.952s, table=0, n_packets=9, n_bytes=486, priority=1,tcp,
    nw_src=192.168.1.1,nw_dst=192.168.1.3,tp_dst=80 actions=output:"eth-H3"
cookie=0x0, duration=11.950s, table=0, n_packets=9, n_bytes=486, priority=1,tcp,
    nw_src=192.168.1.3,nw_dst=192.168.1.1,tp_src=80 actions=output:"eth-H1"
cookie=0x0, duration=125.172s, table=0, n_packets=15, n_bytes=934, priority=0
    actions=CONTROLLER:65535
```

Se altri pacchetti vengono inviati all'infuori di questi percorsi consentiti, verranno eliminati dal controller e non ci sarà alcuna comunicazione seguendo il comportamento di Deny predefinito, esempi sono:

```
1 sudo ip netns exec H1 ping -c 3 192.168.1.3
2 sudo ip netns exec H2 ping -c 3 192.168.1.3
3
4 sudo ip netns exec H1 nping --tcp -p 8080 192.168.1.3
5 sudo ip netns exec H3 nping --tcp -g 80 192.168.1.2
```