USIU AFRICA

United States International University-Africa

Education to take you places

Name: Enid Shilwatso

Student ID: 671055

Lab4Instructor: Professor Dennis Kaburu

Assignment: Lab 5.3.7 - Introduction to wireshark

Title: Advanced Information System Security

Course: ISC6120

Due: Summer 2025

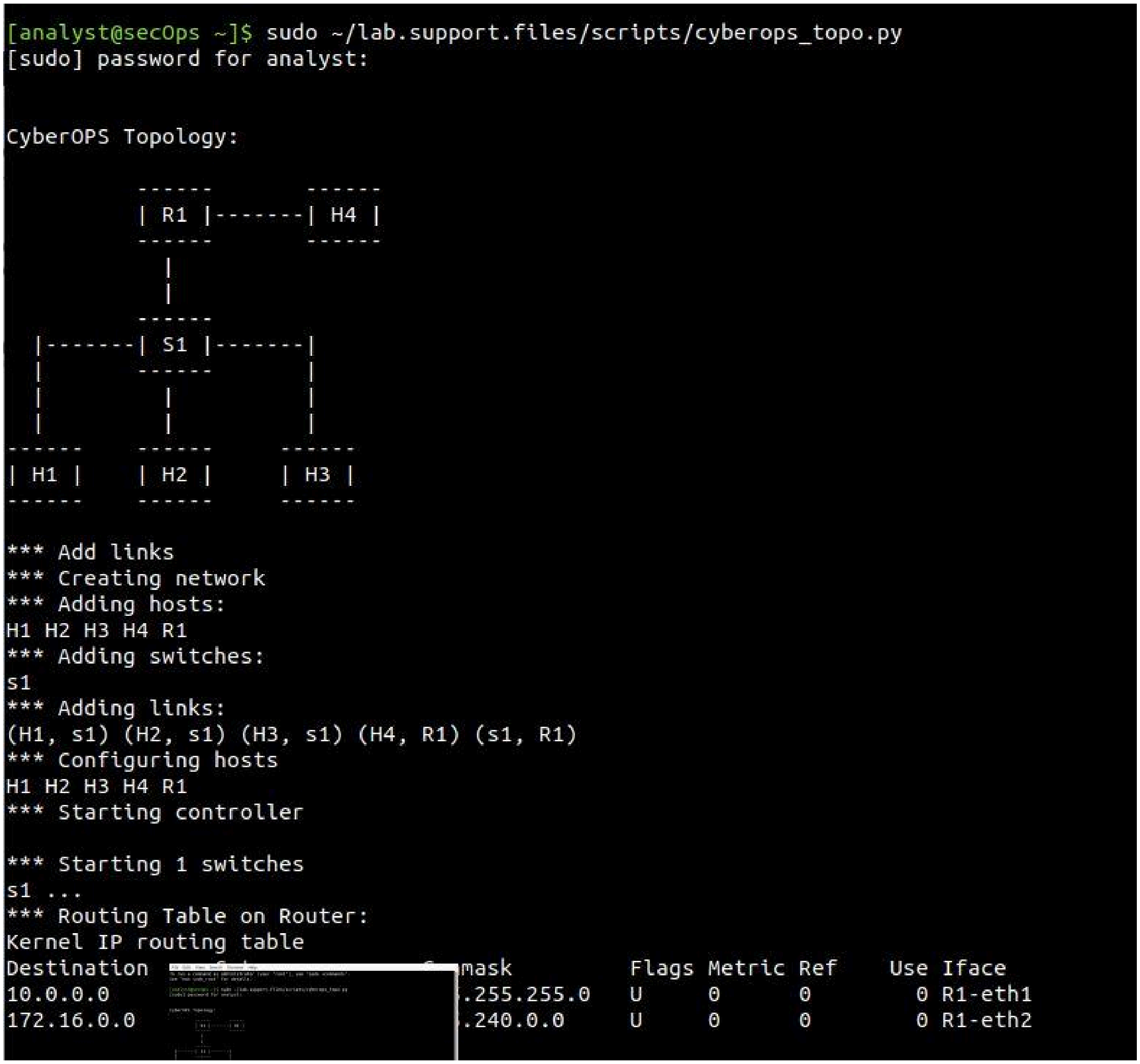
## Part 1: Install and Verify the Mininet Topology

In this part, you will use a Python script to set up the Mininet Topology inside the CyberOps VM. You will then record the IP and MAC addresses for H1 and H2.

a. Start and log into your CyberOps Workstation that you have installed

b. Run the Python script to install the Mininet Topology

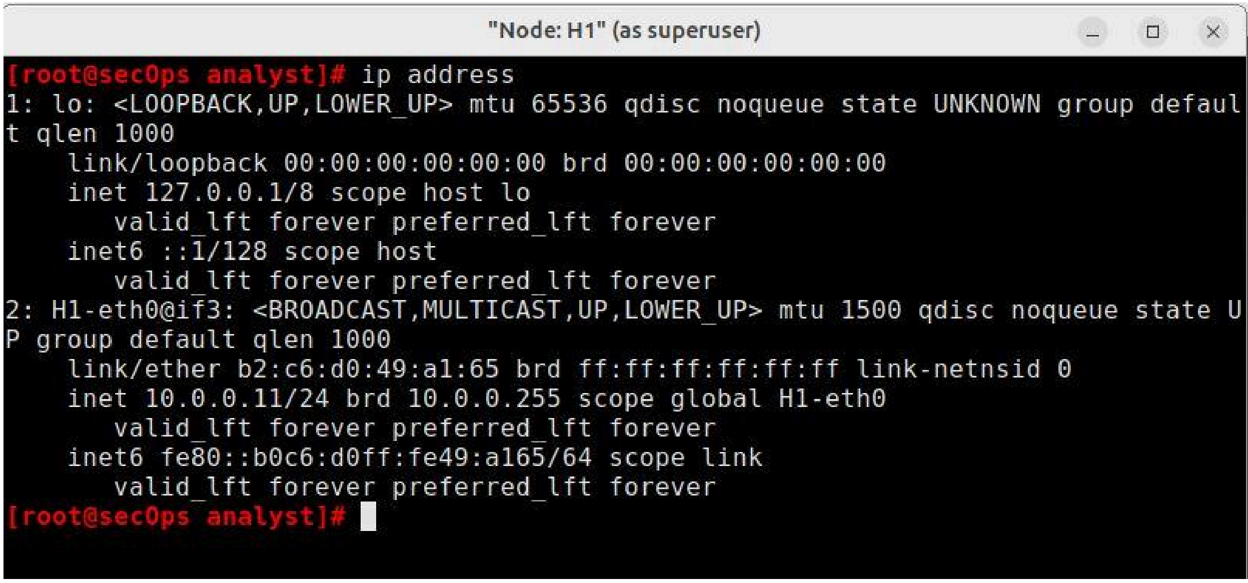
   - Open a terminal emulator to start Mininet and enter the following command at the prompt. When prompted, enter cyberops as the password.
   - [analyst@secOps ~]$ sudo ~/lab.support.files/scripts/cyberops_topo.py



```
[analyst@secOps ~]$ sudo ~/lab.support.files/scripts/cyberops_topo.py
[sudo] password for analyst:


CyberOPS Topology:

          ------       ------
          | R1 |-------| H4 |
          ------       ------
            |
            |
          ------
  |-------| S1 |-------|
  |       ------       |
  |         |          |
  |         |          |
------    ------     ------
| H1 |    | H2 |     | H3 |
------    ------     ------

*** Add links
*** Creating network
*** Adding hosts:
H1 H2 H3 H4 R1
*** Adding switches:
s1
*** Adding links:
(H1, s1) (H2, s1) (H3, s1) (H4, R1) (s1, R1)
*** Configuring hosts
H1 H2 H3 H4 R1
*** Starting controller

*** Starting 1 switches
s1 ...
*** Routing Table on Router:
Kernel IP routing table
Destination                        mask         Flags Metric Ref    Use Iface
10.0.0.0                           .255.255.0   U     0      0        0 R1-eth1
172.16.0.0                         .240.0.0     U     0      0        0 R1-eth2
```

c. Record IP and MAC addresses for H1 and H2

- At the mininet prompt, start terminal windows on hosts H1 and H2. This will open separate windows for these hosts. Each host will have a separate configuration for the network including unique IP and MAC addresses.
- Starting CLI: mininet> xterm H1 mininet> xterm H2
- At the prompt on Node: H1, enter ip address to verify the IPv4 address and record the MAC address. Do the same for Node: H2. The IPv4 address and MAC address are highlighted below for reference.
- [root@secOps analyst]# ip address



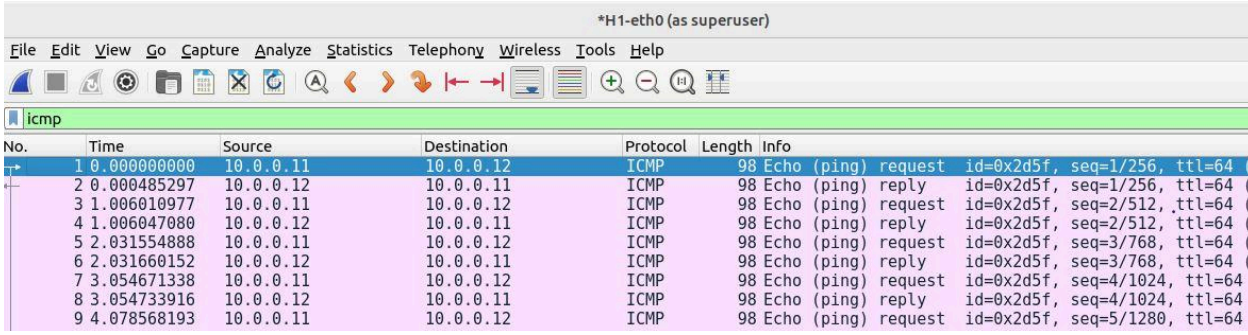
```
[root@secOps analyst]# ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: H1-eth0@if3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state U
P group default qlen 1000
    link/ether b2:c6:d0:49:a1:65 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.0.0.11/24 brd 10.0.0.255 scope global H1-eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::b0c6:d0ff:fe49:a165/64 scope link
       valid_lft forever preferred_lft forever
[root@secOps analyst]#
```

**Part 2: Capture and Analyze ICMP Data in Wireshark**

In this part, you will ping between two hosts in the Mininet and capture ICMP requests and replies in Wireshark.

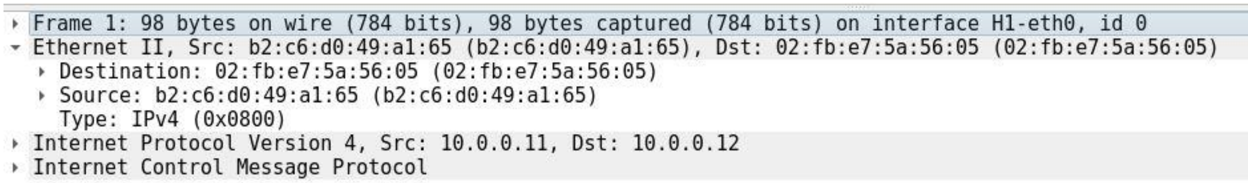
a. Examine the captured data on the same LAN.In this step, you will examine the data that was generated by the ping requests of your team member’s PC.
    - On Node: H1, enter wireshark & to start Wireshark (The pop-up warning is not important for this lab.). Click OK to continue.
    - In the Wireshark window, under the Capture heading, select the H1-eth0 interface. Click Start to capture the data traffic.

- On Node: H1, press the Enter key, if necessary, to get a prompt. Then type ping -c 5 10.0.0.12 to ping H2 five times. The command option -c specifies the count or number of pings. The 5 specifies that five pings should be sent. The pings will all be successful.
- Navigate to the Wireshark window, click Stop to stop the packet capture.
- A filter can be applied to display only the interested traffic. Type icmp in the Filter field and click Apply.
- If necessary, click the first ICMP request PDU frames in the top section of Wireshark. Notice that the Source column has H1's IP address, and the Destination column has H2's IP address.


*H1-eth0 (as superuser)
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
icmp

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 1 | 0.000000000 | 10.0.0.11 | 10.0.0.12 | ICMP | 98 | Echo (ping) request id=0x2d5f, seq=1/256, ttl=64 |
| 2 | 0.000485297 | 10.0.0.12 | 10.0.0.11 | ICMP | 98 | Echo (ping) reply id=0x2d5f, seq=1/256, ttl=64 |
| 3 | 1.006010977 | 10.0.0.11 | 10.0.0.12 | ICMP | 98 | Echo (ping) request id=0x2d5f, seq=2/512, ttl=64 |
| 4 | 1.006047080 | 10.0.0.12 | 10.0.0.11 | ICMP | 98 | Echo (ping) reply id=0x2d5f, seq=2/512, ttl=64 |
| 5 | 2.031554888 | 10.0.0.11 | 10.0.0.12 | ICMP | 98 | Echo (ping) request id=0x2d5f, seq=3/768, ttl=64 |
| 6 | 2.031660152 | 10.0.0.12 | 10.0.0.11 | ICMP | 98 | Echo (ping) reply id=0x2d5f, seq=3/768, ttl=64 |
| 7 | 3.054671338 | 10.0.0.11 | 10.0.0.12 | ICMP | 98 | Echo (ping) request id=0x2d5f, seq=4/1024, ttl=64 |
| 8 | 3.054733916 | 10.0.0.12 | 10.0.0.11 | ICMP | 98 | Echo (ping) reply id=0x2d5f, seq=4/1024, ttl=64 |
| 9 | 4.078568193 | 10.0.0.11 | 10.0.0.12 | ICMP | 98 | Echo (ping) request id=0x2d5f, seq=5/1280, ttl=64 |

- With this PDU frame still selected in the top section, navigate to the middle section. Click the arrow to the left of the Ethernet II row to view the Destination and Source MAC addresses.

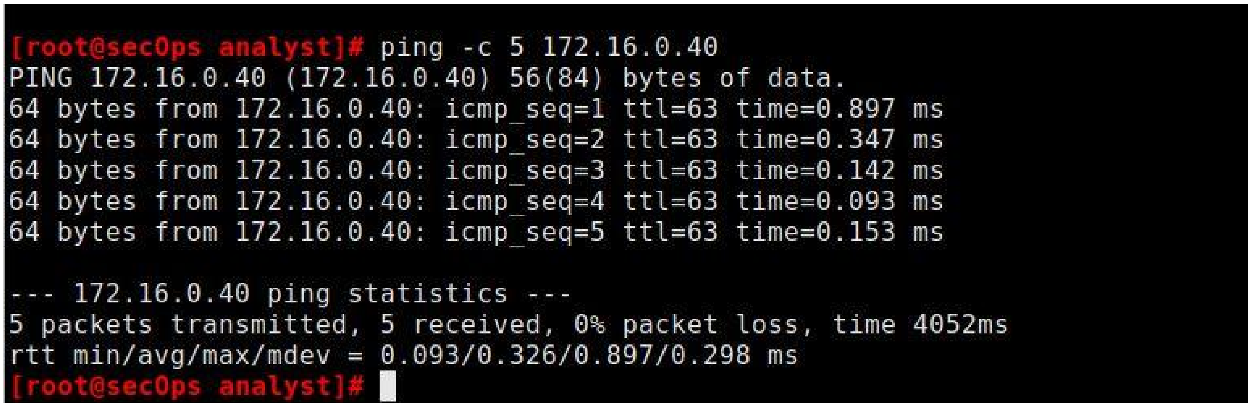


```
▸ Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface H1-eth0, id 0
▾ Ethernet II, Src: b2:c6:d0:49:a1:65 (b2:c6:d0:49:a1:65), Dst: 02:fb:e7:5a:56:05 (02:fb:e7:5a:56:05)
  ▸ Destination: 02:fb:e7:5a:56:05 (02:fb:e7:5a:56:05)
  ▸ Source: b2:c6:d0:49:a1:65 (b2:c6:d0:49:a1:65)
    Type: IPv4 (0x0800)
▸ Internet Protocol Version 4, Src: 10.0.0.11, Dst: 10.0.0.12
▸ Internet Control Message Protocol
```

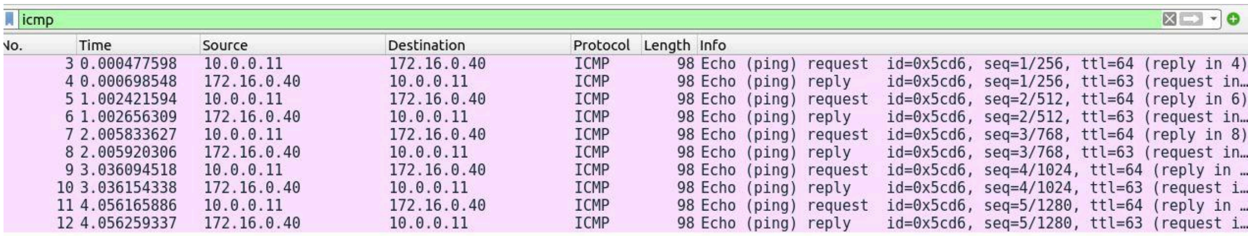
b. Examine the captured data on the remote LAN.

- ping remote hosts (hosts not on the LAN) and examine the generated data from those pings.
- At the mininet prompt, start terminal windows on hosts H4 and R1.
- At the prompt on Node: H4, enter ip address to verify the IPv4 address and record the MAC address. Do the same for the Node: R1.

- Start a new Wireshark capture on H1 by selecting Capture > Start. You can also click the Start button or type Ctrl-E Click Continue without Saving to start a new capture.
- H4 is a simulated remote server. Ping H4 from H1. The ping should be successful. [root@secOps analyst]# ping -c 5 172.16.0.40



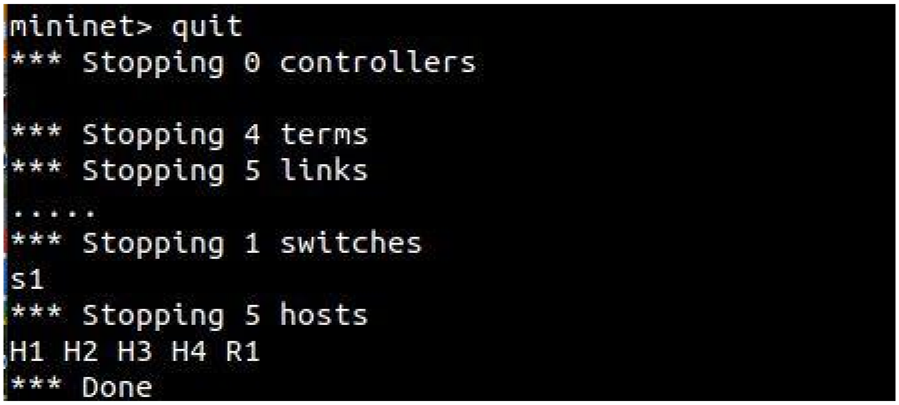
```
[root@secOps analyst]# ping -c 5 172.16.0.40
PING 172.16.0.40 (172.16.0.40) 56(84) bytes of data.
64 bytes from 172.16.0.40: icmp_seq=1 ttl=63 time=0.897 ms
64 bytes from 172.16.0.40: icmp_seq=2 ttl=63 time=0.347 ms
64 bytes from 172.16.0.40: icmp_seq=3 ttl=63 time=0.142 ms
64 bytes from 172.16.0.40: icmp_seq=4 ttl=63 time=0.093 ms
64 bytes from 172.16.0.40: icmp_seq=5 ttl=63 time=0.153 ms

--- 172.16.0.40 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4052ms
rtt min/avg/max/mdev = 0.093/0.326/0.897/0.298 ms
[root@secOps analyst]#
```



icmp

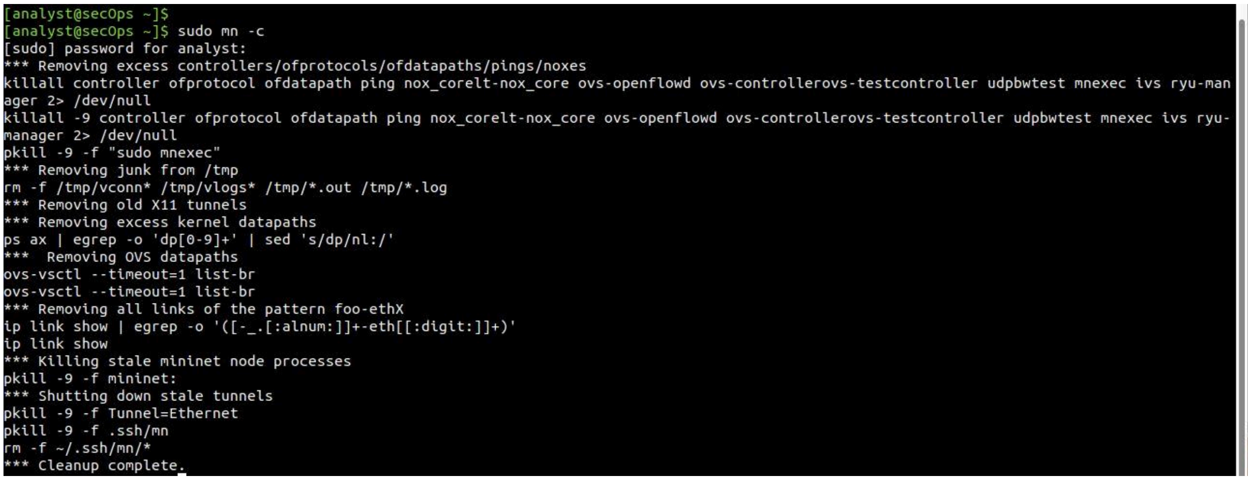
| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 3 | 0.000477598 | 10.0.0.11 | 172.16.0.40 | ICMP | 98 | Echo (ping) request  id=0x5cd6, seq=1/256, ttl=64 (reply in 4) |
| 4 | 0.000698548 | 172.16.0.40 | 10.0.0.11 | ICMP | 98 | Echo (ping) reply    id=0x5cd6, seq=1/256, ttl=63 (request in… |
| 5 | 1.002421594 | 10.0.0.11 | 172.16.0.40 | ICMP | 98 | Echo (ping) request  id=0x5cd6, seq=2/512, ttl=64 (reply in 6) |
| 6 | 1.002656309 | 172.16.0.40 | 10.0.0.11 | ICMP | 98 | Echo (ping) reply    id=0x5cd6, seq=2/512, ttl=63 (request in… |
| 7 | 2.005833627 | 10.0.0.11 | 172.16.0.40 | ICMP | 98 | Echo (ping) request  id=0x5cd6, seq=3/768, ttl=64 (reply in 8) |
| 8 | 2.005920306 | 172.16.0.40 | 10.0.0.11 | ICMP | 98 | Echo (ping) reply    id=0x5cd6, seq=3/768, ttl=63 (request in… |
| 9 | 3.036094518 | 10.0.0.11 | 172.16.0.40 | ICMP | 98 | Echo (ping) request  id=0x5cd6, seq=4/1024, ttl=64 (reply in … |
| 10 | 3.036154338 | 172.16.0.40 | 10.0.0.11 | ICMP | 98 | Echo (ping) reply    id=0x5cd6, seq=4/1024, ttl=63 (request i… |
| 11 | 4.056165886 | 10.0.0.11 | 172.16.0.40 | ICMP | 98 | Echo (ping) request  id=0x5cd6, seq=5/1280, ttl=64 (reply in … |
| 12 | 4.056259337 | 172.16.0.40 | 10.0.0.11 | ICMP | 98 | Echo (ping) reply    id=0x5cd6, seq=5/1280, ttl=63 (request i… |

- In the main CyberOps VM window, enter quit to stop Mininet.



```
mininet> quit
*** Stopping 0 controllers

*** Stopping 4 terms
*** Stopping 5 links
.....
*** Stopping 1 switches
s1
*** Stopping 5 hosts
H1 H2 H3 H4 R1
*** Done
```

- To clean up all the processes that were used by Mininet, enter the sudo mn -c command at the prompt. analyst@secOps ~]$ sudo mn -c



```
[analyst@secOps ~]$
[analyst@secOps ~]$ sudo mn -c
[sudo] password for analyst:
*** Removing excess controllers/ofprotocols/ofdatapaths/pings/noxes
killall controller ofprotocol ofdatapath ping nox_corelt-nox_core ovs-openflowd ovs-controllerovs-testcontroller udpbwtest mnexec ivs ryu-man
ager 2> /dev/null
killall -9 controller ofprotocol ofdatapath ping nox_corelt-nox_core ovs-openflowd ovs-controllerovs-testcontroller udpbwtest mnexec ivs ryu-
manager 2> /dev/null
pkill -9 -f "sudo mnexec"
*** Removing junk from /tmp
rm -f /tmp/vconn* /tmp/vlogs* /tmp/*.out /tmp/*.log
*** Removing old X11 tunnels
*** Removing excess kernel datapaths
ps ax | egrep -o 'dp[0-9]+' | sed 's/dp/nl:/'
***  Removing OVS datapaths
ovs-vsctl --timeout=1 list-br
ovs-vsctl --timeout=1 list-br
*** Removing all links of the pattern foo-ethX
ip link show | egrep -o '([-_.[:alnum:]]+-eth[[:digit:]]+)'
ip link show
*** Killing stale mininet node processes
pkill -9 -f mininet:
*** Shutting down stale tunnels
pkill -9 -f Tunnel=Ethernet
pkill -9 -f .ssh/mn
rm -f ~/.ssh/mn/*
*** Cleanup complete.
```