Security+ Lab Series

Lab 17: Capturing Network Traffic

Document Version: 2020-12-10

Contents

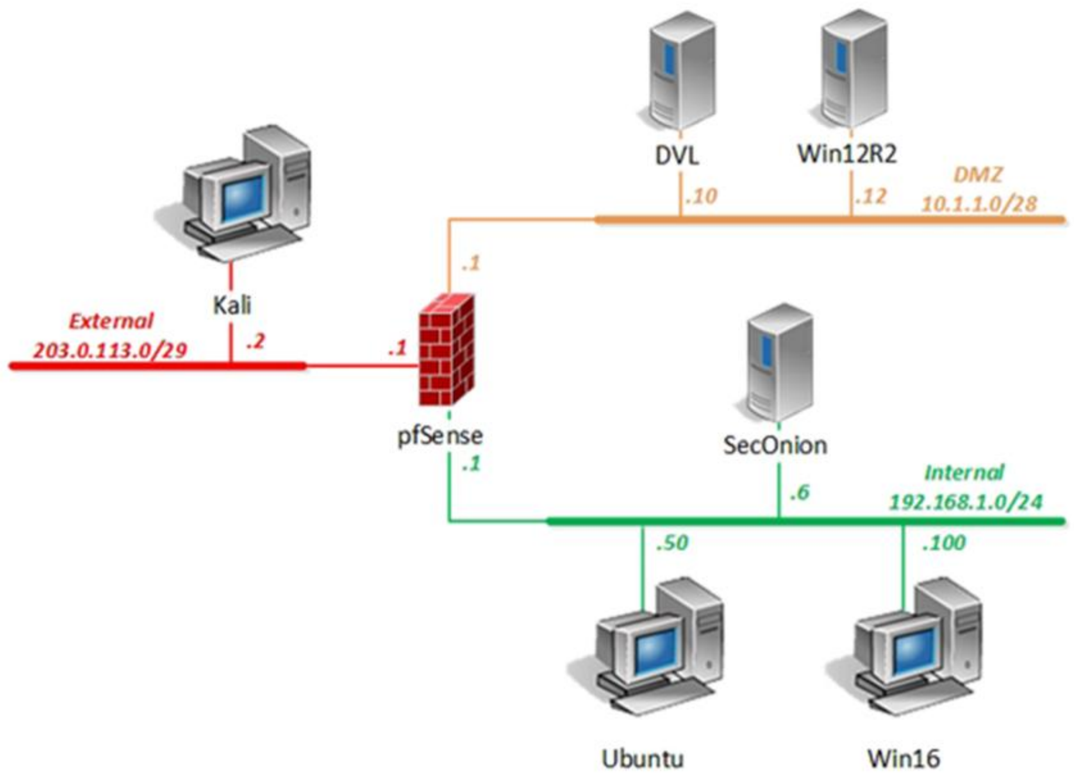
Introduction

In this lab, you will be conducting network security practices using various tools.

Objectives

- Given a scenario, appropriate software tools to assess the security posture of an organization

Lab Topology



Lab Settings

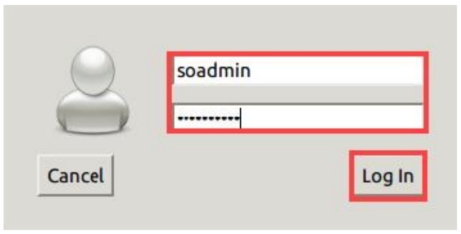
The information in the table below will be needed to complete the lab. The task sections below provide details on the use of this information.

Virtual Machine	IP Address	Account	Password
DVL	10.1.1.10 /28	root	toor
Kali	203.0.113.2 /29	root	toor
pfSense	eth0: 192.168.1.1 /24 eth1: 10.1.1.1 /28 eth2: 203.0.113.1 /29	admin	pfsense
SecOnion	eth0: 192.168.1.6 /24	soadmin	mypassword
		root	mypassword
Ubuntu	192.168.1.50 /24	student	securepassword
		root	securepassword
Win12R2	10.1.1.12 /28	administrator	Train1ng$
Win16	192.168.1.100 /24	lab-user	Train1ng$
		Administrator	Train1ng$

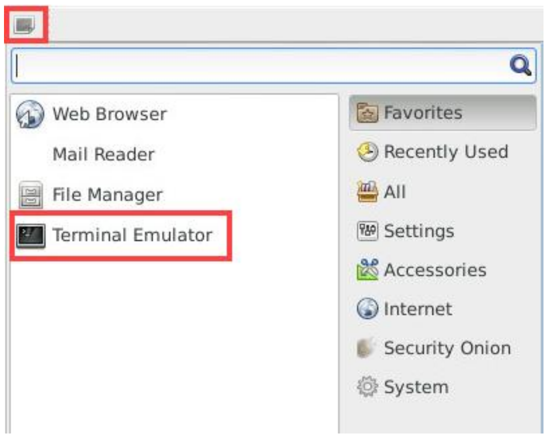
1 Using tcpdump to Capture and Analyze Network Traffic

1.1 Using tcpdump to Capture ICMP Traffic

1. Launch the **SecOnion** virtual machine.
2. On the login screen, type `soadmin` as the username and `mypassword` as the password. Click **Log In**.



3. Once logged in, click the start button followed by clicking on **Terminal Emulator** to launch a new *terminal*.



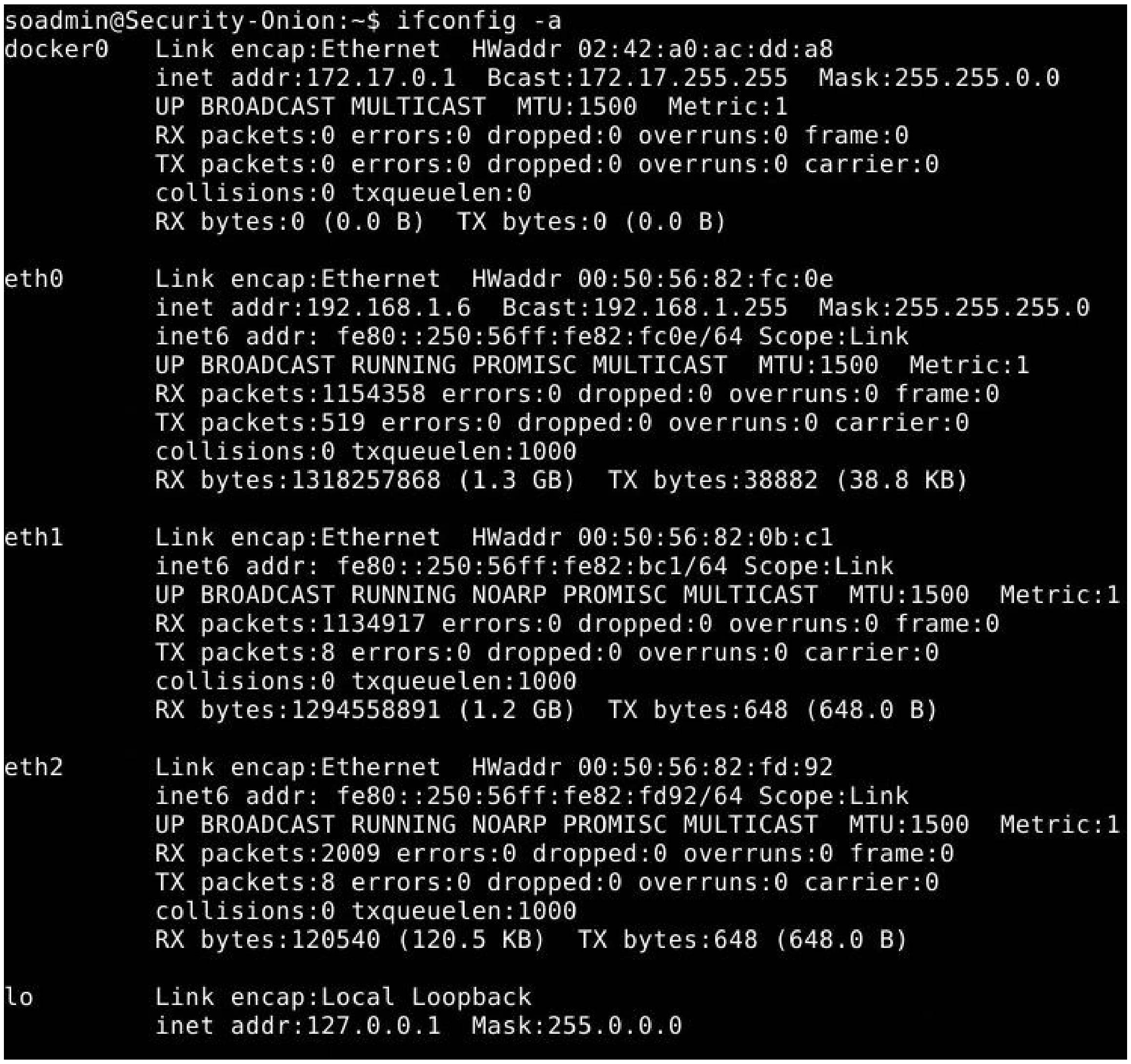
4. Type the command below followed by pressing the **Enter** key. If prompted, enter `mypassword` for root privileges.

```
soadmin@Security-Onion:~$ sudo service nsm status
```

If *nsm status* reports back with all modules as *OK*, proceed to the next step. If not, then initiate the *service nsm start/restart* command.

5. Type the command below to view all available interfaces on the system.

```
soadmin@Security-Onion:~$ ifconfig -a
```

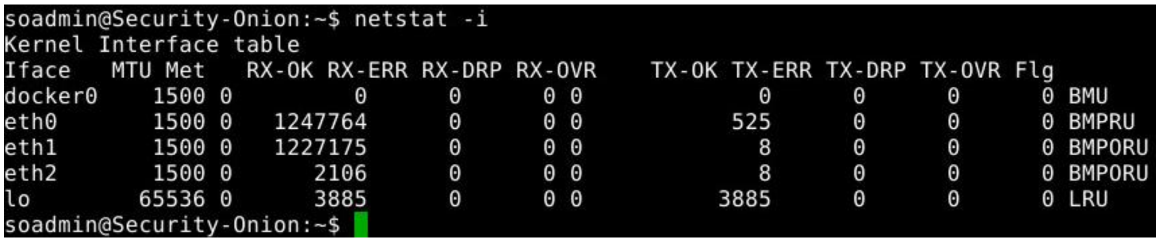


```
soadmin@Security-Onion:~$ ifconfig -a
docker0   Link encap:Ethernet  HWaddr 02:42:a0:ac:dd:a8
          inet addr:172.17.0.1  Bcast:172.17.255.255  Mask:255.255.0.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

eth0      Link encap:Ethernet  HWaddr 00:50:56:82:fc:0e
          inet addr:192.168.1.6  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::250:56ff:fe82:fc0e/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:1154358 errors:0 dropped:0 overruns:0 frame:0
          TX packets:519 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1318257868 (1.3 GB)  TX bytes:38882 (38.8 KB)

eth1      Link encap:Ethernet  HWaddr 00:50:56:82:0b:c1
          inet6 addr: fe80::250:56ff:fe82:bc1/64 Scope:Link
          UP BROADCAST RUNNING NOARP PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:1134917 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1294558891 (1.2 GB)  TX bytes:648 (648.0 B)

eth2      Link encap:Ethernet  HWaddr 00:50:56:82:fd:92
          inet6 addr: fe80::250:56ff:fe82:fd92/64 Scope:Link
          UP BROADCAST RUNNING NOARP PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:2009 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:120540 (120.5 KB)  TX bytes:648 (648.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
```

The *SecOnion* system has three interfaces, each assigned to a different network.

6. Issue the command below to identify which flags are configured for each interface:

```
soadmin@Security-Onion:~$ netstat -i
```

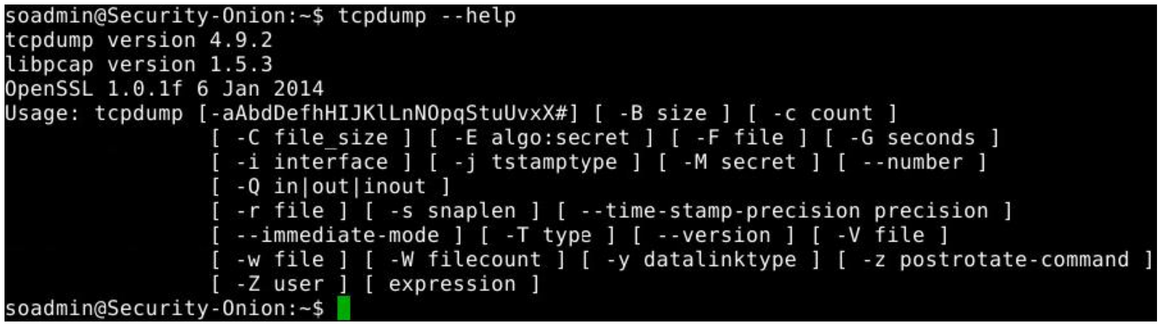


```
soadmin@Security-Onion:~$ netstat -i
Kernel Interface table
Iface   MTU Met   RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
docker0    1500 0       0      0      0 0           0      0      0      0 BMU
eth0       1500 0  1247764      0      0 0         525      0      0      0 BMPRU
eth1       1500 0  1227175      0      0 0           8      0      0      0 BMPORU
eth2       1500 0     2106      0      0 0           8      0      0      0 BMPORU
lo        65536 0     3885      0      0 0        3885      0      0      0 LRU
soadmin@Security-Onion:~$
```

Notice how *BMPRU* is set for the *eth0* physical interface under the *Flg* column. Notice that *BMPORU* is set for both *eth1* and *eth2*. For a quick overview: *B* flag is for broadcast, *M* flag is for multicast, *P* flag is for promiscuous mode, *O* flag is for no *ARP* (*Address Resolution Protocol*) requests, *R* flag is for running and *U* flag is for up. Also, notice that *LRU* is set for *lo*; the *L* flag means that the specified interface is a loopback device.

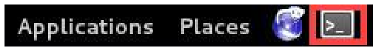
7. To familiarize yourself with the *tcpdump* utility, type the following command to view several available options for *tcpdump*:

```
soadmin@Security-Onion:~$ tcpdump --help
```



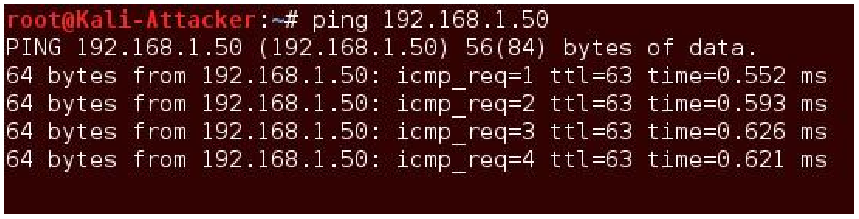
```
soadmin@Security-Onion:~$ tcpdump --help
tcpdump version 4.9.2
libpcap version 1.5.3
OpenSSL 1.0.1f 6 Jan 2014
Usage: tcpdump [-aAbdDefhHIJKlLnNOpqStuUvxX#] [ -B size ] [ -c count ]
                [ -C file_size ] [ -E algo:secret ] [ -F file ] [ -G seconds ]
                [ -i interface ] [ -j tstamptype ] [ -M secret ] [ --number ]
                [ -Q in|out|inout ]
                [ -r file ] [ -s snaplen ] [ --time-stamp-precision precision ]
                [ --immediate-mode ] [ -T type ] [ --version ] [ -V file ]
                [ -w file ] [ -W filecount ] [ -y datalinktype ] [ -z postrotate-command ]
                [ -Z user ] [ expression ]
soadmin@Security-Onion:~$
```

8. Launch the **Kali** virtual machine to access the graphical login screen.
9. Log in as `root` with `toor` as the password.
10. Click on the **terminal** icon located in the top menu bar.



11. Type the following command to initiate a continuous ping to the *Ubuntu* system. Leave the pings running in the background and proceed to the next step.

```
root@Kali-Attacker:~# ping 192.168.1.50
```

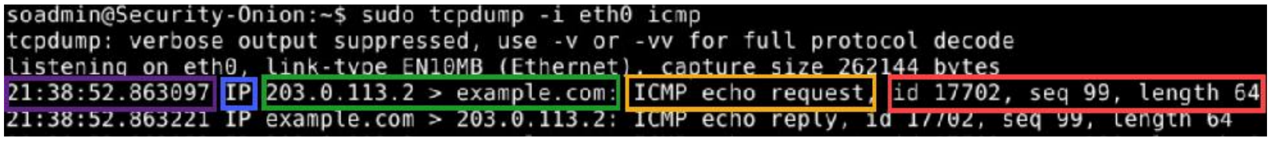


```
root@Kali-Attacker:~# ping 192.168.1.50
PING 192.168.1.50 (192.168.1.50) 56(84) bytes of data.
64 bytes from 192.168.1.50: icmp_req=1 ttl=63 time=0.552 ms
64 bytes from 192.168.1.50: icmp_req=2 ttl=63 time=0.593 ms
64 bytes from 192.168.1.50: icmp_req=3 ttl=63 time=0.626 ms
64 bytes from 192.168.1.50: icmp_req=4 ttl=63 time=0.621 ms
```

12. Switch back to the **SecOnion** system. In a terminal, run **tcpdump** on the *internal network* by entering the command below. If prompted with a password, enter `mypassword`.

```
soadmin@Security-Onion:~$ sudo tcpdump –i eth0 icmp
```

13. Notice the output that tcpdump provides: *HH:MM:SS.mmmmmm IP src > dst: ptype, id, seq, len*. Also, take note that for each echo request, there is a reply.

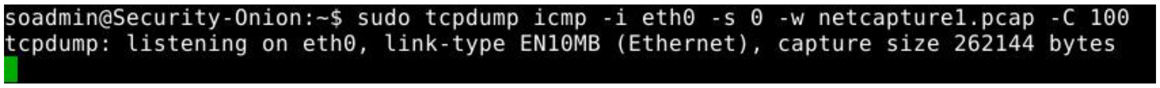


```
soadmin@Security-Onion:~$ sudo tcpdump -i eth0 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
21:38:52.863097 IP 203.0.113.2 > example.com: ICMP echo request, id 17702, seq 99, length 64
21:38:52.863221 IP example.com > 203.0.113.2: ICMP echo reply, id 17702, seq 99, length 64
```

HH:MM:SS.mmmmmm	Timestamp in hours, minutes, seconds and microseconds
IP	Internet Protocol
src > dst	Source and destination IP addresses
ptype	Packet type
id, seq, len	IP headers; identification, protocol (1=ICMP), total length

14. After a minute, press **CTRL+C** to stop *tcpdump* from running and discontinue the network capture.
15. From an administrator's standpoint, we may want to save the output from a *tcpdump capture* and save it automatically into a compatible file to view later with a program such as *Wireshark*. Initiate the command below to capture traffic on the **192.168.1.0/24** network and sending it to a file. If prompted with a password, enter `mypassword`.

```
soadmin@Security-Onion:~$ sudo tcpdump icmp –i eth0 –s 0 –w netcapture1.pcap –C
100
```

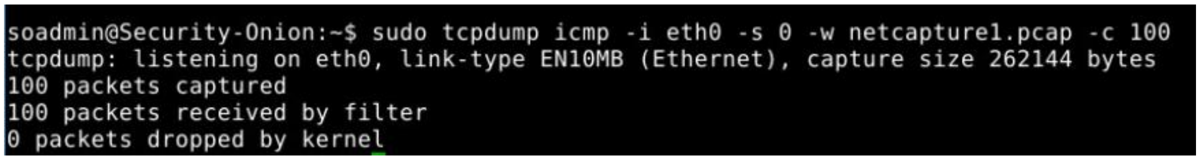


```
soadmin@Security-Onion:~$ sudo tcpdump icmp -i eth0 -s 0 -w netcapture1.pcap -C 100
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

The following table lists details of the options used with the *tcpdump* command:

icmp	Captures only ICMP packets (works for tcp, udp and icmp)
-i eth0	Use interface zero
-s 0	Disables default packet size, date and time format
-w	Write to a capture file, instead of displaying to the screen
-C	Split the captures into files of this size

16. Wait for about 1-2 minutes until all 100 packets are captured.

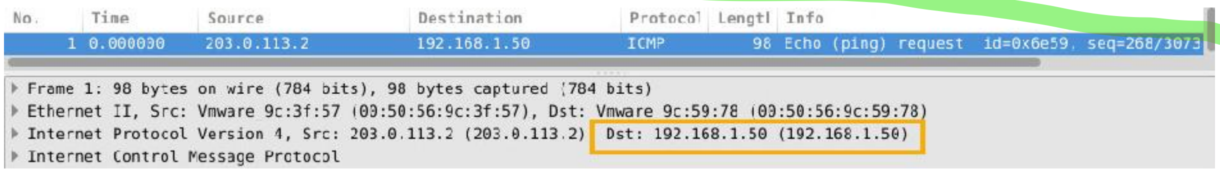


```
soadmin@Security-Onion:~$ sudo tcpdump icmp -i eth0 -s 0 -w netcapture1.pcap -c 100
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
100 packets captured
100 packets received by filter
0 packets dropped by kernel
```

17. To view the captured file in a graphical user interface like *Wireshark*, enter the command below in the *SecOnion terminal*. If prompted with a password, enter **`mypassword`**.

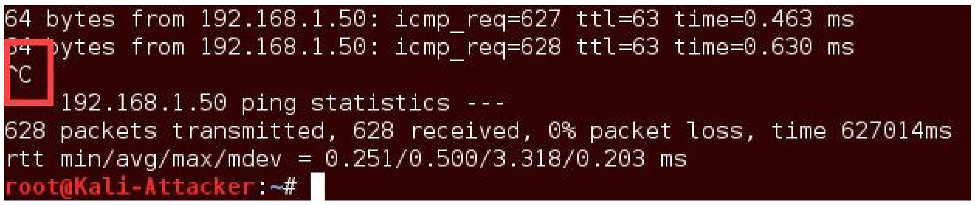
```
soadmin@Security-Onion:~$ sudo wireshark netcapture1.pcap
```

18. If prompted with a warning message, click **OK** to continue.
19. If another warning message appears, click **OK** again.
20. Notice the traffic listed that takes place on the *192.168.1.0/24* network.



```
No.   Time        Source        Destination    Protocol Lengtl Info
    1 0.000000    203.0.113.2   192.168.1.50   ICMP         98 Echo (ping) request  id=0x6e59, seq=268/3073

▸ Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits)
▸ Ethernet II, Src: Vmware_9c:3f:57 (00:50:56:9c:3f:57), Dst: Vmware_9c:59:78 (00:50:56:9c:59:78)
▸ Internet Protocol Version 4, Src: 203.0.113.2 (203.0.113.2) Dst: 192.168.1.50 (192.168.1.50)
▸ Internet Control Message Protocol
```

21. Close **Wireshark**.
22. Switch to the **Kali** machine and press **CTRL+C** to stop the continuous pings.

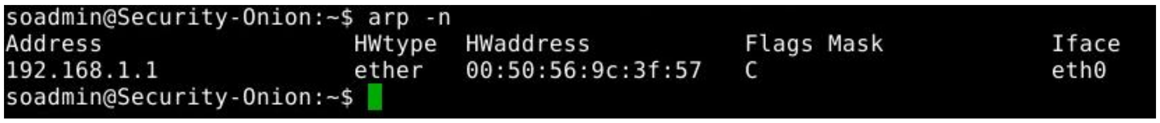


```
64 bytes from 192.168.1.50: icmp_req=627 ttl=63 time=0.463 ms
64 bytes from 192.168.1.50: icmp_req=628 ttl=63 time=0.630 ms
^C
    192.168.1.50 ping statistics ---
628 packets transmitted, 628 received, 0% packet loss, time 627014ms
rtt min/avg/max/mdev = 0.251/0.500/3.318/0.203 ms
root@Kali-Attacker:~#
```

1.2 Using tcpdump to Capture ARP Traffic

1. Change focus to the **SecOnion** system.
2. In a terminal window, enter the *ARP* command below and take note of the results.

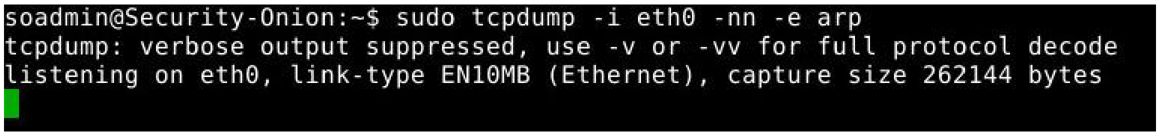
```
soadmin@Security-Onion:~$ arp -n
```



```
soadmin@Security-Onion:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.1.1              ether   00:50:56:9c:3f:57   C                     eth0
soadmin@Security-Onion:~$
```

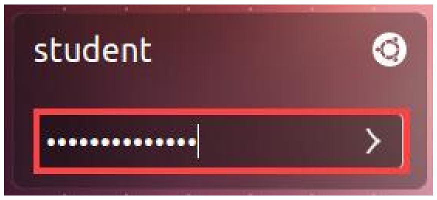
3. Enter the command below to capture *ARP* packets. If prompted with a password, enter `mypassword`.

```
soadmin@Security-Onion:~$ sudo tcpdump –i eth0 –nn –e arp
```

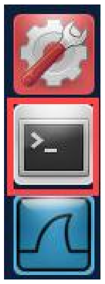


```
soadmin@Security-Onion:~$ sudo tcpdump -i eth0 -nn -e arp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

4. Launch the **Ubuntu** virtual machine to access the graphical login screen.
5. Log in as `student` with `securepassword` as the password.

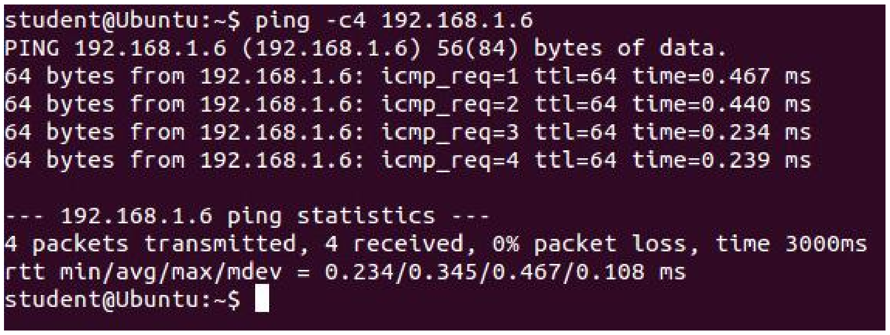


6. Open a terminal window by clicking on the **terminal** icon located in the left menu pane.



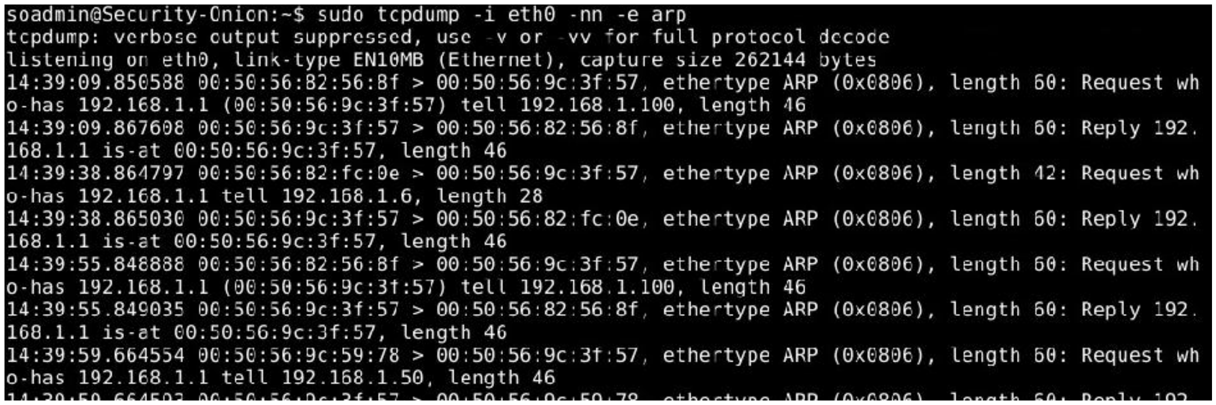
7. Type the *ping* command below:

```
student@Ubuntu:~$ ping –c4 192.168.1.6
```



```
student@Ubuntu:~$ ping -c4 192.168.1.6
PING 192.168.1.6 (192.168.1.6) 56(84) bytes of data.
64 bytes from 192.168.1.6: icmp_req=1 ttl=64 time=0.467 ms
64 bytes from 192.168.1.6: icmp_req=2 ttl=64 time=0.440 ms
64 bytes from 192.168.1.6: icmp_req=3 ttl=64 time=0.234 ms
64 bytes from 192.168.1.6: icmp_req=4 ttl=64 time=0.239 ms

--- 192.168.1.6 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.234/0.345/0.467/0.108 ms
student@Ubuntu:~$
```

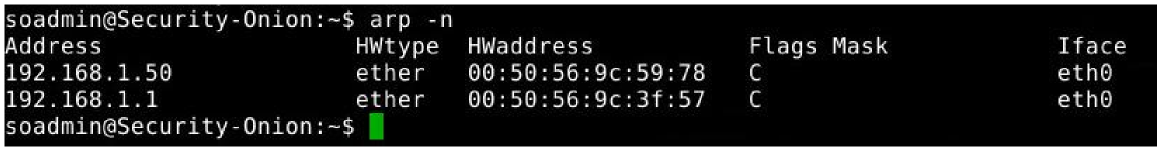
8. Switch back to **SecOnion** and press **CTRL+C** to stop the *tcpdump* capture. Notice the *ARP* output: *HH:MM:SS:mmmmmm srcMAC > dstMAC: ptype, len, request/response, length*.



```
soadmin@Security-Onion:~$ sudo tcpdump -i eth0 -nn -e arp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
14:39:09.850588 00:50:56:82:56:8f > 00:50:56:9c:3f:57, ethertype ARP (0x0806), length 60: Request wh
o-has 192.168.1.1 (00:50:56:9c:3f:57) tell 192.168.1.100, length 46
14:39:09.867608 00:50:56:9c:3f:57 > 00:50:56:82:56:8f, ethertype ARP (0x0806), length 60: Reply 192.
168.1.1 is-at 00:50:56:9c:3f:57, length 46
14:39:38.864797 00:50:56:82:fc:0e > 00:50:56:9c:3f:57, ethertype ARP (0x0806), length 42: Request wh
o-has 192.168.1.1 tell 192.168.1.6, length 28
14:39:38.865030 00:50:56:9c:3f:57 > 00:50:56:82:fc:0e, ethertype ARP (0x0806), length 60: Reply 192.
168.1.1 is-at 00:50:56:9c:3f:57, length 46
14:39:55.848888 00:50:56:82:56:8f > 00:50:56:9c:3f:57, ethertype ARP (0x0806), length 60: Request wh
o-has 192.168.1.1 (00:50:56:9c:3f:57) tell 192.168.1.100, length 46
14:39:55.849035 00:50:56:9c:3f:57 > 00:50:56:82:56:8f, ethertype ARP (0x0806), length 60: Reply 192.
168.1.1 is-at 00:50:56:9c:3f:57, length 46
14:39:59.664554 00:50:56:9c:59:78 > 00:50:56:9c:3f:57, ethertype ARP (0x0806), length 60: Request wh
o-has 192.168.1.1 tell 192.168.1.50, length 46
14:39:59.664593 00:50:56:9c:3f:57 > 00:50:56:9c:59:78, ethertype ARP (0x0806), length 60: Reply 192.
```

9. Type the command shown below to display the ARP table. Notice the ARP entry for the IP address *192.168.1.50*.

```
soadmin@Security-Onion:~$ arp -n
```

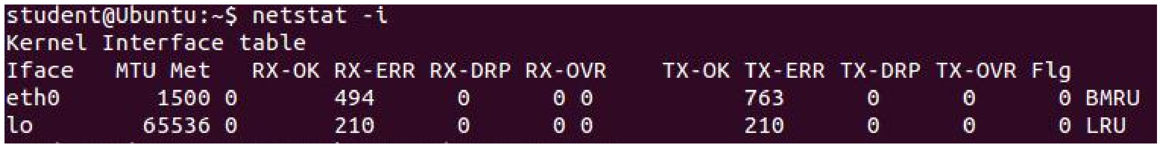


```
soadmin@Security-Onion:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.1.50             ether   00:50:56:9c:59:78   C                     eth0
192.168.1.1              ether   00:50:56:9c:3f:57   C                     eth0
soadmin@Security-Onion:~$
```

1.3 Using arpspoof to Spoof Network Traffic

1. Change focus to the **Ubuntu** system.
2. In the *terminal* window, enter the command below to see whether *eth0* is in *promiscuous mode*.

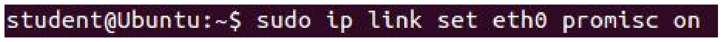
```
student@Ubuntu:~$ netstat -i
```



```
student@Ubuntu:~$ netstat -i
Kernel Interface table
Iface    MTU Met   RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0       1500 0       494      0      0 0           763      0      0      0 BMRU
lo        65536 0       210      0      0 0           210      0      0      0 LRU
```

3. Notice that *eth0* is not in promiscuous mode. To change this, enter the command below. If prompted with a password, enter `securepassword`.

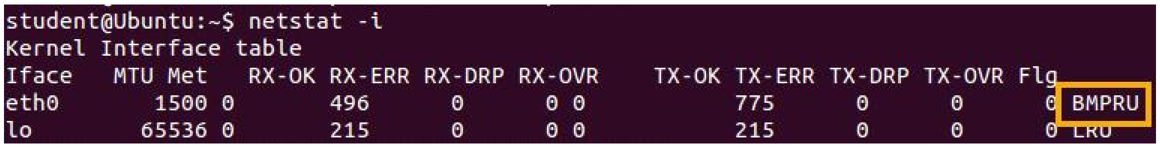
```
student@Ubuntu:~$ sudo ip link set eth0 promisc on
```



```
student@Ubuntu:~$ sudo ip link set eth0 promisc on
```

4. Enter the netstat command once more to confirm the changes took effect.

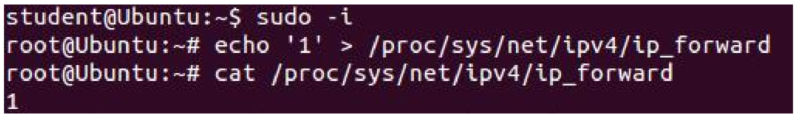
```
student@Ubuntu:~$ netstat -i
```



```
student@Ubuntu:~$ netstat -i
Kernel Interface table
Iface   MTU Met   RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0       1500 0       496      0      0 0         775      0      0      0 BMPRU
lo        65536 0       215      0      0 0         215      0      0      0 LRU
```

5. Once confirmed, configure the *Ubuntu* system to act as a router between the *pfSense* router and the victim (in this case the *SecOnion* system). Type the commands below to change the value from '**0**' to '**1**'. This will help by not modifying the source address of packets going through. If prompted with a password, enter `securepassword`.

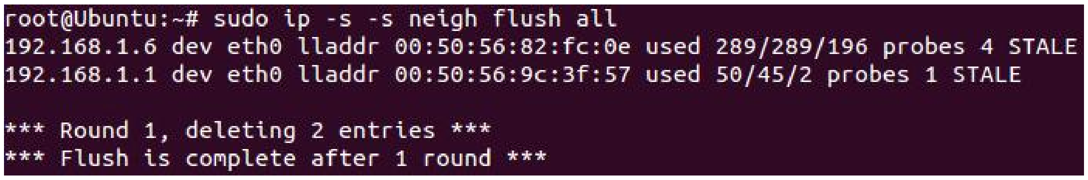
```
student@Ubuntu:~$ sudo -i
root@Ubuntu:~# echo '1' > /proc/sys/net/ipv4/ip_forward
root@Ubuntu:!# cat /proc/sys/net/ipv4/ip_forward
```



```
student@Ubuntu:~$ sudo -i
root@Ubuntu:~# echo '1' > /proc/sys/net/ipv4/ip_forward
root@Ubuntu:~# cat /proc/sys/net/ipv4/ip_forward
1
```

6. Flush out the entire *ARP* table by entering the command below. If prompted with a password, enter `securepassword`.

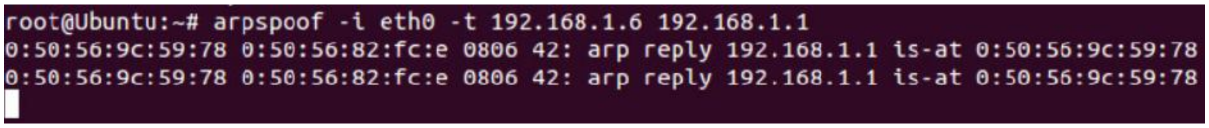
```
root@Ubuntu:!# sudo ip –s –s neigh flush all
```



```
root@Ubuntu:~# sudo ip -s -s neigh flush all
192.168.1.6 dev eth0 lladdr 00:50:56:82:fc:0e used 289/289/196 probes 4 STALE
192.168.1.1 dev eth0 lladdr 00:50:56:9c:3f:57 used 50/45/2 probes 1 STALE

*** Round 1, deleting 2 entries ***
*** Flush is complete after 1 round ***
```

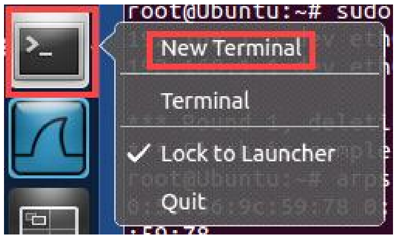
7. Initiate the command below and leave it running in the background. This command will essentially spoof the host's *MAC* on the switch.

```
root@Ubuntu:!# arpspoof –i eth0 –t 192.168.1.6 192.168.1.1
```



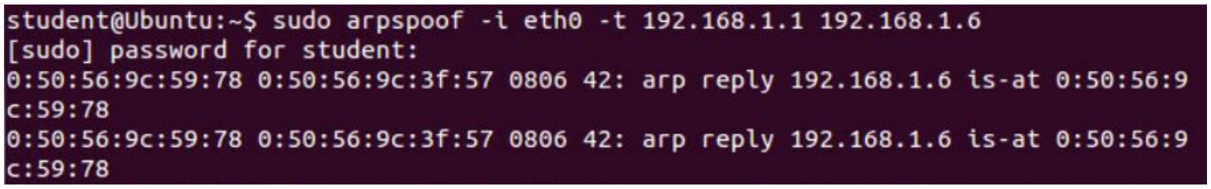
```
root@Ubuntu:~# arpspoof -i eth0 -t 192.168.1.6 192.168.1.1
0:50:56:9c:59:78 0:50:56:82:fc:e 0806 42: arp reply 192.168.1.1 is-at 0:50:56:9c:59:78
0:50:56:9c:59:78 0:50:56:82:fc:e 0806 42: arp reply 192.168.1.1 is-at 0:50:56:9c:59:78
```

8. In the left menu pane, click on the **terminal** icon and select **New Terminal**.



9. In the new *terminal*, enter the command below to spoof the *MAC* on the host. Leave it running in the background. If prompted with a password, enter `securepassword`.

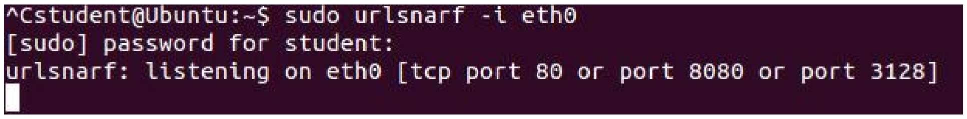
```
student@Ubuntu:~$ sudo arpspoof –i eth0 –t 192.168.1.1 192.168.1.6
```



```
student@Ubuntu:~$ sudo arpspoof -i eth0 -t 192.168.1.1 192.168.1.6
[sudo] password for student:
0:50:56:9c:59:78 0:50:56:9c:3f:57 0806 42: arp reply 192.168.1.6 is-at 0:50:56:9
c:59:78
0:50:56:9c:59:78 0:50:56:9c:3f:57 0806 42: arp reply 192.168.1.6 is-at 0:50:56:9
c:59:78
```

10. Open another new terminal by right-clicking on the terminal icon in the left pane and selecting **New Terminal**. In this *terminal* window, type the **urlsnarf** command below. With this command, a man-in-the-middle attack can sniff the wire actively and monitor what information passes through from the victim. In this case, we are sniffing website data that the victim is entering in their web browser. If prompted with a password, enter `securepassword`.

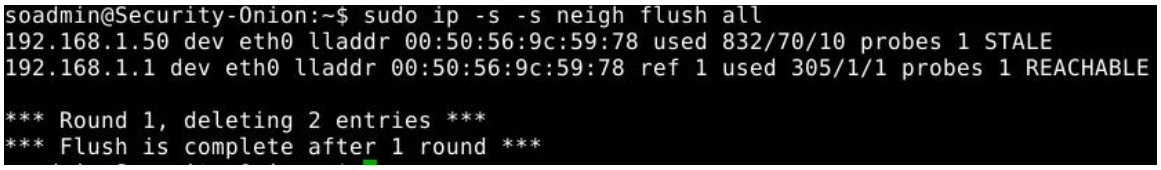
```
student@Ubuntu:~$ sudo urlsnarf –i eth0
```



```
^Cstudent@Ubuntu:~$ sudo urlsnarf -i eth0
[sudo] password for student:
urlsnarf: listening on eth0 [tcp port 80 or port 8080 or port 3128]
```

11. Switch to the **SecOnion** system and enter the command below into a *terminal* window to flush out the *ARP* table. If prompted for a password, enter `mypassword`.

```
soadmin@Security-Onion:~$ sudo ip –s –s neigh flush all
```



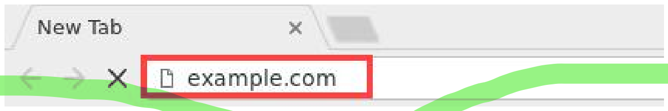
```
soadmin@Security-Onion:~$ sudo ip -s -s neigh flush all
192.168.1.50 dev eth0 lladdr 00:50:56:9c:59:78 used 832/70/10 probes 1 STALE
192.168.1.1 dev eth0 lladdr 00:50:56:9c:59:78 ref 1 used 305/1/1 probes 1 REACHABLE

*** Round 1, deleting 2 entries ***
*** Flush is complete after 1 round ***
```

12. In the same terminal, enter the command below to launch a web browser.

```
soadmin@Security-Onion:~$ chromium-browser
```

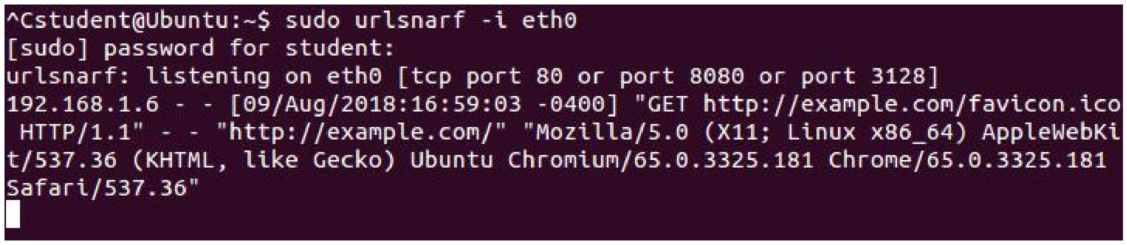
If the web browser does not open immediately, wait 1-2 minutes for it to launch.

13. In the address bar, type `example.com`. Press **Enter**.



2

14. After the webpage loads, switch back to the **Ubuntu** system. View the terminal with **urlsnarf** running and observe the output from the command.



```
^Cstudent@Ubuntu:~$ sudo urlsnarf -i eth0
[sudo] password for student:
urlsnarf: listening on eth0 [tcp port 80 or port 8080 or port 3128]
192.168.1.6 - - [09/Aug/2018:16:59:03 -0400] "GET http://example.com/favicon.ico
 HTTP/1.1" - - "http://example.com/" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKi
t/537.36 (KHTML, like Gecko) Ubuntu Chromium/65.0.3325.181 Chrome/65.0.3325.181
Safari/537.36"
```

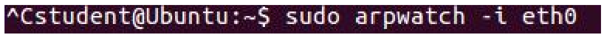
Notice the *GET* entries and how easily it is to spoof *ARP* entries when on the same medium.

15. Press **CTRL+C** to stop the *urlsnarf* process.
16. Leave the *Ubuntu* viewer open to continue with the next task.

1.4 Using arpwatch to Mitigate Spoofed Network Traffic

1. In a terminal, enter the command below to initiate *arpwatch*, a tool that actively seeks any *MAC* address changes on the system's interface. If prompted for a password, enter `securepassword`.

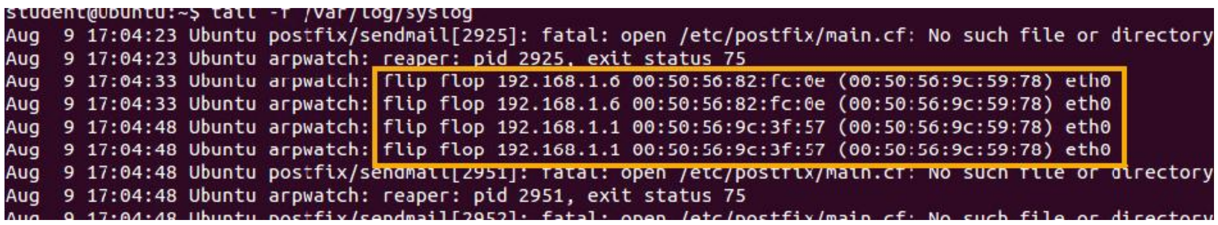
```
student@Ubuntu:~$ sudo arpwatch –i eth0
```



```
^Cstudent@Ubuntu:~$ sudo arpwatch -i eth0
```

2. View the output by typing the command below. Notice the entries from *arpwatch*. This helps mitigate the *ARP* spoofing attack by informing the user when a MAC change has occurred.

```
student@Ubuntu:~$ tail –f /var/log/syslog
```



```
student@Ubuntu:~$ tail -f /var/log/syslog
Aug  9 17:04:23 Ubuntu postfix/sendmail[2925]: fatal: open /etc/postfix/main.cf: No such file or directory
Aug  9 17:04:23 Ubuntu arpwatch: reaper: pid 2925, exit status 75
Aug  9 17:04:33 Ubuntu arpwatch: flip flop 192.168.1.6 00:50:56:82:fc:0e (00:50:56:9c:59:78) eth0
Aug  9 17:04:33 Ubuntu arpwatch: flip flop 192.168.1.6 00:50:56:82:fc:0e (00:50:56:9c:59:78) eth0
Aug  9 17:04:48 Ubuntu arpwatch: flip flop 192.168.1.1 00:50:56:9c:3f:57 (00:50:56:9c:59:78) eth0
Aug  9 17:04:48 Ubuntu arpwatch: flip flop 192.168.1.1 00:50:56:9c:3f:57 (00:50:56:9c:59:78) eth0
Aug  9 17:04:48 Ubuntu postfix/sendmail[2951]: fatal: open /etc/postfix/main.cf: No such file or directory
Aug  9 17:04:48 Ubuntu arpwatch: reaper: pid 2951, exit status 75
Aug  9 17:04:48 Ubuntu postfix/sendmail[2952]: fatal: open /etc/postfix/main.cf: No such file or directory
```

Please Note

If you do not see the “flip flop” occur in the syslog right away, you may have to wait 1-2 minutes before you see the event happen in real time.

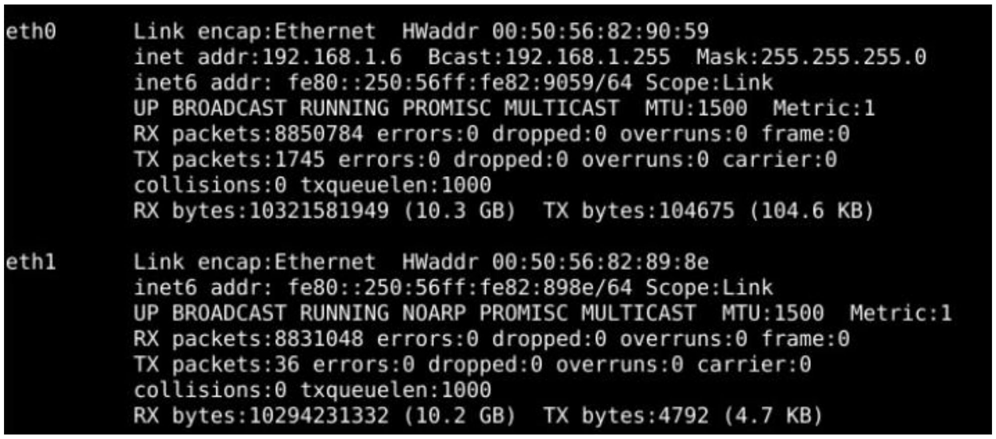
3. Press **CTRL+C** to stop the process.
4. Navigate to the other **terminal** and press **CTRL+C** to stop the *arpspoof* process. Do the same for the other **terminal** with *arpsoof* running.
5. Close all *terminals*.

2 Using Wireshark to Capture & Analyze Network Traffic

2.1 Using Wireshark to Capture FTP Traffic

1. Change focus to the **SecOnion** system.
2. Open a *terminal* window and confirm that both *eth0* and *eth1* are up.

```
soadmin@Security-Onion:~$ ifconfig -a
```

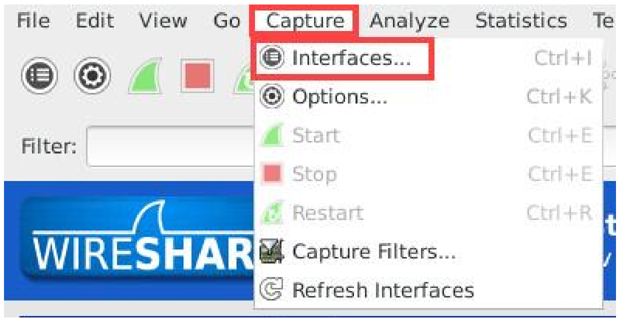


```
eth0      Link encap:Ethernet  HWaddr 00:50:56:82:90:59
          inet addr:192.168.1.6  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::250:56ff:fe82:9059/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:8850784 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1745 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:10321581949 (10.3 GB)  TX bytes:104675 (104.6 KB)

eth1      Link encap:Ethernet  HWaddr 00:50:56:82:89:8e
          inet6 addr: fe80::250:56ff:fe82:898e/64 Scope:Link
          UP BROADCAST RUNNING NOARP PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:8831048 errors:0 dropped:0 overruns:0 frame:0
          TX packets:36 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:10294231332 (10.2 GB)  TX bytes:4792 (4.7 KB)
```

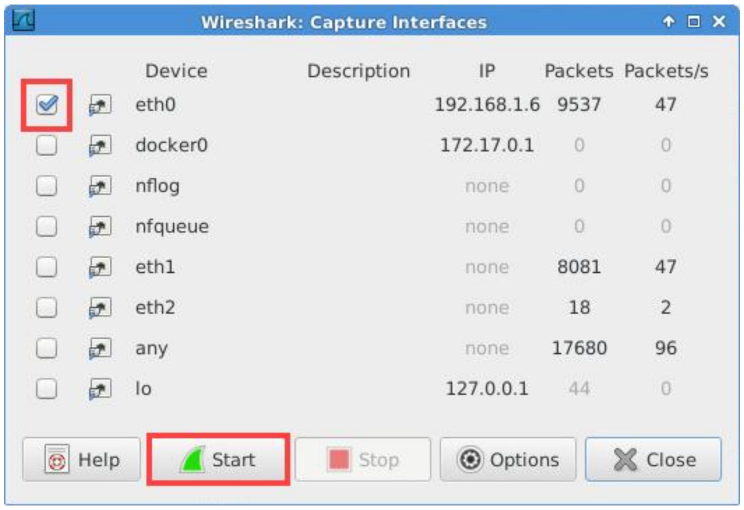
3. If either interface is down, bring them back up by using the **sudo ifconfig eth0 up** command.
4. Type the command below to run *Wireshark* as root. If prompted for a password, enter `mypassword`.

```
soadmin@Security-Onion:~$ sudo wireshark
```

5. If prompted with an error message and a message stating that running *Wireshark* can be dangerous to run while in root, select **OK** for both messages to proceed.
6. Start capturing traffic by clicking the top menu option **Capture > Interfaces**.

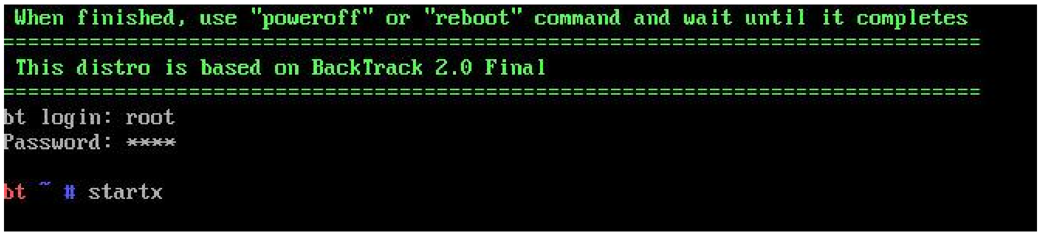


7. In the new *Capture Interfaces* window, select the **eth0** interface and click the **Start** button.



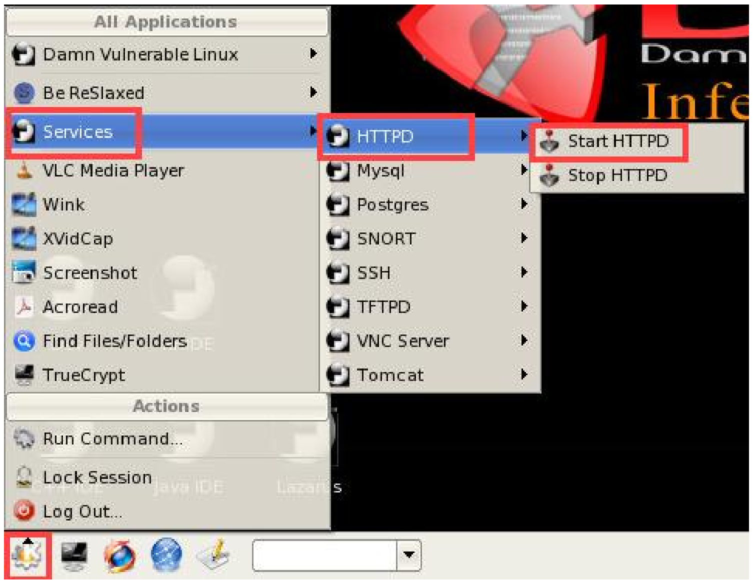
	Device	Description	IP	Packets	Packets/s
✓	eth0		192.168.1.6	9537	47
	docker0		172.17.0.1	0	0
	nflog		none	0	0
	nfqueue		none	0	0
	eth1		none	8081	47
	eth2		none	18	2
	any		none	17680	96
	lo		127.0.0.1	44	0

8. Launch the **DVL** virtual machine.
9. On the login screen, type `root` followed by pressing the **Enter** key.
10. When prompted for a password, type `toor` and press **Enter** again.
11. When presented with the user prompt, type `startx` and then press **Enter**.

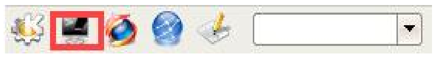


```
 When finished, use "poweroff" or "reboot" command and wait until it completes
================================================================================
 This distro is based on BackTrack 2.0 Final
================================================================================
bt login: root
Password: ****

bt ~ # startx
```

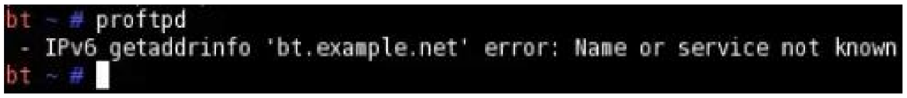
12. Click on the **Application Menu** and navigate to **Services > HTTPD > Start HTTPD** to start the web service.



13. When the message stating that the *Apache* server started, click **OK**.
14. Open a new **terminal** window by clicking on the icon located on the bottom menu pane.



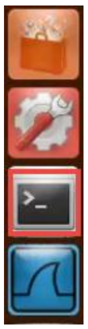
15. Type `proftpd` followed by pressing **Enter** to initiate the *FTP* server.



```
bt ~ # proftpd
 - IPv6 getaddrinfo 'bt.example.net' error: Name or service not known
bt ~ #
```

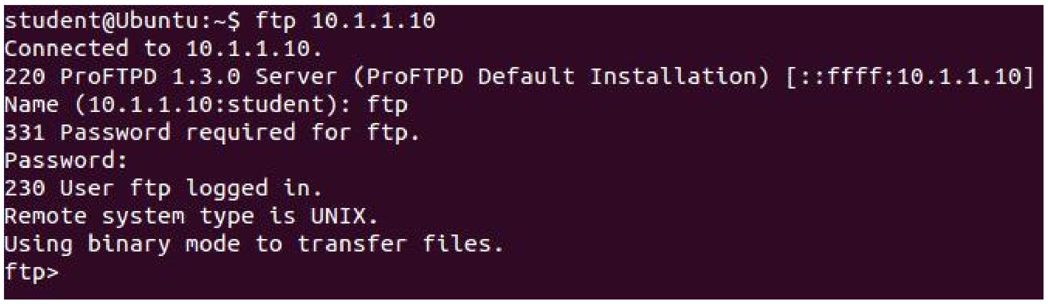
Ignore the error message and continue to the next step.

16. Switch to the **Ubuntu** system and open a **terminal** window.



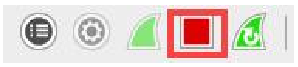
17. Type the command below to connect to the FTP server located on the *DVL Server*. When prompted for a username and password, enter: `ftp` as the user and `ftp` again as the password.

```
student@Ubuntu:~$ ftp 10.1.1.10
```

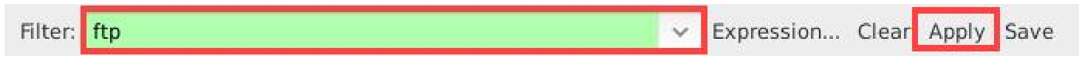


```
student@Ubuntu:~$ ftp 10.1.1.10
Connected to 10.1.1.10.
220 ProFTPD 1.3.0 Server (ProFTPD Default Installation) [::ffff:10.1.1.10]
Name (10.1.1.10:student): ftp
331 Password required for ftp.
Password:
230 User ftp logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

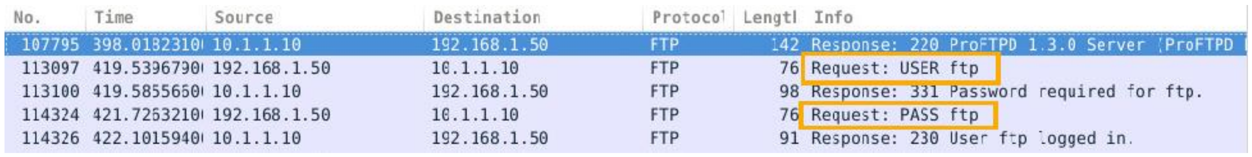
12. Once connected to the *FTP* server, switch back to the **SecOnion** system and click the **Stop Capture** button in the *Wireshark* interface.



13. Type `ftp` in the filter pane and click **Apply**.



14. Now that the packet focus is on *FTP* only, locate the initial request. Here, you can see the username *ftp* and the password of *ftp* in clear text.

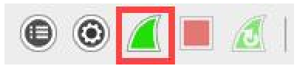


No.	Time	Source	Destination	Protocol	Length	Info
107795	398.0182310	10.1.1.10	192.168.1.50	FTP	142	Response: 220 ProFTPD 1.3.0 Server (ProFTPD
113097	419.5396790	192.168.1.50	10.1.1.10	FTP	76	Request: USER ftp
113100	419.5855650	10.1.1.10	192.168.1.50	FTP	98	Response: 331 Password required for ftp.
114324	421.7253210	192.168.1.50	10.1.1.10	FTP	76	Request: PASS ftp
114326	422.1015940	10.1.1.10	192.168.1.50	FTP	91	Response: 230 User ftp logged in.

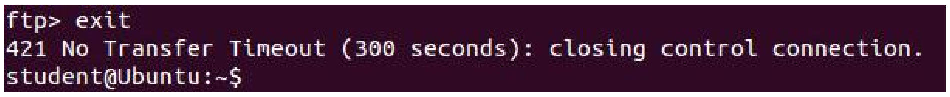
Using *FTP* has gone a long way; since it is no longer a secure channel to use, we will show how using *VSFTP* (*Very Secure FTP*) can be more secure in the following task.

2.2 Using Wireshark to Capture SFTP Traffic

1. Start a new capture by clicking on the Start a new live capture button.



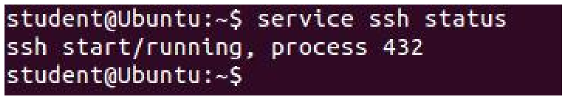
2. If prompted to save the capture file, select **Continue without Saving**.
3. Click on the **Clear** button to clear the filter pane.
4. Switch to the **Ubuntu** system.
5. While on the *terminal* window, type `exit` followed by pressing **Enter** to log out from the *FTP* server.



```
ftp> exit
421 No Transfer Timeout (300 seconds): closing control connection.
student@Ubuntu:~$
```

6. Type the command below to verify that the SSH service is running.

```
student@Ubuntu:~$ service ssh status
```



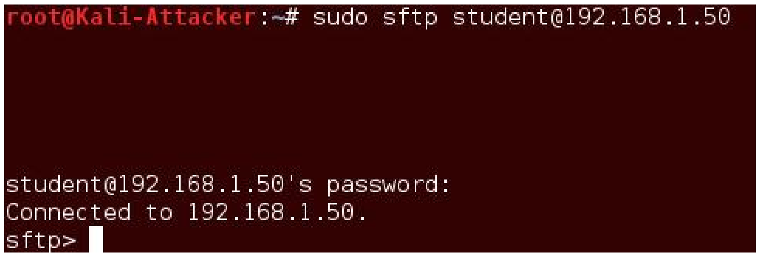
```
student@Ubuntu:~$ service ssh status
ssh start/running, process 432
student@Ubuntu:~$
```

7. If the *SSH* service is not running, type the command below followed by **Enter**:

```
student@Ubuntu:~$:~# sudo service ssh start
```

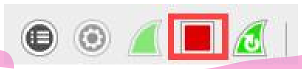
8. Change focus to the **Kali** system.
9. Open the terminal window and enter the command below. If prompted for a password, enter `securepassword`.

```
root@Kali-Attacker:~# sftp student@192.168.1.50
```



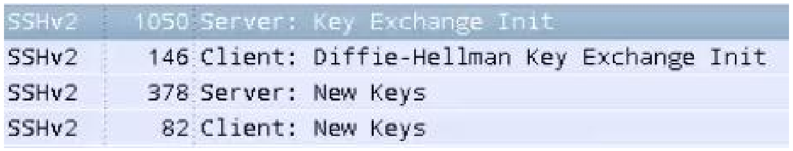
```
root@Kali-Attacker:~# sudo sftp student@192.168.1.50



student@192.168.1.50's password:
Connected to 192.168.1.50.
sftp>
```

10. Switch to the **SecOnion** system and stop the capture by clicking the **Stop Capture** button.



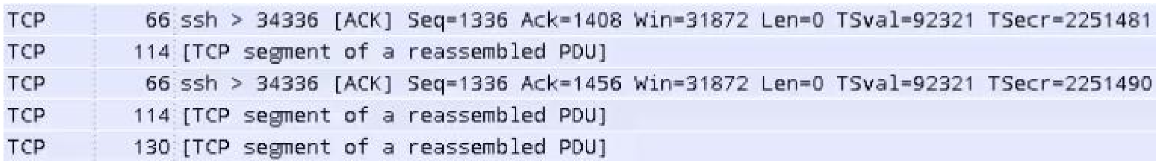
3

11. Locate the *Diffie-Hellman key exchange* between the client and the *SFTP* service.



```
SSHv2   1050 Server: Key Exchange Init
SSHv2    146 Client: Diffie-Hellman Key Exchange Init
SSHv2    378 Server: New Keys
SSHv2     82 Client: New Keys
```

12. Notice after the key exchange, the TCP packets that follow are encrypted over the medium.



```
TCP      66 ssh > 34336 [ACK] Seq=1336 Ack=1408 Win=31872 Len=0 TSval=92321 TSecr=2251481
TCP     114 [TCP segment of a reassembled PDU]
TCP      66 ssh > 34336 [ACK] Seq=1336 Ack=1456 Win=31872 Len=0 TSval=92321 TSecr=2251490
TCP     114 [TCP segment of a reassembled PDU]
TCP     130 [TCP segment of a reassembled PDU]
```

Notice that you can no longer see the username and password in clear text when compared to using *FTP*.

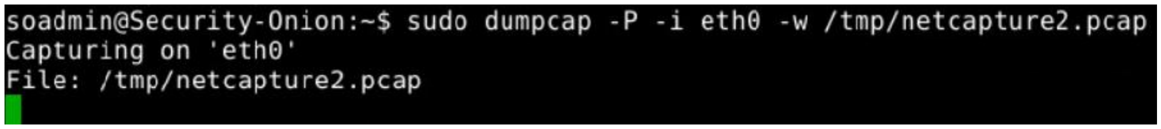
13. Close the **Wireshark** application.
14. Leave the *SecOnion* viewer open to continue with the next task.

3 Capturing and Analyzing HTTP Traffic

3.1 Using dumpcap to Capture HTTP Traffic

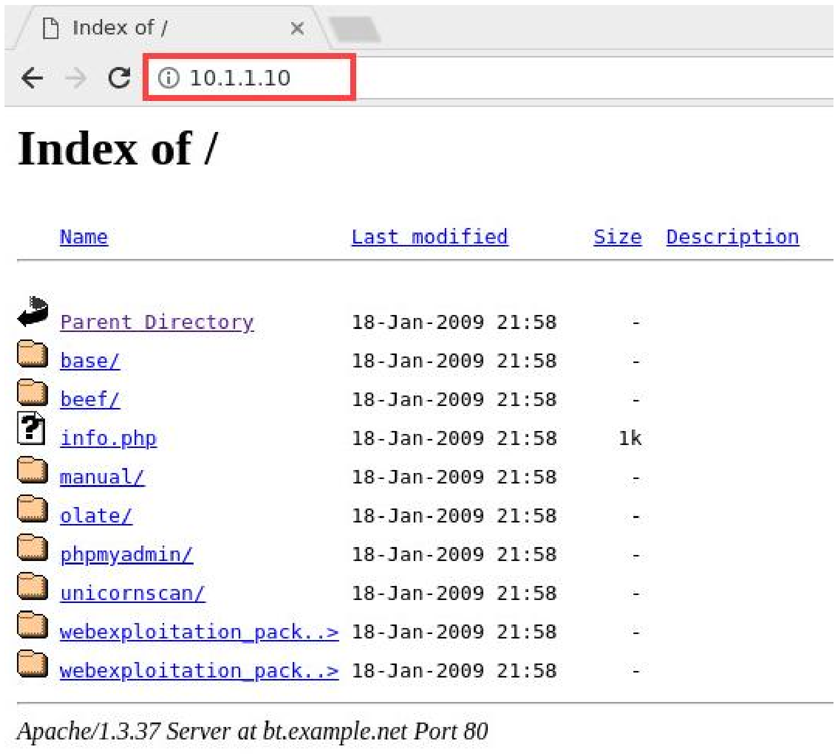
1. While on the *SecOnion* system, focus on the terminal window and enter the command below. If prompted for a password, enter `mypassword`.

```
soadmin@Security-Onion:~$ sudo dumpcap -P –i eth0 –w /tmp/netcapture2.pcap
```



```
soadmin@Security-Onion:~$ sudo dumpcap -P -i eth0 -w /tmp/netcapture2.pcap
Capturing on 'eth0'
File: /tmp/netcapture2.pcap
```

3. Switch focus to the web browser and type `10.1.1.10` into address bar followed by pressing the **Enter** key. Wait until the page finishes loading.



Index of /

```
   Name                        Last modified      Size  Description
   Parent Directory            18-Jan-2009 21:58     -
   base/                       18-Jan-2009 21:58     -
   beef/                       18-Jan-2009 21:58     -
   info.php                    18-Jan-2009 21:58    1k
   manual/                     18-Jan-2009 21:58     -
   olate/                      18-Jan-2009 21:58     -
   phpmyadmin/                 18-Jan-2009 21:58     -
   unicornscan/                18-Jan-2009 21:58     -
   webexploitation_pack..>     18-Jan-2009 21:58     -
   webexploitation_pack..>     18-Jan-2009 21:58     -
```

Apache/1.3.37 Server at bt.example.net Port 80

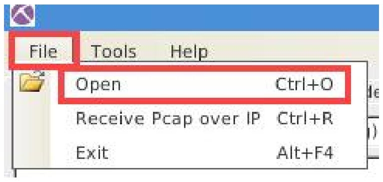
4. Switch back to the terminal running *dumpcap* and press **CTRL+C** to stop the running process.
5. Leave the *terminal* open to continue with the next task.

3.2 Using Network Miner to Capture HTTP Traffic

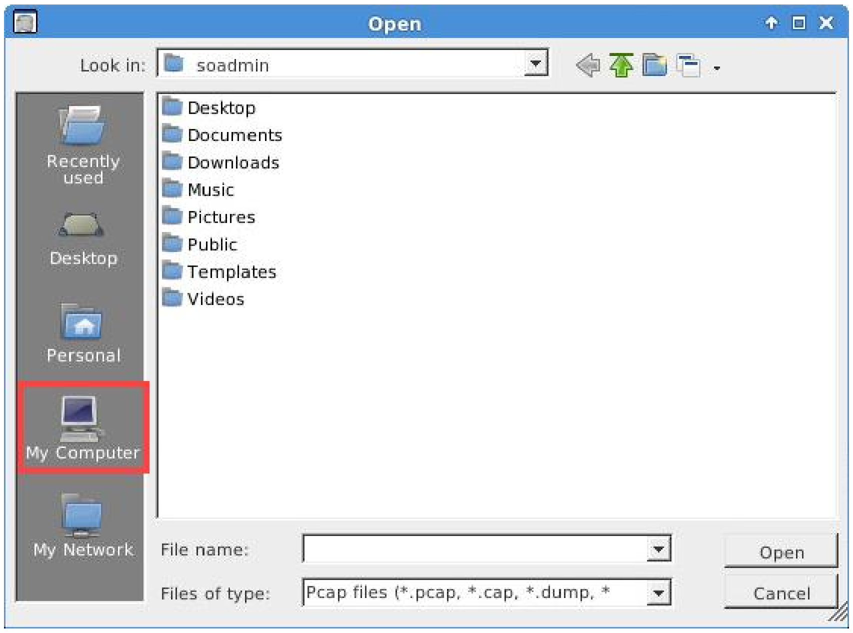
1. While in the *terminal*, enter the command below to open the program **Network Miner**.

```
soadmin@Security-Onion:~$ sudo /opt/networkminer/networkminer
```

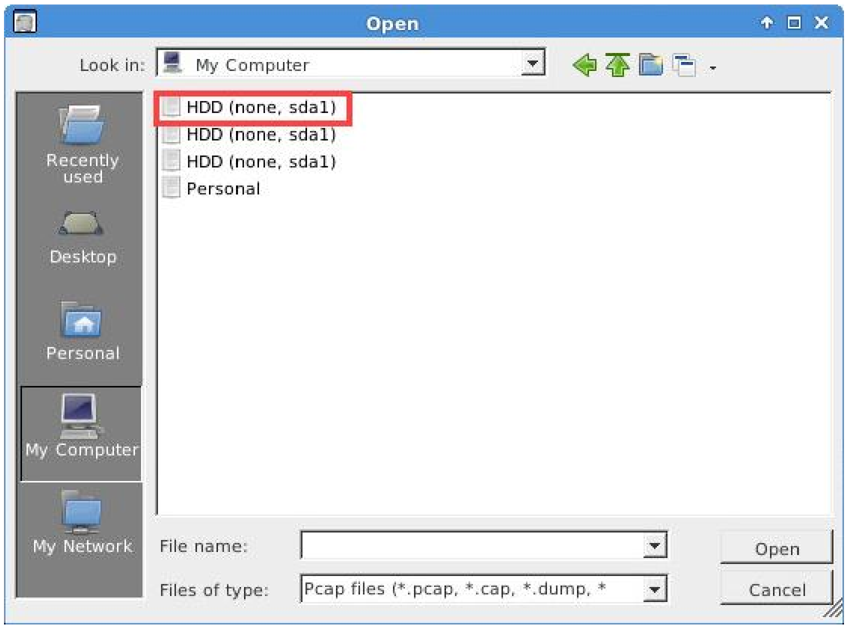
2. On the *Network Miner* application window, navigate to **File > Open**.



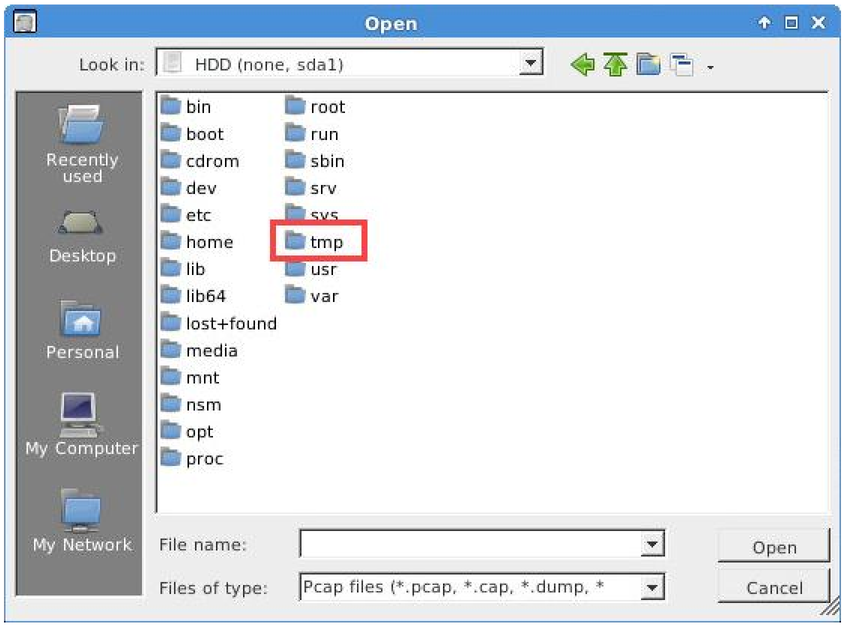
3. Select the **My Computer** icon from the left menu.



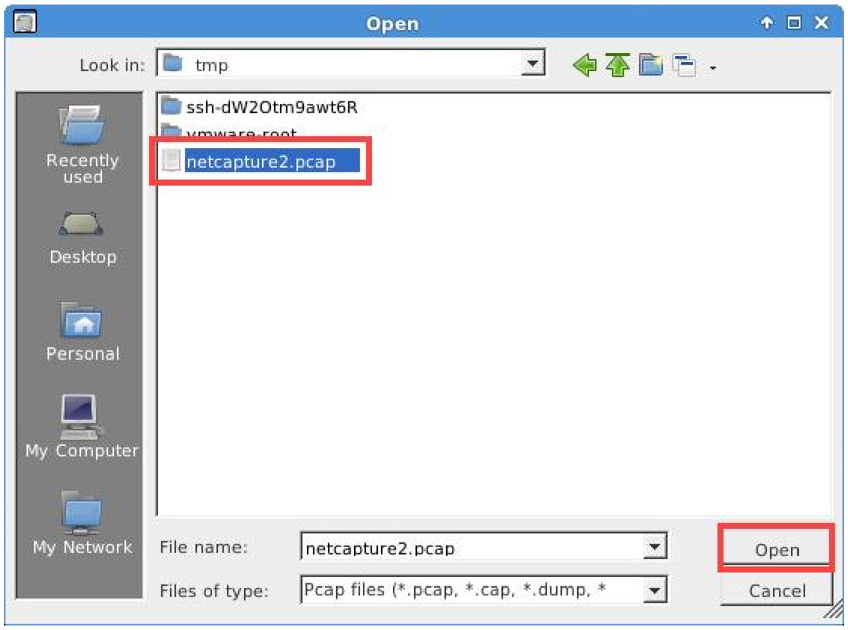
4. Double-click the first listed **HDD** entry.



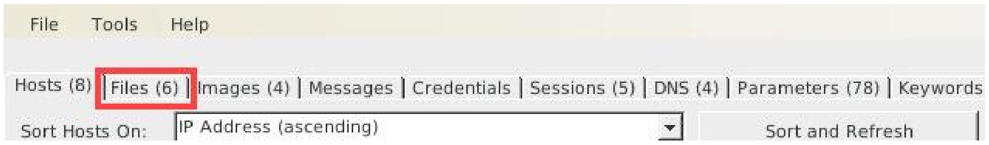
5. Next, double-click the **tmp** directory.



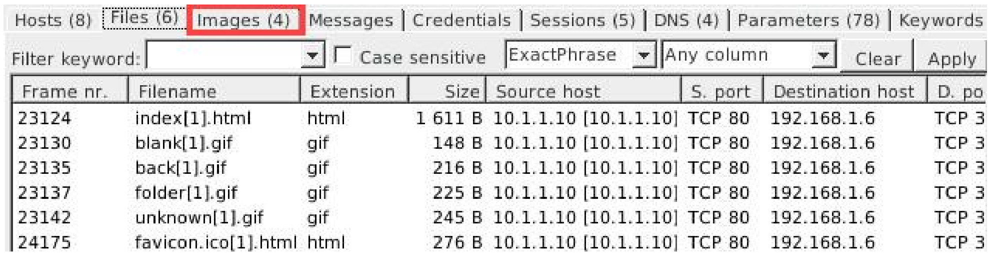
6. Select the **netcapture2.pcap** file and select **Open**.



7. Once the *PCAP* is finished loading, click on the **Files** tab within the *Network Miner* program.

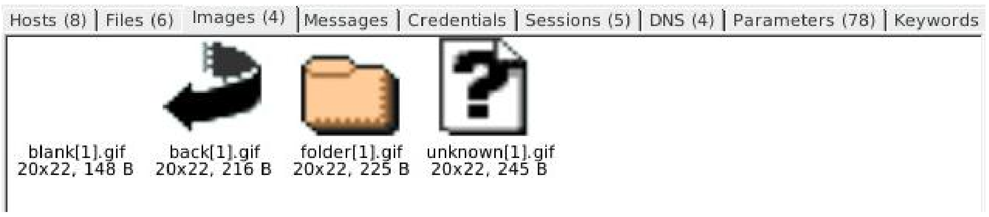


8. Notice the list of files acquired. Click on the **Images** tab.


Hosts (8) | Files (6) | Images (4) | Messages | Credentials | Sessions (5) | DNS (4) | Parameters (78) | Keywords
Filter keyword:
Case sensitive
ExactPhrase
Any column
Clear
Apply

Frame nr.	Filename	Extension	Size	Source host	S. port	Destination host	D. po
23124	index[1].html	html	1 611 B	10.1.1.10 [10.1.1.10]	TCP 80	192.168.1.6	TCP 3
23130	blank[1].gif	gif	148 B	10.1.1.10 [10.1.1.10]	TCP 80	192.168.1.6	TCP 3
23135	back[1].gif	gif	216 B	10.1.1.10 [10.1.1.10]	TCP 80	192.168.1.6	TCP 3
23137	folder[1].gif	gif	225 B	10.1.1.10 [10.1.1.10]	TCP 80	192.168.1.6	TCP 3
23142	unknown[1].gif	gif	245 B	10.1.1.10 [10.1.1.10]	TCP 80	192.168.1.6	TCP 3
24175	favicon.ico[1].html	html	276 B	10.1.1.10 [10.1.1.10]	TCP 80	192.168.1.6	TCP 3

9. Notice the images that are captured.


Hosts (8) | Files (6) | Images (4) | Messages | Credentials | Sessions (5) | DNS (4) | Parameters (78) | Keywords
blank[1].gif
20x22, 148 B
back[1].gif
20x22, 216 B
folder[1].gif
20x22, 225 B
unknown[1].gif
20x22, 245 B

10. The lab is now complete; you may end the reservation.