Scenario

You're a network analyst who needs to use `tcpdump` to capture and analyze live network traffic from a Linux virtual machine.

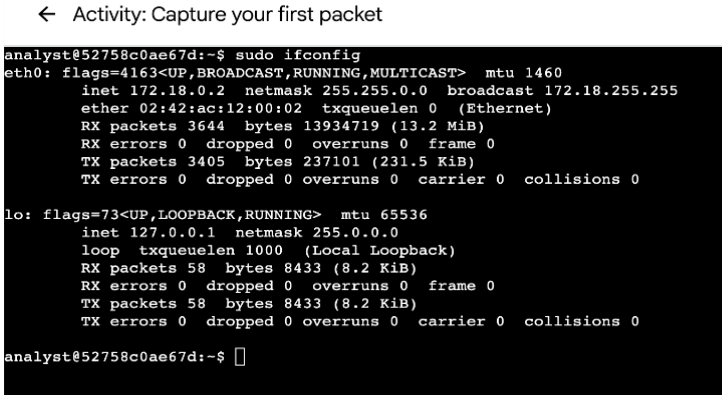
The lab starts with your user account, called `analyst`, already logged in to a Linux terminal.

Your Linux user's home directory contains a sample packet capture file that you will use at the end of the lab to answer a few questions about the network traffic that it contains.

Task 1. Identify network interfaces

In this task, you must identify the network interfaces that can be used to capture network packet data.

1. Use `ifconfig` to identify the interfaces that are available:



Next, Identify the interface options available for packet capture:

```
analyst@52758c0ae67d:~$ sudo tcpdump -D
1.eth0 [Up, Running]
2.any (Pseudo-device that captures on all interfaces) [Up, Running]
3.lo [Up, Running, Loopback]
4.nflog (Linux netfilter log (NFLOG) interface)
5.nfqueue (Linux netfilter queue (NFQUEUE) interface)
analyst@52758c0ae67d:~$
```

2. Inspect the network traffic of a network interface with tcpdump.

- Use `sudo tcpdump -i eth0 -v -c5` to filter live network packet data:

```
analyst@52758c0ae67d:~$ sudo tcpdump -i eth0 -v -c5
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 262144 by
tes
18:41:41.464427 IP (tos 0x0, ttl 64, id 31657, offset 0, flags [DF], proto TCP
(6), length 115)
    52758c0ae67d.5000 > nginx-us-central1-b.c.qwiklabs-terminal-vms-prod-00.int
ernal.52964: Flags [P.], cksum 0x588d (incorrect -> 0xb7b5), seq 2596934417:259
6934480, ack 2009877808, win 492, options [nop,nop,TS val 3548894881 ecr 232390
7120], length 63
18:41:41.464622 IP (tos 0x0, ttl 63, id 57800, offset 0, flags [DF], proto TCP
(6), length 52)
    nginx-us-central1-b.c.qwiklabs-terminal-vms-prod-00.internal.52964 > 52758c
0ae67d.5000: Flags [.], cksum 0x8e83 (correct), ack 63, win 507, options [nop,n
op,TS val 2323907557 ecr 3548894881], length 0
18:41:41.475119 IP (tos 0x0, ttl 64, id 31658, offset 0, flags [DF], proto TCP
(6), length 146)
    52758c0ae67d.5000 > nginx-us-central1-b.c.qwiklabs-terminal-vms-prod-00.int
ernal.52964: Flags [P.], cksum 0x58ac (incorrect -> 0x4f99), seq 63:157, ack 1,
 win 492, options [nop,nop,TS val 3548894891 ecr 2323907557], length 94
18:41:41.475496 IP (tos 0x0, ttl 63, id 57801, offset 0, flags [DF], proto TCP
(6), length 52)
    nginx-us-central1-b.c.qwiklabs-terminal-vms-prod-00.internal.52964 > 52758c
0ae67d.5000: Flags [.], cksum 0x8e11 (correct), ack 157, win 507, options [nop,
nop,TS val 2323907567 ecr 3548894891], length 0
18:41:41.490331 IP (tos 0x0, ttl 64, id 10697, offset 0, flags [DF], proto UDP
(17), length 69)
    52758c0ae67d.48552 > metadata.google.internal.domain: 44489+ PTR? 2.0.17.17
2.in-addr.arpa. (41)
5 packets captured
10 packets received by filter
0 packets dropped by kernel
analyst@52758c0ae67d:~$
```

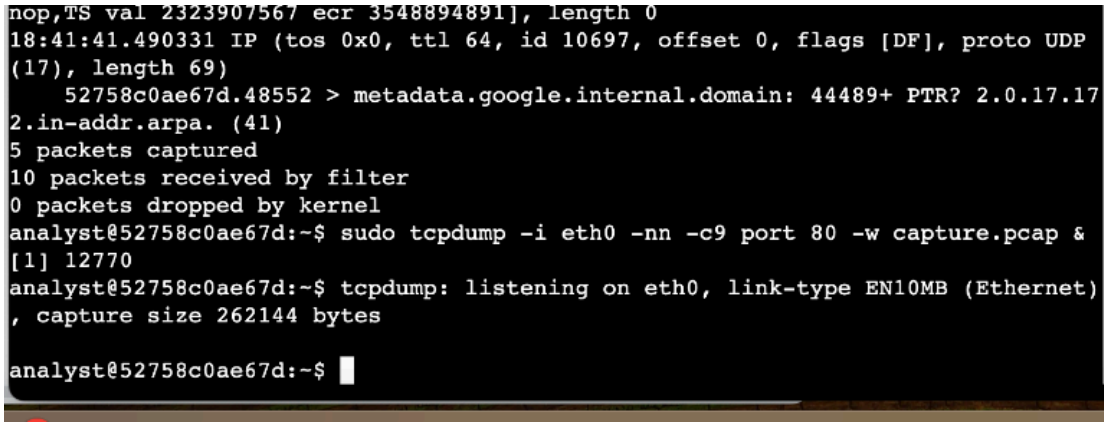
-i eth0: Capture data specifically from the eth0 interface.
-v: Display detailed packet data.
-c5: Capture 5 packets of data.

3. Capture Network traffic:

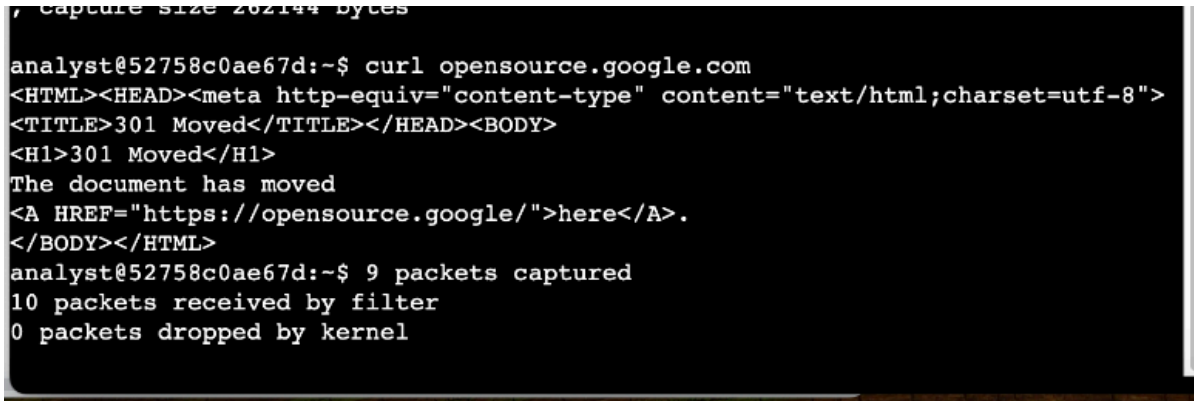
- Capture packet data into a file named called `capture.pcap`: `sudo tcpdump -i eth0 -nn -c9 port 80 -w capture.pcap &`.

`-i eth0`: Capture data from the eth0 interface.
`-nn`: Do not attempt to resolve IP addresses or ports to names.This is best practice from a security perspective, as the lookup data may not be valid. It also prevents malicious actors from being alerted to an investigation.
`-c9`: Capture 9 packets of data and then exit.
`port 80`: Filter only port 80 traffic. This is the default HTTP port.
`-w capture.pcap`: Save the captured data to the named file.

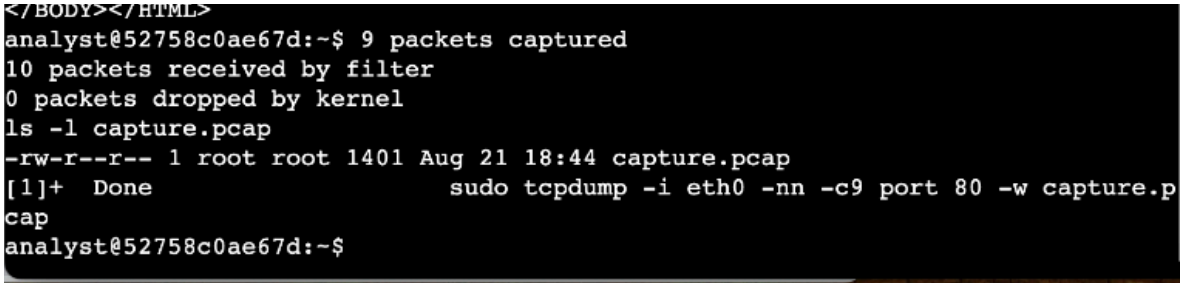
`&`: This is an instruction to the Bash shell to run the command in the background.



- Use `curl` to generate some HTTP (port 80) traffic: `curl opensource.google.com`.



Verify the packet data has been captured: `ls -l capture.pcap`



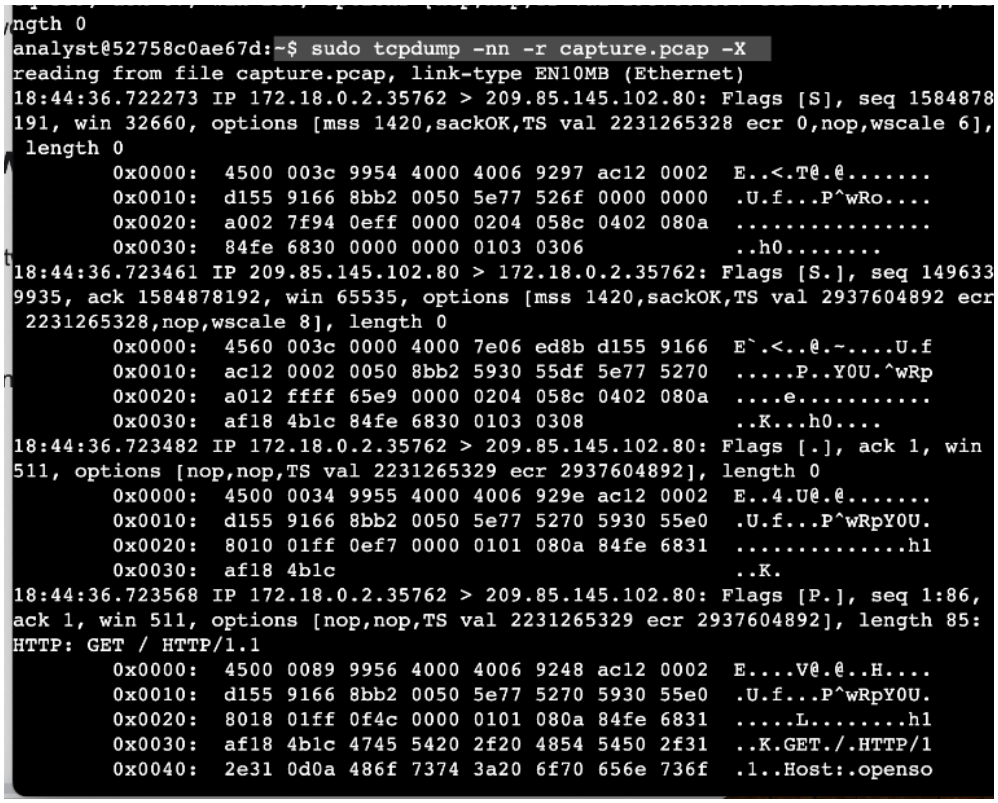
4. Filter the captured packet data.
   - Filter the packet header data from the `capture.pcap` capture file: `sudo tcpdump -nn -r capture.pcap -v`.

```
cap
analyst@52758c0ae67d:~$ sudo tcpdump -nn -r capture.pcap -v
reading from file capture.pcap, link-type EN10MB (Ethernet)
18:44:36.722273 IP (tos 0x0, ttl 64, id 39252, offset 0, flags [DF], proto TCP
(6), length 60)
    172.18.0.2.35762 > 209.85.145.102.80: Flags [S], cksum 0x0eff (incorrect ->
 0x8fac), seq 1584878191, win 32660, options [mss 1420,sackOK,TS val 2231265328
 ecr 0,nop,wscale 6], length 0
18:44:36.723461 IP (tos 0x60, ttl 126, id 0, offset 0, flags [DF], proto TCP (6
), length 60)
    209.85.145.102.80 > 172.18.0.2.35762: Flags [S.], cksum 0x65e9 (correct), s
eq 1496339935, ack 1584878192, win 65535, options [mss 1420,sackOK,TS val 29376
04892 ecr 2231265328,nop,wscale 8], length 0
18:44:36.723482 IP (tos 0x0, ttl 64, id 39253, offset 0, flags [DF], proto TCP
(6), length 52)
    172.18.0.2.35762 > 209.85.145.102.80: Flags [.], cksum 0x0ef7 (incorrect ->
 0x928e), ack 1, win 511, options [nop,nop,TS val 2231265329 ecr 2937604892], l
ength 0
18:44:36.723568 IP (tos 0x0, ttl 64, id 39254, offset 0, flags [DF], proto TCP
(6), length 137)
    172.18.0.2.35762 > 209.85.145.102.80: Flags [P.], cksum 0x0f4c (incorrect -
> 0x0142), seq 1:86, ack 1, win 511, options [nop,nop,TS val 2231265329 ecr 293
7604892], length 85: HTTP, length: 85
        GET / HTTP/1.1
        Host: opensource.google.com
        User-Agent: curl/7.64.0
        Accept: */*

18:44:36.723994 IP (tos 0x60, ttl 126, id 0, offset 0, flags [DF], proto TCP (6
), length 52)
    209.85.145.102.80 > 172.18.0.2.35762: Flags [.], cksum 0x9337 (correct), ac
k 86, win 256, options [nop,nop,TS val 2937604893 ecr 2231265329], length 0
18:44:36.726160 IP (tos 0x80, ttl 126, id 0, offset 0, flags [DF], proto TCP (6
), length 590)
    209.85.145.102.80 > 172.18.0.2.35762: Flags [P.], cksum 0x5e53 (correct), s
eq 1:539, ack 86, win 256, options [nop,nop,TS val 2937604895 ecr 2231265329],
length 538: HTTP, length: 538
        HTTP/1.1 301 Moved Permanently
        Location: https://opensource.google/
        Content-Type: text/html; charset=UTF-8
        X-Content-Type-Options: nosniff
```

`-nn`: Disable port and protocol name lookup.
`-r`: Read capture data from the named file.
`-v`: Display detailed packet data.

- Filter the extended packet data from the `capture.pcap` capture file: `sudo tcpdump -nn -r capture.pcap -X`.



`-nn`: Disable port and protocol name lookup.
`-r`: Read capure data from the named file.
`-X`: Display the hexadecimal and ASCII output format packet data. Security analysts can analyze hexadecimal and ASCII output to detect patterns or anomalies during malware analysis or forensic analysis.