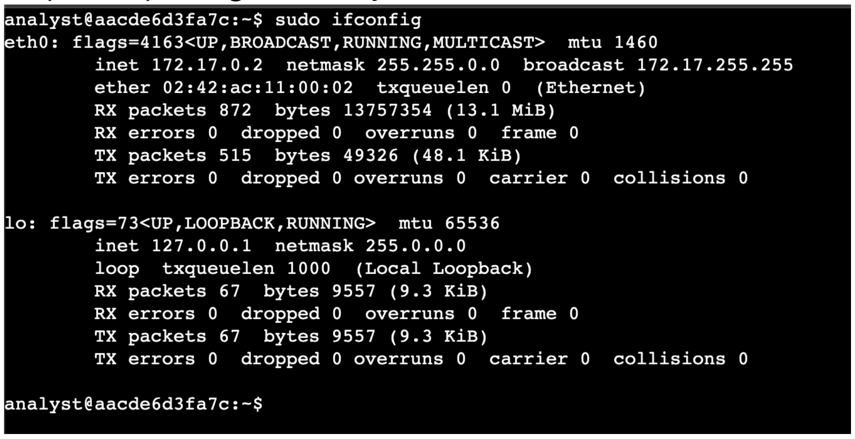
Project

In this project, I analyzed network packets using TCPdump, examining the captured traffic to understand network communication patterns, protocol details, and potential security insights.

I used psudo ipconfig to identify the interfaces that are available

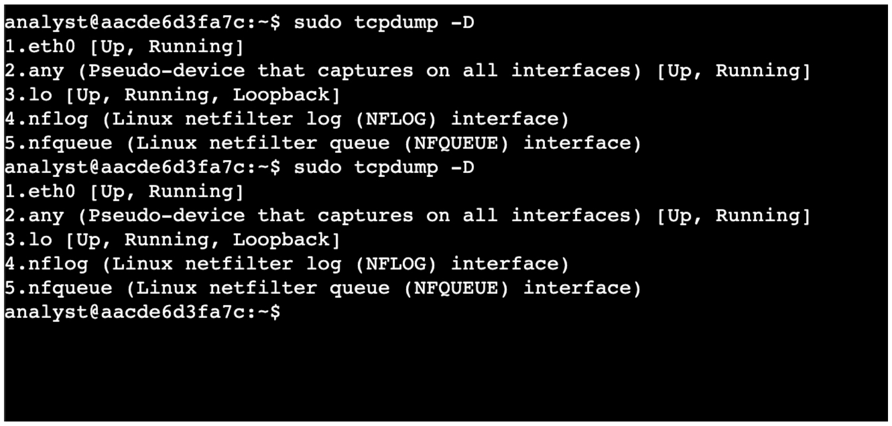


```
analyst@aacde6d3fa7c:~$ sudo ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1460
        inet 172.17.0.2  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:ac:11:00:02  txqueuelen 0  (Ethernet)
        RX packets 872  bytes 13757354 (13.1 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 515  bytes 49326 (48.1 KiB)
        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
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 67  bytes 9557 (9.3 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 67  bytes 9557 (9.3 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

analyst@aacde6d3fa7c:~$
```

In this project, I identified the Ethernet network interface by locating the entry with the "eth" prefix. I specifically used eth0 as the interface for capturing network packet data in the subsequent tasks. To identify all available interface options for packet capture, I executed the command sudo tcpdump -D, which provided a comprehensive list of interfaces that could be used with tcpdump.

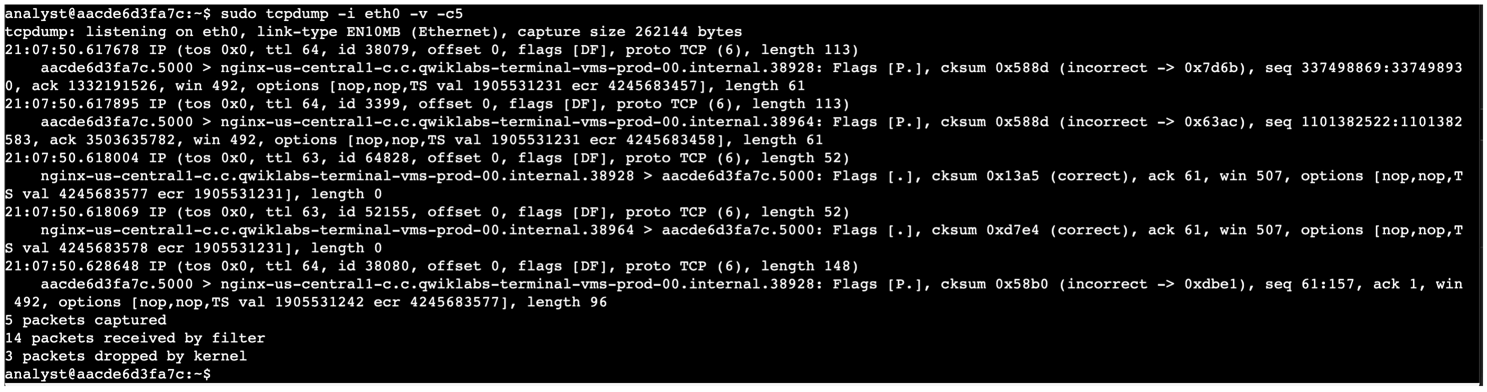


```
analyst@aacde6d3fa7c:~$ sudo tcpdump -D
1.eth0 [Up, Running]
2.any (Pseudo-device that captures on all interfaces) [Up, Running]
3.lo [Up, Running, Loopback]
4.nflog (Linux netfilter log (NFLOG) interface)
5.nfqueue (Linux netfilter queue (NFQUEUE) interface)
analyst@aacde6d3fa7c:~$ sudo tcpdump -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@aacde6d3fa7c:~$
```

In this project, I used tcpdump to filter and inspect live network packet traffic on the eth0 interface. I executed the command sudo tcpdump -i eth0 -v -c5, which allowed me to capture and analyze five verbose packets from the eth0 interface. By specifying the interface with -i eth0, I ensured that only traffic on the Ethernet interface was captured, while the -c5 option limited the capture to five packets, providing a manageable sample of network activity for analysis.

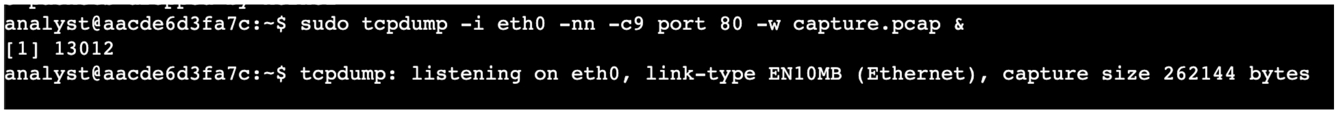
This command will run tcpdump with the following options:

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



```
analyst@aacde6d3fa7c:~$ sudo tcpdump -i eth0 -v -c5
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
21:07:50.617678 IP (tos 0x0, ttl 64, id 38079, offset 0, flags [DF], proto TCP (6), length 113)
    aacde6d3fa7c.5000 > nginx-us-central1-c.c.qwiklabs-terminal-vms-prod-00.internal.38928: Flags [P.], cksum 0x588d (incorrect -> 0x7d6b), seq 337498869:33749893
0, ack 1332191526, win 492, options [nop,nop,TS val 1905531231 ecr 4245683457], length 61
21:07:50.617895 IP (tos 0x0, ttl 64, id 3399, offset 0, flags [DF], proto TCP (6), length 113)
    aacde6d3fa7c.5000 > nginx-us-central1-c.c.qwiklabs-terminal-vms-prod-00.internal.38964: Flags [P.], cksum 0x588d (incorrect -> 0x63ac), seq 1101382522:1101382
583, ack 3503635782, win 492, options [nop,nop,TS val 1905531231 ecr 4245683458], length 61
21:07:50.618004 IP (tos 0x0, ttl 63, id 64828, offset 0, flags [DF], proto TCP (6), length 52)
    nginx-us-central1-c.c.qwiklabs-terminal-vms-prod-00.internal.38928 > aacde6d3fa7c.5000: Flags [.], cksum 0x13a5 (correct), ack 61, win 507, options [nop,nop,T
S val 4245683577 ecr 1905531231], length 0
21:07:50.618069 IP (tos 0x0, ttl 63, id 52155, offset 0, flags [DF], proto TCP (6), length 52)
    nginx-us-central1-c.c.qwiklabs-terminal-vms-prod-00.internal.38964 > aacde6d3fa7c.5000: Flags [.], cksum 0xd7e4 (correct), ack 61, win 507, options [nop,nop,T
S val 4245683578 ecr 1905531231], length 0
21:07:50.628648 IP (tos 0x0, ttl 64, id 38080, offset 0, flags [DF], proto TCP (6), length 148)
    aacde6d3fa7c.5000 > nginx-us-central1-c.c.qwiklabs-terminal-vms-prod-00.internal.38928: Flags [P.], cksum 0x58b0 (incorrect -> 0xdbe1), seq 61:157, ack 1, win
 492, options [nop,nop,TS val 1905531242 ecr 4245683577], length 96
5 packets captured
14 packets received by filter
3 packets dropped by kernel
analyst@aacde6d3fa7c:~$
```

In this step I saved captured network data to a packet capture file by executing the command sudo tcpdump -i eth0 -nn -c9 port 80 -w capture.pcap. This command captured only web traffic (TCP port 80) from the eth0 interface, limited the capture to 9 packets, and saved the data to a file named 'capture.pcap' for further analysis.

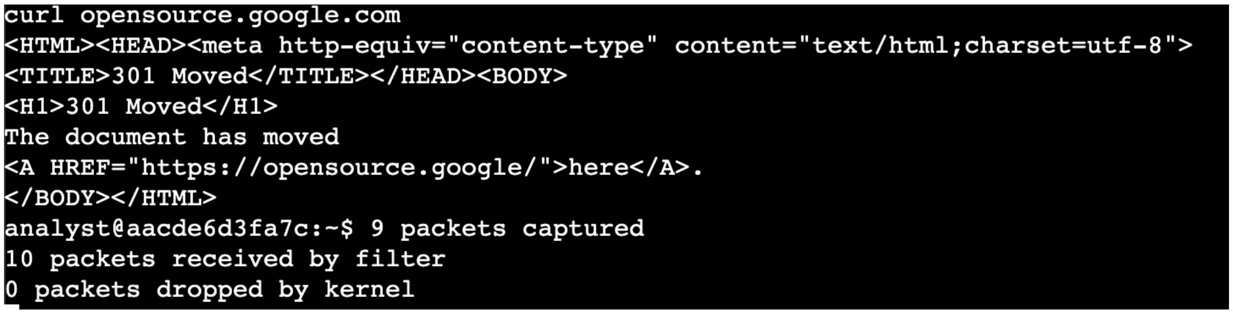


```
analyst@aacde6d3fa7c:~$ sudo tcpdump -i eth0 -nn -c9 port 80 -w capture.pcap &
[1] 13012
analyst@aacde6d3fa7c:~$ tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

This command will run tcpdump in the background with the following options:

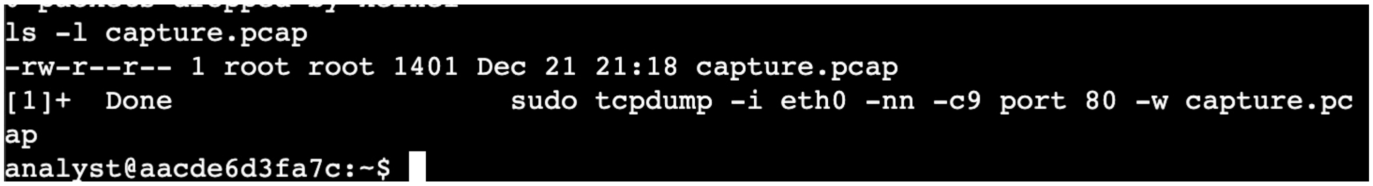
1. -i eth0: Capture data from the eth0 interface.
2. -nn: Do not attempt to resolve IP addresses or ports to names.This is best practice from a security perspective, as the lookup data may not be valid. It also prevents malicious actors from being alerted to an investigation.
3. -c9: Capture 9 packets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

In this step, I generated HTTP (port 80) traffic by executing the command curl opensource.google.com. This command simulated a simple HTTP GET request, allowing me to analyze the resulting network activity and capture relevant packet data for further inspection.



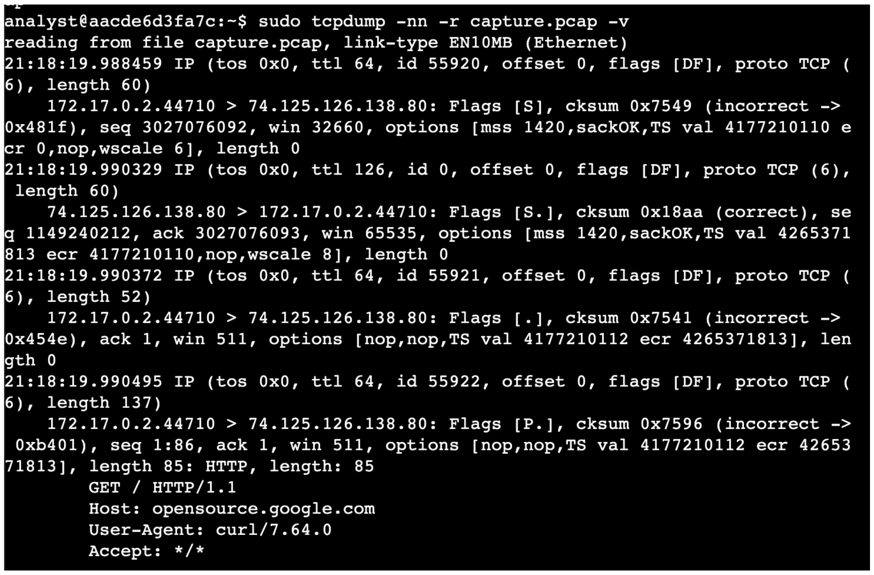
```
curl opensource.google.com
<HTML><HEAD><meta http-equiv="content-type" content="text/html;charset=utf-8">
<TITLE>301 Moved</TITLE></HEAD><BODY>
<H1>301 Moved</H1>
The document has moved
<A HREF="https://opensource.google/">here</A>.
</BODY></HTML>
analyst@aacde6d3fa7c:~$ 9 packets captured
10 packets received by filter
0 packets dropped by kernel
```

In this step, I verified that the packet data was successfully captured by listing the contents of the directory and confirming the presence of the file using the command ls -l capture.pcap. This ensured that the captured data was saved correctly for further analysis.

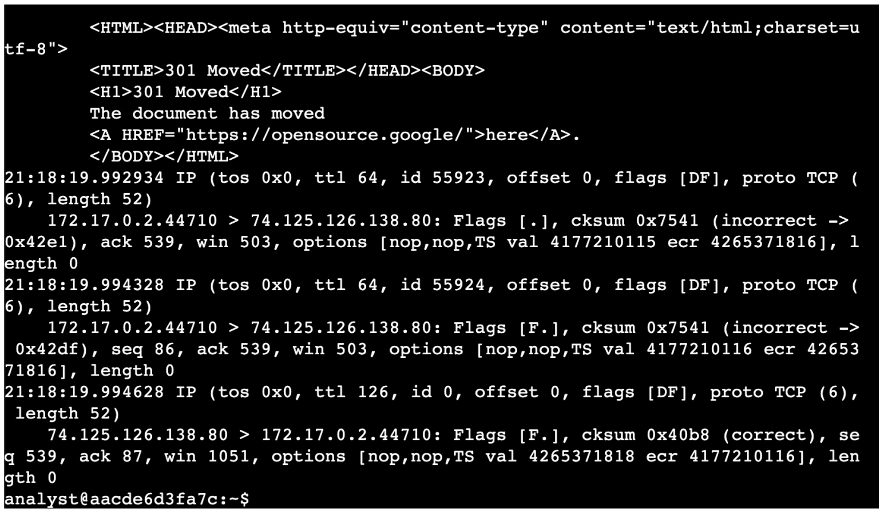


```
ls -l capture.pcap
-rw-r--r-- 1 root root 1401 Dec 21 21:18 capture.pcap
[1]+  Done                    sudo tcpdump -i eth0 -nn -c9 port 80 -w capture.pc
ap
analyst@aacde6d3fa7c:~$
```

In this step, I filtered packet header data from the previously saved capture.pcap file. I executed the command sudo tcpdump -nn -r capture.pcap -v, which allowed me to read the file and analyze the packet headers in a verbose format without resolving hostnames or ports.



```
analyst@aacde6d3fa7c:~$ sudo tcpdump -nn -r capture.pcap -v
reading from file capture.pcap, link-type EN10MB (Ethernet)
21:18:19.988459 IP (tos 0x0, ttl 64, id 55920, offset 0, flags [DF], proto TCP (
6), length 60)
    172.17.0.2.44710 > 74.125.126.138.80: Flags [S], cksum 0x7549 (incorrect ->
0x481f), seq 3027076092, win 32660, options [mss 1420,sackOK,TS val 4177210110 e
cr 0,nop,wscale 6], length 0
21:18:19.990329 IP (tos 0x0, ttl 126, id 0, offset 0, flags [DF], proto TCP (6),
 length 60)
    74.125.126.138.80 > 172.17.0.2.44710: Flags [S.], cksum 0x18aa (correct), se
q 1149240212, ack 3027076093, win 65535, options [mss 1420,sackOK,TS val 4265371
813 ecr 4177210110,nop,wscale 8], length 0
21:18:19.990372 IP (tos 0x0, ttl 64, id 55921, offset 0, flags [DF], proto TCP (
6), length 52)
    172.17.0.2.44710 > 74.125.126.138.80: Flags [.], cksum 0x7541 (incorrect ->
0x454e), ack 1, win 511, options [nop,nop,TS val 4177210112 ecr 4265371813], len
gth 0
21:18:19.990495 IP (tos 0x0, ttl 64, id 55922, offset 0, flags [DF], proto TCP (
6), length 137)
    172.17.0.2.44710 > 74.125.126.138.80: Flags [P.], cksum 0x7596 (incorrect ->
 0xb401), seq 1:86, ack 1, win 511, options [nop,nop,TS val 4177210112 ecr 42653
71813], length 85: HTTP, length: 85
        GET / HTTP/1.1
        Host: opensource.google.com
        User-Agent: curl/7.64.0
        Accept: */*
```



```
        <HTML><HEAD><meta http-equiv="content-type" content="text/html;charset=u
tf-8">
        <TITLE>301 Moved</TITLE></HEAD><BODY>
        <H1>301 Moved</H1>
        The document has moved
        <A HREF="https://opensource.google/">here</A>.
        </BODY></HTML>
21:18:19.992934 IP (tos 0x0, ttl 64, id 55923, offset 0, flags [DF], proto TCP (
6), length 52)
    172.17.0.2.44710 > 74.125.126.138.80: Flags [.], cksum 0x7541 (incorrect ->
0x42e1), ack 539, win 503, options [nop,nop,TS val 4177210115 ecr 4265371816], l
ength 0
21:18:19.994328 IP (tos 0x0, ttl 64, id 55924, offset 0, flags [DF], proto TCP (
6), length 52)
    172.17.0.2.44710 > 74.125.126.138.80: Flags [F.], cksum 0x7541 (incorrect ->
 0x42df), seq 86, ack 539, win 503, options [nop,nop,TS val 4177210116 ecr 42653
71816], length 0
21:18:19.994628 IP (tos 0x0, ttl 126, id 0, offset 0, flags [DF], proto TCP (6),
 length 52)
    74.125.126.138.80 > 172.17.0.2.44710: Flags [F.], cksum 0x40b8 (correct), se
q 539, ack 87, win 1051, options [nop,nop,TS val 4265371818 ecr 4177210116], len
gth 0
analyst@aacde6d3fa7c:~$
```

This command will run tcpdump with the following options:

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