Assignment 2

Anjishnu Mukherjee B05-511017020 (510517086)

Relevant flags	Corresponding tcpdump functionality
-D	Display available interfaces.
-n	Don't convert addresses (i.e., host addresses, port numbers, etc.) to names.
-nn	Stop Domain Name translation and lookups (Host names or port names).
-c count	Exit after receving count number of packets. (Here, count = 20)
-w file	Write the raw packets to file rather than parsing and printing them out.
-r file	Read packets from file (which was created with the -w option)
-ttt	Print a delta (micro-second resolution) between current and previous line on each dump line.
-A	Print packet information in Ascii format. Handy for capturing web pages.
host	Capture packets from specific hosts.
src	Capture packets from specific source.
dst	Capture packets from specific destination.
[port]	Capture packets from specific port.
-s snaplen	Snarf snaplen bytes of data from each packet.Setting snaplen to 0 sets it to the default of 262144.
	Setting snaplen to 0 sets it to the default of 262144.
-e	Print the link-level header on each dump line. This can be used, for example,
	to print MAC layer addresses for protocols such as Ethernet and IEEE 802.11.
-i interface	Listen on interface. If unspecified, tcpdump searches the system interface list
	for the lowest numbered, configured up interface (excluding loopback), which may be, for eg, eth0.
expression	Selects which packets will be dumped. If no expression is given, all packets on the net will be dumped.
	Otherwise, packets for which expression is true will be dumped.For expression syntax, see pcap-filter(7).

2a.1. **root@ubuntu-s-1vcpu-1gb-nyc1-01:~# tcpdump –version**

```
tcpdump version 4.9.2
libpcap version 1.8.1
OpenSSL 1.1.1  11 Sep 2018
```

2a.2. **root@ubuntu-s-1vcpu-1gb-nyc1-01:~# 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)
6.usbmon1 (USB bus number 1)

There are thus 6 interfaces available.
```

2a.3. **root@ubuntu-s-1vcpu-1gb-nyc1-01:~# ifconfig eth0 promisc**

```
Switch the primary ethernet interface to promiscuous mode.
```

2b.1. **root@ubuntu-s-1vcpu-1gb-nyc1-01:~# tcpdump -c 20 -n -w results_with_n.pcap**

```
Listen to the promiscuous mode interface of host with -n flag.
```

2b.2. **root@ubuntu-s-1vcpu-1gb-nyc1-01:~# tcpdump -c 20 -w results_without_n.pcap**

```
Listen to the promiscuous mode interface of host without -n flag.
```

2c.1. **root@ubuntu-s-1vcpu-1gb-nyc1-01:~# tcpdump -r results_with_n.pcap**

```
Read results_with_n.pcap file
```

2c.2. **root@ubuntu-s-1vcpu-1gb-nyc1-01:~# tcpdump -r results_without_n.pcap**

```
Read results_without_n.pcap file
```

2c.3. **Identifying the different fields present in TCP/IP packets captured by tcpdump.** :

```
1. Packet arrival time as per local clock
2. Internet Protocol - IP for IPv4 and IP6 for IPv6
3. Source IP address and port
4. TCP flags
5. Destination IP address and port
6. Destination network layer protocol
7. Packet length
```

2d. **Extract packet arrival time, source IP address, destination IP address and port**

```
Considering the file results_with_n.pcap,
we take any one entry in the file for demonstration.
The result will show arrivial time, source IP address,
destination IP address and port in the form of
[arrivial time][source IP].[port]>[destination IP].[port]
--------------------------------------------------------------------------
01:52:01.676462 IP ubuntu-s-1vcpu-1gb-nyc1-01.ssh > 115.96.128.146.50243 ...
--------------------------------------------------------------------------
Thus, arrival time is 01:52:01.676462 relative to local clock.
Source IP is ubuntu-s-1vcpu-1gb-nyc1-01 and port is ssh.
Destination IP is 115.96.128.146 and port is 50243.
The reason why source still displays a name and not a IP address
is because I am using a cloud machine which implements some measures
to ensure that -n flag doesn't reveal its IP address.
```

2e. **root@ubuntu-s-1vcpu-1gb-nyc1-01:~# tcpdump -c 1 -e**

```
Extract source MAC address and destination MAC address using the -e flag.
In the result, source MAC address and destination MAC address is shown
in the form of [source MAC address] > [destination MAC address]
-----------------------------------------------------------------------
... 6e:b5:09:98:23:a3 (oui Unknown) > 00:00:5e:00:01:6e (oui IANA) ...
-----------------------------------------------------------------------
MAC address of source is 6e:b5:09:98:23:a3
MAC address of destination is 00:00:5e:00:01:6e

Comparing with the output of question 2d, I have observed
that the source IP is ubuntu-s-1vcpu-1gb-nyc1-01.ssh
and destination IP is 115.96.128.146.50243.
```

2f. **root@ubuntu-s-1vcpu-1gb-nyc1-01:~# tcpdump -c 5 -ttt**

```
...
00:00:00.000000 ...
00:00:00.000058 ...
00:00:00.000050 ...
00:00:00.000038 ...
00:00:00.001187 ...
...

To get the inter-arrival times while capturing packets, I use the fact that the
-ttt flag prints a delta (micro-second resolution) between current and previous
line on each dump line. In the output displayed above, I have only included
the first colum of the output corresponding to these deltas.
```

2g. **root@laptop:~# tcpdump host www.google.com -A -c 5**

```
 www.google.com needs to be opened in the browser window for this
command to work. We need to run tcpdump first and then browse
something on google.com to get network traffic. -A flag ensures output
is in ASCII format.

...

20:28:40.784103 IP laptop.38450 > del11s05-in-f4.1e100.net.443: UDP,
length 724 E.....@.@.G......:.d.2......@...................'......J
<u...I..w.......5T.&O........p_.p.....6..}+.2....b...Z...j.V.=i.CU.i.E.0
..Ft#.~.,b.Q..../.3E=.lw.e..:...2....,.s    ..KXZ..;M.i.B.P..\:k..+....
{g.......$..    v7..N!....k`j..$.RV.0..?h.+.n..........3.^.*...
(.W7N...k...a..<G..DC.-.2.#x...f...N..X.~...@K.[.{.z....")..^.......a.z..jo
..<O.X.....R.A.G......}00....%.J.97yx.*..!jaP   }..o..:..L...e..].ucIp;H..
F.s#........@F.B.24..v..=Y...;."+.......8L..D.&...R...L.3.&.t=.o=6...h..4M
..:.E...L....e....M.......rh......$.=.........../P..0.t...q..Y
.p'......c.....f......K|...*....XuY...N.A..C.Q..pm.Q.!E.}.^....
[DY,:;.(..b2..B....]6V .Y..V.M.V...(.!.CX[BC....J=..9.......K.;
x.....u....X......y.>Y.-N:.gc....S.sR.^.OZ..m..+.)e........2.....c9.
O......j.....N.w
20:28:40.821607 IP del11s05-in-f4.1e100.net.443 > laptop.38450: UDP,
length 21 E..1..@.:....:.d.......2..    .A.....K..^...n..G..^.
20:28:40.894790 IP del11s05-in-f4.1e100.net.443 > laptop.38450: UDP,
length 59 E..W..@.:....:.d.......2.C..A..H.&.J@.>.
\i..&.A`.D.c......,..........0 U...@....BBa`RO.
20:28:40.894848 IP del11s05-in-f4.1e100.net.443 > laptop.38450: UDP,
length 17 E..-..@.:....:.d.......2....A.........-m8t...
20:28:40.903268 IP laptop.38450 > del11s05-in-f4.1e100.net.443: UDP,
length 29 E..9..@.@.J......:.d.2...%K.@.........zsA.."n.!...tX..,.8

...
```

2h. **The different fields of each request and reply for 2h.1.b, 2h.2.b, 2h.3.b:**

- Packet arrival time
- Internet protocol, IPv4 in this case
- Source IP address
- Destination IP address
- Destination network layer protocol with status
- Ping request ID
- Sequence number of captured packet
- Packet length
- Flags

2h.1.a **root@ubuntu-s-1vcpu-1gb-nyc1-01:~# ping 8.8.8.8**

```
We let ping run in one terminal.

PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=56 time=1.36 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=56 time=0.896 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=56 time=0.890 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=56 time=0.870 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=56 time=0.904 ms
...
```

2h.1.b **root@ubuntu-s-1vcpu-1gb-nyc1-01:~# tcpdump -c 5 icmp**

```
We run this tcpdump command to capture the packets associated with ping
in a separate terminal and let tcpdump listen to traffic
from traceroute. traceroute and ping both use icmp protocol.

...
13:23:49.642813 IP ubuntu-s-1vcpu-1gb-nyc1-01 > dns.google: ICMP
echo request, id 8530, seq 2, length 64
13:23:49.643680 IP dns.google > ubuntu-s-1vcpu-1gb-nyc1-01: ICMP
echo reply, id 8530, seq 2, length 64
13:23:50.644122 IP ubuntu-s-1vcpu-1gb-nyc1-01 > dns.google: ICMP
echo request, id 8530, seq 3, length 64
13:23:50.644980 IP dns.google > ubuntu-s-1vcpu-1gb-nyc1-01: ICMP
echo reply, id 8530, seq 3, length 64
13:23:51.645379 IP ubuntu-s-1vcpu-1gb-nyc1-01 > dns.google: ICMP
echo request, id 8530, seq 4, length 64
...
```

2h.2.a **root@ubuntu-s-1vcpu-1gb-nyc1-01:~# wget https://wordpress.org/latest.zip**

```
We let wget run in one terminal.

--2020-01-19 13:32:29--  https://wordpress.org/latest.zip
Resolving wordpress.org (wordpress.org)... 198.143.164.252
Connecting to wordpress.org (wordpress.org)|198.143.164.252|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 13360988 (13M) [application/zip]
Saving to: ‘latest.zip’

latest.zip.5 100%[=================================================
                 ========================>]
                 12.74M  51.9MB/s    in 0.2s

2020-01-19 13:32:30 (51.9 MB/s) - ‘latest.zip.5’ saved [13360988/13360988]
```

2h.2.b **root@ubuntu-s-1vcpu-1gb-nyc1-01:~# tcpdump -c 5 -i eth0 not icmp**

```
We run this tcpdump command to capture the packets associated with wget
in a separate terminal and let tcpdump listen to traffic
from traceroute.

...
13:32:30.316194 IP ubuntu-s-1vcpu-1gb-nyc1-01.ssh > 115.96.128.99.50842:
Flags [P.], seq 3563088372:3563088480, ack 1172209409, win 501, length 108
13:32:30.316247 IP ubuntu-s-1vcpu-1gb-nyc1-01.ssh > 115.96.128.99.50842:
Flags [P.], seq 108:144, ack 1, win 501, length 36
13:32:30.316294 IP ubuntu-s-1vcpu-1gb-nyc1-01.ssh > 115.96.128.99.50842:
Flags [P.], seq 144:252, ack 1, win 501, length 108
13:32:30.316333 IP ubuntu-s-1vcpu-1gb-nyc1-01.ssh > 115.96.128.99.50842:
Flags [P.], seq 252:288, ack 1, win 501, length 36
13:32:30.317360 IP ubuntu-s-1vcpu-1gb-nyc1-01.ssh > 115.96.128.99.50842:
Flags [P.], seq 288:476, ack 1, win 501, length 188
...
```

2h.3.a **root@ubuntu-s-1vcpu-1gb-nyc1-01:~# traceroute www.google.com**

```
We let traceroute run in one terminal.

traceroute to www.google.com (172.217.10.36), 30 hops max, 60 byte packets
1  157.245.240.253 (157.245.240.253)  0.797 ms 157.245.240.254 (157.245.240.254)
   0.741 ms 157.245.240.253 (157.245.240.253)  0.740 ms
2  138.197.251.72 (138.197.251.72)  1.036 ms 138.197.251.90 (138.197.251.90)
   0.976 ms  1.328 ms
3  138.197.251.108 (138.197.251.108)  0.770 ms 138.197.251.112 (138.197.251.112)
   0.731 ms 138.197.251.104 (138.197.251.104)  0.729 ms
4  138.197.244.9 (138.197.244.9)  0.789 ms 138.197.244.11 (138.197.244.11)
   0.849 ms 138.197.244.13 (138.197.244.13)  0.823 ms
5  162.243.191.241 (162.243.191.241)  1.165 ms 162.243.191.243 (162.243.191.243)
   1.221 ms  1.247 ms
6  108.170.248.97 (108.170.248.97)  1.222 ms 108.170.248.33 (108.170.248.33)
   1.846 ms *
7  216.239.62.171 (216.239.62.171)  0.944 ms 216.239.62.169 (216.239.62.169)
   0.925 ms  0.903 ms
8  lga34s13-in-f4.1e100.net (172.217.10.36)  0.865 ms 216.239.62.169
   (216.239.62.169) 0.991 ms lga34s13-in-f4.1e100.net (172.217.10.36)  0.793 ms
```

2h.3.b **root@ubuntu-s-1vcpu-1gb-nyc1-01:~# tcpdump -c 5 -n 'icmp[icmptype] != icmp-echo and icmp[icmptype] !=icmp-echoreply'**

```
We run this tcpdump command to capture the packets associated with traceroute
in a separate terminal and let tcpdump listen to traffic
from traceroute.

...
13:47:32.191851 IP 172.217.10.36 > 157.245.240.68: ICMP 172.217.10.36
udp port 33457 unreachable, length 36
13:47:32.191870 IP 172.217.10.36 > 157.245.240.68: ICMP 172.217.10.36
udp port 33458 unreachable, length 36
13:47:32.191880 IP 172.217.10.36 > 157.245.240.68: ICMP 172.217.10.36
udp port 33460 unreachable, length 36
13:47:32.191961 IP 172.217.10.36 > 157.245.240.68: ICMP 172.217.10.36
udp port 33461 unreachable, length 36
13:47:32.192008 IP 216.239.62.169 > 157.245.240.68: ICMP
time exceeded in-transit, length 68
...
```

2i.1. **root@laptop:~# tcpdump -c 5 -n src 192.168.42.85 and dst 10.2.1.40 and tcp**

```
 The tcpdump command that captures packets containing TCP
packets with a specific IP address as both source and destination.

...
12:19:00.048184 IP 192.168.42.85.40378 > 10.2.1.40.22: Flags [S], seq
1508922276, win 64240, options [mss 1460,sackOK,TS val 2718616759
ecr 0,nop,wscale 7], length 0
12:19:01.072902 IP 192.168.42.85.40378 > 10.2.1.40.22: Flags [S], seq
1508922276, win 64240, options [mss 1460,sackOK,TS val 2718617784
ecr 0,nop,wscale 7], length 0
12:19:03.088853 IP 192.168.42.85.40378 > 10.2.1.40.22: Flags [S], seq
1508922276, win 64240, options [mss 1460,sackOK,TS val 2718619800
ecr 0,nop,wscale 7], length 0
12:19:07.152860 IP 192.168.42.85.40378 > 10.2.1.40.22: Flags [S], seq
1508922276, win 64240, options [mss 1460,sackOK,TS val 2718623864
ecr 0,nop,wscale 7], length 0
12:19:15.344831 IP 192.168.42.85.40378 > 10.2.1.40.22: Flags [S], seq
1508922276, win 64240, options [mss 1460,sackOK,TS val 2718632057
ecr 0,nop,wscale 7], length 0
...
```

2i.2. **root@laptop:~# tcpdump -c 5 -n src 192.168.42.85 and tcp**

```
 The tcpdump command that captures packets containing TCP
packets with a specific IP address as only source.

...
12:20:05.008881 IP 192.168.42.85.40378 > 10.2.1.40.22: Flags [S], seq
1508922276, win 64240, options [mss 1460,sackOK,TS val 2718681726
ecr 0,nop,wscale 7], length 0
12:20:15.248865 IP 192.168.42.85.45062 > 18.211.35.165.443: Flags
[.], ack 891777754, win 501, options [nop,nop,TS val 979285033 ecr
119756224], length 0
12:20:15.717923 IP 192.168.42.85.40382 > 10.2.1.40.22: Flags [S], seq
801386797, win 64240, options [mss 1460,sackOK,TS val 2718692438
ecr 0,nop,wscale 7], length 0
12:20:16.724833 IP 192.168.42.85.40382 > 10.2.1.40.22: Flags [S], seq
801386797, win 64240, options [mss 1460,sackOK,TS val 2718693446
ecr 0,nop,wscale 7], length 0
12:20:18.736918 IP 192.168.42.85.40382 > 10.2.1.40.22: Flags [S], seq
801386797, win 64240, options [mss 1460,sackOK,TS val 2718695458
ecr 0,nop,wscale 7], length 0
...
```

2i.3. **root@laptop:~# tcpdump -c 5 -n dst 10.2.1.40 and tcp**

```
 The tcpdump command that captures packets containing TCP
packets with a specific IP address as only destination.

...
12:20:56.490963 IP 192.168.42.85.40386 > 10.2.1.40.22: Flags [S], seq
970139261, win 64240, options [mss 1460,sackOK,TS val 2718733222
ecr 0,nop,wscale 7], length 0
12:20:57.520853 IP 192.168.42.85.40386 > 10.2.1.40.22: Flags [S], seq
970139261, win 64240, options [mss 1460,sackOK,TS val 2718734253
ecr 0,nop,wscale 7], length 0
12:20:59.536852 IP 192.168.42.85.40386 > 10.2.1.40.22: Flags [S], seq
970139261, win 64240, options [mss 1460,sackOK,TS val 2718736269
ecr 0,nop,wscale 7], length 0
12:21:00.873843 IP 192.168.42.85.40388 > 10.2.1.40.22: Flags [S], seq
1753229232, win 64240, options [mss 1460,sackOK,TS val 2718737606
ecr 0,nop,wscale 7], length 0
12:21:01.904864 IP 192.168.42.85.40388 > 10.2.1.40.22: Flags [S], seq
1753229232, win 64240, options [mss 1460,sackOK,TS val 2718738638
ecr 0,nop,wscale 7], length 0
...
```

2j. **root@laptop:~# tcpdump -c 5 -n src 192.168.42.85 and dst 172.217.27.206 and icmp**

```
 The tcpdump command that captures packets containing ICMP
packets between two hosts with different IP addresses.

...
08:53:55.994743 IP 192.168.42.85 > 172.217.167.36: ICMP echo
request, id 7275, seq 349, length 64
08:53:56.995712 IP 192.168.42.85 > 172.217.167.36: ICMP echo
request, id 7275, seq 350, length 64
08:53:57.995890 IP 192.168.42.85 > 172.217.167.36: ICMP echo
request, id 7275, seq 351, length 64
08:53:58.996396 IP 192.168.42.85 > 172.217.167.36: ICMP echo
request, id 7275, seq 352, length 64
08:53:59.996976 IP 192.168.42.85 > 172.217.167.36: ICMP echo
request, id 7275, seq 353, length 64
...
```

2k. **root@laptop:~# tcpdump -n -c 5 src 192.168.42.85 and dst 10.2.1.40 and port 22**

```
 The tcpdump command to capture packets containing SSH request
and reply between two specific IP addresses
(using port number 22 for SSH)

...
08:56:32.904027 IP 192.168.42.85.52642 > 10.2.1.40.22: Flags [S], seq
3014443262, win 64240, options [mss 1460,sackOK,TS val 3916743373
ecr 0,nop,wscale 7], length 0
08:56:33.926133 IP 192.168.42.85.52642 > 10.2.1.40.22: Flags [S], seq
3014443262, win 64240, options [mss 1460,sackOK,TS val 3916744395
ecr 0,nop,wscale 7], length 0
08:56:35.942244 IP 192.168.42.85.52642 > 10.2.1.40.22: Flags [S], seq
3014443262, win 64240, options [mss 1460,sackOK,TS val 3916746411
ecr 0,nop,wscale 7], length 0
08:56:39.974163 IP 192.168.42.85.52642 > 10.2.1.40.22: Flags [S], seq
3014443262, win 64240, options [mss 1460,sackOK,TS val 3916750443
ecr 0,nop,wscale 7], length 0
08:56:48.166248 IP 192.168.42.85.52642 > 10.2.1.40.22: Flags [S], seq
3014443262, win 64240, options [mss 1460,sackOK,TS val 3916758635
ecr 0,nop,wscale 7], length 0
...
```

Note: *For 2i,2j,2k src IP is 192.168.42.85. It is the IP for 'eno1' for my local machine named 'laptop'. dst IP is 10.2.1.40 and it is the IP of the 'hamsa' server of the department. To generate traffic for 2i and 2j, I use the 'ping' command from a separate terminal and for 2k, I 'ssh' into 'hamsa' from a separate terminal. I could not get these commands to work for the cloud machine 'ubuntu-s-1vcpu-1gb-nyc1-01' which I have used for the rest of the exercises. I also did exercise 2g from the local machine 'laptop' rather than the cloud machine.*

Note: *Throughout the exercises, I have used the ellipses symbol(...) to indicate that those portions of the output are not significant or necessary for the demonstration purposes*