

## ASSIGNMENT 7: TCPDUMP

AIM: Study of Packet Sniffer tool TCPDUMP. Use it to capture and analyze the packet.

LO MAPPED: LO3

THEORY:

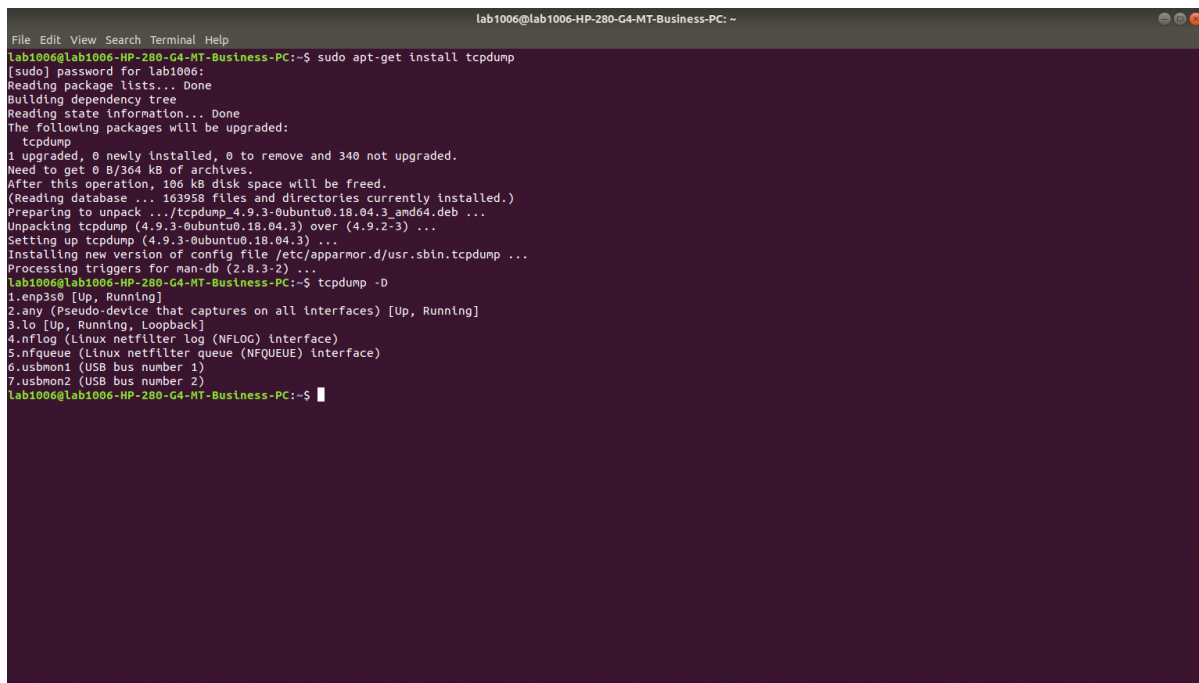
1. To install tcpdump

\$ sudo apt-get install tcpdump

2. Choosing an interface:

By default, tcpdump captures packets on all interfaces. To view a summary of available interfaces, run

**# tcpdump -D**

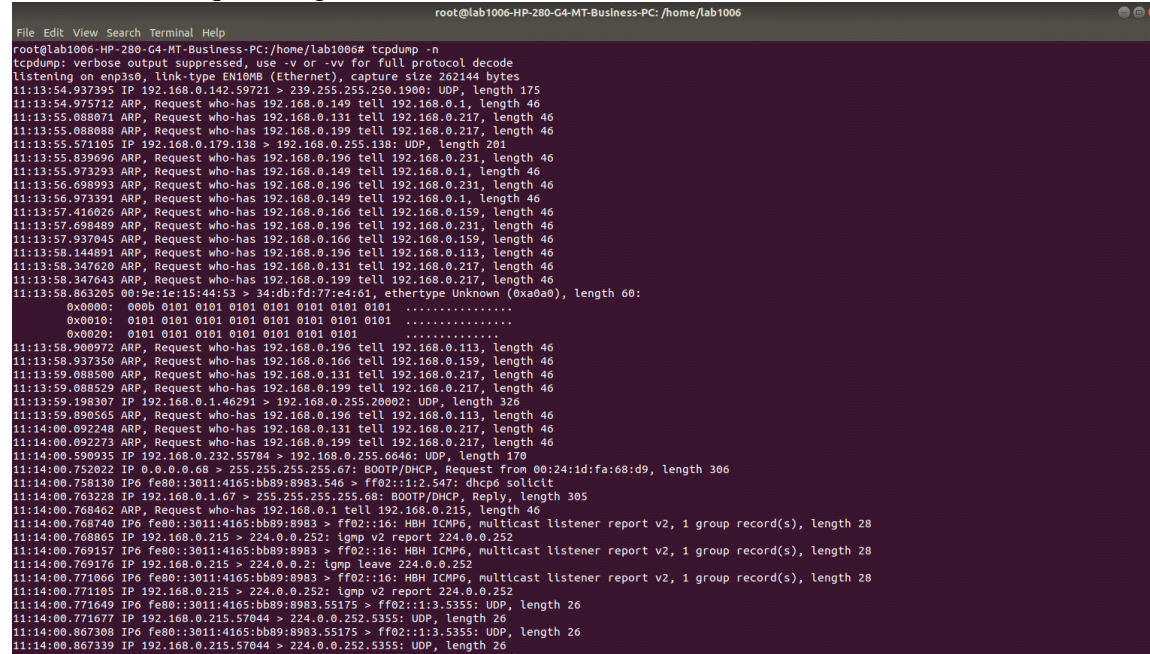


```
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~  
File Edit View Search Terminal Help  
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo apt-get install tcpdump  
[sudo] password for lab1006:  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following packages will be upgraded:  
  tcpdump  
1 upgraded, 0 newly installed, 0 to remove and 340 not upgraded.  
Need to get 0 B/364 kB of archives.  
After this operation, 106 kB disk space will be freed.  
(Reading database ... 163958 files and directories currently installed.)  
Preparing to unpack .../tcpdump_4.9.3-0ubuntu0.18.04.3_amd64.deb ...  
Unpacking tcpdump (4.9.3-0ubuntu0.18.04.3) over (4.9.2-3) ...  
Setting up tcpdump (4.9.3-0ubuntu0.18.04.3) ...  
Installing new version of config file /etc/apparmor.d/usr.sbin.tcpdump ...  
Processing triggers for man-db (2.8.3-2) ...  
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ tcpdump -D  
1.enp3s0 [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)  
7.usbmon2 (USB bus number 2)  
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$
```

3. Basic command for sniffing

**# tcpdump -n**

The -n parameter is given to stop tcpdump from resolving ip addresses to hostnames, which take look and not required right now.

A screenshot of a terminal window titled 'root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006'. The terminal shows the command 'tcpdump -n' being executed. The output displays network traffic in a compact format, including timestamps, IP addresses, ports, and protocol details. For example, it shows ARP requests and IP packets between 192.168.0.149 and 192.168.0.1. The output is truncated with '...' at the end of some lines.

Consider the output line

11:11:48.755303 IP 172.16.92.5.43780 > 106.10.184.41.443: Flags [..], ack 263857829, win 367, options [nop,nop,TS val 2072096 ecr 29009846], length 0

11:11:48.755303 is the time stamp with microsecond precision. Next is the protocol of the packet called IP (stands for Internet protocol and it is under this protocol that most of the internet communication goes on). Next is the source ip address joined with the source port. Following next is the destination port and then some information about the packet.

Now lets increase the display resolution of this packet, or get more details about it. The verbose switch comes in handy. Here is a quick example

#### 4. tcpdump -v -n

Now with the verbose switch lots of additional details about the packet are also being displayed. And these include the ttl, id, tcp flags, packet length etc.



```

root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006# tcpdump -n tcp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
14:18:55.930801 IP 192.168.0.202.50852 > 192.168.0.178.7680: Flags [S], seq 2671915345, win 64240, options [mss 1460,nop,wscale 8,nop,nop,sackOK], length 0
14:18:56.931321 IP 192.168.0.202.50852 > 192.168.0.178.7680: Flags [S], seq 2671915345, win 64240, options [mss 1460,nop,wscale 8,nop,nop,sackOK], length 0
14:18:58.937326 IP 192.168.0.202.50852 > 192.168.0.178.7680: Flags [S], seq 2671915345, win 64240, options [mss 1460,nop,wscale 8,nop,nop,sackOK], length 0
14:19:02.939518 IP 192.168.0.202.50852 > 192.168.0.178.7680: Flags [S], seq 2671915345, win 64240, options [mss 1460,nop,wscale 8,nop,nop,sackOK], length 0

```

# tcpdump -n icmp

## 7. Particular host or port

Expressions can be used to specify source ip, destination ip, and port numbers. The next example picks up all those packets with source address 172.16.92.1

# tcpdump -n src 172.16.92.1

```

root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006# tcpdump -n src 192.168.0.102
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
14:20:36.093475 IP 192.168.0.102 > 192.168.0.115: ICMP echo request, id 1, seq 13, length 40
14:20:37.105792 IP 192.168.0.102 > 192.168.0.115: ICMP echo request, id 1, seq 14, length 40
14:20:38.114367 IP 192.168.0.102 > 192.168.0.115: ICMP echo request, id 1, seq 15, length 40
14:20:39.119721 IP 192.168.0.102 > 192.168.0.115: ICMP echo request, id 1, seq 16, length 40
14:20:39.992971 ARP, Reply 192.168.0.102 is-at 04:0e:3c:1a:64:38, length 46
14:20:40.847446 ARP, Request who-has 192.168.0.115 (04:0e:3c:1a:5f:16) tell 192.168.0.102, length 46
14:20:41.398023 ARP, Request who-has 192.168.0.173 tell 192.168.0.102, length 46
14:20:42.335603 ARP, Request who-has 192.168.0.173 tell 192.168.0.102, length 46
14:20:43.335388 ARP, Request who-has 192.168.0.173 tell 192.168.0.102, length 46

```

# tcpdump -n dst 172.16.92.1

```

root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006# tcpdump -n dst 192.168.0.102
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
14:21:14.904295 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 152, length 64
14:21:15.928515 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 153, length 64
14:21:16.956367 ARP, Request who-has 192.168.0.102 tell 192.168.0.115, length 28
14:21:16.952466 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 154, length 64
14:21:17.976524 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 155, length 64
14:21:19.000392 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 156, length 64
14:21:20.024406 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 157, length 64
14:21:21.048405 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 158, length 64
14:21:22.072431 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 159, length 64
14:21:23.096504 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 160, length 64
14:21:24.120511 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 161, length 64
14:21:25.144396 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 162, length 64
14:21:26.168415 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 163, length 64
14:21:27.192271 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 164, length 64
14:21:28.216390 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 165, length 64
14:21:29.240391 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 166, length 64
14:21:30.264501 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 167, length 64
14:21:31.288405 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 168, length 64
14:21:32.312404 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 169, length 64
14:21:33.336405 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 170, length 64
14:21:34.360275 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 171, length 64
14:21:35.384314 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 172, length 64
14:21:36.408408 IP 192.168.0.115 > 192.168.0.102: ICMP echo request, id 3009, seq 173, length 64

```

**CONCLUSION:** In this assignment we have studied Packet Sniffer tool TCPDUMP. Used it to capture and analyze the packet.