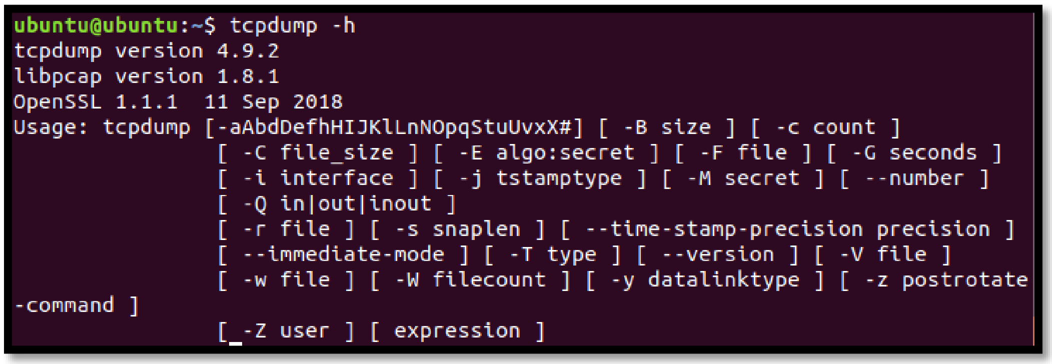
TCPDUMP

What is tcpdump?

tcpdump is a data-network analyzer program that runs in the command line interface. It allows us to capture and analyze traffic over a network to which the computer is connected. This tool is used to troubleshoot computer networks.

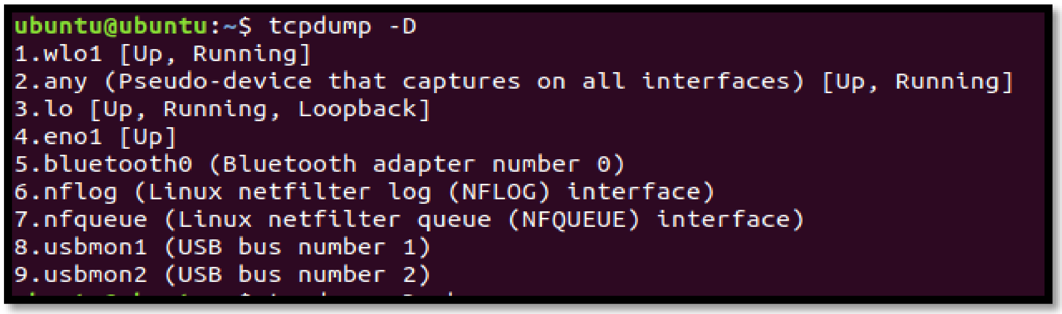
Using tcpdump:

tcpdump comes installed in Ubuntu (and several other Linux distributions). To see all the options available in tcpdump we type **tcpdump –h** in the command prompt.



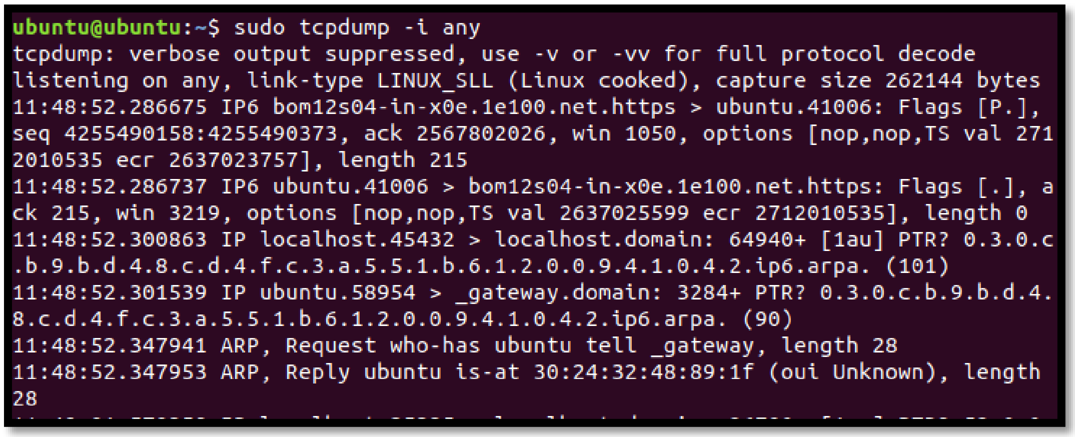
```
ubuntu@ubuntu:~$ tcpdump -h
tcpdump version 4.9.2
libpcap version 1.8.1
OpenSSL 1.1.1  11 Sep 2018
Usage: tcpdump [-aAbdDefhHIJKlLnNOpqStuUvxX#] [ -B size ] [ -c count ]
                [ -C file_size ] [ -E algo:secret ] [ -F file ] [ -G seconds ]
                [ -i interface ] [ -j tstamptype ] [ -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 ]
```

To check all the available interfaces for the packet capture we use the command **tcpdump -D**



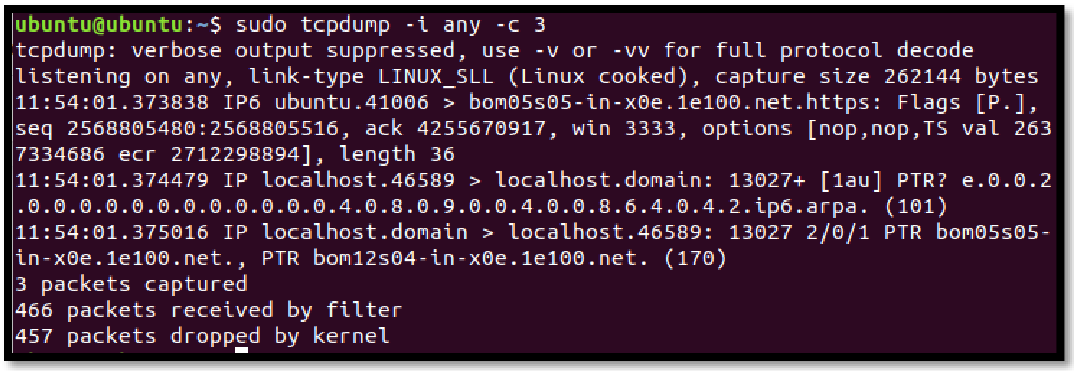
```
ubuntu@ubuntu:~$ tcpdump -D
1.wlo1 [Up, Running]
2.any (Pseudo-device that captures on all interfaces) [Up, Running]
3.lo [Up, Running, Loopback]
4.eno1 [Up]
5.bluetooth0 (Bluetooth adapter number 0)
6.nflog (Linux netfilter log (NFLOG) interface)
7.nfqueue (Linux netfilter queue (NFQUEUE) interface)
8.usbmon1 (USB bus number 1)
9.usbmon2 (USB bus number 2)
```

We can select an interface by using the **-i** option. To capture packets from all the interfaces we can use **tcpdump -i any**. This goes on capturing the packets until we hit control + c. And to capture packets, we need sudo permission so we use the keyword **sudo** at the start of the command.



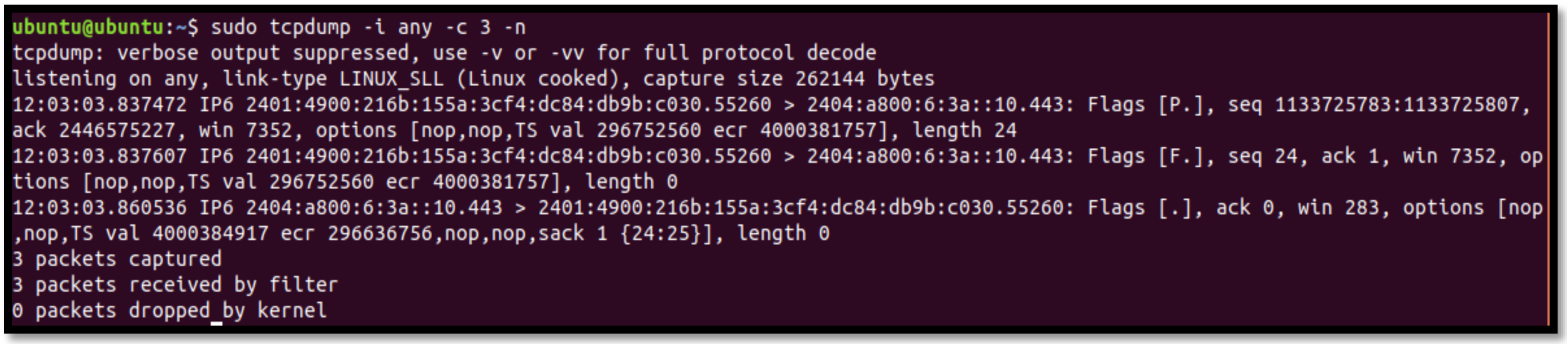
```
ubuntu@ubuntu:~$ sudo tcpdump -i any
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
11:48:52.286675 IP6 bom12s04-in-x0e.1e100.net.https > ubuntu.41006: Flags [P.],
seq 4255490158:4255490373, ack 2567802026, win 1050, options [nop,nop,TS val 271
2010535 ecr 2637023757], length 215
11:48:52.286737 IP6 ubuntu.41006 > bom12s04-in-x0e.1e100.net.https: Flags [.], a
ck 215, win 3219, options [nop,nop,TS val 2637025599 ecr 2712010535], length 0
11:48:52.300863 IP localhost.45432 > localhost.domain: 64940+ [1au] PTR? 0.3.0.c
.b.9.b.d.4.8.c.d.4.f.c.3.a.5.5.1.b.6.1.2.0.0.9.4.1.0.4.2.ip6.arpa. (101)
11:48:52.301539 IP ubuntu.58954 > _gateway.domain: 3284+ PTR? 0.3.0.c.b.9.b.d.4.
8.c.d.4.f.c.3.a.5.5.1.b.6.1.2.0.0.9.4.1.0.4.2.ip6.arpa. (90)
11:48:52.347941 ARP, Request who-has ubuntu tell _gateway, length 28
11:48:52.347953 ARP, Reply ubuntu is-at 30:24:32:48:89:1f (oui Unknown), length
28
```

To halt tcpdump after capturing a specific number of packets we use the lower case c "**-c**" option. To stop tcpdump after capturing 3 packets from any interface the command is **sudo tcpdump -i any -c 3**



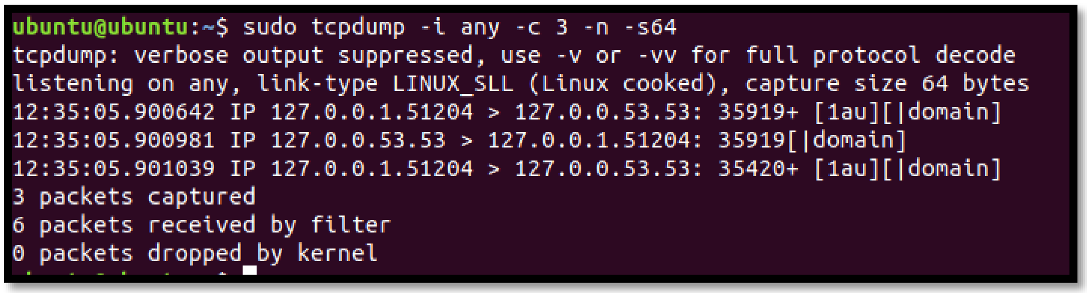
```
ubuntu@ubuntu:~$ sudo tcpdump -i any -c 3
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
11:54:01.373838 IP6 ubuntu.41006 > bom05s05-in-x0e.1e100.net.https: Flags [P.],
seq 2568805480:2568805516, ack 4255670917, win 3333, options [nop,nop,TS val 263
7334686 ecr 2712298894], length 36
11:54:01.374479 IP localhost.46589 > localhost.domain: 13027+ [1au] PTR? e.0.0.2
.0.0.0.0.0.0.0.0.0.0.0.0.4.0.8.0.9.0.0.4.0.0.8.6.4.0.4.2.ip6.arpa. (101)
11:54:01.375016 IP localhost.domain > localhost.46589: 13027 2/0/1 PTR bom05s05-
in-x0e.1e100.net., PTR bom12s04-in-x0e.1e100.net. (170)
3 packets captured
466 packets received by filter
457 packets dropped by kernel
```

In the above output, we can see that the IP addresses are resolved. The **-n** option prevents reverse DNS lookup. So we get IP addresses only. And the option **-v** (verbose) displays the full output. The options **-vv** and **-vvv** give even more detailed output.



```
ubuntu@ubuntu:~$ sudo tcpdump -i any -c 3 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
12:03:03.837472 IP6 2401:4900:216b:155a:3cf4:dc84:db9b:c030.55260 > 2404:a800:6:3a::10.443: Flags [P.], seq 1133725783:1133725807,
ack 2446575227, win 7352, options [nop,nop,TS val 296752560 ecr 4000381757], length 24
12:03:03.837607 IP6 2401:4900:216b:155a:3cf4:dc84:db9b:c030.55260 > 2404:a800:6:3a::10.443: Flags [F.], seq 24, ack 1, win 7352, op
tions [nop,nop,TS val 296752560 ecr 4000381757], length 0
12:03:03.860536 IP6 2404:a800:6:3a::10.443 > 2401:4900:216b:155a:3cf4:dc84:db9b:c030.55260: Flags [.], ack 0, win 283, options [nop
,nop,TS val 4000384917 ecr 296636756,nop,nop,sack 1 {24:25}], length 0
3 packets captured
3 packets received by filter
0 packets dropped by kernel
```

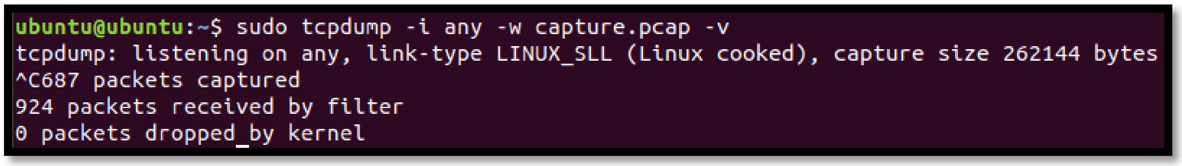
In the above output, we can see that the capture size is 262144 bytes i.e., more than the length of a packet. This means it captures a full packet. This size can be modified by option **-s**. To capture the header (maximum 60 bytes) and some data, we can set size as 64 bytes. Then the command is **sudo tcpdump -i any -c 3 -n -s 64**



```
ubuntu@ubuntu:~$ sudo tcpdump -i any -c 3 -n -s64
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 64 bytes
12:35:05.900642 IP 127.0.0.1.51204 > 127.0.0.53.53: 35919+ [1au][|domain]
12:35:05.900981 IP 127.0.0.53.53 > 127.0.0.1.51204: 35919[|domain]
12:35:05.901039 IP 127.0.0.1.51204 > 127.0.0.53.53: 35420+ [1au][|domain]
3 packets captured
6 packets received by filter
0 packets dropped by kernel
```

The output can be saved in a file using the option **-w**. To capture in a filename capture.pcap the command is **sudo tcpdump -i any -w capture.pcap -v**. The -v option is to know how many packets are captured.

Existing captured files can be read with the **-r** option. The same output will be displayed as packets were directly seen on the screen.



```
ubuntu@ubuntu:~$ sudo tcpdump -i any -w capture.pcap -v
tcpdump: listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
^C687 packets captured
924 packets received by filter
0 packets dropped by kernel
```

Understanding the output:

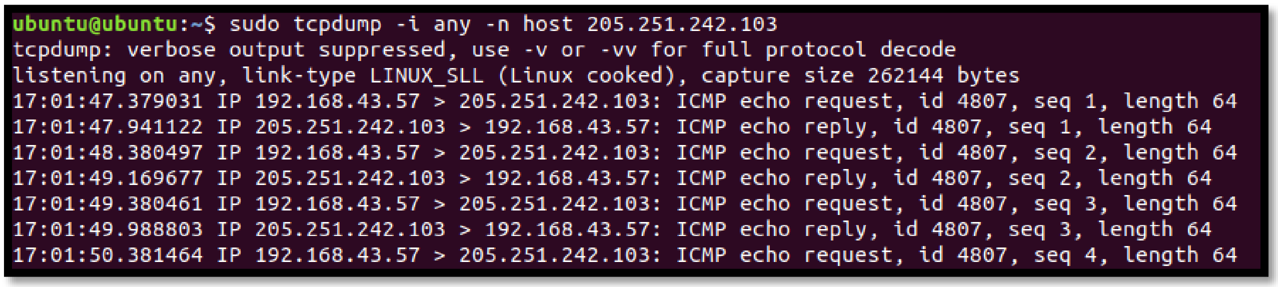
The output line indicates the following

- First is the time field, this is the timestamp of the received packet as per the local clock.
- The next field represents whether it is IPv4 or IPv6, if it is IPv4 then it displays IP or if it is IPv6 then it displays IP6.
- Then the source IP address and the source port number followed by the destination IP and the destination port number.
- After that, we can find the flags, which represent TCP flags.
    - S is for connection start
    - F is for connection finish
    - P is for data push
    - R is for connection reset
    - '.' is for acknowledgment
- Next is the sequence number. For the first packet captured it is the absolute number and the subsequent packets use the relative number.
- The next field is the acknowledgment number. The sender has ack 1 and for the receiver, the ack number is the next expected byte.
- Next is the window size (win <some number>) that represents the number of bytes available in the receiving buffer.

- ❏ The next field is the TCP options such as MSS, window scaling, selective acknowledgments, timestamps.
- ❏ Next is the packet length. This represents the length in bytes of the payload data.

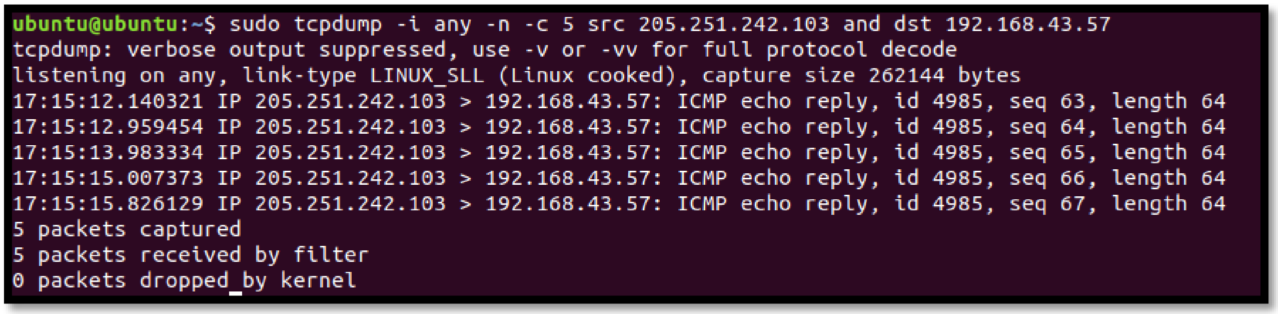
Filters:

Filters are used to see the traffic that we are interested in and ignore the rest.
Host keyword: Using the host keyword followed by an IP address, we can filter the traffic that is going to or from the given IP address.



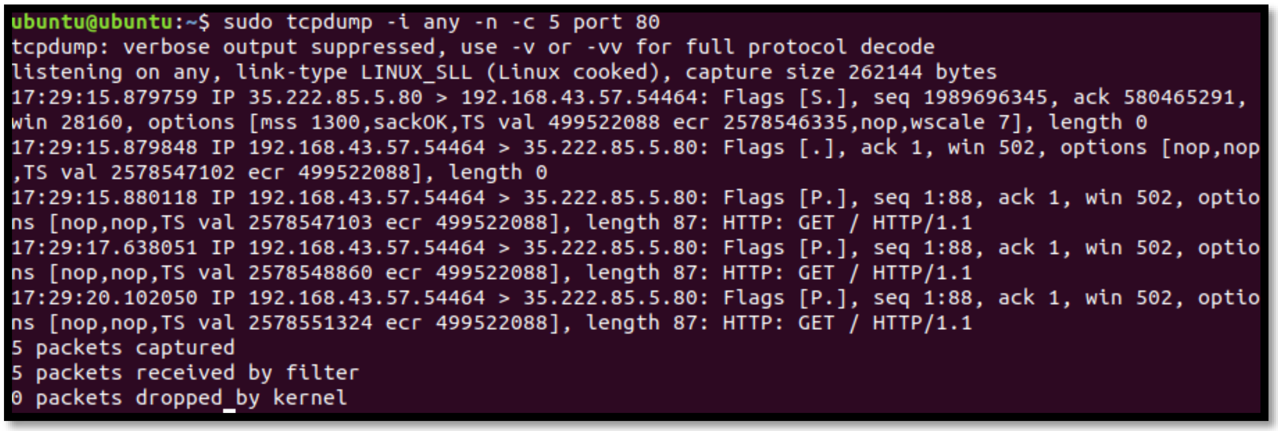
```
ubuntu@ubuntu:~$ sudo tcpdump -i any -n host 205.251.242.103
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
17:01:47.379031 IP 192.168.43.57 > 205.251.242.103: ICMP echo request, id 4807, seq 1, length 64
17:01:47.941122 IP 205.251.242.103 > 192.168.43.57: ICMP echo reply, id 4807, seq 1, length 64
17:01:48.380497 IP 192.168.43.57 > 205.251.242.103: ICMP echo request, id 4807, seq 2, length 64
17:01:49.169677 IP 205.251.242.103 > 192.168.43.57: ICMP echo reply, id 4807, seq 2, length 64
17:01:49.380461 IP 192.168.43.57 > 205.251.242.103: ICMP echo request, id 4807, seq 3, length 64
17:01:49.988803 IP 205.251.242.103 > 192.168.43.57: ICMP echo reply, id 4807, seq 3, length 64
17:01:50.381464 IP 192.168.43.57 > 205.251.242.103: ICMP echo request, id 4807, seq 4, length 64
```

src and dst keyword: we can filter the traffic that is going from an IP using **src <IP address>** or we can filter traffic that is going to an IP using **dst <IP address>** or we can filter the traffic going from an IP address to an Ip address using **src <Ip address> and dst <IP address>**



```
ubuntu@ubuntu:~$ sudo tcpdump -i any -n -c 5 src 205.251.242.103 and dst 192.168.43.57
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
17:15:12.140321 IP 205.251.242.103 > 192.168.43.57: ICMP echo reply, id 4985, seq 63, length 64
17:15:12.959454 IP 205.251.242.103 > 192.168.43.57: ICMP echo reply, id 4985, seq 64, length 64
17:15:13.983334 IP 205.251.242.103 > 192.168.43.57: ICMP echo reply, id 4985, seq 65, length 64
17:15:15.007373 IP 205.251.242.103 > 192.168.43.57: ICMP echo reply, id 4985, seq 66, length 64
17:15:15.826129 IP 205.251.242.103 > 192.168.43.57: ICMP echo reply, id 4985, seq 67, length 64
5 packets captured
5 packets received by filter
0 packets dropped by kernel
```

Similar to the src and dst keywords we have a port keyword. With this keyword, only traffic in the specified port will be filtered.



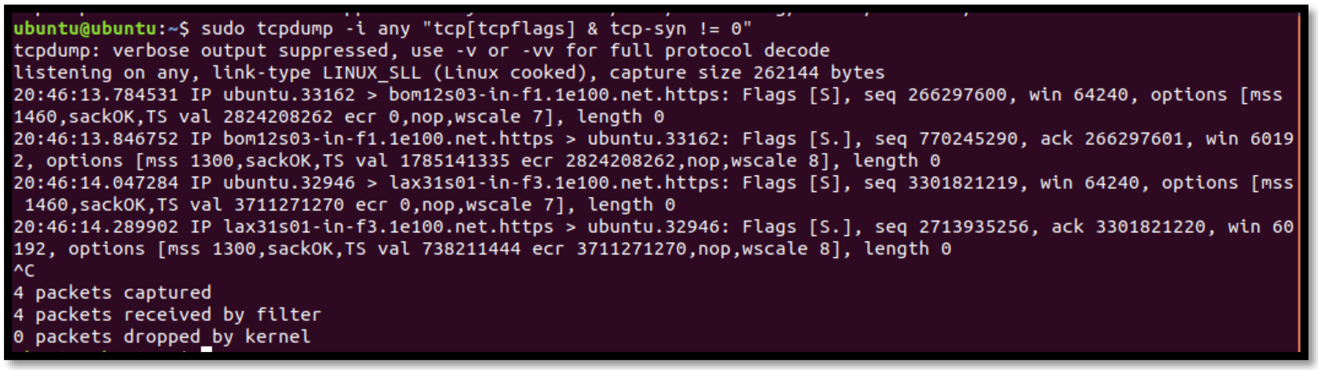
```
ubuntu@ubuntu:~$ sudo tcpdump -i any -n -c 5 port 80
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
17:29:15.879759 IP 35.222.85.5.80 > 192.168.43.57.54464: Flags [S.], seq 1989696345, ack 580465291,
win 28160, options [mss 1300,sackOK,TS val 499522088 ecr 2578546335,nop,wscale 7], length 0
17:29:15.879848 IP 192.168.43.57.54464 > 35.222.85.5.80: Flags [.], ack 1, win 502, options [nop,nop
,TS val 2578547102 ecr 499522088], length 0
17:29:15.880118 IP 192.168.43.57.54464 > 35.222.85.5.80: Flags [P.], seq 1:88, ack 1, win 502, optio
ns [nop,nop,TS val 2578547103 ecr 499522088], length 87: HTTP: GET / HTTP/1.1
17:29:17.638051 IP 192.168.43.57.54464 > 35.222.85.5.80: Flags [P.], seq 1:88, ack 1, win 502, optio
ns [nop,nop,TS val 2578548860 ecr 499522088], length 87: HTTP: GET / HTTP/1.1
17:29:20.102050 IP 192.168.43.57.54464 > 35.222.85.5.80: Flags [P.], seq 1:88, ack 1, win 502, optio
ns [nop,nop,TS val 2578551324 ecr 499522088], length 87: HTTP: GET / HTTP/1.1
5 packets captured
5 packets received by filter
0 packets dropped by kernel
```

Compound expressions:

We can apply multiple filters in the same expression. For example the expression **sudo tcpdump -i any -n “host 192.168.1.91 and (port 80 or port 443)”** captures all the packets from or to the IP 192.168.1.91 and through the port number 80 or 443.

With the option -e we can get the mac address. With the ip6 keyword, we can filter only IPv6 traffic.

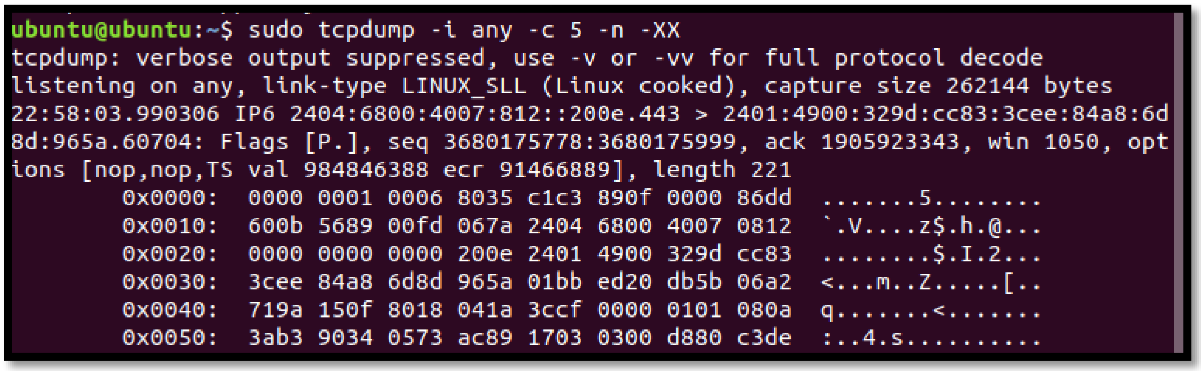
Similarly, we can filter the packets based on the TCP flags. The expression **sudo tcpdump -i any “tcp[tcpflags] & tcp-syn != 0”** filters only packets that have TCP ‘S’ flag.



```
ubuntu@ubuntu:~$ sudo tcpdump -i any "tcp[tcpflags] & tcp-syn != 0"
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
20:46:13.784531 IP ubuntu.33162 > bom12s03-in-f1.1e100.net.https: Flags [S], seq 266297600, win 64240, options [mss
1460,sackOK,TS val 2824208262 ecr 0,nop,wscale 7], length 0
20:46:13.846752 IP bom12s03-in-f1.1e100.net.https > ubuntu.33162: Flags [S.], seq 770245290, ack 266297601, win 6019
2, options [mss 1300,sackOK,TS val 1785141335 ecr 2824208262,nop,wscale 8], length 0
20:46:14.047284 IP ubuntu.32946 > lax31s01-in-f3.1e100.net.https: Flags [S], seq 3301821219, win 64240, options [mss
 1460,sackOK,TS val 3711271270 ecr 0,nop,wscale 7], length 0
20:46:14.289902 IP lax31s01-in-f3.1e100.net.https > ubuntu.32946: Flags [S.], seq 2713935256, ack 3301821220, win 60
192, options [mss 1300,sackOK,TS val 738211444 ecr 3711271270,nop,wscale 8], length 0
^C
4 packets captured
4 packets received by filter
0 packets dropped by kernel
```

And to capture all the packets with TCP flag **r** we write **sudo tcpdump -i any “tcp[tcpflags] & tcp-rst != 0”**

With the -XX option we can get more about the packet in hex and ASCII format.



```
ubuntu@ubuntu:~$ sudo tcpdump -i any -c 5 -n -XX
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
22:58:03.990306 IP6 2404:6800:4007:812::200e.443 > 2401:4900:329d:cc83:3cee:84a8:6d
8d:965a.60704: Flags [P.], seq 3680175778:3680175999, ack 1905923343, win 1050, opt
ions [nop,nop,TS val 984846388 ecr 91466889], length 221
        0x0000:  0000 0001 0006 8035 c1c3 890f 0000 86dd  .......5........
        0x0010:  600b 5689 00fd 067a 2404 6800 4007 0812  `.V....z$.h.@...
        0x0020:  0000 0000 0000 200e 2401 4900 329d cc83  ........$.I.2...
        0x0030:  3cee 84a8 6d8d 965a 01bb ed20 db5b 06a2  <...m..Z.....[..
        0x0040:  719a 150f 8018 041a 3ccf 0000 0101 080a  q.......<.......
        0x0050:  3ab3 9034 0573 ac89 1703 0300 d880 c3de  :..4.s..........
```

Similarly, if we use the lowercase -xx option we get only hex data. We have -A option to get data in ASCII

There are many more options in the tcpdump. tcpdump is a very useful tool in analyzing and troubleshooting computer networks.