21/01/2024

Telecommunication Software

Report 1

Task 1 – Page 2

Task 1.1 – Page 2

Task 1.2- Page 7

Task 1.3 – Page 13

Task 2 – Page 17

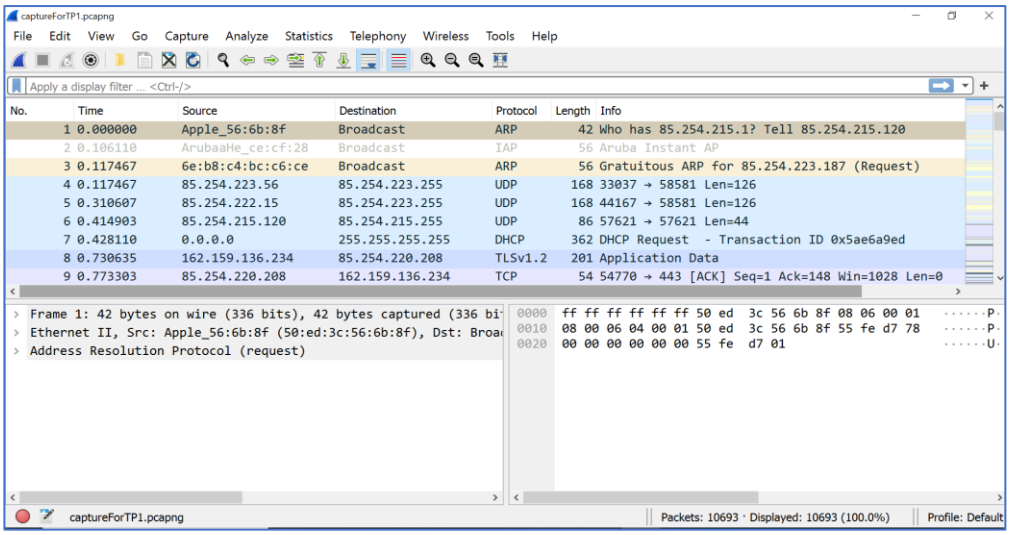
Jehanne Sarrazin

230AEM053

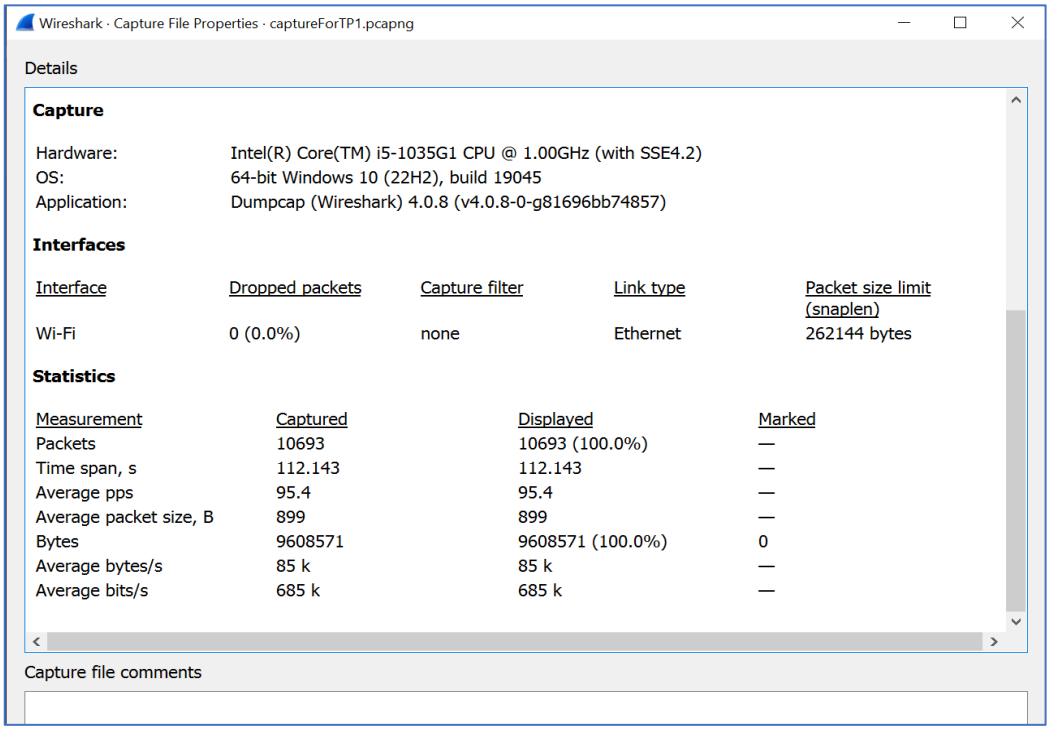
TASK 1 - 1)

My main work support for this task is the pdf “Packet Sniffing and Wireshark Guidebook 02”. In it, I filled all the tables of values required to complete the work. I will be adding screenshots of the completed tables here to ensure a clear report, with additional screenshots and comments I made that I felt should be present in the work.

First, after making a capture in Wireshark, this is what my capture looks like:



3.1) After going to Statistics/Capture File Properties, we obtain this screen:



Which allows us to fill the table 3.1 of the report as follows:

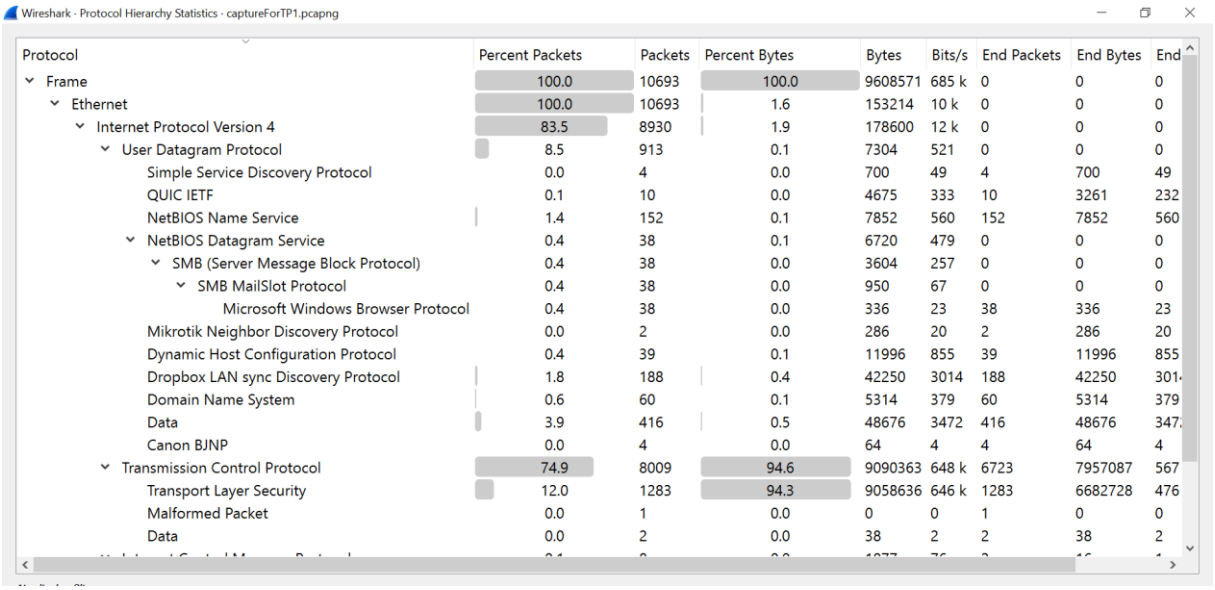
3.1. Capture File Properties

Fill in the table. For initial data use the **Statistics/Capture File Properties**.

Nr	Parametr	Value
1	Time of capture, min	01:52
2	Packets	10693
3	Bytes, MiB	9608571 bytes ~ 9.16 MiB
4	Average packet size, B	899
5	Average packets per seconds, pps	95.4
6	Average bytes per seconds, B/s	85k

7. Determine the relative network load L (in%) for the control period T by formula:
L = (Traffic [Mbits] / T [sec]) / (Bandwidth [Mbits/sec])
Bandwidth = 100 Mbits/sec
L = Your Ansver 9608571 bytes ~ 76.9 Mbits and 1:52min = 112sec so :
L = (76.9/112)/100 = 6.87 x10^(-3)

3.2) Next, after going to Statistics/Protocol Hierarchy, we obtain this screen:



Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End
Frame	100.0	10693	100.0	9608571	685 k	0	0	0
Ethernet	100.0	10693	1.6	153214	10 k	0	0	0
Internet Protocol Version 4	83.5	8930	1.9	178600	12 k	0	0	0
User Datagram Protocol	8.5	913	0.1	7304	521	0	0	0
Simple Service Discovery Protocol	0.0	4	0.0	700	49	4	700	49
QUIC IETF	0.1	10	0.0	4675	333	10	3261	232
NetBIOS Name Service	1.4	152	0.1	7852	560	152	7852	560
NetBIOS Datagram Service	0.4	38	0.1	6720	479	0	0	0
SMB (Server Message Block Protocol)	0.4	38	0.0	3604	257	0	0	0
SMB MailSlot Protocol	0.4	38	0.0	950	67	0	0	0
Microsoft Windows Browser Protocol	0.4	38	0.0	336	23	38	336	23
Mikrotik Neighbor Discovery Protocol	0.0	2	0.0	286	20	2	286	20
Dynamic Host Configuration Protocol	0.4	39	0.1	11996	855	39	11996	855
Dropbox LAN sync Discovery Protocol	1.8	188	0.4	42250	3014	188	42250	301
Domain Name System	0.6	60	0.1	5314	379	60	5314	379
Data	3.9	416	0.5	48676	3472	416	48676	347
Canon BJNP	0.0	4	0.0	64	4	4	64	4
Transmission Control Protocol	74.9	8009	94.6	9090363	648 k	6723	7957087	567
Transport Layer Security	12.0	1283	94.3	9058636	646 k	1283	6682728	476
Malformed Packet	0.0	1	0.0	0	0	1	0	0
Data	0.0	2	0.0	38	2	2	38	2

And I can fill the table 3.2 accordingly:

It is interesting to note that in my case, the sum of traffic in % doesn't add up to a 100%, perhaps because other types of protocols are used.

3.2. Ethernet Traffic Distribution by Protocols

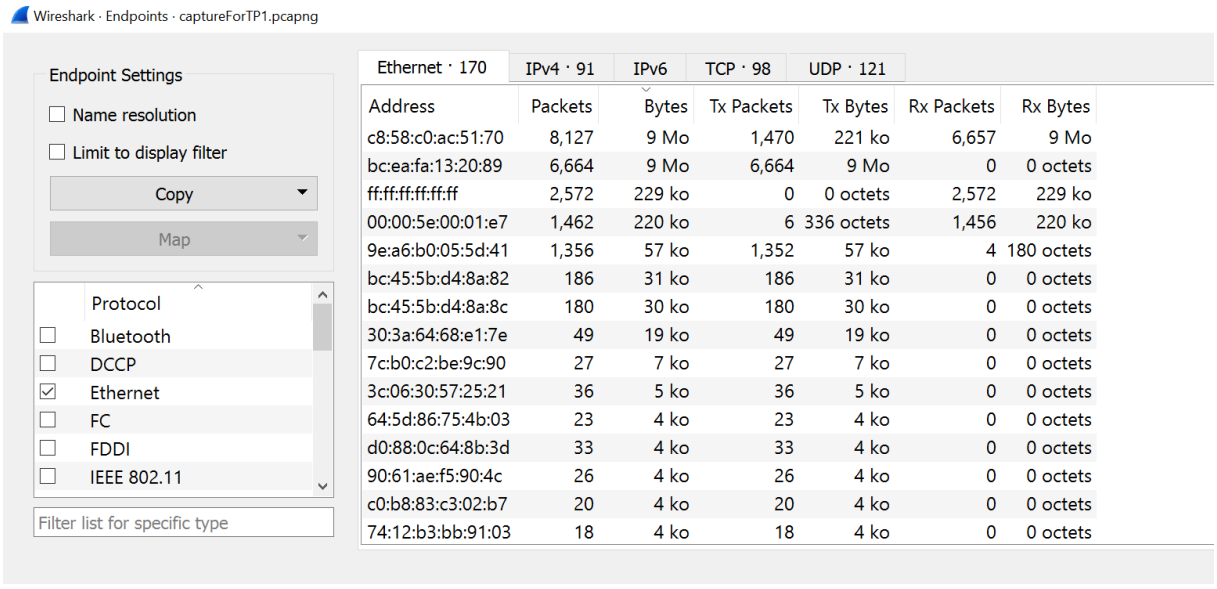
Fill in the table. For initial data use the **Statistics/Protocol Hierarchy**.

Nr	Protocol	Traffic, MiB	Traffic, %
1	IPv6	/	/
2	IPv4	0.17	83.5
3	--UDP	0.00697	8.5
4	--TCP	8.67	74.9
5	--ICMP	0.00103	0.1
6	ARP	0.0487	16.3
7	802.1X	/	/
	SUMM	8.90	83.5 ~~100~~

8. What is the ratio of the numbers of application (http, mail, ftp, …) to numbers of service (dns, icmp, arp, …) protocols?

Anr / Snr = Your Ansver 16/7 = 2.29

3.3) In Statistics/Endpoints/Ethernet we access this screen:



Which gives us this output for the 3.3 table:

3.3. Ethernet Traffic Distribution by Nodes

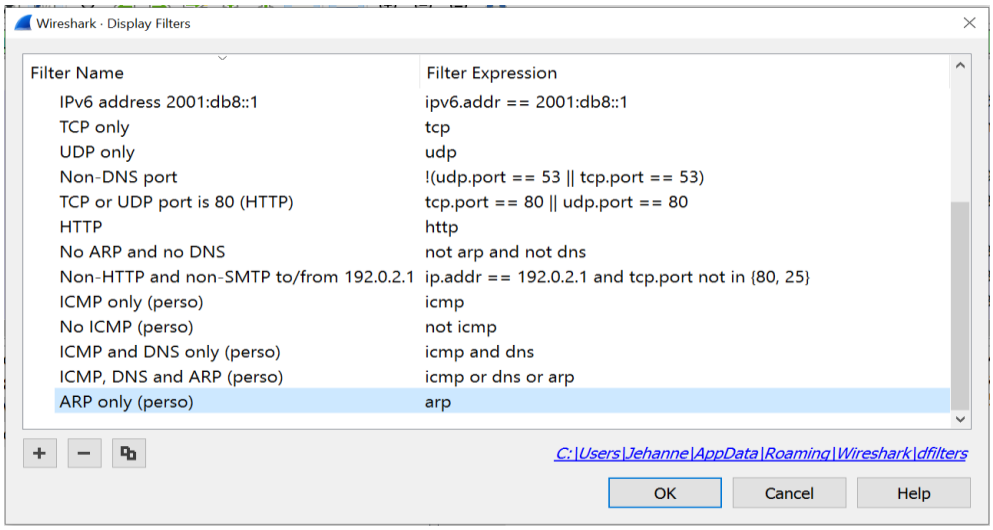
Fill in the table (for the 5 most active network nodes by Bytes). For initial data use the **Statistics/Endpoints/Ethernet**.

Nr	MAC-address	IP- address	Traffic					
			Rx input		Tx output		Overall	
			MiB	%	MiB	%	MiB	%
1.	c8:58:c0:ac:51:70	162.159.136.234	8.6	93	0.21	2	8.6	48
2.	bc:ea:fa:13:20:89	162.159.136.234	0	0	8.6	97	8.6	48
3.	ff:ff:ff:ff:ff:ff	0.0.0.0	0.22	2	0	0	0.22	1
4.	00:00:5e:00:01:e7	85.254.220.208	0.21	2	~0	0	0.21	1
5.	9e:a6:b0:05:5d:41	85.254.220.113	0.17	2	0.05	1	0.17	1
		SUM	9.2	**100**	8.86	**100**	17.8	**100**

6. Which IP nodes are the most loaded, given the direction of traffic?
Incoming – 162.159.136.234
Outgoing – 162.159.136.234
Overall – 162.159.136.234

As you can see in the screenshot, the IP address doesn't appear in the data. To find the corresponding IP address for each MAC address I simply filtered each individual MAC address on the mainstream and collected the corresponding IP address.

3.4) I accessed the filter table and added the last 6 filters:



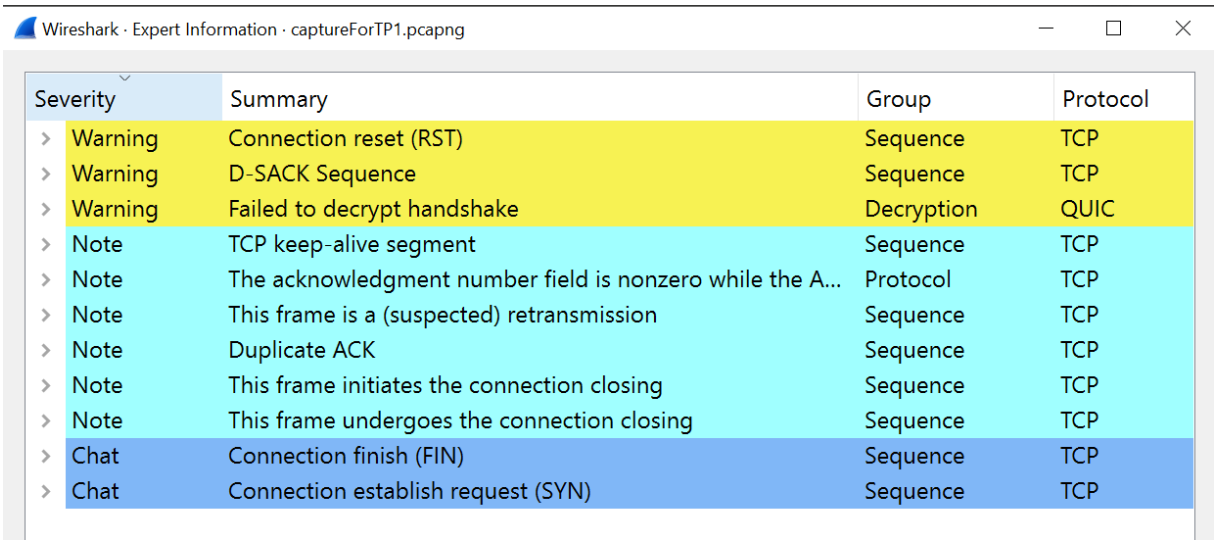
And gave their signification in my table 3.4:

3.4. Display Filters

Fill in the table. Write and test in Wireshark 5 simple search filters (Display Filters) using AND, OR, NO to display packets from (to) a specific node generated by ICMP, DNS, ARP requests (responses) when accessing any server of your choice.

Nr	Display Filter	Description
1	icmp	Displays ICMP only
2	not icmp	Display all but ICMP
3	icmp and dns	Dispay ICMP and DNS only
4	icmp or dns or arp	Display all ICMP, DNS and ARP
5	arp	Display ARP

3.5) Analyze/Expert Information gives us this screen:



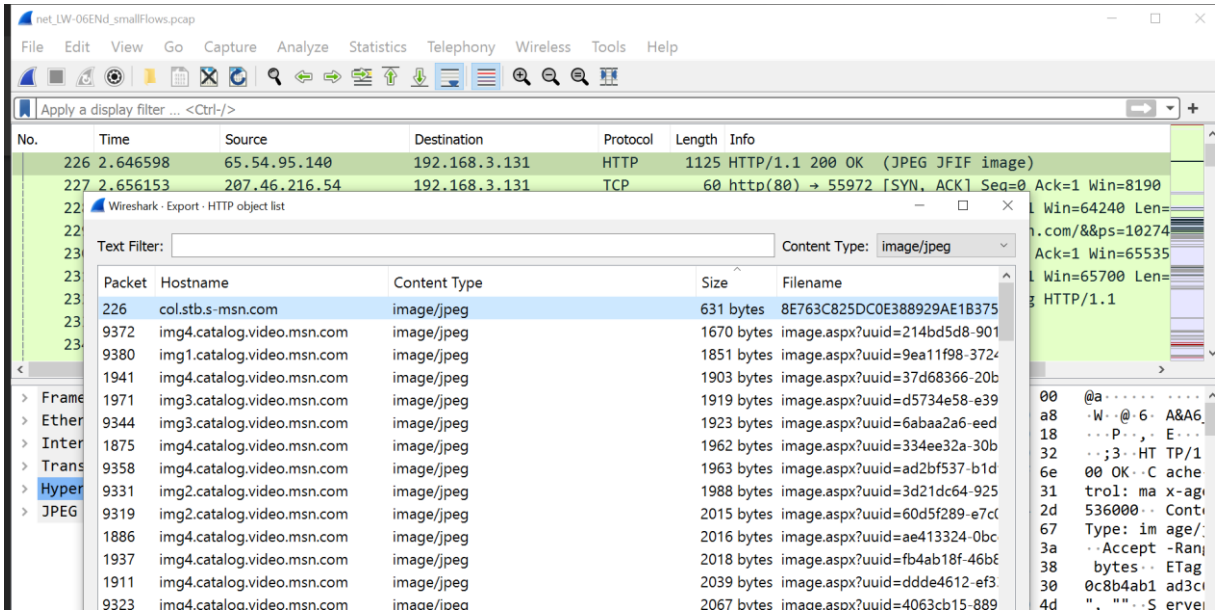
And here is their explanation in table 3.5:

3.5. Network Problem Analyze

Analyze the 5 note/warning/error problems existing on the network. Find and read information about network problems on the Internet.
For initial data use the **Analyze/Expert Information**.

Nr	Expert Information	Severity	Your Short Description (Problem Analyse)
1	Connection reset (RST)	Warning	A problem in TCP communication led to a connexion reset
2	TCP keep-alive segment	Note	Prevent a TCP connexion from timing out
3	... Reassembly error	Error	A problem was detected with the way a network manipulates the data
4	Failed to decrypt handshake	Warning	Couldn't decrypt the TLS handshake between client and server
5	D-SACK sequence	Warning	A duplicate segement was detected

3.6) This exercise was the most interesting in my opinion. This exercise was to be realized using our own capture, unfortunately, there wasn't a jpeg file in the capture I made. So, I downloaded the original capture file that the document was referring to and collected the biggest image (as we have no attributed number in the class). Here is a screenshot of the different files found in the pcap file:



And the corresponding image I found:

A motorcycle ->

2)

For this task I downloaded WSL (Ubuntu) to do terminal exercises. They will include the following tools: ip, ifconfig, route, netstat, ping, nslookup, traceroute, wget, iwlist and iwconfig.

I will provide every time screenshot of what I did, and some explanation of the command.

IP:

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ ip
Usage: ip [ OPTIONS ] OBJECT { COMMAND | help }
       ip [ -force ] -batch filename
where  OBJECT := { address | addrlabel | fou | help | ila | ioam | l2tp | link |
                   macsec | maddress | monitor | mptcp | mroute | mrule |
                   neighbor | neighbour | netconf | netns | nexthop | ntable |
                   ntbl | route | rule | sr | tap | tcpmetrics |
                   token | tunnel | tuntap | vrf | xfrm }
       OPTIONS := { -V[ersion] | -s[tatistics] | -d[etails] | -r[esolve] |
                    -h[uman-readable] | -iec | -j[son] | -p[retty] |
                    -f[amily] { inet | inet6 | mpls | bridge | link } |
                    -4 | -6 | -M | -B | -0 |
                    -l[oops] { maximum-addr-flush-attempts } | -br[ief] |
                    -o[neline] | -t[imestamp] | -ts[hort] | -b[atch] [filename] |
                    -rc[vbuf] [size] | -n[etns] name | -N[umeric] | -a[ll] |
                    -c[olor]}
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$
```

The ip command is used to to display and modify the network configuration of the system.

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:15:5d:da:14:11 brd ff:ff:ff:ff:ff:ff
    inet 172.25.108.20/20 brd 172.25.111.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::215:5dff:feda:1411/64 scope link
       valid_lft forever preferred_lft forever
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$
```

Ip address is used to list all the network interfaces and their IP addresses.

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ ip route show
default via 172.25.96.1 dev eth0 proto kernel
172.25.96.0/20 dev eth0 proto kernel scope link src 172.25.108.20
```

Similarly, ip route show shows the routing table within the system.

IFCONFIG :

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.25.108.20  netmask 255.255.240.0  broadcast 172.25.111.255
        inet6 fe80::215:5dff:feda:1411  prefixlen 64  scopeid 0x20<link>
        ether 00:15:5d:da:14:11  txqueuelen 1000  (Ethernet)
        RX packets 65466  bytes 95993503 (95.9 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8309  bytes 713451 (713.4 KB)
        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
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

The ifconfig command is used to display and modify the network configuration of the system. It is similar to ip, but it is older and less powerful.

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.25.108.20  netmask 255.255.240.0  broadcast 172.25.111.255
        inet6 fe80::215:5dff:feda:1411  prefixlen 64  scopeid 0x20<link>
        ether 00:15:5d:da:14:11  txqueuelen 1000  (Ethernet)
        RX packets 65477  bytes 95995515 (95.9 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8309  bytes 713451 (713.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ sudo ifconfig eth0 down
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ sudo ifconfig eth0
eth0: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        inet 172.25.108.20  netmask 255.255.240.0  broadcast 172.25.111.255
        ether 00:15:5d:da:14:11  txqueuelen 1000  (Ethernet)
        RX packets 65477  bytes 95995515 (95.9 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8309  bytes 713451 (713.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ sudo ifconfig eth0 up
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ eth0
^[[B^[[Beth0: command not found
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.25.108.20  netmask 255.255.240.0  broadcast 172.25.111.255
        inet6 fe80::215:5dff:feda:1411  prefixlen 64  scopeid 0x20<link>
        ether 00:15:5d:da:14:11  txqueuelen 1000  (Ethernet)
        RX packets 65477  bytes 95995515 (95.9 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8317  bytes 714107 (714.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

One can also use ifconfig eth0 up or ifconfig eth0 down to enable or disable an interface (here eth0).

ROUTE :

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ route help
Usage: route [-nNvee] [-FC] [<AF>]          List kernel routing tables
       route [-v] [-FC] {add|del|flush} ...  Modify routing table for AF.

       route {-h|--help} [<AF>]             Detailed usage syntax for specified AF.
       route {-V|--version}                 Display version/author and exit.

        -v, --verbose            be verbose
        -n, --numeric            don't resolve names
        -e, --extend             display other/more information
        -F, --fib                display Forwarding Information Base (default)
        -C, --cache              display routing cache instead of FIB

  <AF>=Use -4, -6, '-A <af>' or '--<af>'; default: inet
  List of possible address families (which support routing):
    inet (DARPA Internet) inet6 (IPv6) ax25 (AMPR AX.25)
    netrom (AMPR NET/ROM) ipx (Novell IPX) ddp (Appletalk DDP)
    x25 (CCITT X.25)
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.25.96.0     0.0.0.0         255.255.240.0   U     0      0        0 eth0
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ route -C
Kernel IP routing cache
Source          Destination     Gateway         Flags Metric Ref    Use Iface
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ route -F
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.25.96.0     0.0.0.0         255.255.240.0   U     0      0        0 eth0
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ route -F -e
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
172.25.96.0     0.0.0.0         255.255.240.0   U         0 0          0 eth0
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$
```

The route command is used to display and modify the routing table of the system. For example, I can use route -n to show the routing table in numeric format, -C to see the cache, -F to see the forwarding information base and -e to display more information about the routing tables.

With add and del, the route table can also be modified in the system.

NETSTAT :

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ netstat -t
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ netstat -i
Kernel Interface table
Iface      MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0      1500    65512      0      0 0          8321      0      0      0 BMRU
lo       65536        0      0      0 0             0      0      0      0 LRU
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ]         DGRAM                    19458    /var/run/chrony/chronyd.sock
unix  2      [ ]         DGRAM                    843      /run/user/1000/systemd/notify
unix  3      [ ]         DGRAM      CONNECTED     22299    /run/systemd/notify
unix  2      [ ]         DGRAM                    22308    /run/systemd/journal/syslog
unix  9      [ ]         DGRAM      CONNECTED     22315    /run/systemd/journal/dev-log
unix  7      [ ]         DGRAM      CONNECTED     22317    /run/systemd/journal/socket
unix  3      [ ]         STREAM     CONNECTED     589
unix  2      [ ]         DGRAM      CONNECTED     717
unix  3      [ ]         STREAM     CONNECTED     17993
unix  3      [ ]         STREAM     CONNECTED     18045    /run/systemd/journal/stdout
unix  3      [ ]         STREAM     CONNECTED     23225
unix  2      [ ]         STREAM     CONNECTED     23139
unix  2      [ ]         DGRAM      CONNECTED     27735
unix  3      [ ]         STREAM     CONNECTED     23259    /run/dbus/system_bus_socket
unix  3      [ ]         STREAM     CONNECTED     22445
unix  3      [ ]         STREAM     CONNECTED     24617
unix  3      [ ]         STREAM     CONNECTED     20005    /run/systemd/journal/stdout
unix  3      [ ]         STREAM     CONNECTED     24010
unix  3      [ ]         STREAM     CONNECTED     24087    /run/dbus/system_bus_socket
unix  3      [ ]         STREAM     CONNECTED     666
unix  3      [ ]         STREAM     CONNECTED     24226
unix  2      [ ]         DGRAM      CONNECTED     19360
```

Netstat is used to display various information about the network connections, routing tables, interface statistics and others. For example, one can use netstat -t to show the TCP connections, or netstat -i to show the interface statistics.

PING :

Ping of course is used to test the connectivity and latency between a system and another host on the network or the internet. Here for example, I try to ping google.com. I also use ping -c 5 google.com to ping Google's server five times and then stop.

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ ping google.com
PING google.com (142.250.74.14) 56(84) bytes of data.
64 bytes from arn09s21-in-f14.1e100.net (142.250.74.14): icmp_seq=1 ttl=56 time=13.9 ms
64 bytes from arn09s21-in-f14.1e100.net (142.250.74.14): icmp_seq=2 ttl=56 time=22.5 ms
64 bytes from arn09s21-in-f14.1e100.net (142.250.74.14): icmp_seq=3 ttl=56 time=17.5 ms
64 bytes from arn09s21-in-f14.1e100.net (142.250.74.14): icmp_seq=4 ttl=56 time=15.7 ms
64 bytes from arn09s21-in-f14.1e100.net (142.250.74.14): icmp_seq=5 ttl=56 time=11.7 ms
^C
--- google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 11.708/16.268/22.476/3.656 ms
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ ping -c 5 google.com
PING google.com (142.250.74.14) 56(84) bytes of data.
64 bytes from arn09s21-in-f14.1e100.net (142.250.74.14): icmp_seq=1 ttl=56 time=12.3 ms
64 bytes from arn09s21-in-f14.1e100.net (142.250.74.14): icmp_seq=2 ttl=56 time=11.9 ms
64 bytes from arn09s21-in-f14.1e100.net (142.250.74.14): icmp_seq=3 ttl=56 time=12.7 ms
64 bytes from arn09s21-in-f14.1e100.net (142.250.74.14): icmp_seq=4 ttl=56 time=13.1 ms
64 bytes from arn09s21-in-f14.1e100.net (142.250.74.14): icmp_seq=5 ttl=56 time=12.9 ms
```

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ ping -a -4 -c 4 google.com
PING google.com (142.250.74.110) 56(84) bytes of data.
64 bytes from arn11s10-in-f14.1e100.net (142.250.74.110): icmp_seq=1 ttl=56 time=14.9 ms
64 bytes from arn11s10-in-f14.1e100.net (142.250.74.110): icmp_seq=2 ttl=56 time=12.3 ms
64 bytes from arn11s10-in-f14.1e100.net (142.250.74.110): icmp_seq=3 ttl=56 time=11.8 ms
64 bytes from arn11s10-in-f14.1e100.net (142.250.74.110): icmp_seq=4 ttl=56 time=12.3 ms

--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 11.780/12.835/14.894/1.209 ms
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ ping -6 -c 4 google.com
ping: connect: Network is unreachable
```

NSLOOKUP :

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ nslookup google.com
Server:         172.25.96.1
Address:        172.25.96.1#53

Non-authoritative answer:
Name:   google.com
Address: 142.250.74.110
Name:   google.com
Address: 2a00:1450:400f:803::200e

jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ nslookup 8.8.8.8
8.8.8.8.in-addr.arpa    name = dns.google.

Authoritative answers can be found from:

jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ nslookup -type=mx google.com
Server:         172.25.96.1
Address:        172.25.96.1#53

Non-authoritative answer:
google.com      mail exchanger = 10 smtp.google.com.

Authoritative answers can be found from:

jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$
```

Nslookup is used to query DNS servers for information about domain names and IP addresses. For example, here I can look up the ip address of google.com or 8.8.8.8. I also use nslookup -type=mx google.com to find out the mail exchange servers for Google's domain name.

TRACEROUTE :

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ traceroute google.com
traceroute to google.com (142.250.74.78), 64 hops max
  1   172.25.96.1  0.438ms  0.290ms  0.330ms
  2   192.168.2.1  1.848ms  1.871ms  2.098ms
  3   192.168.0.1  2.099ms  3.330ms  5.184ms
  4   89.201.0.1  3.555ms  3.266ms  3.858ms
  5   10.220.1.205  3.757ms  3.144ms  2.975ms
  6   213.252.199.197  3.052ms  4.745ms  3.624ms
  7   *  *  *
  8   213.226.130.115  16.960ms  13.726ms  14.739ms
  9   *  *  *
 10   142.251.48.42  14.540ms  13.649ms  15.463ms
 11   142.251.65.81  13.268ms  15.897ms  13.333ms
 12   142.250.74.78  14.581ms  13.201ms  14.989ms
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ traceroute -I google.com
traceroute to google.com (142.250.74.78), 64 hops max
  1   172.25.96.1  0.646ms  0.582ms  0.375ms
  2   192.168.2.1  15.189ms  3.091ms  2.868ms
  3   192.168.0.1  3.614ms  4.613ms  3.675ms
  4   89.201.0.1  5.673ms  4.050ms  3.138ms
  5   10.220.1.205  2.810ms  4.072ms  3.083ms
  6   213.252.199.197  3.033ms  2.859ms  4.953ms
  7   *  *  *
  8   213.226.130.115  13.037ms  13.931ms  13.378ms
  9   108.170.233.55  13.073ms  12.308ms  12.578ms
 10   142.251.65.83  13.106ms  13.378ms  12.256ms
 11   142.250.74.78  11.522ms  12.654ms  11.460ms
```

As the name implies, traceroute is used to trace the route that packets take from one system to another host on the network or the internet. I take once again the example of google.com here and use traceroute -I google.com to use ICMP packets instead of UDP packets for tracing.

WGET:

Wget is used to download files from the internet or a local network using HTTP, HTTPS, or FTP protocols.

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ sudo wget https://www.bing.com
--2023-10-29 11:49:51--  https://www.bing.com/
Resolving www.bing.com (www.bing.com)... 139.45.207.74, 139.45.207.81, 139.45.207.88, ...
Connecting to www.bing.com (www.bing.com)|139.45.207.74|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/html]
index.html: Permission denied

Cannot write to ‘index.html’ (Success).
```

Unfortunately, here I kept having a permission denied for everything I tried. This will be updated later If I find a solution.

IWLIST:

The iwlist command is used to display information about wireless networks that are available or accessible by the system’s wireless interfaces. For example, you can use iwlist wlan0 scan to scan for wireless networks using your wlan0 interface, or iwlist wlan0 channel to show the available channels for your wlan0 interface.

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ sudo iwlist scan
[sudo] password for jehanne:
lo        Interface doesn't support scanning.

eth0      Interface doesn't support scanning.

jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ sudo iwlist channel
lo        no frequency information.

eth0      no frequency information.

jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ sudo iwlist rate
lo        no bit-rate information.

eth0      no bit-rate information.
```

Unfortunately, is seems the wireless interface of my system isn't responding as expected same goes for the next command:

IWCONFIG:

The iwconfig command to display and modify the wireless configuration of your system's wireless interfaces.

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ iwconfig
lo        no wireless extensions.

eth0      no wireless extensions.
```

I will try other solutions later to understand why this isn't working here.

3) Try to do Tcpdump and Tshark terminals practical exercises with different options and compound command combinations. Study libpcap expression syntax format.

For this task I downloaded TCPDump and TShark within WSL.

TCPDump is a command line-based packet tracer. Here are some commands as used to get familiar with it:

First of course, a view of what's available with --help

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ tcpdump --help
tcpdump version 4.99.1
libpcap version 1.10.1 (with TPACKET_V3)
OpenSSL 3.0.2 15 Mar 2022
Usage: tcpdump [-AbdDefhHIJKlLnNOpqStuUvxX#] [ -B size ] [ -c count ] [--count]
                [ -C file_size ] [ -E algo:secret ] [ -F file ] [ -G seconds ]
                [ -i interface ] [ --immediate-mode ] [ -j tstamptype ]
                [ -M secret ] [ --number ] [ --print ] [ -Q in|out|inout ]
                [ -r file ] [ -s snaplen ] [ -T type ] [ --version ]
                [ -V file ] [ -w file ] [ -W filecount ] [ -y datalinktype ]
                [ --time-stamp-precision precision ] [ --micro ] [ --nano ]
                [ -z postrotate-command ] [ -Z user ] [ expression ]
```

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ sudo tcpdump -i eth0 -v
[sudo] password for jehanne:
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
11:13:03.873978 IP (tos 0x0, ttl 4, id 5627, offset 0, flags [none], proto UDP (17), length 165)
    LAPTOP-S2BNJRGD.mshome.net.58555 > 239.255.255.250.1900: UDP, length 137
11:13:03.889641 IP (tos 0x0, ttl 64, id 8841, offset 0, flags [DF], proto UDP (17), length 74)
    172.25.108.20.56963 > LAPTOP-S2BNJRGD.mshome.net.domain: 7778+ PTR? 250.255.255.239.in-addr.arpa. (46)
11:13:03.926480 IP (tos 0x0, ttl 1, id 22283, offset 0, flags [none], proto UDP (17), length 80)
    LAPTOP-S2BNJRGD.mshome.net.mdns > mdns.mcast.net.mdns: 0 PTR (QM)? 250.255.255.239.in-addr.arpa.local. (52)
11:13:03.926921 IP6 (flowlabel 0x82ec7, hlim 1, next-header UDP (17) payload length: 60) LAPTOP-S2BNJRGD.mdns > f
.mdns: [udp sum ok] 0 PTR (QM)? 250.255.255.239.in-addr.arpa.local. (52)
11:13:04.942397 IP (tos 0x0, ttl 1, id 22284, offset 0, flags [none], proto UDP (17), length 80)
    LAPTOP-S2BNJRGD.mshome.net.mdns > mdns.mcast.net.mdns: 0 PTR (QM)? 250.255.255.239.in-addr.arpa.local. (52)
11:13:04.943162 IP6 (flowlabel 0x82ec7, hlim 1, next-header UDP (17) payload length: 60) LAPTOP-S2BNJRGD.mdns > f
.mdns: [udp sum ok] 0 PTR (QM)? 250.255.255.239.in-addr.arpa.local. (52)
11:13:04.951928 IP (tos 0x0, ttl 128, id 40867, offset 0, flags [none], proto UDP (17), length 131)
    LAPTOP-S2BNJRGD.mshome.net.domain > 172.25.108.20.56963: 7778 NXDomain 0/1/0 (103)
11:13:04.952303 IP (tos 0x0, ttl 64, id 50595, offset 0, flags [DF], proto UDP (17), length 70)
    172.25.108.20.41293 > LAPTOP-S2BNJRGD.mshome.net.domain: 39998+ PTR? 1.96.25.172.in-addr.arpa. (42)
11:13:04.954447 IP (tos 0x0, ttl 128, id 40868, offset 0, flags [none], proto UDP (17), length 134)
    LAPTOP-S2BNJRGD.mshome.net.domain > 172.25.108.20.41293: 39998- 1/0/0 1.96.25.172.in-addr.arpa. PTR LAPTOP-S2
mshome.net. (106)
11:13:04.954902 IP (tos 0x0, ttl 64, id 9687, offset 0, flags [DF], proto UDP (17), length 72)
```

I then used -i (interface) and -v (verbose) to have a first detailed view.

➔ We can see that there are numerous packets show, so I continue with using filters (here filtering packets whose host is 172.25.96.1).

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ sudo tcpdump -i eth0 -v host 172.25.96.1
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
11:18:02.330153 IP (tos 0x0, ttl 1, id 5657, offset 0, flags [none], proto UDP (17), length 203)
    LAPTOP-S2BNJRGD.mshome.net.58561 > 239.255.255.250.1900: UDP, length 175
11:18:02.346070 IP (tos 0x0, ttl 1, id 5658, offset 0, flags [none], proto UDP (17), length 203)
    LAPTOP-S2BNJRGD.mshome.net.58564 > 239.255.255.250.1900: UDP, length 175
11:18:02.396755 IP (tos 0x0, ttl 64, id 54040, offset 0, flags [DF], proto UDP (17), length 74)
    172.25.108.20.53401 > LAPTOP-S2BNJRGD.mshome.net.domain: 61826+ PTR? 250.255.255.239.in-addr.arpa. (46)
11:18:02.432380 IP (tos 0x0, ttl 1, id 22300, offset 0, flags [none], proto UDP (17), length 80)
    LAPTOP-S2BNJRGD.mshome.net.mdns > mdns.mcast.net.mdns: 0 PTR (QM)? 250.255.255.239.in-addr.arpa.local. (52)
11:18:03.334411 IP (tos 0x0, ttl 1, id 5659, offset 0, flags [none], proto UDP (17), length 203)
    LAPTOP-S2BNJRGD.mshome.net.58561 > 239.255.255.250.1900: UDP, length 175
11:18:03.349185 IP (tos 0x0, ttl 1, id 5660, offset 0, flags [none], proto UDP (17), length 203)
    LAPTOP-S2BNJRGD.mshome.net.58564 > 239.255.255.250.1900: UDP, length 175
11:18:03.443654 IP (tos 0x0, ttl 1, id 22301, offset 0, flags [none], proto UDP (17), length 80)
    LAPTOP-S2BNJRGD.mshome.net.mdns > mdns.mcast.net.mdns: 0 PTR (QM)? 250.255.255.239.in-addr.arpa.local. (52)
```

And those from which it is the source with src:

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ sudo tcpdump -i eth0 -v src 172.25.96.1
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
11:33:48.973293 IP (tos 0x0, ttl 128, id 5406, offset 0, flags [none], proto UDP (17), length 131)
    LAPTOP-S2BNJRGD.mshome.net.domain > 172.25.108.20.53079: 43271 NXDomain 0/1/0 (103)
11:33:48.996847 IP (tos 0x0, ttl 128, id 5407, offset 0, flags [none], proto UDP (17), length 131)
    LAPTOP-S2BNJRGD.mshome.net.domain > 172.25.108.20.53079: 43271 NXDomain 0/1/0 (103)
11:33:49.010141 IP (tos 0x0, ttl 1, id 44502, offset 0, flags [none], proto UDP (17), length 78)
    LAPTOP-S2BNJRGD.mshome.net.mdns > mdns.mcast.net.mdns: 0 PTR (QM)? 20.108.25.172.in-addr.arpa.local. (50)
11:33:49.040328 IP (tos 0x0, ttl 1, id 44503, offset 0, flags [none], proto UDP (17), length 78)
    LAPTOP-S2BNJRGD.mshome.net.mdns > mdns.mcast.net.mdns: 0 PTR (QM)? 20.108.25.172.in-addr.arpa.local. (50)
11:33:49.782230 IP (tos 0x0, ttl 128, id 5408, offset 0, flags [none], proto UDP (17), length 72)
    LAPTOP-S2BNJRGD.mshome.net.domain > 172.25.108.20.51879: 60847 NXDomain 0/0/0 (44)
11:33:49.810301 IP (tos 0x0, ttl 128, id 5409, offset 0, flags [none], proto UDP (17), length 149)
    LAPTOP-S2BNJRGD.mshome.net.domain > 172.25.108.20.51879: 60847 NXDomain 0/1/0 (121)
11:33:50.013065 IP (tos 0x0, ttl 1, id 44504, offset 0, flags [none], proto UDP (17), length 78)
    LAPTOP-S2BNJRGD.mshome.net.mdns > mdns.mcast.net.mdns: 0 PTR (QM)? 20.108.25.172.in-addr.arpa.local. (50)
11:33:50.044382 IP (tos 0x0, ttl 1, id 44505, offset 0, flags [none], proto UDP (17), length 78)
    LAPTOP-S2BNJRGD.mshome.net.mdns > mdns.mcast.net.mdns: 0 PTR (QM)? 20.108.25.172.in-addr.arpa.local. (50)
11:33:50.050745 IP (tos 0x0, ttl 128, id 5410, offset 0, flags [none], proto UDP (17), length 72)
    LAPTOP-S2BNJRGD.mshome.net.domain > 172.25.108.20.34148: 64477 NXDomain 0/0/0 (44)
```

dst can also be used to find the packets that have this destination.

TSHARK:

TShark is a command line version of Wireshark.

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ tshark -h
TShark (Wireshark) 3.6.2 (Git v3.6.2 packaged as 3.6.2-2)
Dump and analyze network traffic.
See https://www.wireshark.org for more information.

Usage: tshark [options] ...

Capture interface:
  -i <interface>, --interface <interface>
                           name or idx of interface (def: first non-loopback)
  -f <capture filter>      packet filter in libpcap filter syntax
  -s <snaplen>, --snapshot-length <snaplen>
                           packet snapshot length (def: appropriate maximum)
  -p, --no-promiscuous-mode
                           don't capture in promiscuous mode
  -I, --monitor-mode       capture in monitor mode, if available
  -B <buffer size>, --buffer-size <buffer size>
                           size of kernel buffer (def: 2MB)
  -y <link type>, --linktype <link type>
                           link layer type (def: first appropriate)
  --time-stamp-type <type> timestamp method for interface
  -D, --list-interfaces    print list of interfaces and exit
  -L, --list-data-link-types
```

Following this same logic, I start with using -h to get a view of the possibilities, and start with -D to see the list if interfaces.

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ tshark -D
1. ciscodump (Cisco remote capture)
2. dpauxmon (DisplayPort AUX channel monitor capture)
3. randpkt (Random packet generator)
4. sdjournal (systemd Journal Export)
5. sshdump (SSH remote capture)
6. udpdump (UDP Listener remote capture)
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$
```

We can see that there aren't many interfaces. As an admin however, more are available:

```
jehanne@LAPTOP-S2BNJRGD:~$ sudo tshark -D
[sudo] password for jehanne:
Running as user "root" and group "root". This could be dangerous.
1. eth0
2. any
3. lo (Loopback)
4. bluetooth-monitor
5. nflog
6. nfqueue
7. dbus-system
8. dbus-session
9. ciscodump (Cisco remote capture)
10. dpauxmon (DisplayPort AUX channel monitor capture)
11. randpkt (Random packet generator)
12. sdjournal (systemd Journal Export)
13. sshdump (SSH remote capture)
14. udpdump (UDP Listener remote capture)
jehanne@LAPTOP-S2BNJRGD:~$
```

We can now start capturing:

```
jehanne@LAPTOP-S2BNJRGD:~$ sudo tshark -i eth0
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
 ** (tshark:956) 14:11:28.225649 [Main MESSAGE] -- Capture started.
 ** (tshark:956) 14:11:28.225792 [Main MESSAGE] -- File: "/tmp/wireshark_eth0YUFUD2.pcapng"
^Z
[9]+  Stopped                 sudo tshark -i eth0
jehanne@LAPTOP-S2BNJRGD:~$
```

And to have a cleaner result, set parameters such as the capture of 5 packets (-c) then save them (-w) under a given pcap name:

```
jehanne@LAPTOP-S2BNJRGD:/mnt/c/WINDOWS/system32$ cd /tmp
jehanne@LAPTOP-S2BNJRGD:/tmp$ sudo tshark -i eth0 -c 5 -w test.pcap
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
 ** (tshark:1159) 14:14:55.848258 [Main MESSAGE] -- Capture started.
 ** (tshark:1159) 14:14:55.848381 [Main MESSAGE] -- File: "test.pcap"
5
jehanne@LAPTOP-S2BNJRGD:/tmp$
```

We can now read the file that we just captured using -r:

```
jehanne@LAPTOP-S2BNJRGD:/tmp$ sudo tshark -r test.pcap
Running as user "root" and group "root". This could be dangerous.
    1 0.000000000  172.25.96.1 → 172.25.111.255 NBNS 92 Name query NB ETH0<00>
    2 0.000746255  172.25.96.1 → 224.0.0.251  MDNS 70 Standard query 0x0000 A eth0.local, "QM" question
    3 0.001990366 fe80::2a93:984e:5adf:350d → ff02::fb     MDNS 90 Standard query 0x0000 A eth0.local, "QM" question
    4 0.002724843  172.25.96.1 → 224.0.0.251  MDNS 70 Standard query 0x0000 AAAA eth0.local, "QM" question
    5 0.003167684 fe80::2a93:984e:5adf:350d → ff02::fb     MDNS 90 Standard query 0x0000 AAAA eth0.local, "QM" question
jehanne@LAPTOP-S2BNJRGD:/tmp$
```

Or read it under a different format, such as json for example:


```
jehanne@LAPTOP-S2BNJRGD:/tmp$ sudo tshark -r test.pcap -T json
Running as user "root" and group "root". This could be dangerous.
[
  {
    "_index": "packets-2023-11-04",
    "_type": "doc",
    "_score": null,
    "_source": {
      "layers": {
        "frame": {
          "frame.interface_id": "0",
          "frame.interface_id_tree": {
            "frame.interface_name": "eth0"
          },
          "frame.encap_type": "1",
          "frame.time": "Nov  4, 2023 14:15:08.848971182 CET",
          "frame.offset_shift": "0.000000000",
          "frame.time_epoch": "1699103708.848971182",
          "frame.time_delta": "0.000000000",
          "frame.time_delta_displayed": "0.000000000",
          "frame.time_relative": "0.000000000",
          "frame.number": "1",
          "frame.len": "92",
          "frame.cap_len": "92",
          "frame.marked": "0",
          "frame.ignored": "0",
          "frame.protocols": "eth:ethertype:ip:udp:nbns"
        },
        "eth": {
          "eth.dst": "ff:ff:ff:ff:ff:ff",
          "eth.dst_tree": {
            "eth.dst_resolved": "Broadcast",
            "eth.dst.oui": "16777215",
            "eth.addr": "ff:ff:ff:ff:ff:ff",
            "eth.addr_resolved": "Broadcast",
```


We can also apply filters while capturing using -f:

```
jehanne@LAPTOP-S2BNJRGD:/tmp$ sudo tshark -i eth0 -f "tcp port 80"
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
 ** (tshark:1210) 14:20:59.922890 [Main MESSAGE] -- Capture started.
 ** (tshark:1210) 14:20:59.923011 [Main MESSAGE] -- File: "/tmp/wireshark_eth0C2M3D2.pcapng"
```

Task 2 :

Following on the next page, the pdf version of the jupyter notebook used to run some exercises on python using numpy, matplotlib and pandas. The actual notebook is in the folder “TASK 2”. For every practical task you can also find all the files in my GitHub repository: https://github.com/Niennaaa/TelecommunicationSoftware

In [1]:
```
## Import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:
```
## Import CSV with pandas
data = pd.read_csv("C:\\Users\\Jehanne\\Desktop\\RTU\\Telecommunication Software\\R
```

In [6]:
```
## Let's analyze the data roughly

print("\n\n\n###### Shape : ")
print(data.shape)

print("\n\n\n###### Columns : ")
print(data.columns)

print("\n\n\n###### Their type : ")
pd.set_option('display.max_rows', None) #by default it will only show a handful so
print(data.dtypes)
pd.set_option('display.max_rows',10)

print("\n\n\n###### First values : ")
print(data.head())

print("\n\n\n###### Statistics : ")
print(data.describe())
```

```
###### Shape :
(4234, 65)



###### Columns :
Index(['id_flow', 'nw_src', 'tp_src', 'nw_dst', 'tp_dst', 'nw_proto',
       'forward_pc', 'forward_bc', 'forward_pl', 'forward_piat', 'forward_pps',
       'forward_bps', 'forward_pl_mean', 'forward_piat_mean',
       'forward_pps_mean', 'forward_bps_mean', 'forward_pl_var',
       'forward_piat_var', 'forward_pps_var', 'forward_bps_var',
       'forward_pl_q1', 'forward_pl_q3', 'forward_piat_q1', 'forward_piat_q3',
       'forward_pl_max', 'forward_pl_min', 'forward_piat_max',
       'forward_piat_min', 'forward_pps_max', 'forward_pps_min',
       'forward_bps_max', 'forward_bps_min', 'forward_duration',
       'forward_size_packets', 'forward_size_bytes', 'reverse_pc',
       'reverse_bc', 'reverse_pl', 'reverse_piat', 'reverse_pps',
       'reverse_bps', 'reverse_pl_mean', 'reverse_piat_mean',
       'reverse_pps_mean', 'reverse_bps_mean', 'reverse_pl_var',
       'reverse_piat_var', 'reverse_pps_var', 'reverse_bps_var',
       'reverse_pl_q1', 'reverse_pl_q3', 'reverse_piat_q1', 'reverse_piat_q3',
       'reverse_pl_max', 'reverse_pl_min', 'reverse_piat_max',
       'reverse_piat_min', 'reverse_pps_max', 'reverse_pps_min',
       'reverse_bps_max', 'reverse_bps_min', 'reverse_duration',
       'reverse_size_packets', 'reverse_size_bytes', 'category'],
      dtype='object')



###### Their type :
id_flow                 object
nw_src                  object
tp_src                   int64
nw_dst                  object
tp_dst                   int64
nw_proto                 int64
forward_pc               int64
forward_bc               int64
forward_pl             float64
forward_piat           float64
forward_pps            float64
forward_bps            float64
forward_pl_mean        float64
forward_piat_mean      float64
forward_pps_mean       float64
forward_bps_mean       float64
forward_pl_var         float64
forward_piat_var       float64
forward_pps_var        float64
forward_bps_var         object
forward_pl_q1          float64
forward_pl_q3          float64
forward_piat_q1        float64
forward_piat_q3        float64
```

```
forward_pl_max          float64
forward_pl_min          float64
forward_piat_max        float64
forward_piat_min        float64
forward_pps_max         float64
forward_pps_min         float64
forward_bps_max         float64
forward_bps_min         float64
forward_duration          int64
forward_size_packets      int64
forward_size_bytes        int64
reverse_pc                int64
reverse_bc              float64
reverse_pl              float64
reverse_piat            float64
reverse_pps             float64
reverse_bps             float64
reverse_pl_mean         float64
reverse_piat_mean       float64
reverse_pps_mean        float64
reverse_bps_mean        float64
reverse_pl_var          float64
reverse_piat_var        float64
reverse_pps_var         float64
reverse_bps_var         float64
reverse_pl_q1           float64
reverse_pl_q3           float64
reverse_piat_q1         float64
reverse_piat_q3         float64
reverse_pl_max          float64
reverse_pl_min          float64
reverse_piat_max        float64
reverse_piat_min        float64
reverse_pps_max         float64
reverse_pps_min         float64
reverse_bps_max         float64
reverse_bps_min         float64
reverse_duration          int64
reverse_size_packets      int64
reverse_size_bytes        int64
category                 object
dtype: object



###### First values :
                           id_flow         nw_src  tp_src         nw_dst  \
0  b2bb77a570fcfa9325eb9e51b6116d2a  172.16.25.104   41402  34.107.221.82
1  f07977b0d1d6645c4fe1e9efea080ff3  172.16.25.104   41406  34.107.221.82
2  e4026ba9b6c1957516e92bdd0d04878f  172.16.25.104   38232    52.84.77.43
3  e2d747932e41500b1463fe8ae4299ecb  172.16.25.104   38234    52.84.77.43
4  56325703391225ad65e013e7a2b02fac  172.16.25.104   60166    52.32.34.32

   tp_dst  nw_proto  forward_pc  forward_bc  forward_pl  forward_piat  ...  \
0      80         6           5         300       60.00           6.0  ...
1      80         6           5         300       60.00           6.0  ...
```

```
2     443         6           3         198       66.00          10.0  ...
3     443         6           3         198       66.00          10.0  ...
4     443         6           4         265       66.25           7.5  ...

   reverse_piat_max  reverse_piat_min  reverse_pps_max  reverse_pps_min  \
0         10.333333              6.00         0.166667         0.096774
1         10.000000              6.20         0.161290         0.100000
2         10.333333             10.00         0.100000         0.096774
3         10.333333             10.00         0.100000         0.096774
4          7.750000              7.75         0.129032         0.129032

   reverse_bps_max  reverse_bps_min  reverse_duration  reverse_size_packets  \
0        15.133333         5.806452               121                    15
1        15.133333         6.000000               121                    15
2         6.000000         5.806452                91                     9
3         6.000000         5.806452                91                     9
4         8.548387         8.548387                31                     4

   reverse_size_bytes category
0                1114      WWW
1                1114      WWW
2                 540      WWW
3                 540      WWW
4                 265      WWW

[5 rows x 65 columns]



###### Statistics :
             tp_src        tp_dst     nw_proto     forward_pc    forward_bc  \
count   4234.000000   4234.000000  4234.000000    4234.000000  4.234000e+03
mean   39994.956542   8540.046528     6.660132    3835.848370  7.356521e+06
std    17331.881734  17575.486397     3.815368   18375.794566  3.585172e+07
min        0.000000      0.000000     1.000000       0.000000  0.000000e+00
25%    35248.500000     80.000000     6.000000       2.000000  1.200000e+02
50%    44009.000000    443.000000     6.000000       3.000000  1.980000e+02
75%    52130.250000    443.000000     6.000000       6.000000  3.850000e+02
max    65534.000000  60949.000000    17.000000  181104.000000  3.558093e+08

          forward_pl  forward_piat   forward_pps   forward_bps  \
count    4234.000000   4234.000000  4.234000e+03  4.234000e+03
mean      316.336560     15.261581  4.788105e+02  2.576202e+06
std      3732.045349    182.065520  2.021312e+04  1.200390e+08
min         0.000000      0.000000  0.000000e+00  0.000000e+00
25%        60.000000      0.048051  6.451613e-02  4.000000e+00
50%        66.000000      3.500000  1.666667e-01  1.260000e+01
75%        79.811688      7.500000  5.161290e-01  4.600000e+01
max    154375.000000   4125.000000  1.303625e+06  7.422774e+09

       forward_pl_mean  ...  reverse_pl_min  reverse_piat_max  \
count      4234.000000  ...     4234.000000       4234.000000
mean       1582.814224  ...       54.418871         23.652912
std        9644.341190  ...      269.495303        229.416470
min           0.000000  ...        0.000000          0.000000
25%          43.000000  ...        0.000938          0.000433
```

```
50%          61.250000  ...       15.500000          7.500000
75%          98.000000  ...       60.000000         15.500000
max      162975.000000  ...     5573.208202       4125.000000

       reverse_piat_min  reverse_pps_max  reverse_pps_min  reverse_bps_max  \
count      4.234000e+03     4.234000e+03     4.234000e+03     4.234000e+03
mean       5.189081e+02     1.263424e+03     6.683260e+04     1.270755e+05
std        2.792340e+04     4.689801e+04     2.674774e+06     4.139731e+06
min        0.000000e+00     0.000000e+00     0.000000e+00     0.000000e+00
25%        3.225807e-02     3.030303e-02     3.125000e-02     1.935484e+00
50%        6.559140e-01     9.677419e-02     1.000000e-01     6.026316e+00
75%        8.500000e+00     2.903226e-01     2.325000e+01     4.167742e+01
max        1.816375e+06     2.316875e+06     1.556534e+08     1.707531e+08

       reverse_bps_min  reverse_duration  reverse_size_packets  \
count     4.234000e+03       4234.000000          4.234000e+03
mean      6.747949e+04       3224.000000          2.750047e+05
std       3.034402e+06      20429.627234          1.519335e+06
min       0.000000e+00          0.000000          0.000000e+00
25%       2.242424e+00          5.000000          2.000000e+00
50%       9.258333e+00         30.000000          1.800000e+01
75%       4.900000e+01         60.000000          6.860000e+02
max       1.488506e+08     232137.000000          1.717689e+07

       reverse_size_bytes
count        4.234000e+03
mean         2.592156e+05
std          2.875554e+06
min          0.000000e+00
25%          0.000000e+00
50%          0.000000e+00
75%          2.460000e+02
max          1.214242e+08

[8 rows x 60 columns]
```

In [16]:

```
## Now that we have a better view of the data, we can manipulate it.
## The data is already nicely formatted so not many operations are necessary.

print(data["category"].unique())


data_category = data.groupby("category")[["tp_dst", "forward_size_packets", "revers
data_id_src = data.groupby("nw_src")[["tp_dst", "forward_size_packets", "reverse_si
data_id_dst = data.groupby("nw_dst")[["tp_dst", "forward_size_packets", "reverse_si


print("\n\n\n###### per category : ")
print(data_category.head())

print("\n\n\n###### per source : ")
print(data_id_src.head())

print("\n\n\n###### per destination : ")
print(data_id_dst.head())
```

```
print("\n\n\n###### stats per category : ")
print(data_category.describe())

print("\n\n\n###### stats per source : ")
print(data_id_src.describe())

print("\n\n\n###### stats per destination : ")
print(data_id_dst.describe())
```

```
['WWW' 'DNS' 'VOIP' 'ICMP' 'FTP' 'P2P']
```

```
###### per category :
  category    tp_dst  forward_size_packets  reverse_size_packets
0      DNS     75713             296924721               9044166
1      FTP    208803           16731539682             382059857
2     ICMP    625608            6164534717             176377710
3      P2P  31405590             263289124               6465986
4     VOIP   1527226            3399064747              82085318
```

```
###### per source :
        nw_src  tp_dst  forward_size_packets  reverse_size_packets
0  1.01136E+11   38705                     0                     3
1  1.02129E+11   50275                     0                     1
2  1.02134E+11   60309                     0                     2
3  1.02165E+11   37945                     0                     1
4  1.02222E+11  145385                     0                     5
```

```
###### per destination :
        nw_dst  tp_dst  forward_size_packets  reverse_size_packets
0  1.04198E+11     886                    11                     8
1  1.04237E+11     443                   145                   180
2  1.07155E+11     160                     4                     0
3  1.14141E+11       0                  5586                     0
4  1.30225E+11       0                  6083                  6084
```

```
###### stats per category :
             tp_dst  forward_size_packets  reverse_size_packets
count  6.000000e+00          6.000000e+00          6.000000e+00
mean   6.026426e+06          7.074529e+09          1.940617e+08
std    1.246226e+07          7.381980e+09          2.080323e+08
min    7.571300e+04          2.632891e+08          6.465986e+06
25%    3.130042e+05          1.072460e+09          2.730445e+07
50%    1.076417e+06          4.781800e+09          1.292315e+08
75%    2.118519e+06          1.323500e+10          3.306393e+08
max    3.140559e+07          1.673154e+10          5.083369e+08
```

```
###### stats per source :
              tp_dst  forward_size_packets  reverse_size_packets
count     483.000000          4.830000e+02          4.830000e+02
mean    74862.436853          8.788235e+07          2.410704e+06
std     83313.921056          1.393963e+09          3.186155e+07
min     32783.000000          0.000000e+00          0.000000e+00
25%     43980.000000          0.000000e+00          1.000000e+00
50%     53723.000000          0.000000e+00          2.000000e+00
75%     85348.000000          4.000000e+00          4.000000e+00
```

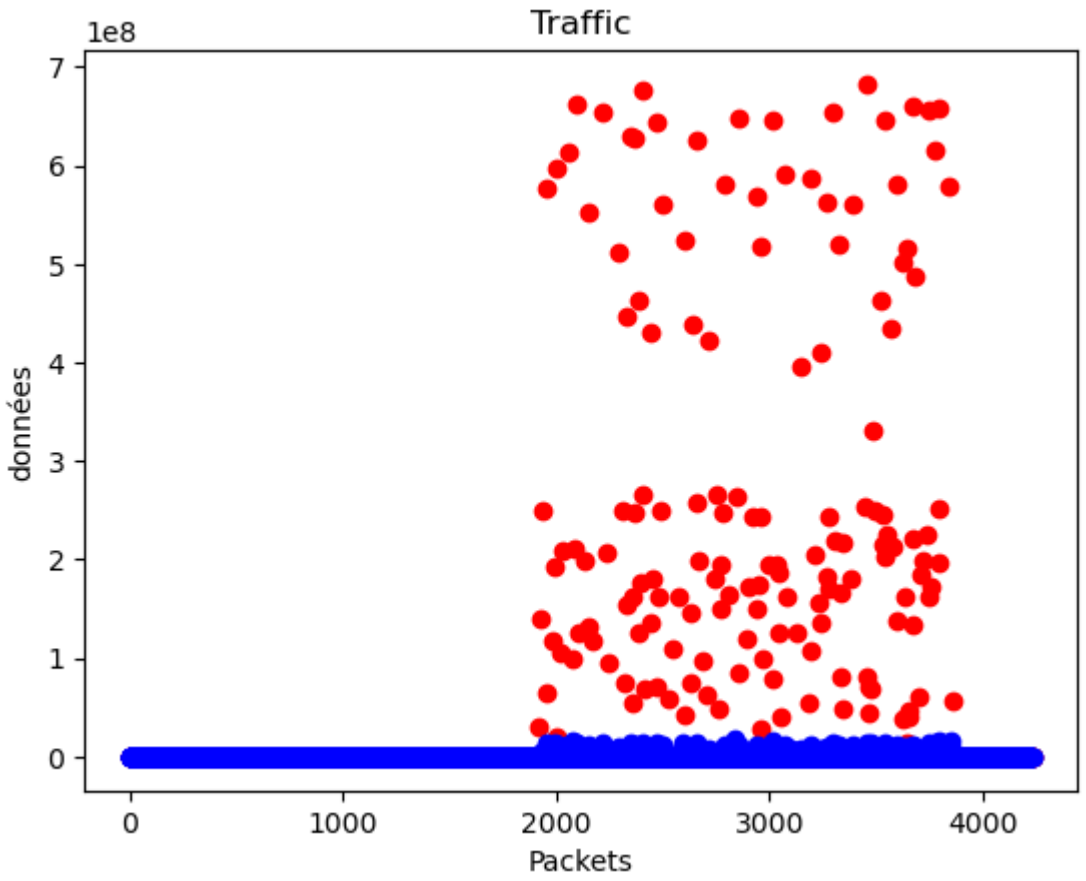
```
max    940337.000000          2.574205e+10          5.463225e+08


###### stats per destination :
            tp_dst  forward_size_packets  reverse_size_packets
count  8.800000e+02          8.800000e+02          8.800000e+02
mean   4.108927e+04          4.823543e+07          1.323148e+06
std    1.101733e+06          1.427903e+09          3.013184e+07
min    0.000000e+00          0.000000e+00          0.000000e+00
25%    4.430000e+02          2.300000e+01          1.400000e+01
50%    4.430000e+02          3.845000e+02          2.725000e+02
75%    8.860000e+02          4.208000e+03          5.134250e+03
max    3.265328e+07          4.235854e+10          8.923850e+08
```

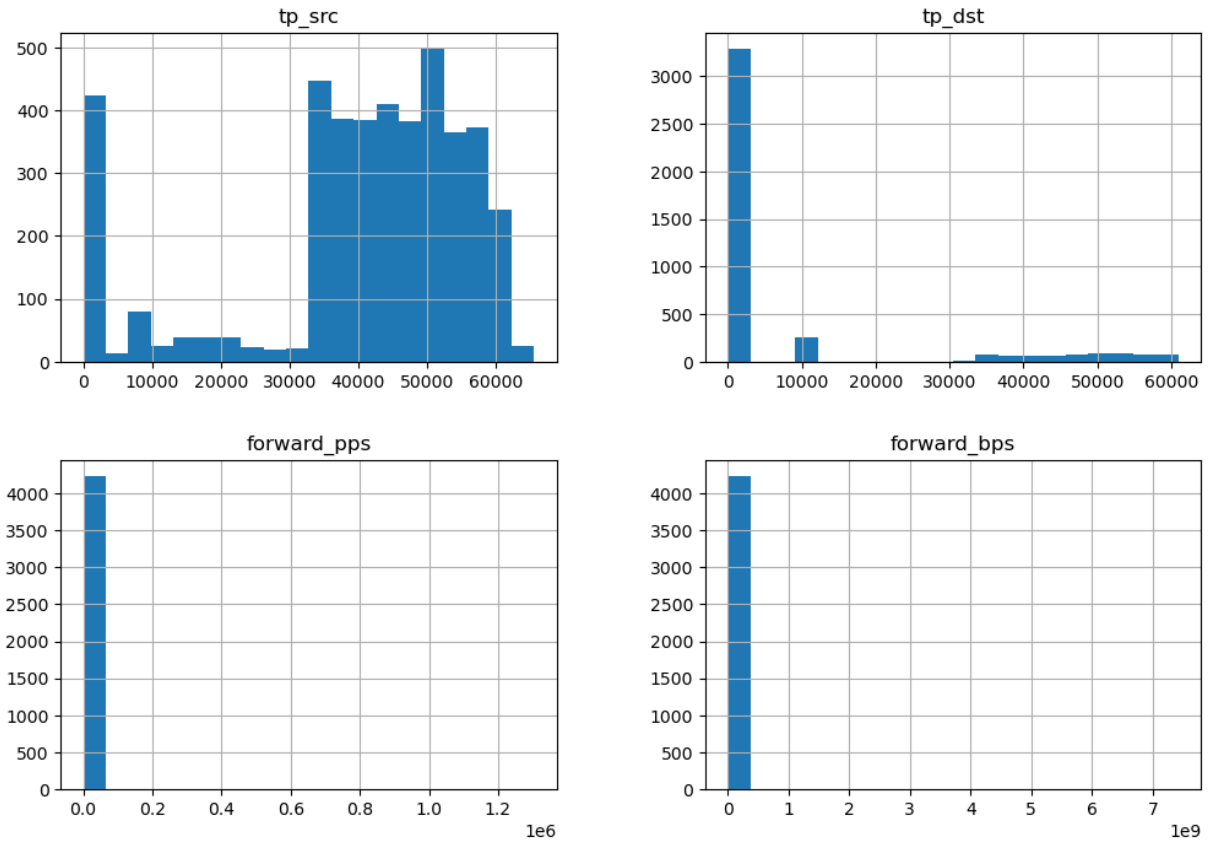
In [27]:
```
#Some matplotlib graphs:

plt.scatter(np.arange(0, len(data)), data["forward_size_packets"], color = "r", lab
plt.scatter(np.arange(0, len(data)), data["reverse_size_packets"], color = "b", lab
plt.title("Traffic")
plt.xlabel("Packets")
plt.ylabel("données")
```

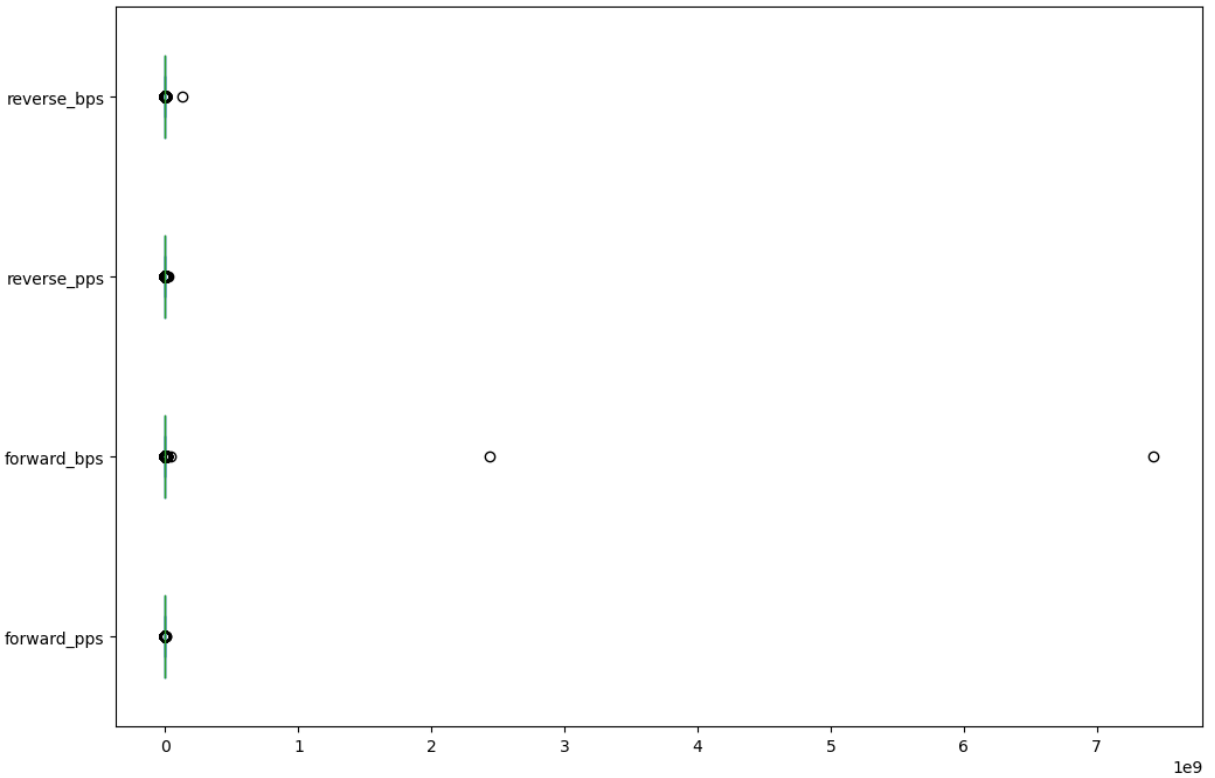
Out[27]:
```
Text(0, 0.5, 'données')
```



In [19]:
```
data.hist(column=['tp_src', 'tp_dst', 'forward_pps', 'forward_bps'], bins=20, figsi
plt.show()
```

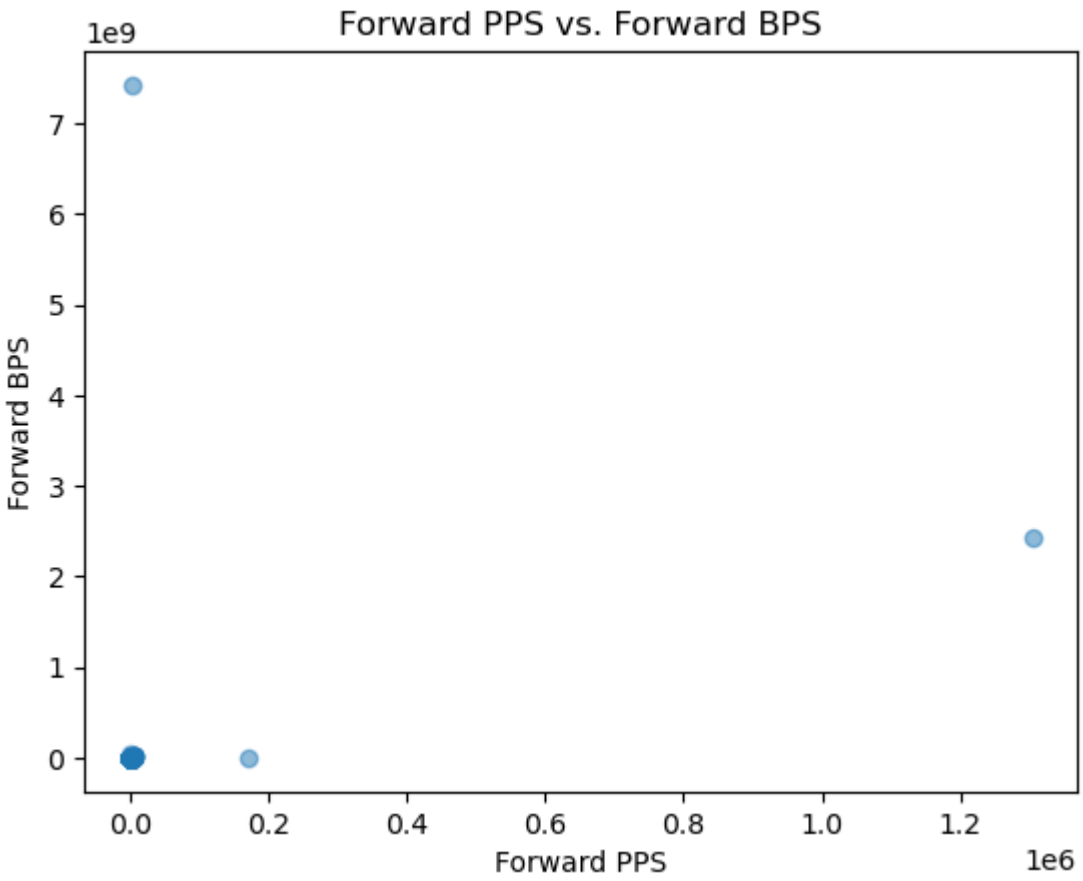


In [20]:
```
data[['forward_pps', 'forward_bps', 'reverse_pps', 'reverse_bps']].plot(kind='box',
plt.show()
```

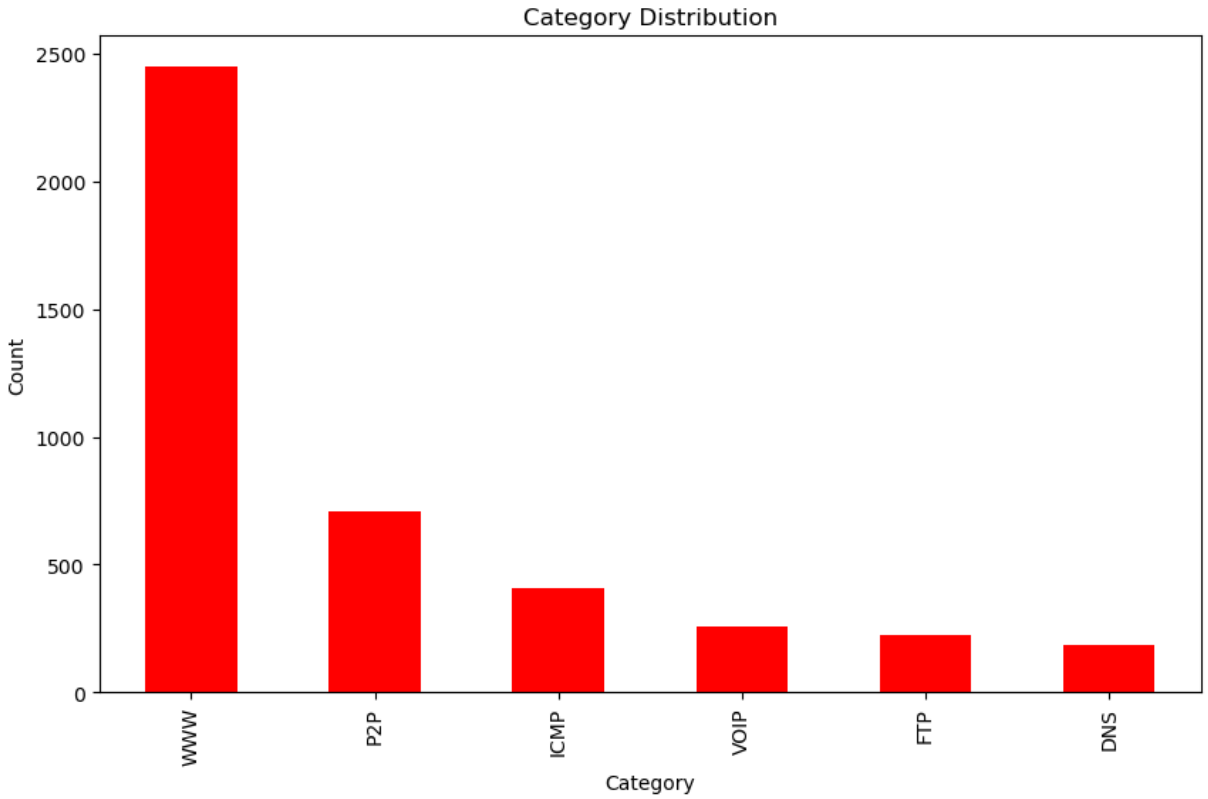


In [21]:
```
plt.scatter(data['forward_pps'], data['forward_bps'], alpha=0.5)
plt.title('Forward PPS vs. Forward BPS')
plt.xlabel('Forward PPS')
```

```
plt.ylabel('Forward BPS')
plt.show()
```



In [26]:
```
category_counts = data['category'].value_counts()
category_counts.plot(kind='bar', figsize=(10, 6), color='red')
plt.title('Category Distribution')
plt.xlabel('Category')
plt.ylabel('Count')
plt.show()
```


Category Distribution
2500
2000
1500
1000
500
0
Count
WWW
P2P
ICMP
VOIP
FTP
DNS
Category

In [ ]: