**Assignment 1**

1. **Do the following before you start attempting the exercise questions, here:**

**(a) Start up your web browser - not in the https mode.**

**(b) Start up the Wireshark packet sniffer, but don't yet begin packet capture. Enter "http" (just the letters, not the quotation marks) in the display-filter-specification window, so that only captured HTTP messages will be displayed later in the packet-listing window. (We're only interested in the HTTP protocol here, and don't want to see the clutter of all captured packets).**

**(c) Wait a bit more than one minute (we'll see why shortly), and then begin Wireshark packet capture.**

**(d) Enter the following to your browser http://gaia.cs.umass.edu/wireshark-labs/**

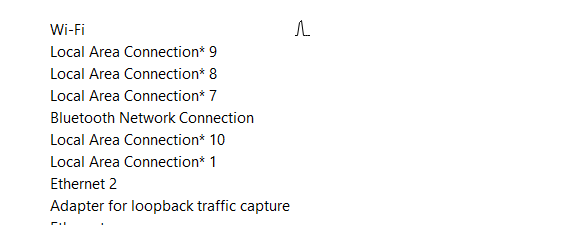
**HTTP-wireshark-file1.html Your browser should display the very simple, one-line**

**HTML file.**

**(e) Stop Wireshark packet capture.**

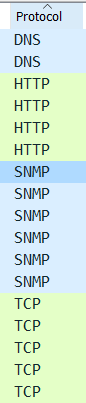
1. **What are the network interfaces available on your computer? Which network did you eventually select in your experiments.**

Available network interfaces:

****

For this experiment, I have selected Wi-Fi interface.

1. **Which application layer protocol is used in this case?**

****

Application layer protocol used: DNS, HTTP, SNMP

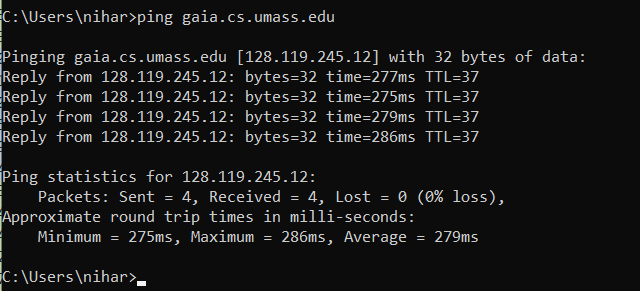
1. **What are the other protocols used and displayed in the unfiltered packet listing window of Wireshark, besides the one that you answered in Q(b)?**

Other protocols used: TCP

1. **What is the IPA of your machine? What is the IPA of the destination machine? Is there any way by which you can ascertain that the IPA of the destination indeed is the same as that you observed in Wireshark? If so, how?**

Source IPA: 192.168.1.102

Destination IPA: 128.119.245.12



Yes, we can ascertain by sending ping request on that domain and we can see the ip address of that domain in response.

1. **What is the class of the IPA of the source machine? That of destination machine?**

Range of 1st octet of class A = [1, 127]

Range of 1st octet of class B = [128, 191]

Range of 1st octet of class C = [192, 223]

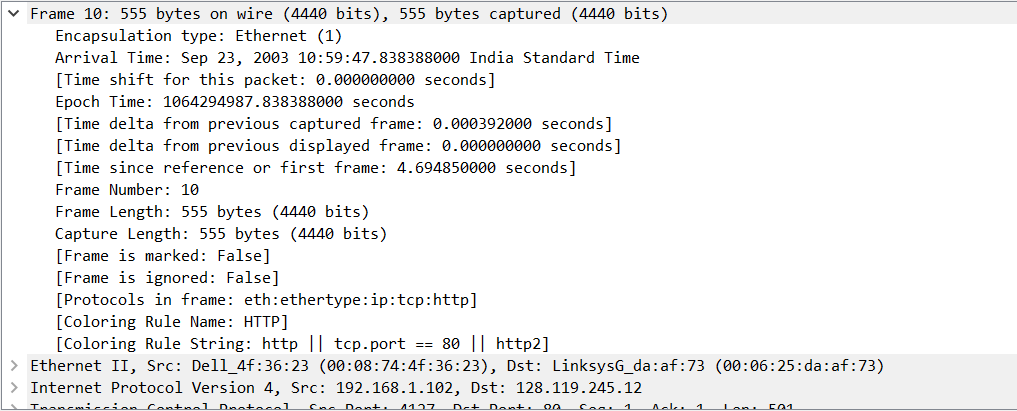
Range of 1st octet of class D = [224, 239]

Range of 1st octet of class E = [240, 255]

Class of Source machine is C.

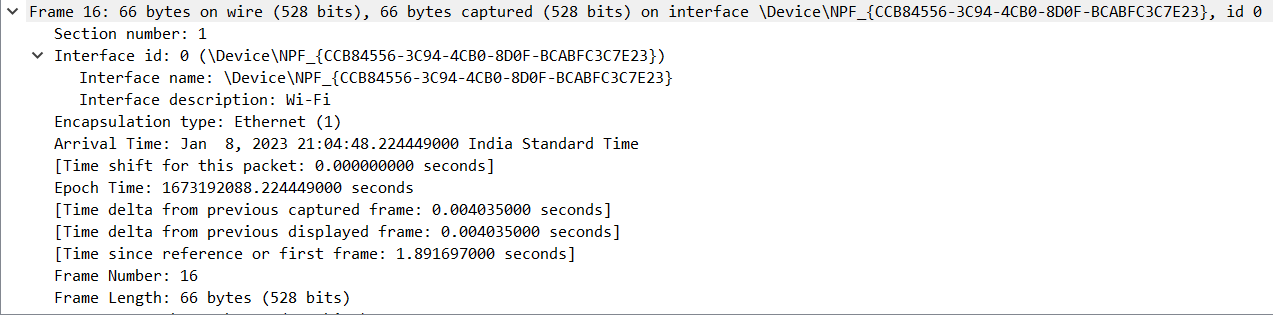
Class of destination machine is B.

1. **How many bits were captured in this packet? At what time was this packet captured?**

****

4440 bits were captured in this packet at Sep 23, 2003 10:59:47.838388000 India Standard Time.

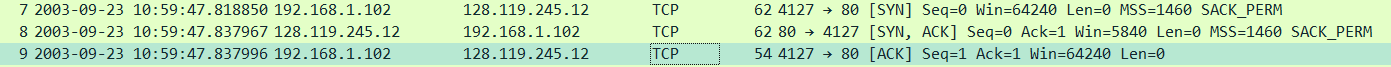
1. **What is the interface id used? What is the address of the interface?**

****

Interface id: 0

Interface address: \Device\NPF\_{CCB84556-3C94-4CB0-8D0F-BCABFC3C7E23}

1. **Which packets are forming the TCP 3-way handshake for connection establishment? What are the SYN and ACK in each of the three packets?**

****

Packet 1

SYN: 1

ACK: 0

Packet 2

SYN: 1

ACK: 1

Packet 3

SYN: 0

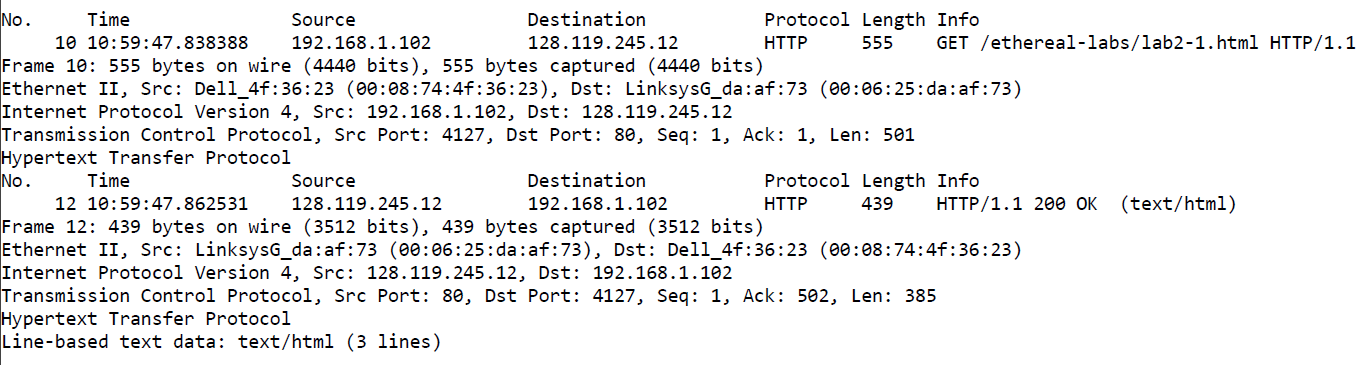
ACK: 1

1. **How long did it take from when the HTTP GET message was sent until the HTTP OK reply was received? (By default, the value of the Time column in the packet-listing window is the amount of time, in seconds, since Wireshark tracing began. To display the Time field in time-of-day format, select the Wireshark View pull down menu, then select Time Display Format, then select Time-of-day.)**

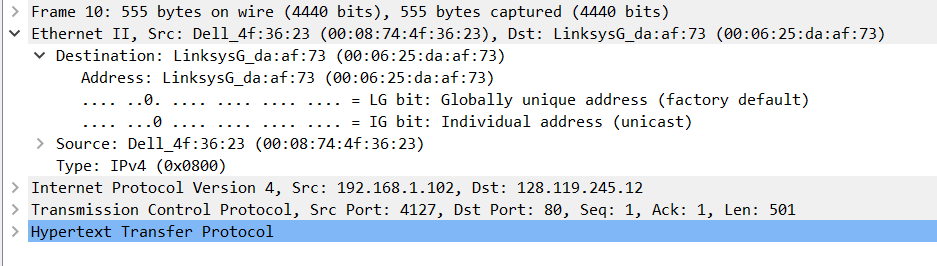


Time taken: 0.001704 seconds

1. **Print the two HTTP messages (GET and OK) referred to in question above. To do so, select Print from the Wireshark File command menu, and select the "Selected Packet Only" and "Print as displayed" radial buttons, and then click OK.**

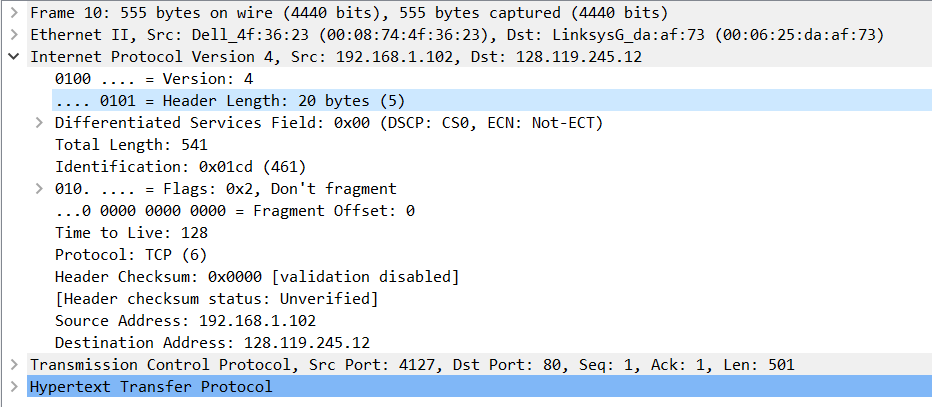
****

1. **What is the destination physical address of the first packet captured? What device does it belong to? Show where in the capture would you find this information.**

****

Destination Physical Address: 00:06:25:da:af:73

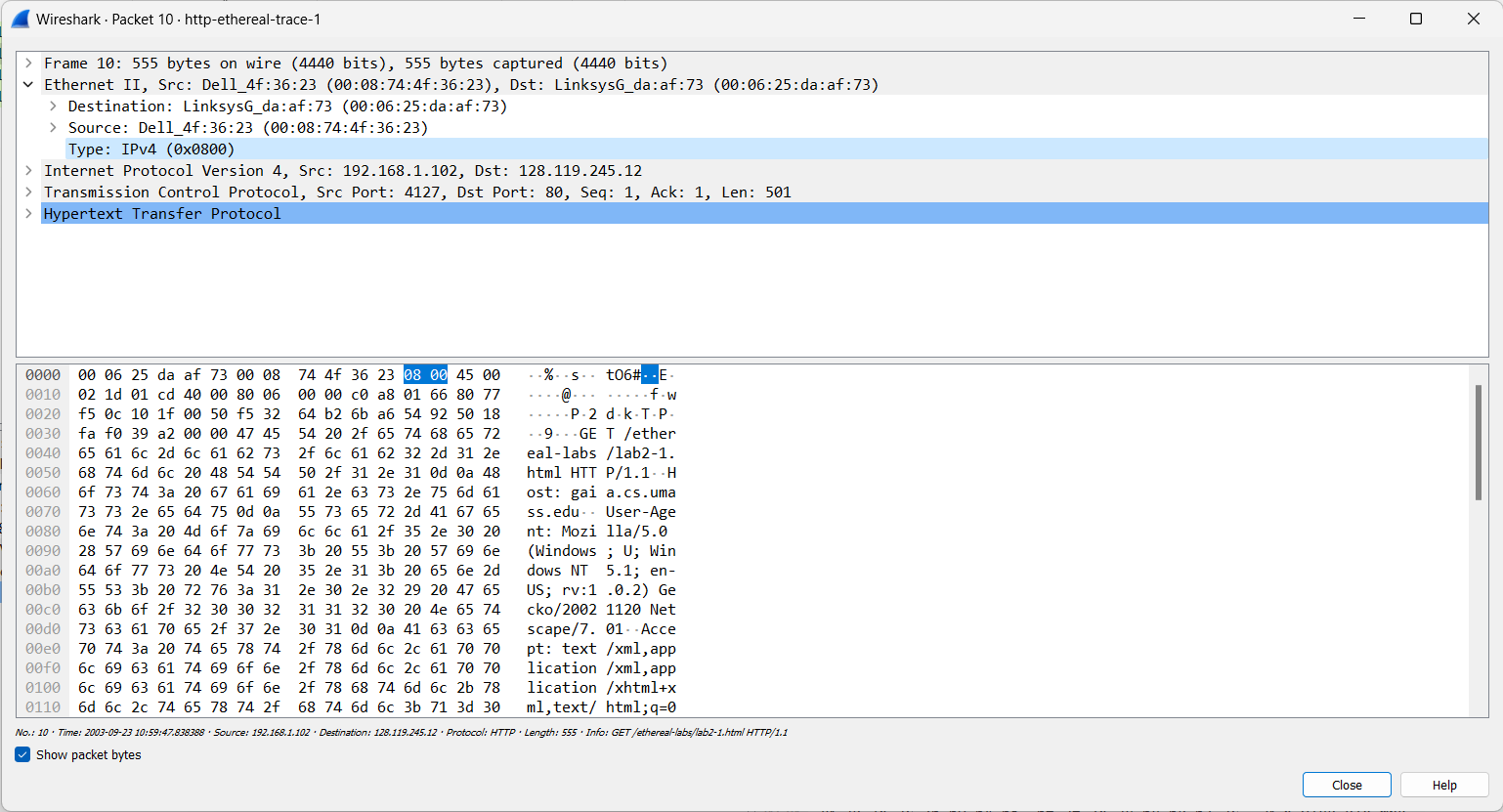
1. **How many bytes of header does the first frame sent have? Show where in the capture would you find this information.**

****

Header Length: 20 bytes

1. **By looking at the Ethernet header of a frame, can we determine if it contains an IP packet? Show where in the capture would you find this information.**

Yes, we can determine if a frame contains an IP packet by looking at the Ethernet Header of a frame.  
The following is the Ethernet Header of a HTTP packet

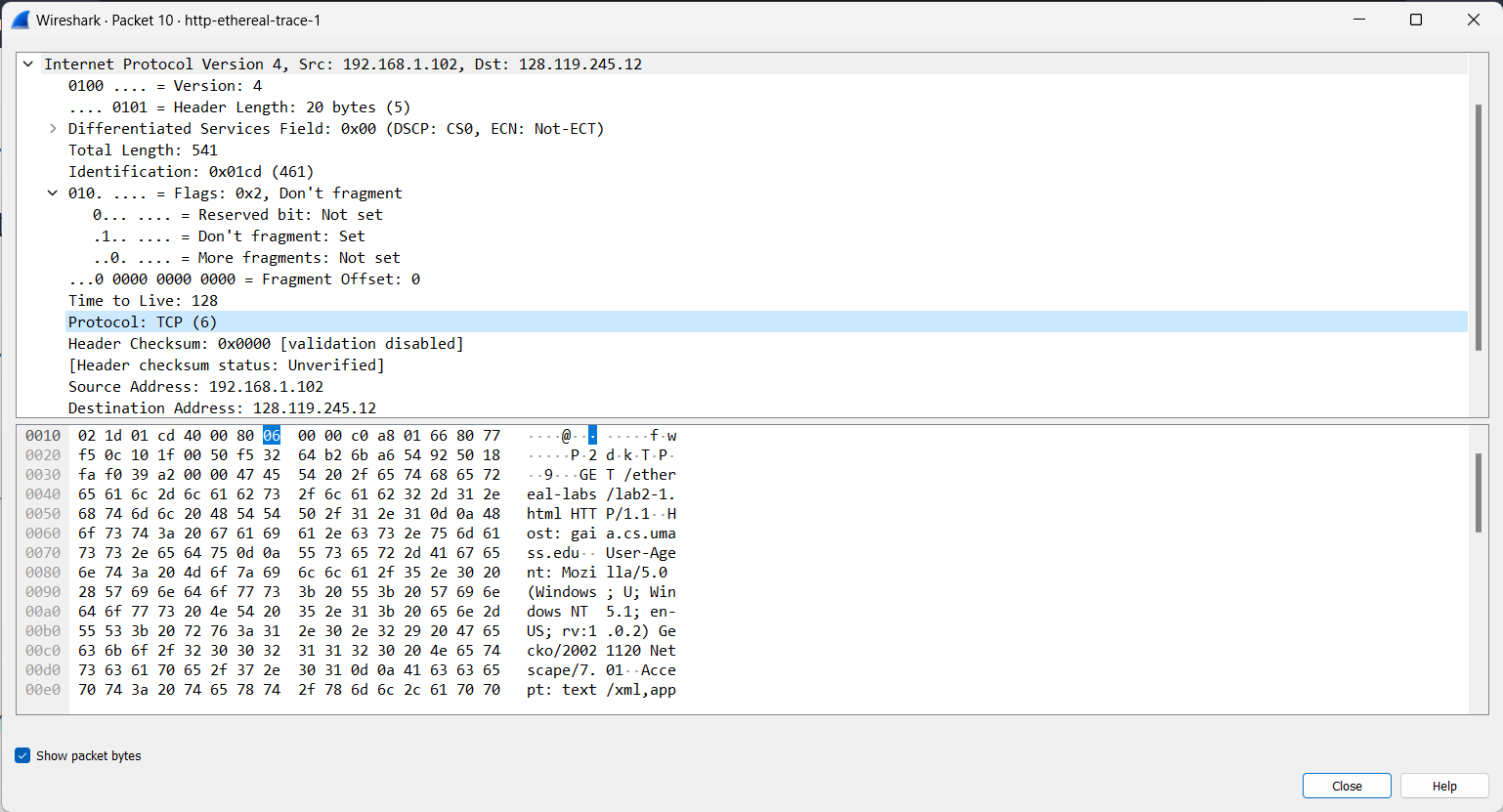


The packet shows IPv4 as the Type

1. **Is it possible to know if the first packet captured has TCP or UDP as transport protocol by looking at the IP header? Explain and show where in the capture would you find this information.**

Yes, it is possible to see if the packet has TCP or UDP by looking at the IP header.

The following is the first packet captured, which has TCP.

****

1. **In the SYN, ACK. What are the source and destination ports? Are these the same for the client and the server? Explain why.**

****

TCP 3-way handshake for connection establishment first SYN packet from client

Source port: 4127

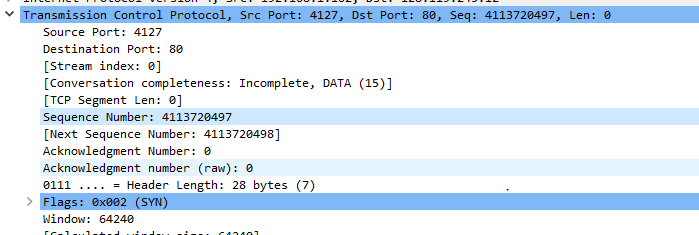
Destination port: 80

TCP 3-way handshake for connection establishment second SYN, ACK packet

Source port: 4127

Destination port: 80

1. **Why does the Server Hello message sent by the server have 1 as a relative sequence number and 185 as a relative acknowledgement number.**
2. **Right-click a TCP capture -> TCP preferences -> Uncheck the box "Show relative sequence number." What is the first sequence number sent by the server to the client? Why is it not the 0 displayed by Wireshark?**

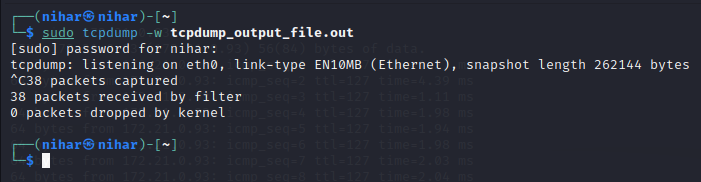


The first sequence number sent by the server to the client: 4113720497

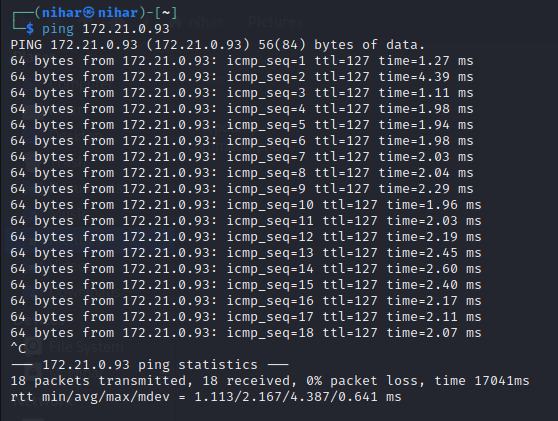
The sequence number is not taken zero for every TCP connection to avoid confusion and overlap between different connections, and prevent easy connection hijacking.

1. **This exercise is a simple exercise that only requires you to capture the tcpdump track. The problem requires you to either use two virtual machines on your laptop or two different machines in the computer lab ask the administrator for the host name of both the machines, if so. Then run the tcpdump command on one machine say PC1 (saving the output for your lab report) so that it monitors all the packets that contain the IP address of PC2 only and none else. Next, open a new terminal window on PC1 and execute ping command to PC2. It may be necessary to press Ctrl + C to terminate the tcpdump session. It may sometimes be best to simply redirect the output of tcpdump straight to a file and view it afterward with the more command or a text editor. Find out how can you do so. Run the command $tcpdump -enx -w tcpdump\_out\_file.out & do you see any output on the screen? Why?**

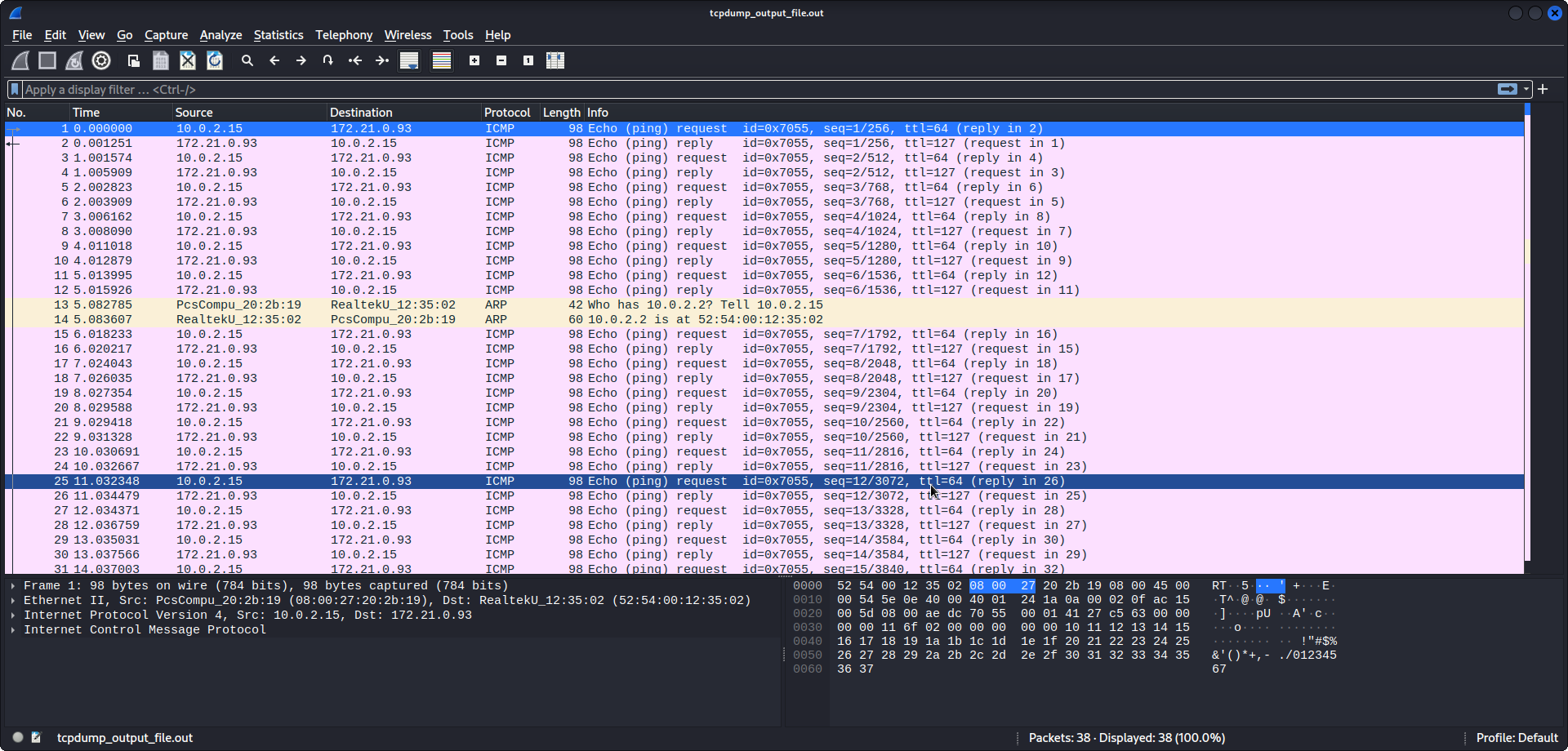
Step – 1 Start tcpdump

****

Step – 2 Ping request started

****

Step – 3 Captured Packets

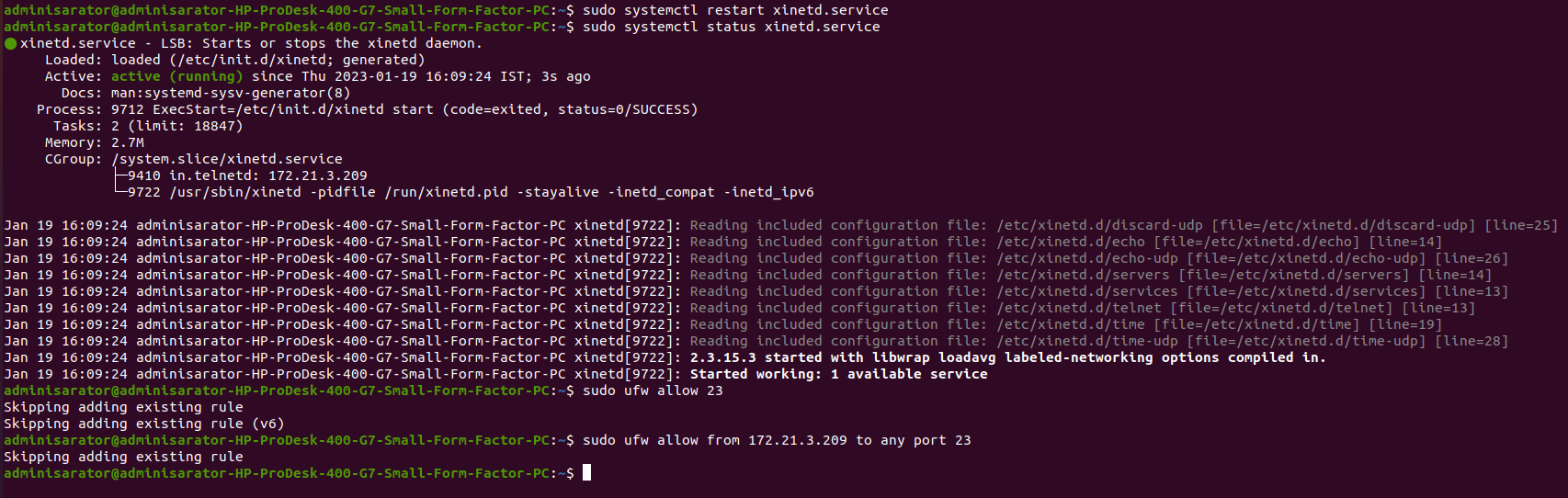
****

1. **This question is in continuation of the question no 2. Run the command "telnet remote host". remotehost is the host name of either another virtual machine in your machine or it is the host name of any other machine in the network used in the lab (Ask the lab technical suport staff about the name of other machine). This command would generate some TCP trac. After you login to the remote machine, terminate the telnet session and terminate the tcpdump program.**

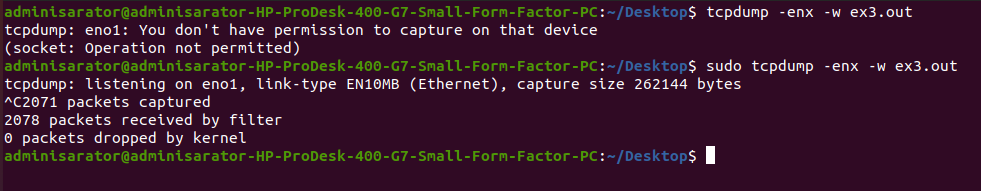
**• Click on a TCP packet from the list of captured packets in the wireshark window. Then go to the Edit menu and choose Mark Frame.**

**• Go to the File menu and choose Print. In the Wireshark: Print dialog that pops up, check File, Plain Text, expand all levels, print detail and supress unmarked frames. Then, enter the output text file name, e.g., headers.txt, and click the OK button. The marked packet is now dumped into the text file, with a detailed list of the name and value of every field in all the three headers.**

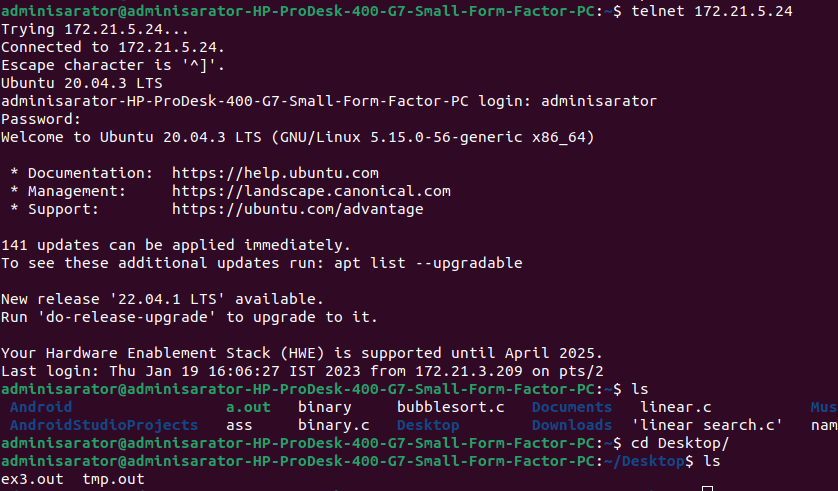
Step 1 – Telnet server creation

****

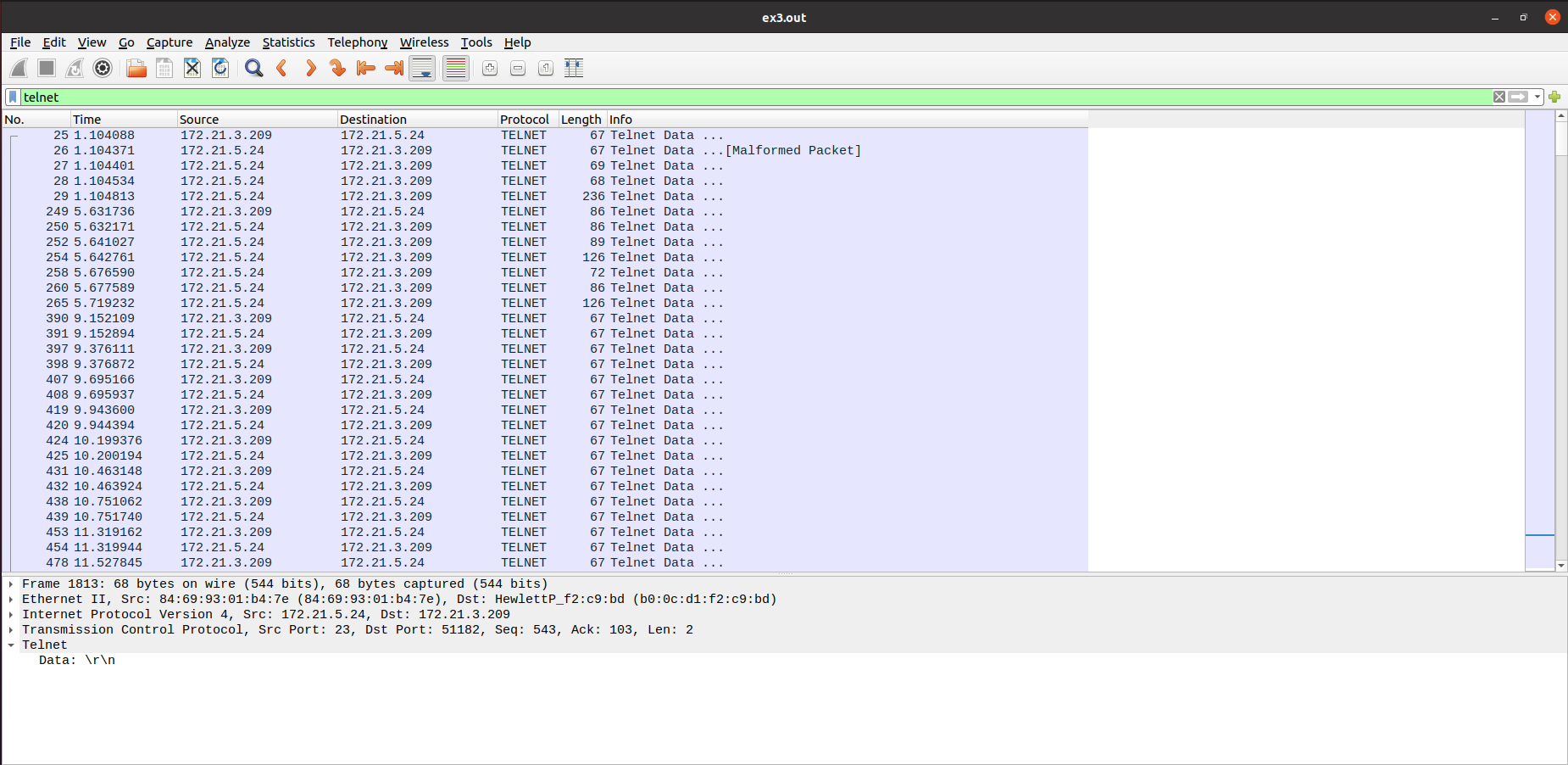
Step 2 – Starting of tcpdump server on server side

****

Step 3 – Trying to connect from client side and getting access of server files

****

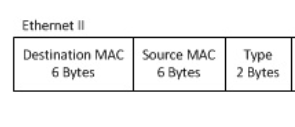
Step 4 – Captured Wireshark package

****

**Now answer the following questions:**

1. **Draw the format of the packet you saved, including the link, IP, and TCP headers, and identify the value of each field in these headers. Express the values in the decimal format.**

**Ethernet Packet Format:**

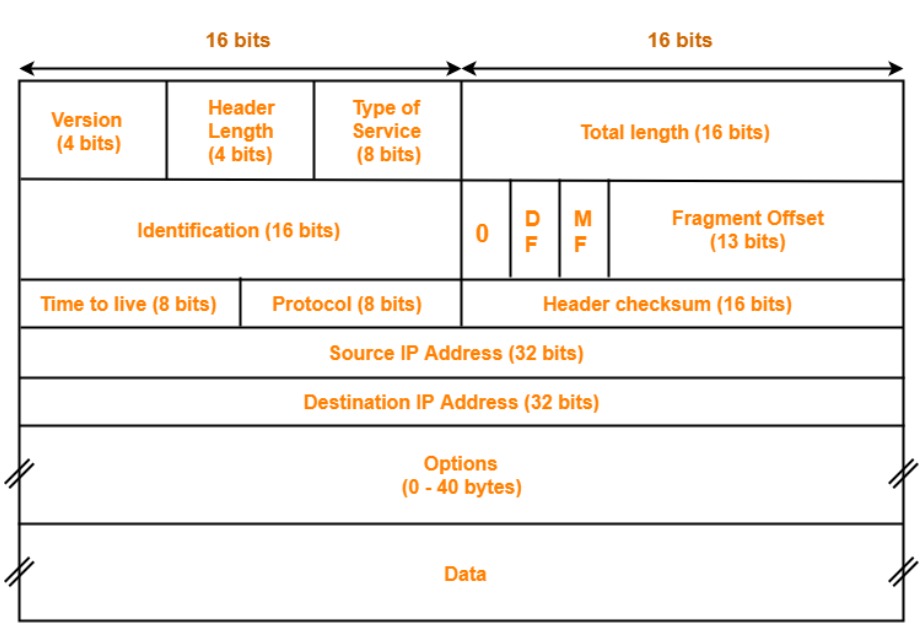


Destination: HP\_01:b4:7e (84:69:93:01:b4:7e)

Source: HewlettP\_f2:c9:bd (b0:0c:d1:f2:c9:bd)

Type: IPv4 (0x0800)

**IP Packet Format:**



Version: 4

Header Length: 20 bytes

Type of Service: 0

Total Length: 53

Identification: 0597111

Flags:

Do not Fragment: 1

More Fragment:

Fragment Offset: 0

Time to Live: 64

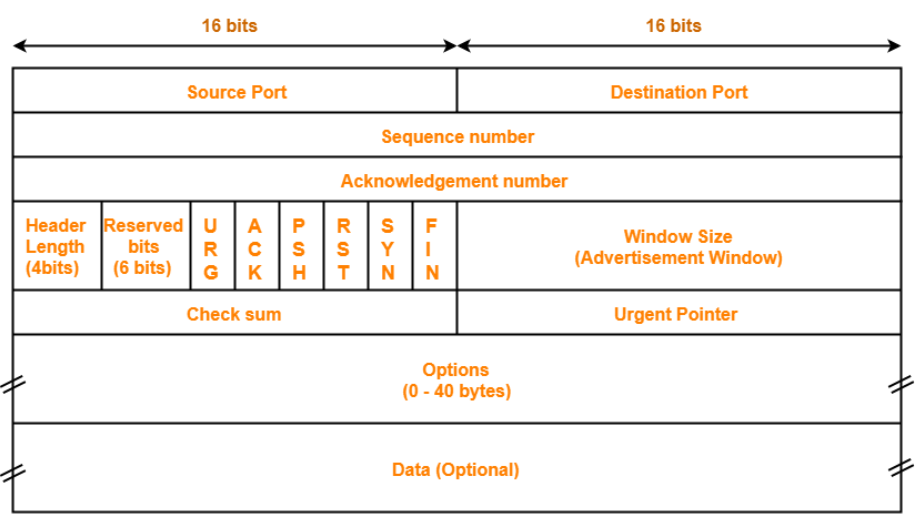
Protocol: TCP (6)

Header Checksum: 61475

Source Address: 172.21.3.209

Destination Address: 172.21.5.24

**TCP Packet Format:**



Source Port: 5182

Destination Port: 23

Sequence Number: 3109923138

Ack Number: 1773035314

Header Length: 32 Bytes

Reserved bits: 0

URG: 0

ACK: 1

PSH: 1

RST: 0

SYN: 0

FIN: 0

Window Size: 501

Check Sum: 37867

Urgent Pointer: 0

Options: 12 Bytes

1. **What is the value of the protocol field in the IP header of the packet you saved? What is the use of the protocol field?**

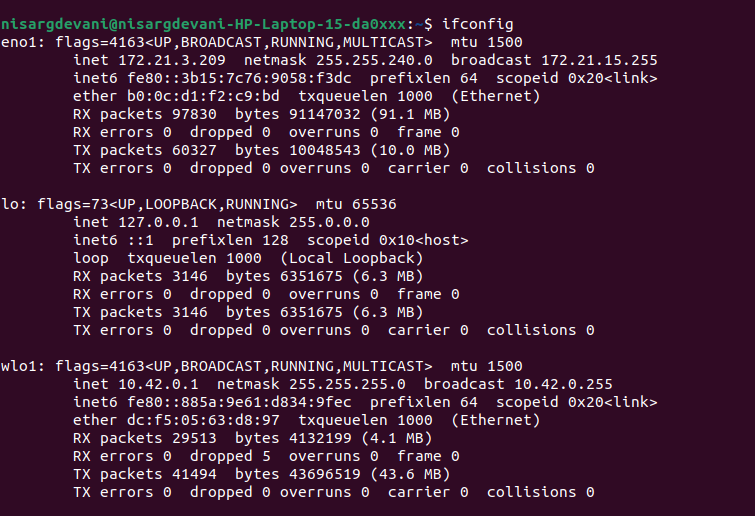
Protocol is an 8-bit field. It tells the network layer at the destination host to which protocol the IP datagram belongs to. In other words, it tells the next level protocol to the network layer at the destination side. Protocol number of ICMP is 1, IGMP is 2, TCP is 6 and UDP is 17.

In this case the protocol field containing number 6 signifies TCP protocol.

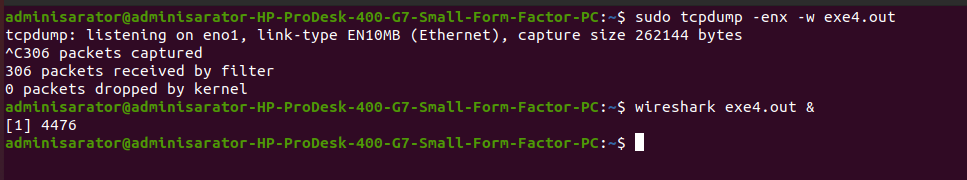
1. **In a manner similar to the Exercise no 3, now run tcpdump to capture an ARP request and an ARP reply and then use Wireshark to analyse the frames. If there are no arp requests and replies in the network, generate some using arpinga - remote - machine: After you see several ARP replies in the arping output, terminate the arping and the tcpdump program. Open the tcpdump trace using $wireshark - r exe4:out &: Print one ARP request and one ARP reply using wireshark.**

ARP request and ARP reply

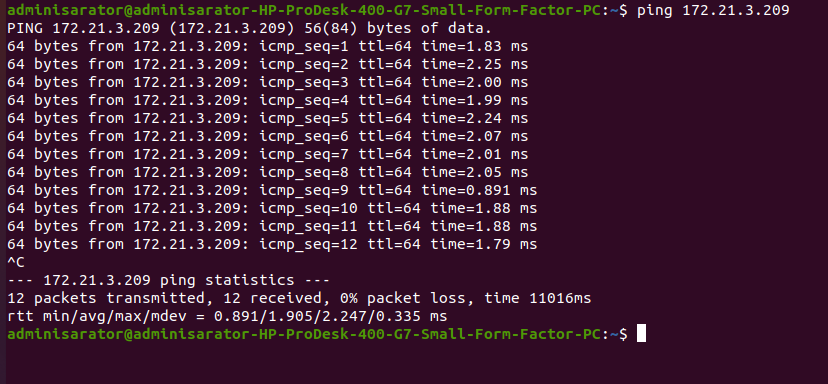
Step - 1: Check configuration of receiver machine.



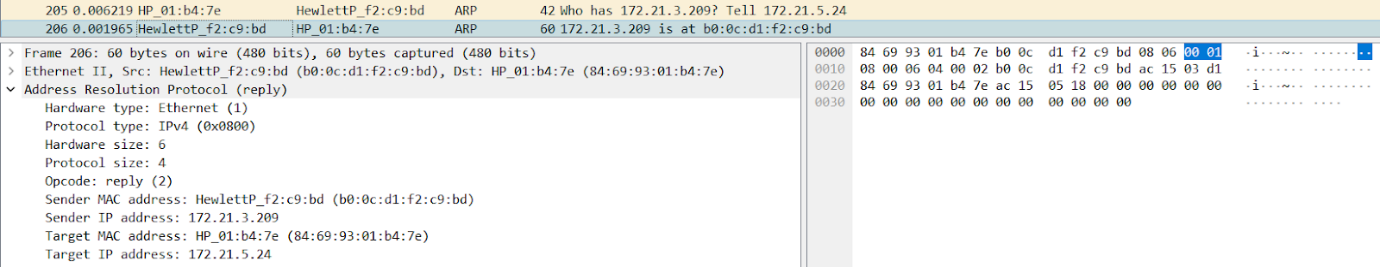
Step - 2: Start tcpdump on sender machine.



Step - 3: Ping to receiver machine from sender machine.



Step - 4: Find ARP request/reply in captured packets.



**Now answer the following questions:**

1. **What is the value of the frame type field in an Ethernet frame carrying an ARP request and in an Ethernet frame carrying an ARP reply, respectively?**

The value of the frame type field in an Ethernet frame carrying an ARP request is ARP and in an Ethernet frame carrying an ARP reply is ARP.

1. **What is the value of the frame type field in an Ethernet frame carrying an IP datagram captured in the previous exercise?**

Frame type: Ipv4

1. **What is the use of the frame type field?**

The frame type field is a field in the link-layer (layer 2) header of a packet that is used to identify the type of packet that is being transmitted. The frame type field is used by network devices to determine how to process a packet and what protocol is being used.

1. **Explain briefly the purposes of the following tcpdump expressions.**
2. **tcpdump udp port 520**

Tcpdump is used to capture and analyze network packets.

So, purpose of given expression is to capture UDP packet with port number 520 which is used by routing information protocol which uses hop count as a routing metric to find the best path between the source and the destination network.

1. **tcpdump -x -s 120 ip proto 89**

-x command : When parsing and printing, in addition to printing the headers of each packet, print the data of each packet (minus its link level header) in hex

-s sLen : Prints slen bytes of data from each packet rather than the default of 262144 bytes

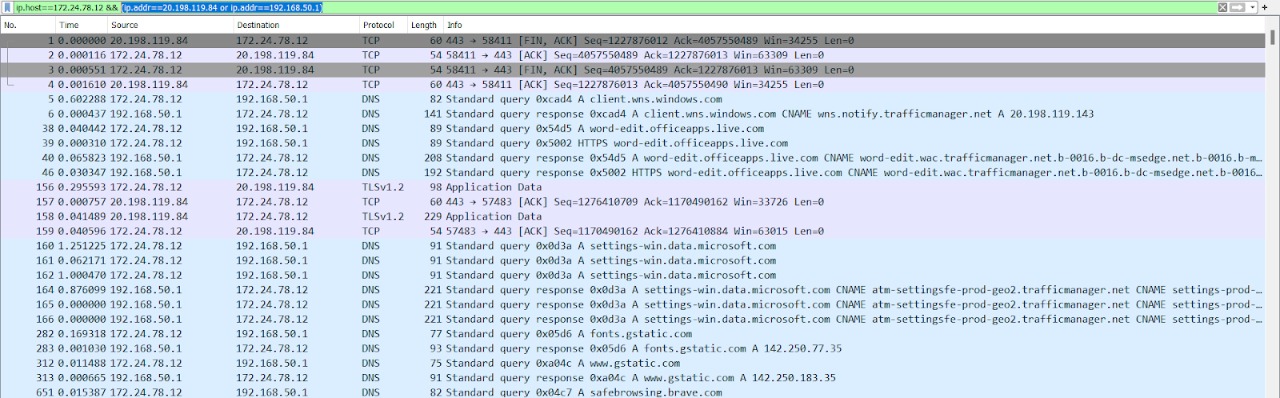
Proto 89 is protocol code for OSPF.

So purpose of given expression is to capture and print 120 bytes data of each packet which is using OSPF protocol

1. **tcpdump -x -s 70 host ip addr1 and (ip addr2 or ip addr3)**

Command host ip\_addr1 will match that address with source or destination address. After matching that command ip\_addr2 or ip\_adrr3 will search that ip in the remainder of the source or destination.

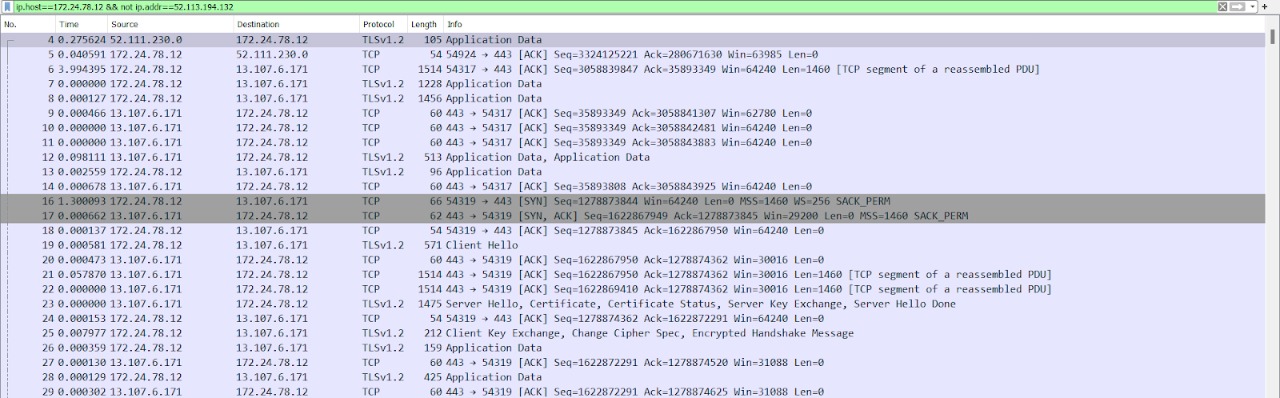
For example,



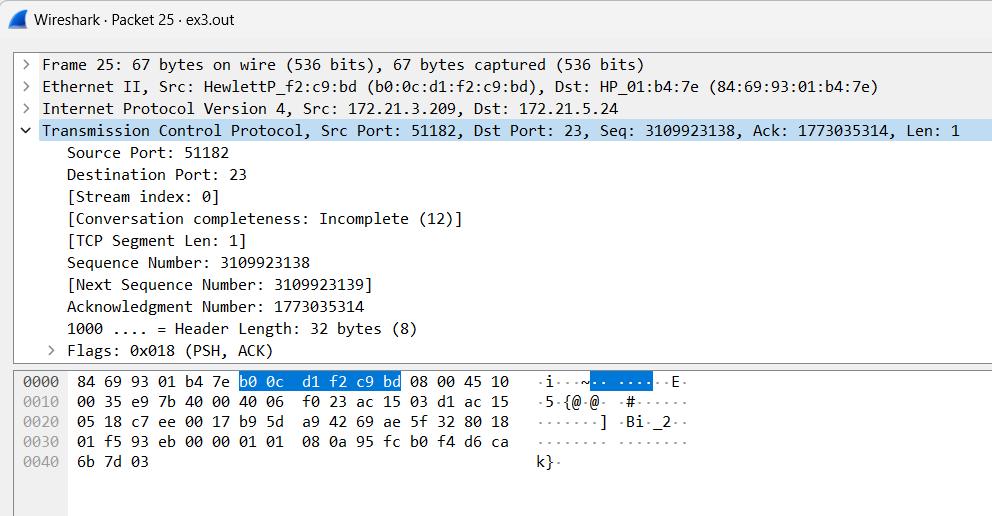
1. **tcpdump -x -s 70 host ip addr1 and not ip addr2**

Command host ip\_addr1 will match that address with source or destination address. After matching that command, ip\_addr2 will search that ip not in remaining of the source or destination.

For example,



1. **Start tcpdump in a command window to capture packets between your machine and a remote host using: tcpdump -n -nn host your -host remote -host. Execute any TCP utility,telnet for example - as in the problem before, in another command window. When you see a TCP packet in the tcpdump output, terminate tcpdump and save its output. Now answer the following question:**

****

1. **What are the port numbers used by the remote and the local computer?**

Port number used by local computer : 51182

Port number used by remote host : 23

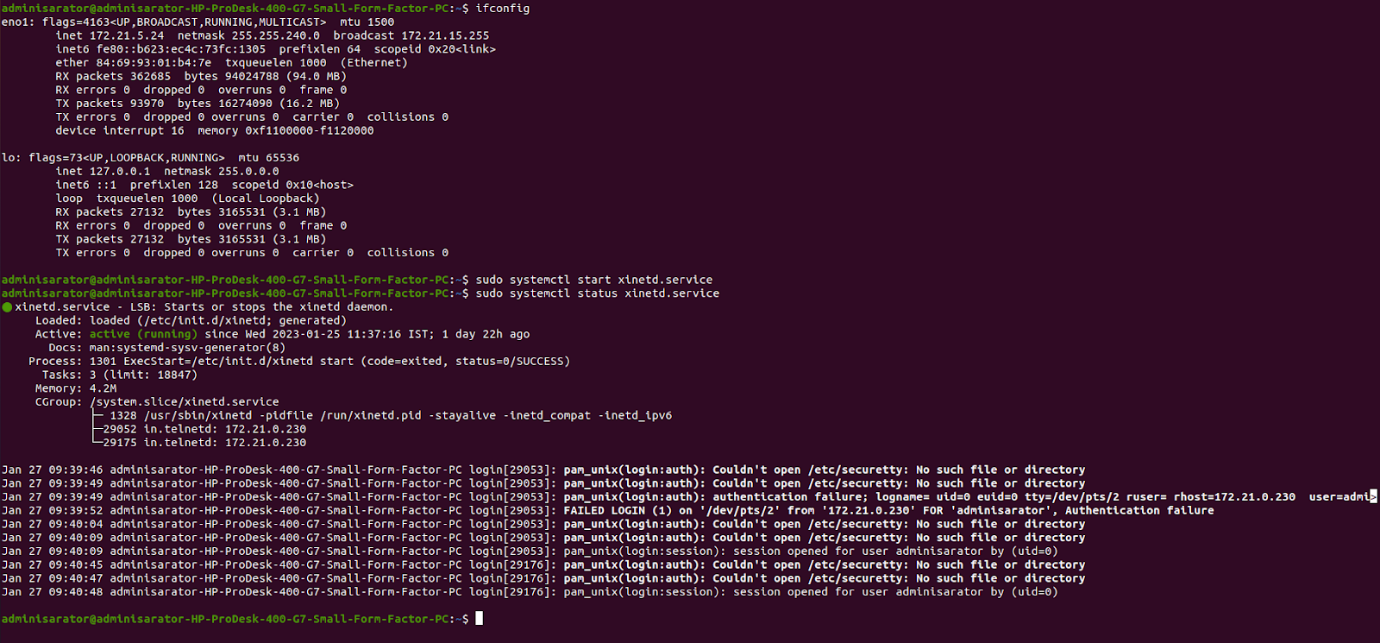
1. **Which machine’s port number matches the port number listed for telnet in the /etc/services file? Note: In case telnet is not listed in the /etc/services file, use ssh utility.**

Remote host’s machine uses port number 23, which is assigned to the telnet protocol.

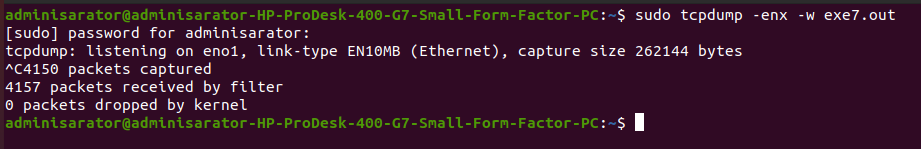
1. **Start tcpdump in one command window using tcpdump -n -nn host your-host remote-host: Then, telnet to the remote host from a second command window by typing telnet remotehost. Again issue the same command from a third command window. Now you are opening two telnet sessions to the same remote host simultaneously, from two different command windows. Check the port numbers being used on both sides of the two connections from the output in the tcpdump window. Save a TCP packet from each of the connections. Now answer the following questions:**

Telnet Configuration

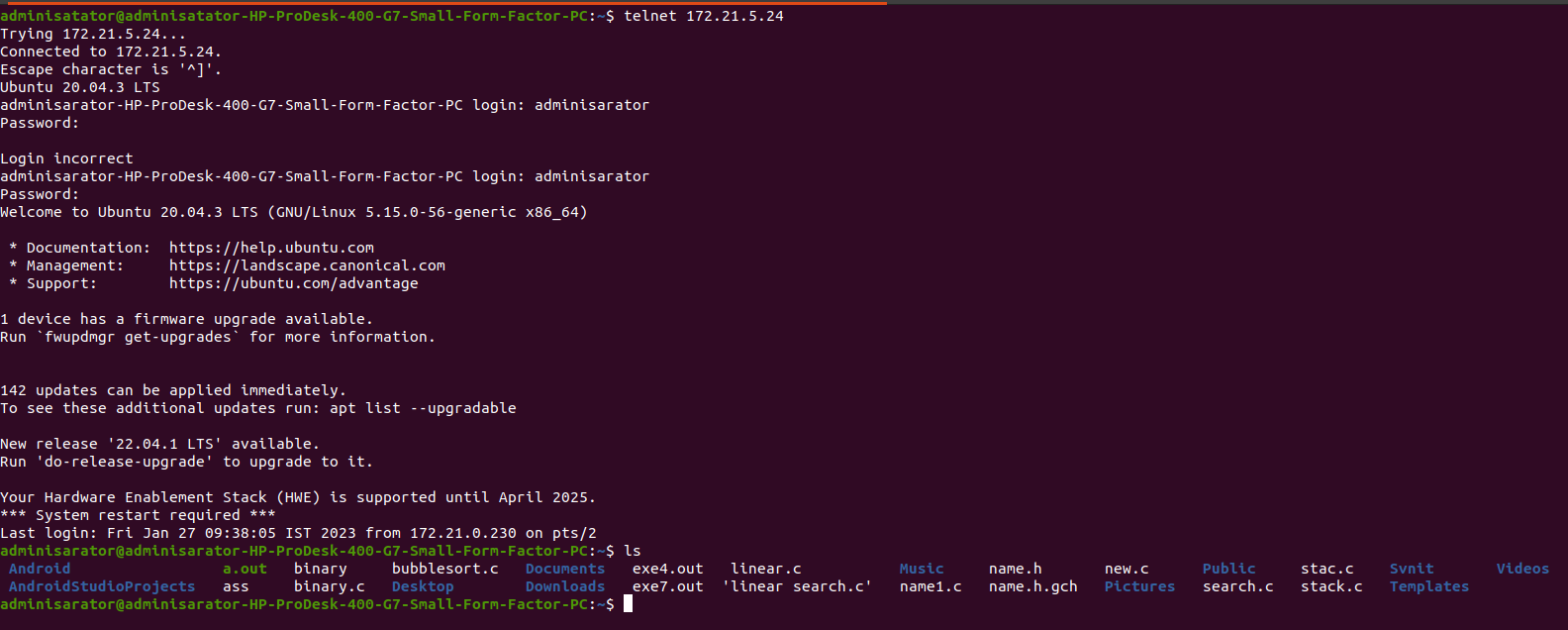
Step - 1: Created Telnet Server

****

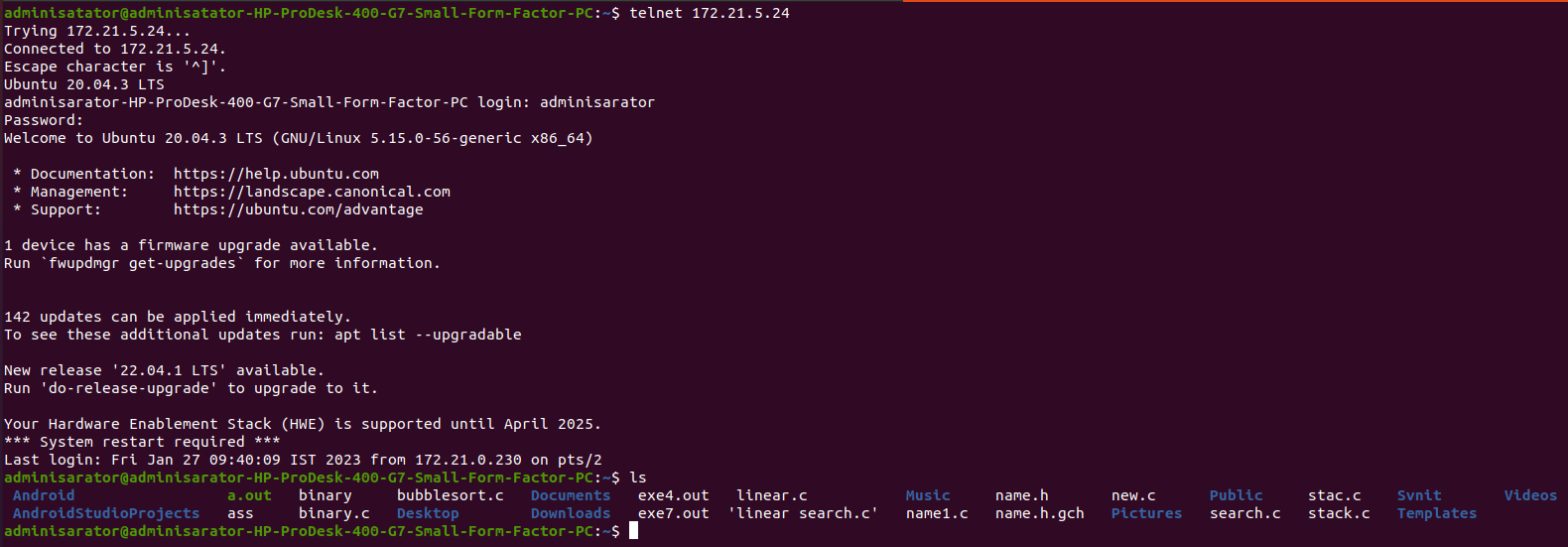
Step - 2: Tcpdump started on the server side.



Step - 3: Trying to connect to telnet server from client window1

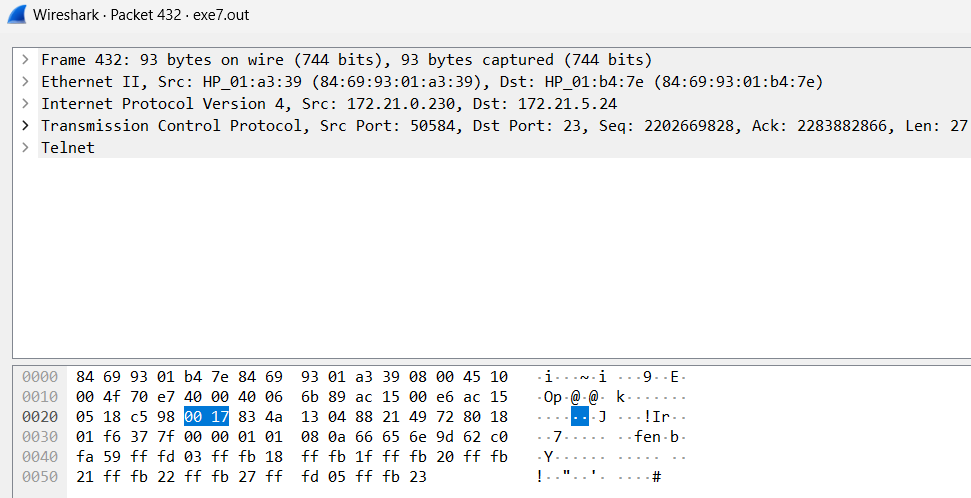


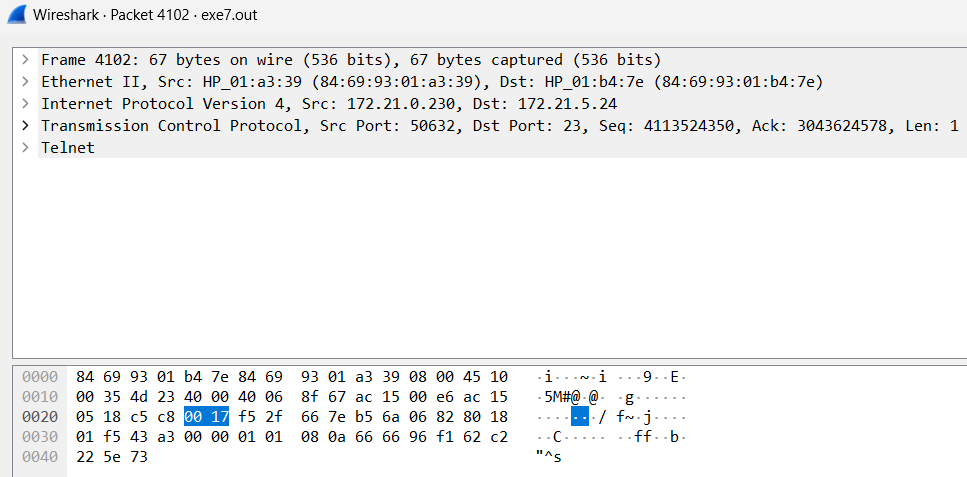
Step - 4: Trying to connect to telnet server from client window2



Step - 5: Analyze captured packets.

**Now answer the following questions:**

****

****

1. **When you have two telnet sessions with your machine, what port number is used on the remote machine? Are both sessions connected to the same port number on the remote machine?**

23 port number is used on the remote machine, when we have two telnet sessions with our machine. Yes, both sessions connected to the same port number on the remote machine.

1. **What port numbers are used in your machine for the first and second telnet, respectively?**

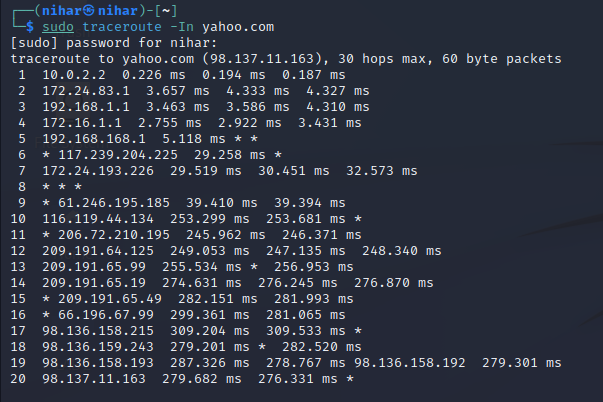
Port number used by the first telnet is 50584.

Port number used for the second telnet is 50632.

1. **What is the range of Internet-wide well-known port numbers? What is the range of well-known port numbers for Unix/Linux specific service? What is the range for a client port number? Compare your answer to the well-known port numbers defined in the /etc/services file. Are they consistent? In case they are not, try to discuss amongst peers and specify your view of the reason why they are not.**

**Note: In case telnet is not listed in the /etc/services file, use ssh.**

1. **Execute the traceroute command with www/yahoo.com as argument. Write down the IP address of yahoo.com that was used for the trace route. Determine the number of iterations required to determine route. Enlist the IP addresses of all the machines between the source and the destination. What is the average round trip time of the packet that reached the destination?**



Ip of yahoo.com: 98.137.11.163

Number of iterations required to determine route: 20

IP addresses of all the machines between the source and the destination:

1 10.0.2.2

2 172.24.83.1

3 192.168.1.1

4 172.16.1.1

5 192.168.168.1

6 \* 117.239.204.225

7 172.24.193.226

8 \* \* \*

9 \* 61.246.195.185

10 116.119.44.134

11 \* 206.72.210.195

12 209.191.64.125

13 209.191.65.99

14 209.191.65.19

15 \* 209.191.65.49

16 \* 66.196.67.99

17 98.136.158.215

18 98.136.159.243

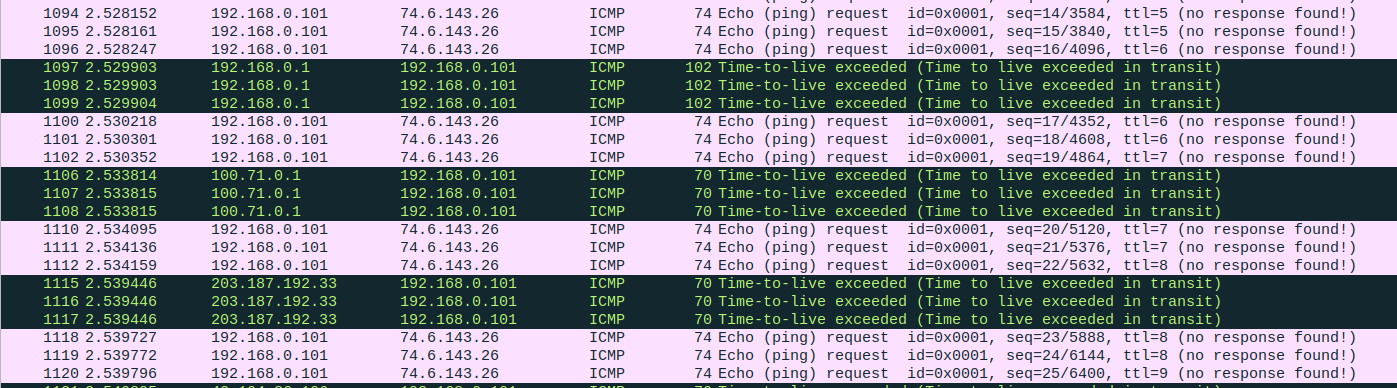
19 98.136.158.193

20 98.137.11.163

average round trip time of the packet: 236.953667

1. **With respect to the question above, run traceroute on one window of your OS and run tcpdump on the other window. Analyze the output of tcpdump. Answer the following questions giving appropriate high lighted snapshots in support of your answer :**
2. **How many packets are sent by traceroute in each iteration ? How can you prove this using the tcpdump output.**

Three packet send by traceroute in each iteration



1. **Consider one specific iteration of traceroute invocation/iteration. For this specific iteration, what are the individual round trip times of each of the three probes sent ? What is the average round trip time ? Does it match with the round trip time returned by traceroute ?**

Round trip time for router: 100.71.0.1

1st trip: 3.596 ms

2nd trip: 3.514 ms

3rd trip: 3.463 ms

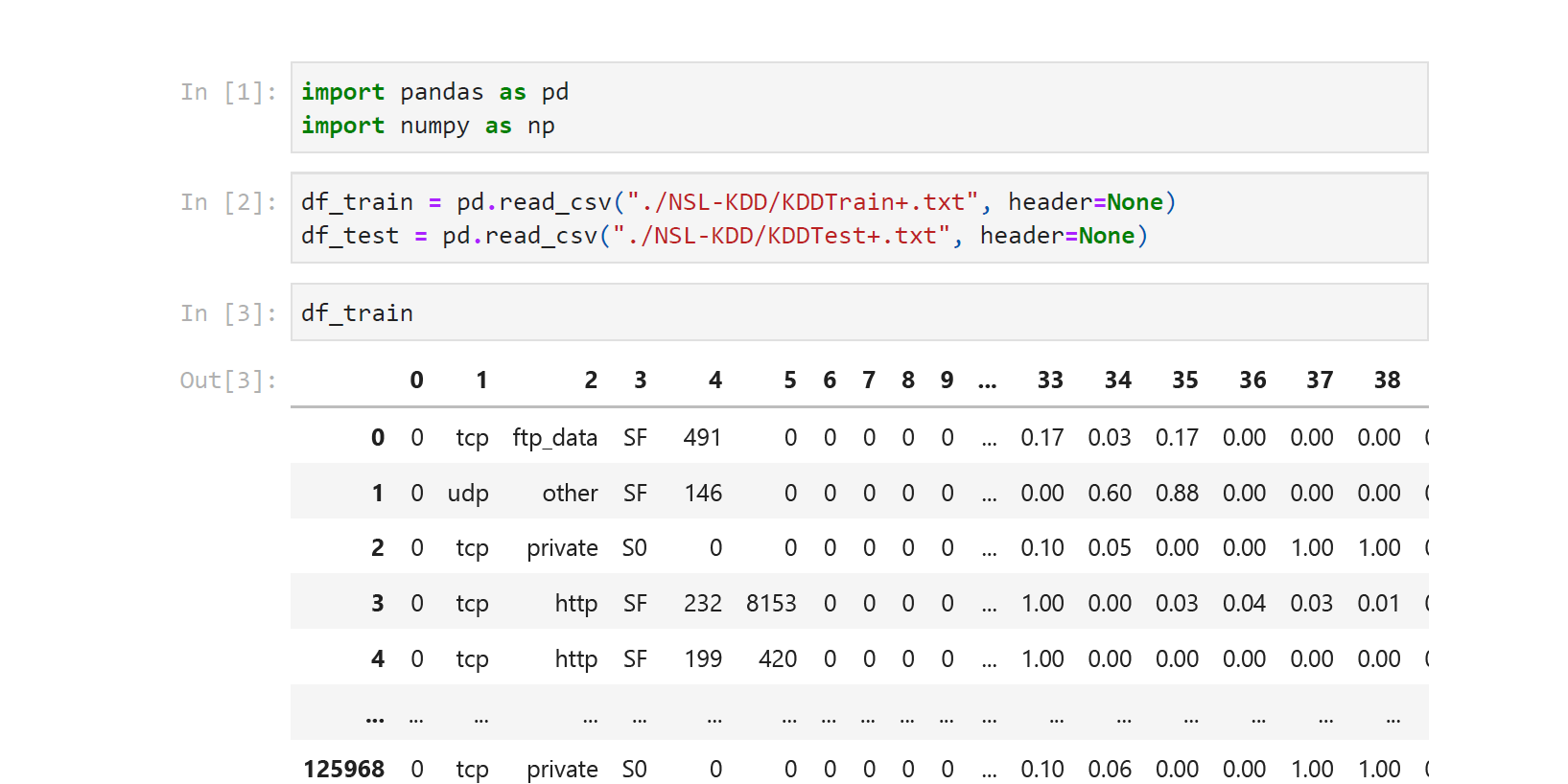
Avg: 3.525 ms

1. **In each iteration of traceroute does it use the same port number for the destination? IF yes, reason why and if no, then also argue why it does so ?**

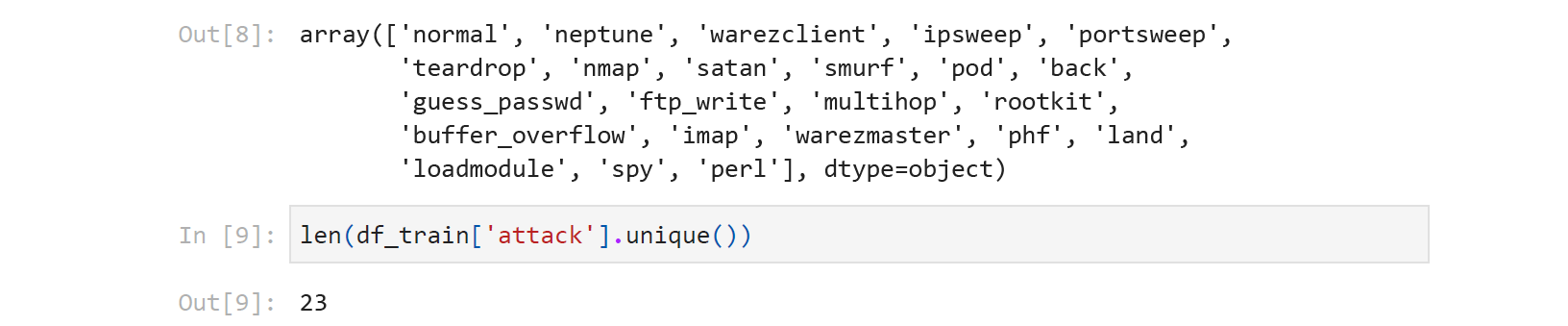
Traceroute send ICMP request and ICMP has no concept of ports

1. **Download, study and analyse the NSL-KDD dataset (ref: https://www.unb.ca/cic/datasets/nsl.html). Is this a labelled dataset or unlabelled dataset? Give reasons for your answer. How many samples are there in the training and that in the testing dataset? Identify various attributes in this dataset. How many attack types are specified in this dataset? Which ones? Use the figure below to learn the attack types and different categories and identify them in the dataset. The study here prepares you with the groundwork to undertake the first problem in the next assignment.**
2. This is labelled dataset: because Dataset have target field attack-type.
3. Training Sample: 125973, Testing Sample: 22544
4. No. attack type in dataset: 23

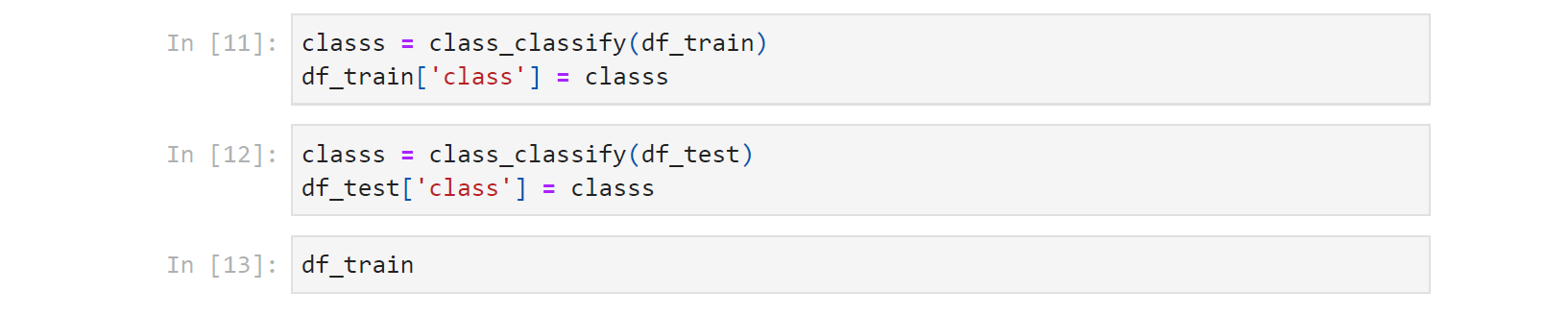
Attacks: normal, neptune, warezclient, ipsweep, portsweep, teardrop, nmap, satan, smurf, pod, back, guess\_passwd, ftp\_write, multihop, rootkit, buffer\_overflow, imap, warezmaster, phf, land, loadmodule, spy, perl

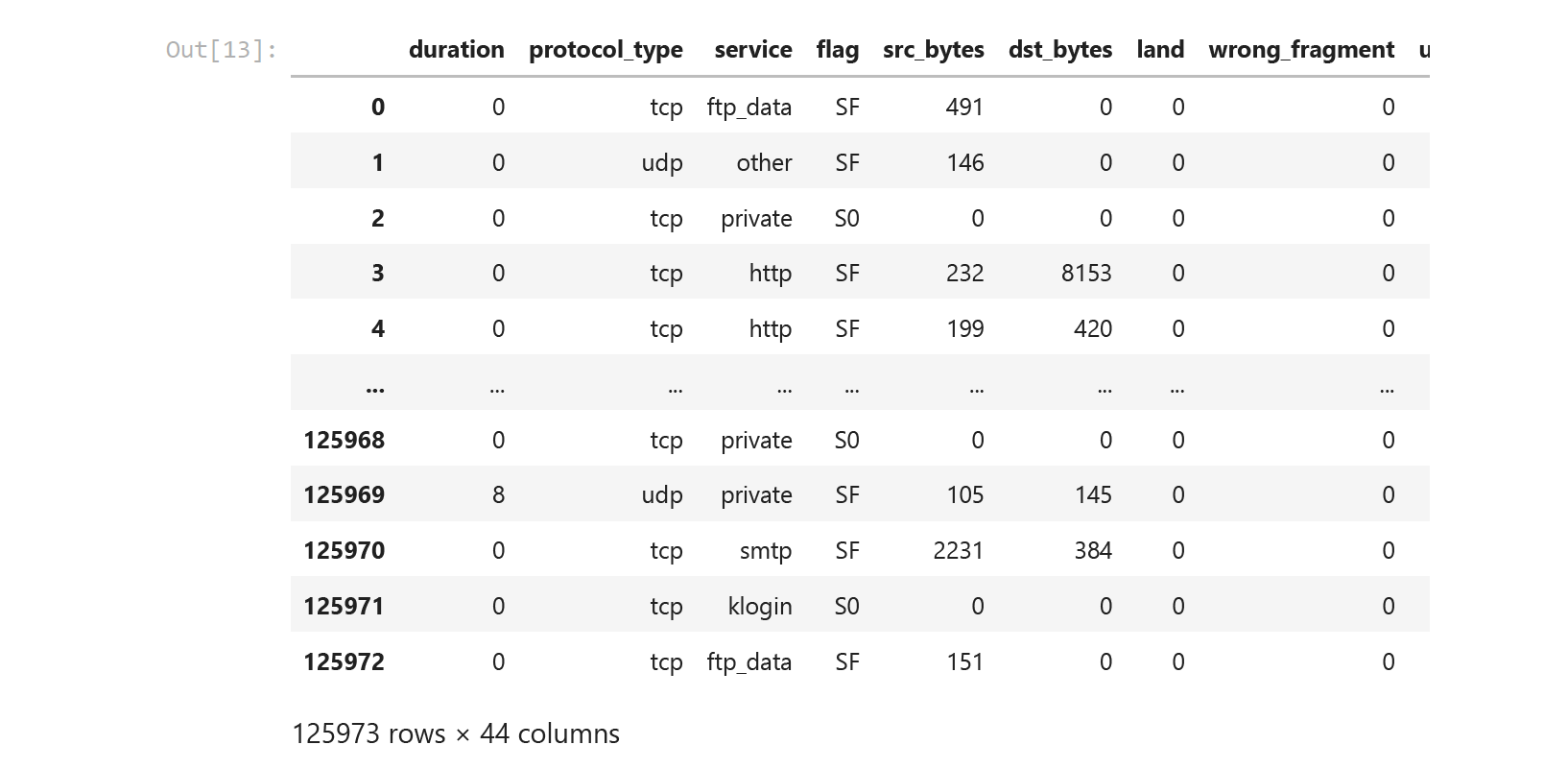


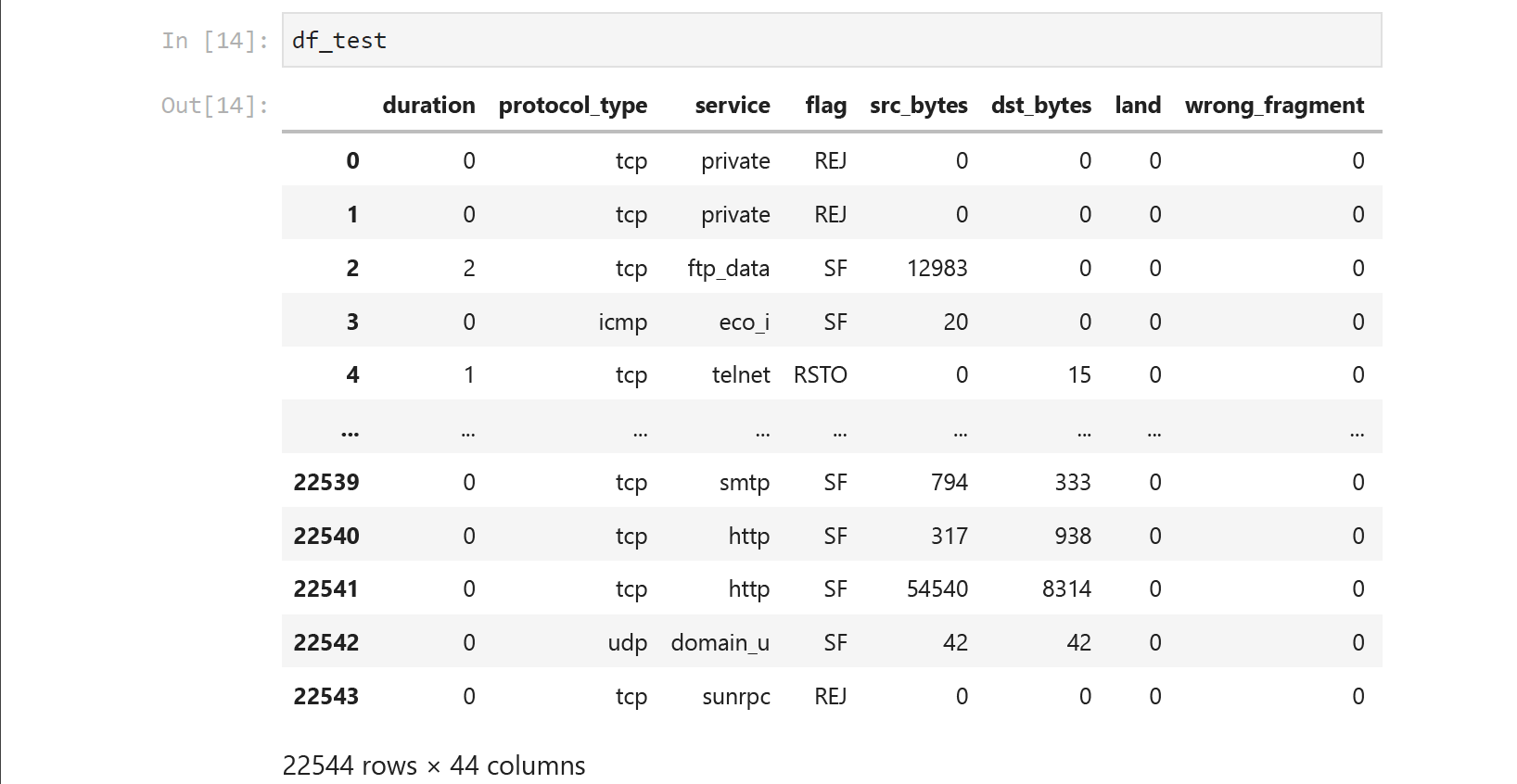
****

****

****

****

****

****