

CSP334: Computer Networks, Lab Assignment No 1, WIRESHARK

Rahul Byas Sherwan
Entry No. : 2016UCS0028

1: The First Problem

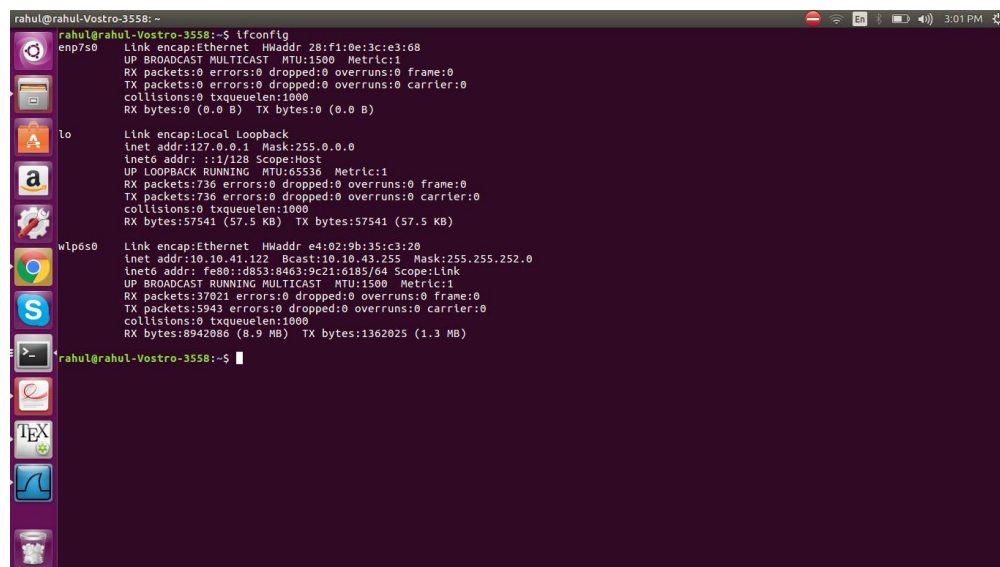
Network Interfaces Available:

Ethernet: enp7s0

Local: lo

Wireless Wifi: wlp6s0

Network selected for the experiment: wlp6s0



```
rahul@rahul-Vostro-3550:~$ ifconfig
enp7s0  Link encap:Ethernet  HWaddr 28:f1:0e:3c:e3:68
        UP BROADCAST MULTICAST  MTU:1500  Metric:1
        RX packets:0 errors:0 dropped:0 overruns:0 frame:0
        TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo      Link encap:Local Loopback
        inet addr:127.0.0.1  Mask:255.0.0.0
        inet6 addr: ::1/128 Scope:Host
        UP LOOPBACK RUNNING  MTU:65536  Metric:1
        RX packets:736 errors:0 dropped:0 overruns:0 frame:0
        TX packets:736 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:57541 (57.5 KB)  TX bytes:57541 (57.5 KB)

wlp6s0  Link encap:Ethernet  HWaddr e4:02:9b:35:c3:20
        inet addr:10.10.41.122  Bcast:10.10.43.255  Mask:255.255.252.0
        inet6 addr: fe80::d853:8463:9c21:6185/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:37021 errors:0 dropped:0 overruns:0 frame:0
        TX packets:5943 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:8942086 (8.9 MB)  TX bytes:1362025 (1.3 MB)

rahul@rahul-Vostro-3550:~$
```

Figure 1: Terminal

We can verify the network interfaces displayed in wireshark using ifconfig as shown above.

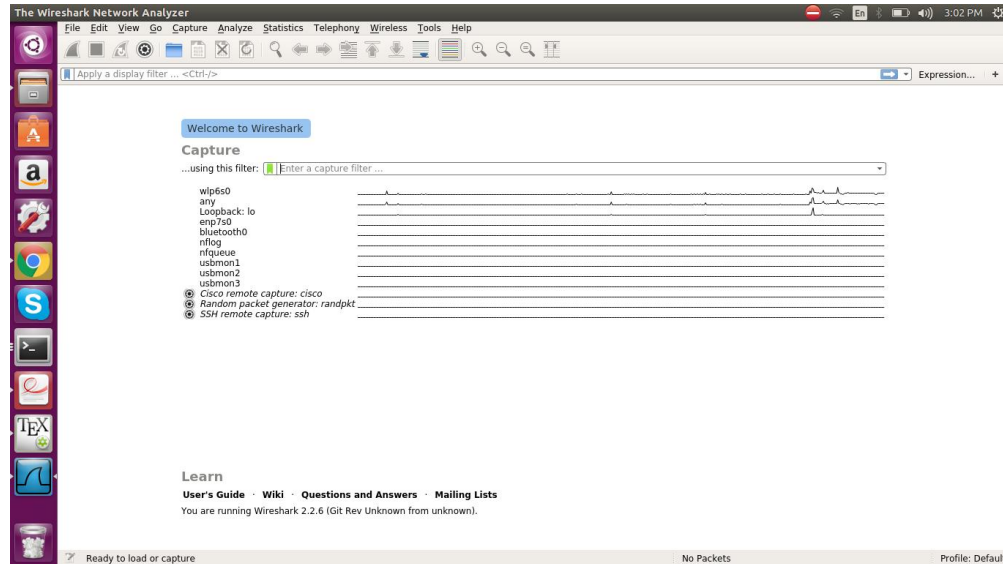


Figure 2: Wireshark Interface

2: The Second Problem

HTTP and DNS is used in this case as we can notice in the screenshot.

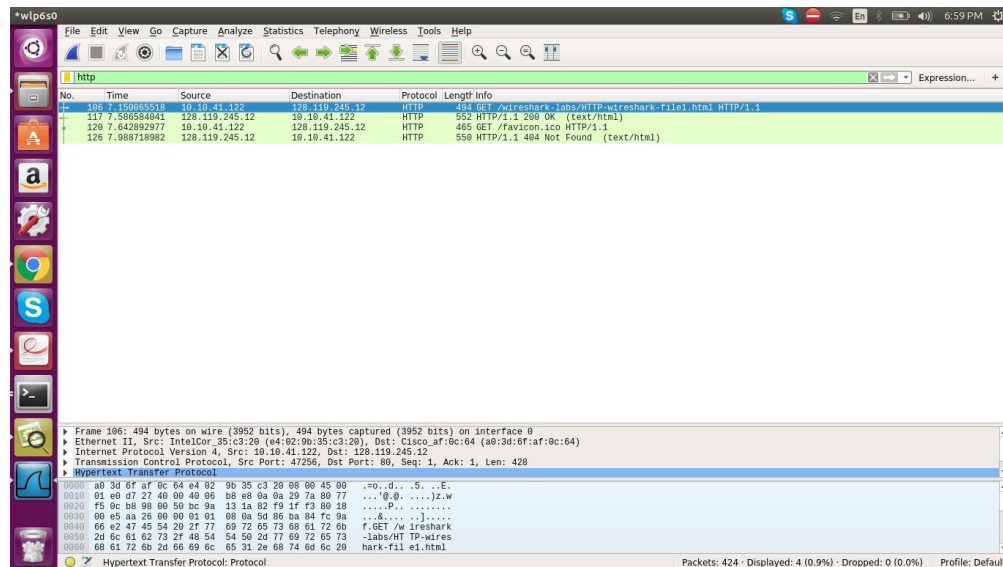


Figure 3: HTTP

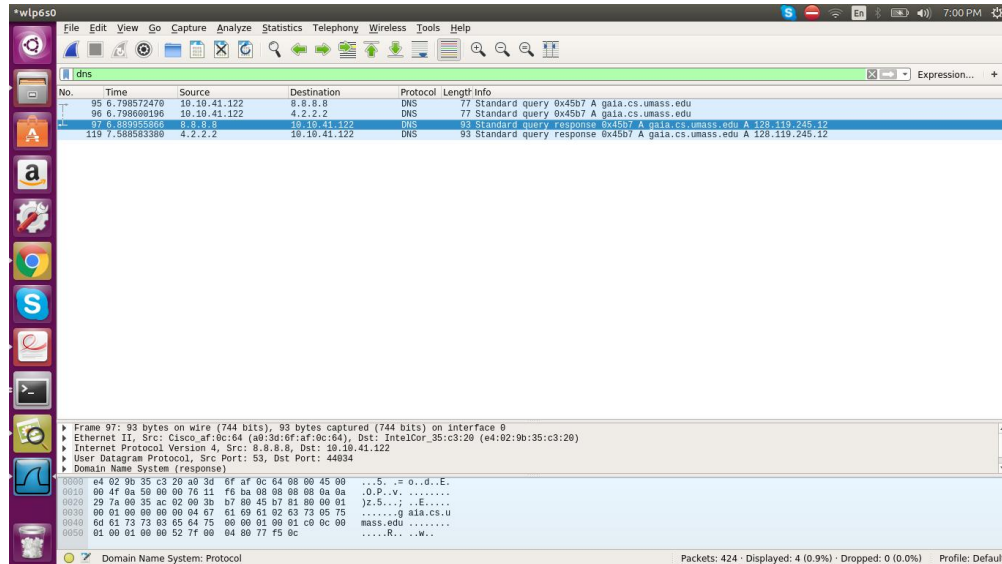


Figure 4: DNS

3: The Third Problem

The other protocols used and displayed in the unfiltered packet listing window of wireshark, besides the one that you answered in Q2 are :

(a) arp

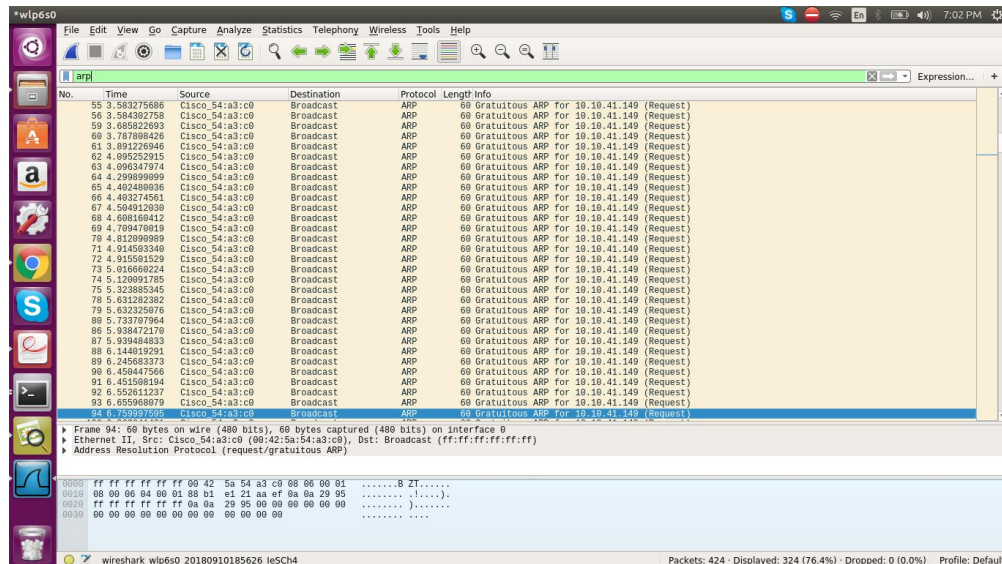


Figure 5: ARP

(b) tcp

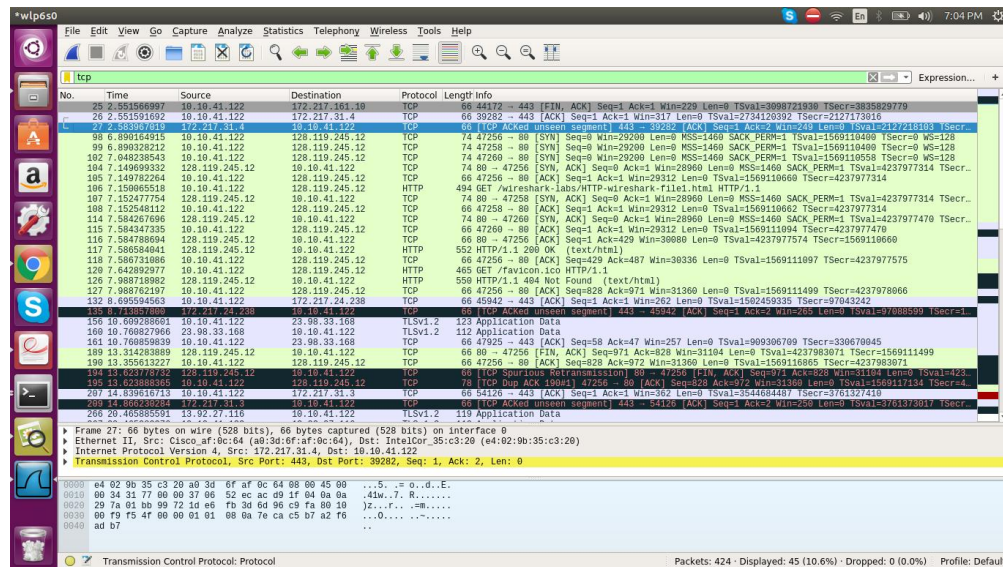


Figure 6: TCP

(c) quic

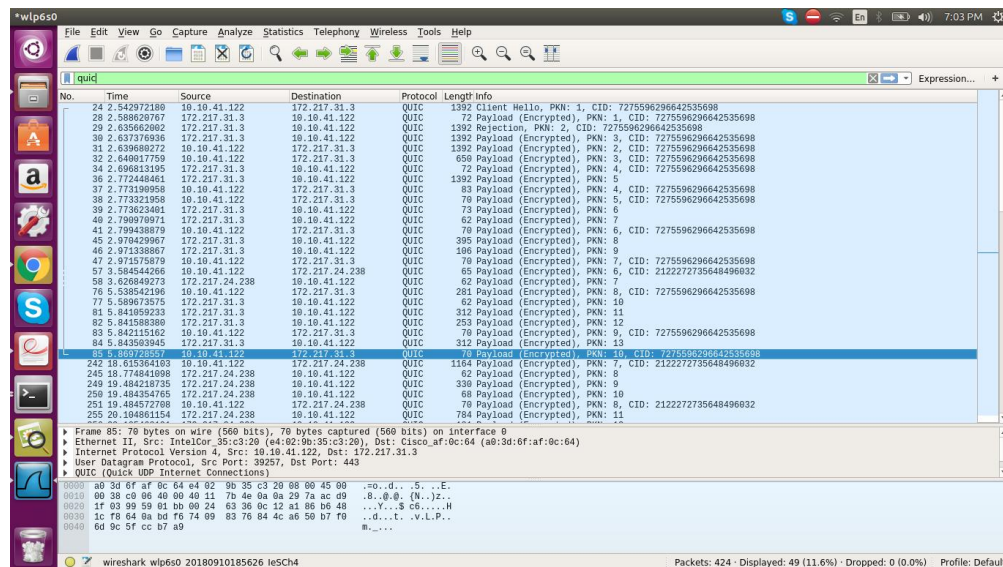
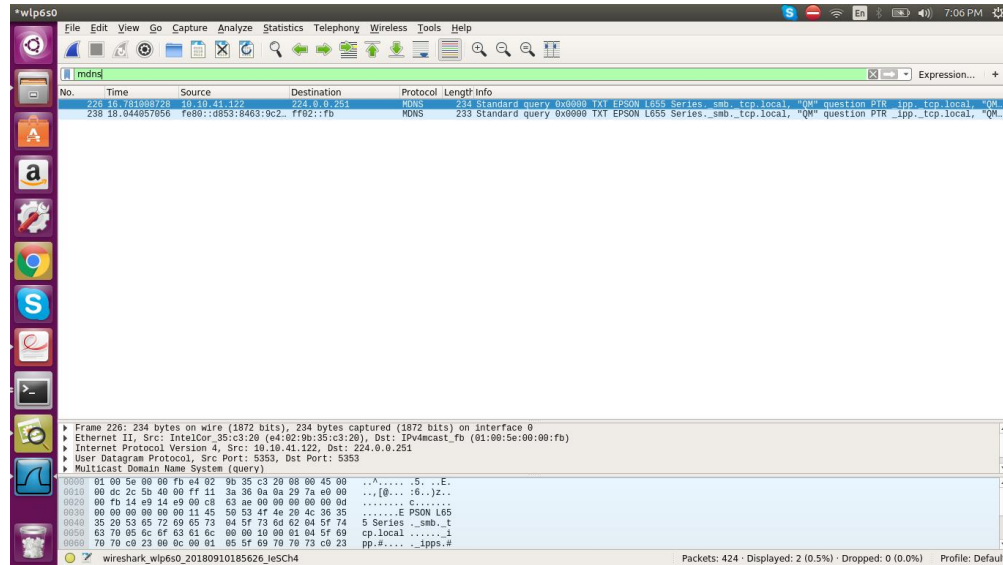
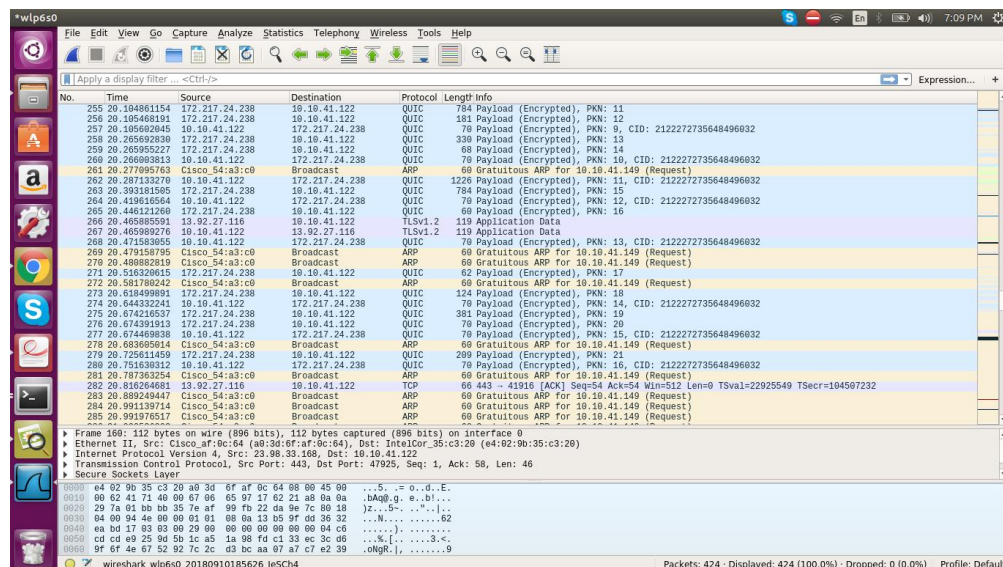


Figure 7: QUIC

(d) mdns



(e) tlsv1.2



4: The Fourth Problem

(a) What is the IPA of your machine? What is the IPA of the destination machine?

IPA of the source machine : 10.10.41.122

IPA of the destination machine :128.119.245.12

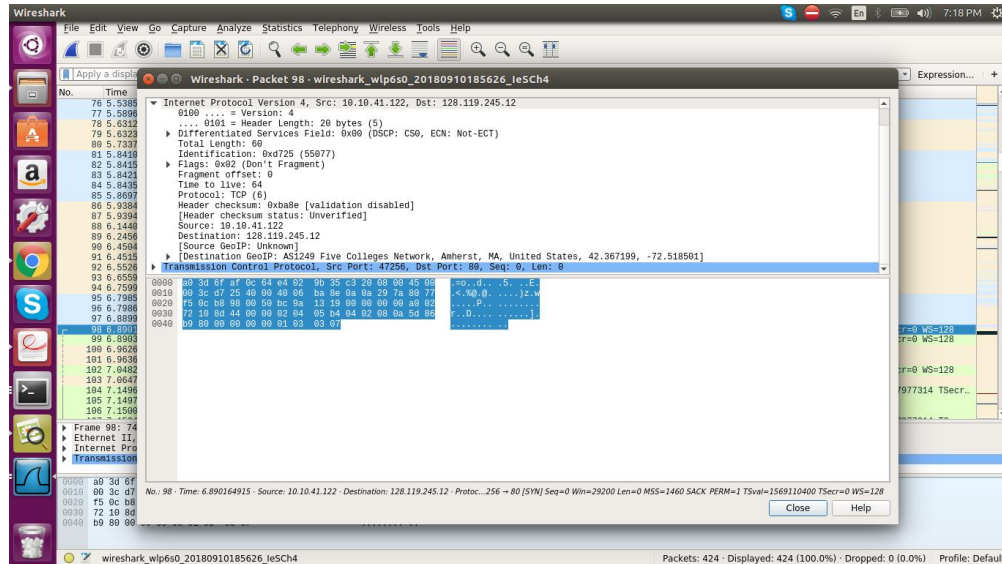


Figure 10: IPAs

(b) 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 ?

Yes , we can verify the IPA of the destination indeed is the same as that you observed in wireshark.

We can run traceroute gaia.cs.umass.edu on the terminal and can observe that the destination's IPA is same as that shown in wireshark.(i.e. 128.119.245.12)

Given below is the screenshot in which both wireshark destination's IPA is visible and the one which is there in the terminal. Both are same.

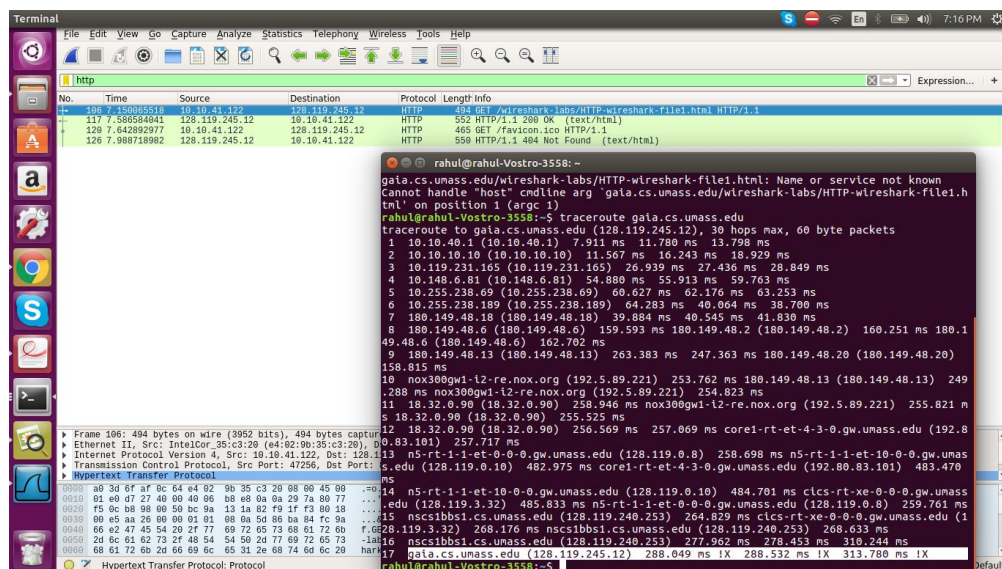


Figure 11: Traceroute

5: The Fifth Problem

IPA class of the source's machine : A (because it is 10.10.41.122 which is between 1.0.0.1 to 126.255.255.254)

IPA class of the destination's machine : B (because it is 128.119.245.12 which is between 128.1.0.1 to 191.255.255.254)

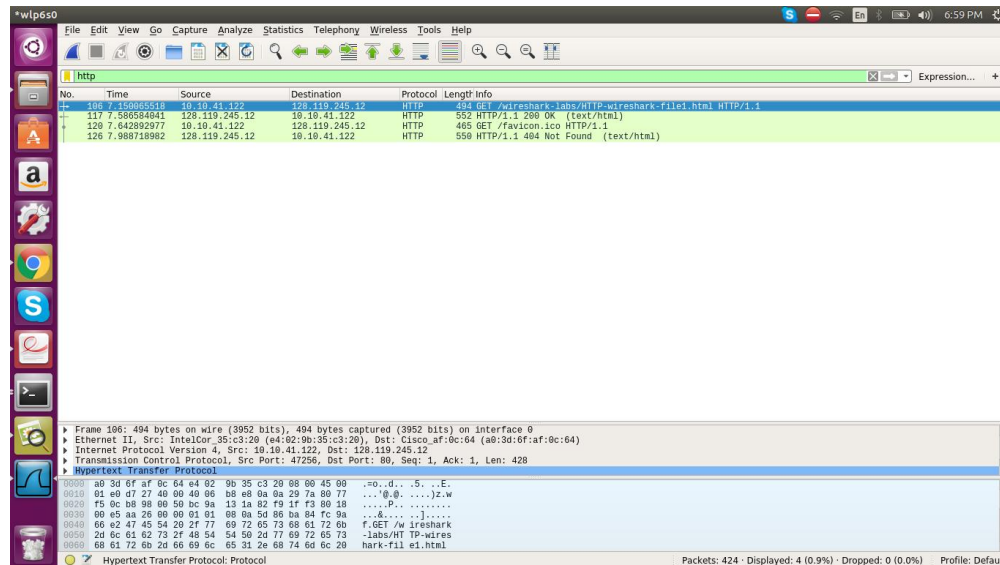


Figure 12: IPA's of src and dst

6: The Sixth Problem

Bits were captured in the packet are 592.

Time at which this packet captured is sept 10, 18:56:33.43 IST.

We can verify the same from screenshot given :

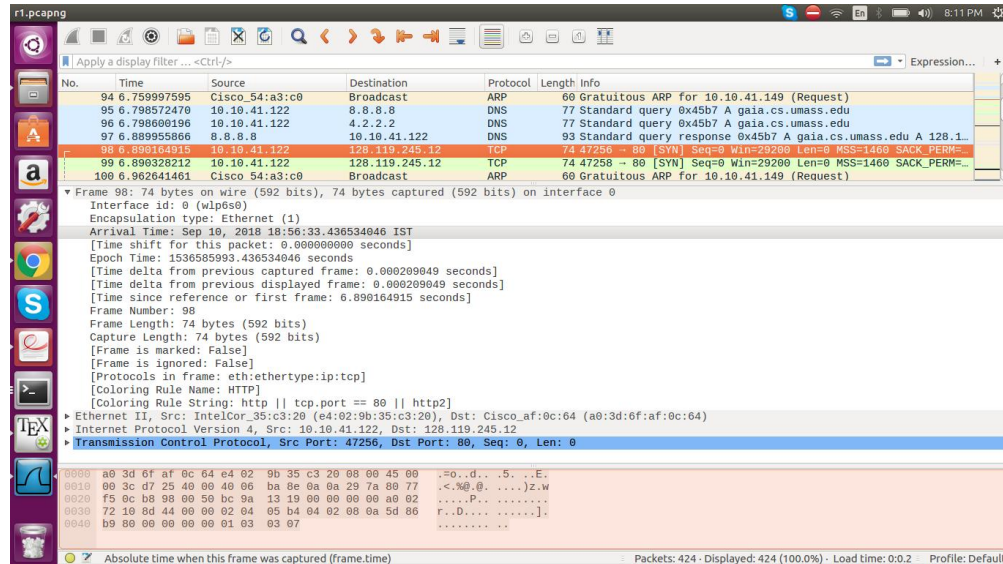


Figure 13: frame

7: The Seventh Problem

the interface id used is : 0 (wlp6s0)
 the address used by the interface is : e4:02:9b:35:c3:20
 (verify from screenshots above).

8: The Eighth Problem

The time network takes from when the HTTP GET message was sent until the HTTP OK reply received is : (Ok response received time) - (GET request time)
 = 18:56:34.132953172 - 18:56:33.696434649
 = 0.436518226 seconds

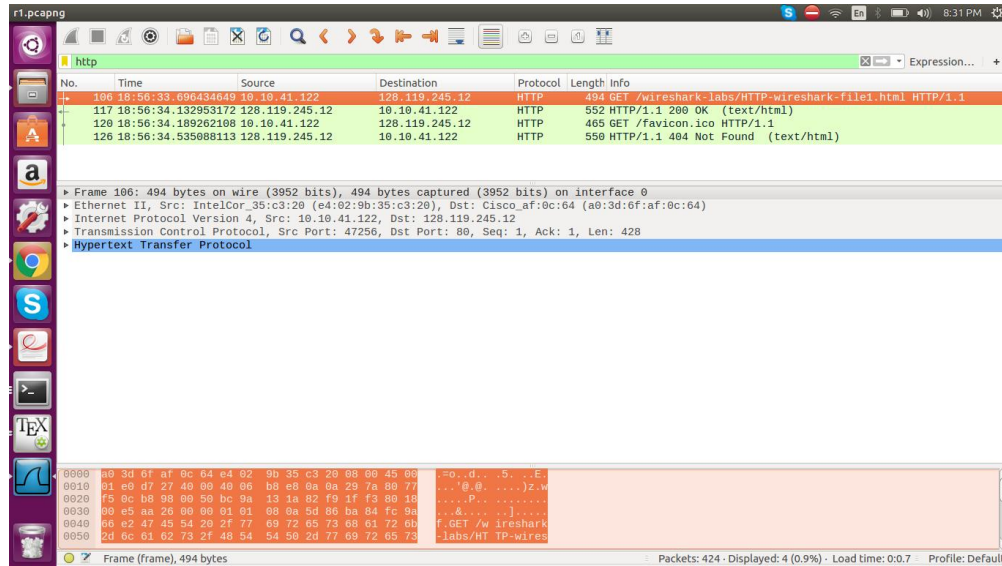


Figure 14: http GET-OK message

9: The Ninth Problem

There is no ninth question in the assignment.

10: The Tenth Problem

(a) Screenshot of HTTP GET message packet :



Figure 15: GET http packet

(b) Screenshot of HTTP OK message packet :



Figure 16: OK http packet

11: The Eleventh Problem

The destination physical address of the first packet captured is : a0:3d:6f:af:0c:64

The device is : cisco af:0c:64

We can find this information in Ethernet part of packet as shown in the screenshot. This is the first packet.

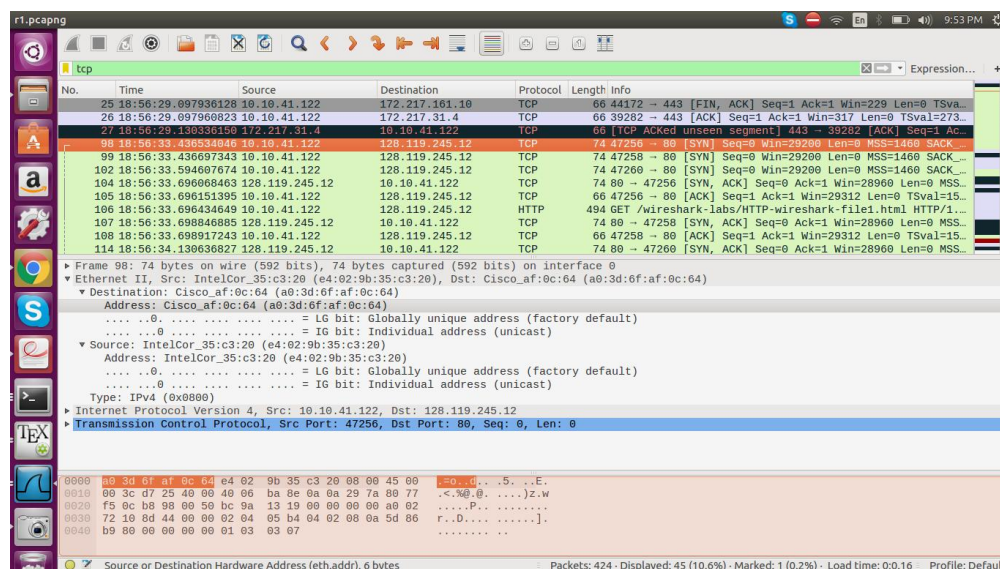


Figure 17: First packet captured

12: The twelfth Problem

74 bytes of header has been sent by the first frame. We can find it in frame header.

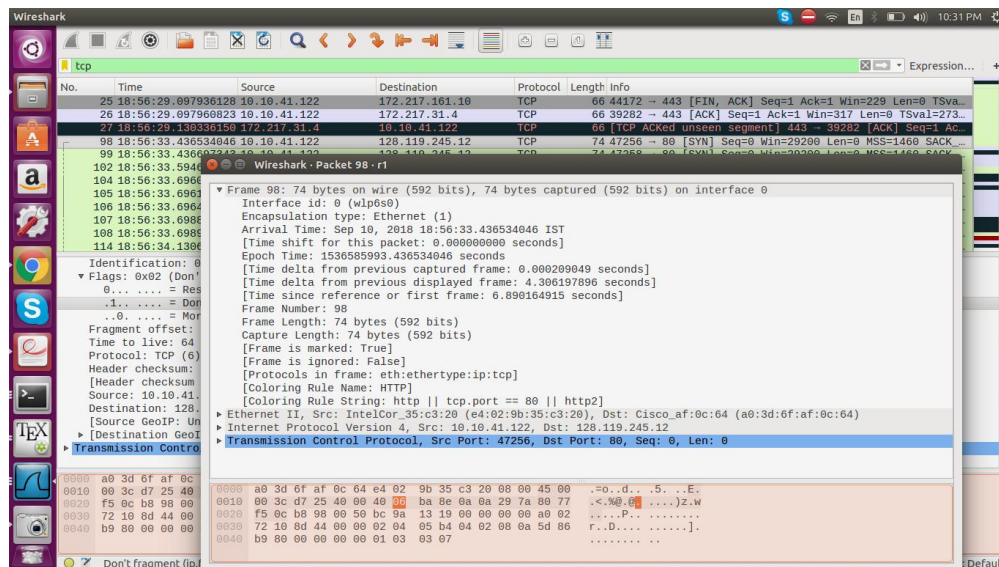


Figure 18: Size of first frame

13: The Thirteenth Problem

By looking at the Ethernet header of a frame, Yes we can determine that it contains an IP packet. As we can see in the screenshot that the Ethernet header contains IPv4 (0X0800) information which helps us in determining that it contains an IP packet.

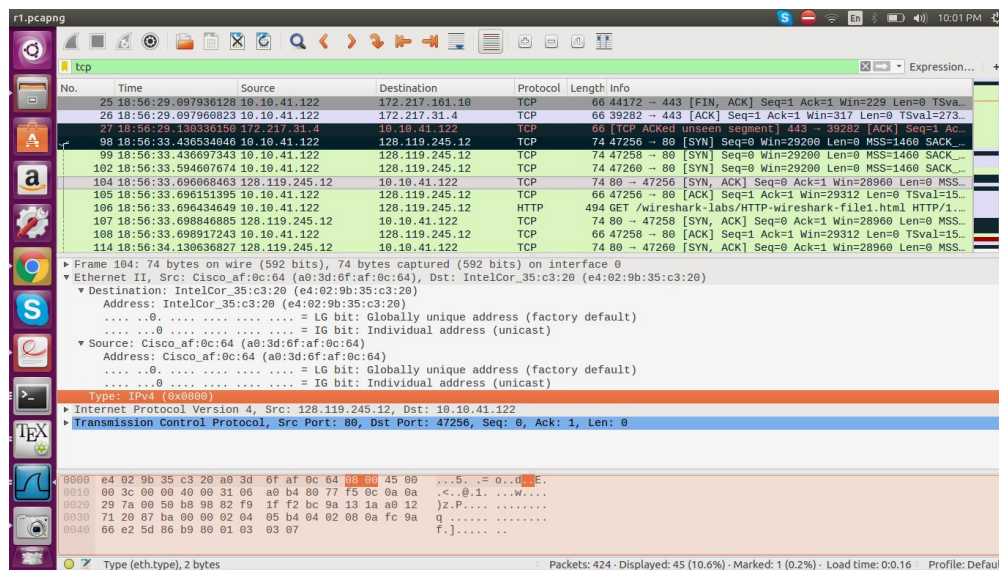


Figure 19: IPv4(0X0800)

14: The Fourteenth Problem

Yes, it is possible to know that the first packet captured has TCP or UDP as transport protocol by looking at the IP header. We can find this information in Protocol section of IP header. It says TCP (6) .

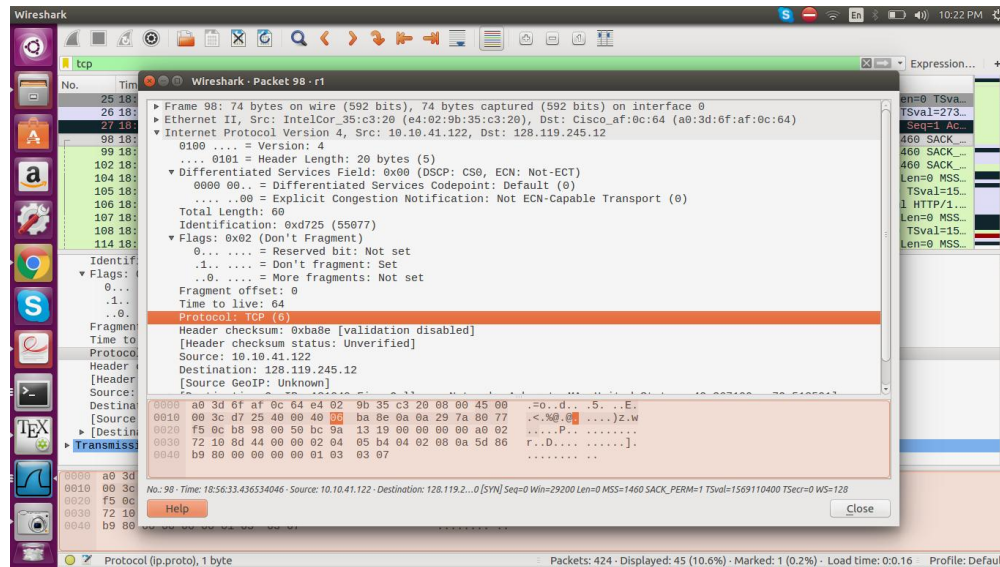


Figure 20: First tcp packet

15: The Fifteenth Problem

Source Port : 80

Destination Port : 47256

No, these are not the same for the client and the server. This is due to the fact that server works on well known port numbers while client work on emperhical port numbers.

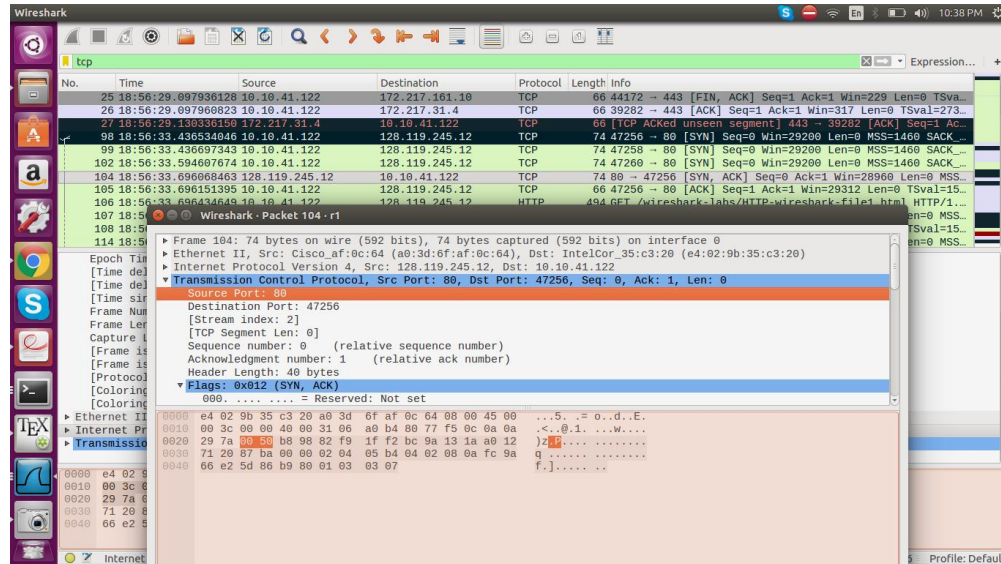


Figure 21: src , dst port

16: The Sixteenth Problem

The Server Hello message sent by the server have 1 sequence number, this is due to the fact that in wireshark systems has 1 as a default and relative sequence number . And the acknowledgment number as a relative of 185 because the number of bytes of packet adds to the sequence number and makes it an acknowledgment number. It implies that the original packet must be of 184 bytes which adds to 1 and make it 185

17: The Seventeenth Problem

sequence number : 2197364722

If every TCP connection started at zero, it would be trivial to generate a sequence number that was in the middle of someone else's connection. This would make connection hijacking easy.(help from quora)

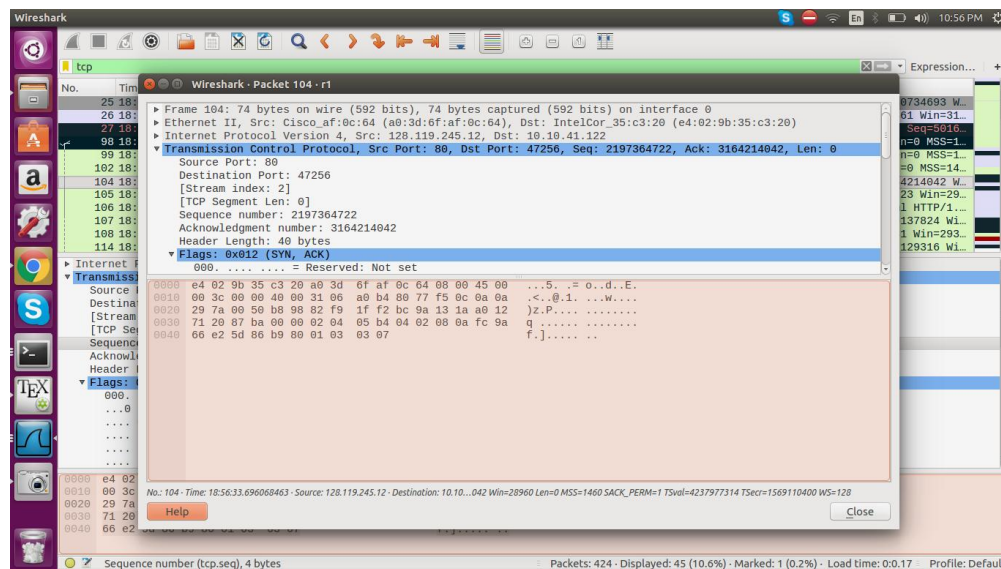


Figure 22: sequence number