Student Information

Full Name: Doruk Gerçel
ID Number: 2310027

Question 1

I couldn't see the whole path to the 'metu.edu.tr' (Figure 1). The IP address of the last hop that the program can reach is 144.122.1.18 (notice it is not the IP address of 'metu.edu.tr'). I reached this address in the ninth hop, but I couldn't see the IP address of the eight hop. The IP address of the eight hop is not seen and it is denoted with asterisks. According to the manual, asterisks are printed when the packets don't return within the expected time. Also traceroute program only shows the hops in the pathway (the network devices which decrement the TTLs of the packages), but may not show the real hosts. Therefore all the path is not seen as I can only see the hops (some of the hops in the path which didn't return packets in the expected time (asterisks) are not visible as-well). Also the program halted when it hit the max, therefore it didn't reach the host (which is the web server of 'metu.edu.tr'). So we didn't see the actual web server in the path as-well. Probably we couldn't reach the web server because of firewall or any other kind of filtering mechanism. (Please note that the program works in default with 30 TTLs)

```
doruk@doruk-HP-Pavilion-Notebook:~$ traceroute metu.edu.tr
traceroute to metu.edu.tr (144.122.145.153), 30 hops max, 60 byte packets
 1  _gateway (192.168.1.1)  5.187 ms  5.135 ms  6.497 ms
 2  host-212-57-0-235.reverse.superonline.net (212.57.0.235)  14.589 ms  22.944 ms  22.939 ms
 3  10.40.22.153 (10.40.22.153)  19.529 ms  20.782 ms  24.543 ms
 4  10.34.255.81 (10.34.255.81)  24.453 ms  24.446 ms  26.542 ms
 5  10.34.255.86 (10.34.255.86)  29.721 ms  32.054 ms  32.481 ms
 6  10.38.207.137 (10.38.207.137)  33.862 ms  5.998 ms  10.545 ms
 7  10.40.145.85 (10.40.145.85)  7.298 ms  11.136 ms  11.120 ms
 8  * * *
 9  144.122.1.18 (144.122.1.18)  29.034 ms  29.003 ms  29.891 ms
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *
```

Figure 1: Screenshot of 'traceroute metu.edu.tr' command from the terminal.

Question 2

The default method of route tracing is sending probe packet which are UDP datagrams. These datagrams include destination port numbers which are called "unlikely" (they are called "unlikely" as they are expected to be unused). These probe's destination port numbers start from port number 33434 and this number is incremented in each probe. In my Wireshark capture (Figure 2) my computer (IP source address 192.168.1.26) starts sending probe packets which are UDP datagrams. The first probe datagram sent (has package number 5) is sent to the port number 33434 as-well. Also this port number is incremented in each probe datagram after the first one. (The destination port number can be observed in the info field (the right hand side of the arrow).)

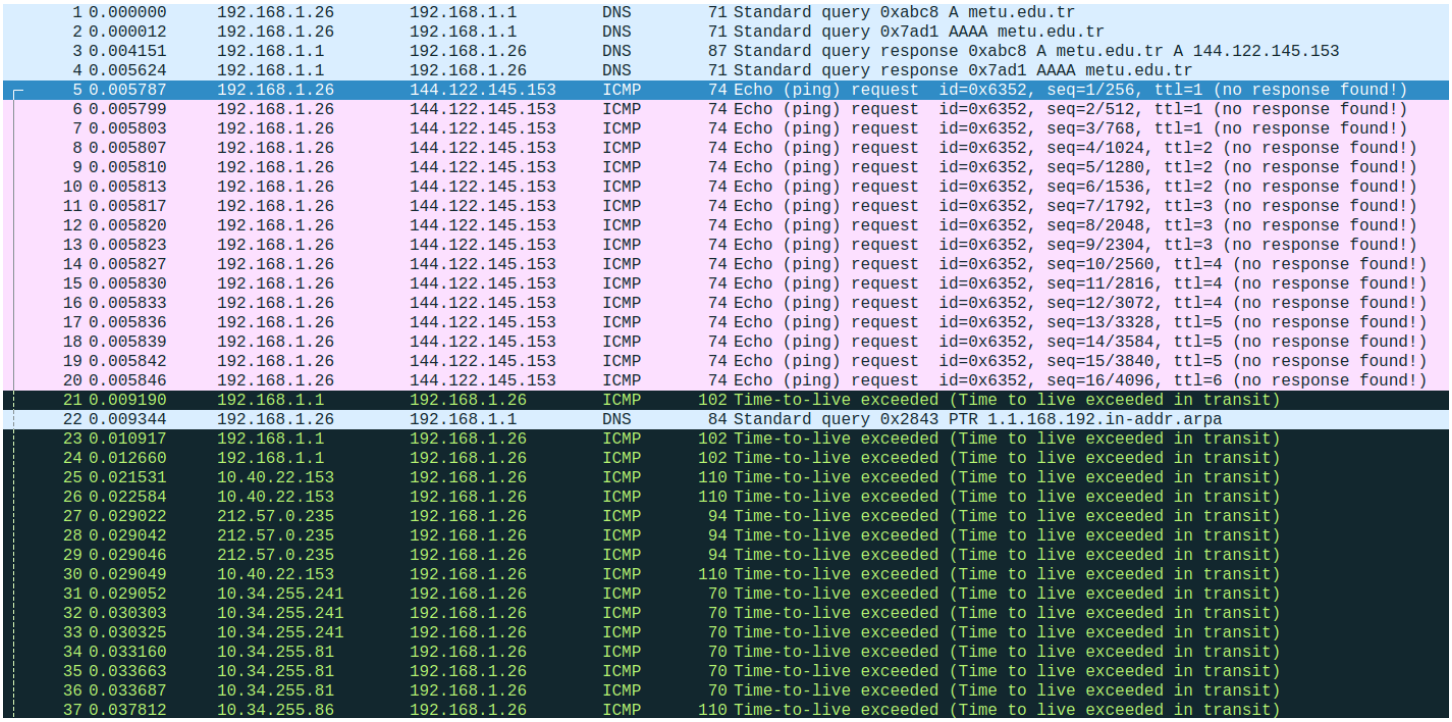
1	0.000000	192.168.1.26	192.168.1.1	DNS	71	Standard query 0x5ae1 A metu.edu.tr
2	0.000035	192.168.1.26	192.168.1.1	DNS	71	Standard query 0xfefc AAAA metu.edu.tr
3	0.007441	192.168.1.1	192.168.1.26	DNS	87	Standard query response 0x5ae1 A metu.edu.tr A 144.122.145.153
4	4.372691	192.168.1.1	192.168.1.26	DNS	122	Standard query response 0xfefc AAAA metu.edu.tr SOA ns1.metu.edu.tr
5	4.373083	192.168.1.26	144.122.145.153	UDP	74	42601 → 33434 Len=32
6	4.373159	192.168.1.26	144.122.145.153	UDP	74	59385 → 33435 Len=32
7	4.373207	192.168.1.26	144.122.145.153	UDP	74	53413 → 33436 Len=32
8	4.373250	192.168.1.26	144.122.145.153	UDP	74	44043 → 33437 Len=32
9	4.373295	192.168.1.26	144.122.145.153	UDP	74	50302 → 33438 Len=32
10	4.373340	192.168.1.26	144.122.145.153	UDP	74	50313 → 33439 Len=32
11	4.373385	192.168.1.26	144.122.145.153	UDP	74	52562 → 33440 Len=32
12	4.373428	192.168.1.26	144.122.145.153	UDP	74	34897 → 33441 Len=32
13	4.373472	192.168.1.26	144.122.145.153	UDP	74	37181 → 33442 Len=32
14	4.373515	192.168.1.26	144.122.145.153	UDP	74	38425 → 33443 Len=32
15	4.373562	192.168.1.26	144.122.145.153	UDP	74	56976 → 33444 Len=32
16	4.373606	192.168.1.26	144.122.145.153	UDP	74	43261 → 33445 Len=32
17	4.373649	192.168.1.26	144.122.145.153	UDP	74	60209 → 33446 Len=32
18	4.373691	192.168.1.26	144.122.145.153	UDP	74	39957 → 33447 Len=32
19	4.373733	192.168.1.26	144.122.145.153	UDP	74	41020 → 33448 Len=32
20	4.373778	192.168.1.26	144.122.145.153	UDP	74	35149 → 33449 Len=32
21	4.378229	192.168.1.1	192.168.1.26	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
22	4.378278	192.168.1.1	192.168.1.26	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
23	4.378650	192.168.1.26	192.168.1.1	DNS	84	Standard query 0x51e6 PTR 1.1.168.192.in-addr.arpa
24	4.379693	192.168.1.1	192.168.1.26	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
25	4.387829	212.57.0.235	192.168.1.26	ICMP	94	Time-to-live exceeded (Time to live exceeded in transit)
26	4.392904	10.40.22.153	192.168.1.26	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
27	4.394200	10.40.22.153	192.168.1.26	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
28	4.396229	212.57.0.235	192.168.1.26	ICMP	94	Time-to-live exceeded (Time to live exceeded in transit)
29	4.396269	212.57.0.235	192.168.1.26	ICMP	94	Time-to-live exceeded (Time to live exceeded in transit)
30	4.397958	10.34.255.81	192.168.1.26	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
31	4.397996	10.34.255.81	192.168.1.26	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
32	4.398005	10.40.22.153	192.168.1.26	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
33	4.400138	10.34.255.81	192.168.1.26	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
34	4.403361	10.34.255.86	192.168.1.26	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
35	4.405736	10.34.255.86	192.168.1.26	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
36	4.406205	10.34.255.86	192.168.1.26	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
37	4.407630	10.38.207.137	192.168.1.26	ICMP	182	Time-to-live exceeded (Time to live exceeded in transit)
38	5.051714	IntelCor_2f:32:44	HuaweiTe_1e:20:64	ARP	42	Who has 192.168.1.1? Tell 192.168.1.26
39	5.053751	HuaweiTe_1e:20:64	IntelCor_2f:32:44	ARP	42	192.168.1.1 is at 24:09:95:1e:20:64

Figure 2: Screenshot of Wireshark capture of 'traceroute metu.edu.tr' command.

Question 3

One of the major differences is that with the '-I' flag, the program sends ICMP echo packets for probes, instead of default UDP datagrams. It is clearly visible in the Wireshark capture (Figure 3) below (protocol field for the probe packages are ICMP). Also we can see that from the terminal window (Figure 4) that, if we run the traceroute program wiht '-I' flag, there is one new destination added to the path. After the hop with IP address '10.40.22.153' the packet moves to the hop with the IP address '10.34.255.241' with the '-I' flag, but without this flag, packet bypasses this new hop and directly moves to the hop with the IP address '10.34.255.81'. Because of this difference, without the flag, we reach the last hop

with a package whose TTL is equal to 9, but with the '-I' flag, we reach it with a package whose TTL is equal to 10. The reason for this situation is that, some of the hops reject the UDP datagrams so they don't send any replies (but they send reply to ICMP probes). Therefore they are not printed in the output. In our situation, the hop with the IP address '10.34.255.241' doesn't send any reply to UDP datagram probes, but send reply to ICMP probes.



1	0.000000	192.168.1.26	192.168.1.1	DNS	71	Standard query 0xabc8 A metu.edu.tr
2	0.000012	192.168.1.26	192.168.1.1	DNS	71	Standard query 0x7ad1 AAAA metu.edu.tr
3	0.004151	192.168.1.1	192.168.1.26	DNS	87	Standard query response 0xabc8 A metu.edu.tr A 144.122.145.153
4	0.005624	192.168.1.1	192.168.1.26	DNS	71	Standard query response 0x7ad1 AAAA metu.edu.tr
5	0.005787	192.168.1.26	144.122.145.153	ICMP	74	Echo (ping) request id=0x6352, seq=1/256, ttl=1 (no response found!)
6	0.005799	192.168.1.26	144.122.145.153	ICMP	74	Echo (ping) request id=0x6352, seq=2/512, ttl=1 (no response found!)
7	0.005803	192.168.1.26	144.122.145.153	ICMP	74	Echo (ping) request id=0x6352, seq=3/768, ttl=1 (no response found!)
8	0.005807	192.168.1.26	144.122.145.153	ICMP	74	Echo (ping) request id=0x6352, seq=4/1024, ttl=2 (no response found!)
9	0.005810	192.168.1.26	144.122.145.153	ICMP	74	Echo (ping) request id=0x6352, seq=5/1280, ttl=2 (no response found!)
10	0.005813	192.168.1.26	144.122.145.153	ICMP	74	Echo (ping) request id=0x6352, seq=6/1536, ttl=2 (no response found!)
11	0.005817	192.168.1.26	144.122.145.153	ICMP	74	Echo (ping) request id=0x6352, seq=7/1792, ttl=3 (no response found!)
12	0.005820	192.168.1.26	144.122.145.153	ICMP	74	Echo (ping) request id=0x6352, seq=8/2048, ttl=3 (no response found!)
13	0.005823	192.168.1.26	144.122.145.153	ICMP	74	Echo (ping) request id=0x6352, seq=9/2304, ttl=3 (no response found!)
14	0.005827	192.168.1.26	144.122.145.153	ICMP	74	Echo (ping) request id=0x6352, seq=10/2560, ttl=4 (no response found!)
15	0.005830	192.168.1.26	144.122.145.153	ICMP	74	Echo (ping) request id=0x6352, seq=11/2816, ttl=4 (no response found!)
16	0.005833	192.168.1.26	144.122.145.153	ICMP	74	Echo (ping) request id=0x6352, seq=12/3072, ttl=4 (no response found!)
17	0.005836	192.168.1.26	144.122.145.153	ICMP	74	Echo (ping) request id=0x6352, seq=13/3328, ttl=5 (no response found!)
18	0.005839	192.168.1.26	144.122.145.153	ICMP	74	Echo (ping) request id=0x6352, seq=14/3584, ttl=5 (no response found!)
19	0.005842	192.168.1.26	144.122.145.153	ICMP	74	Echo (ping) request id=0x6352, seq=15/3840, ttl=5 (no response found!)
20	0.005846	192.168.1.26	144.122.145.153	ICMP	74	Echo (ping) request id=0x6352, seq=16/4096, ttl=6 (no response found!)
21	0.009190	192.168.1.1	192.168.1.26	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
22	0.009344	192.168.1.26	192.168.1.1	DNS	84	Standard query 0x2843 PTR 1.1.168.192.in-addr.arpa
23	0.010917	192.168.1.1	192.168.1.26	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
24	0.012660	192.168.1.1	192.168.1.26	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
25	0.021531	10.40.22.153	192.168.1.26	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
26	0.022584	10.40.22.153	192.168.1.26	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
27	0.029022	212.57.0.235	192.168.1.26	ICMP	94	Time-to-live exceeded (Time to live exceeded in transit)
28	0.029042	212.57.0.235	192.168.1.26	ICMP	94	Time-to-live exceeded (Time to live exceeded in transit)
29	0.029046	212.57.0.235	192.168.1.26	ICMP	94	Time-to-live exceeded (Time to live exceeded in transit)
30	0.029049	10.40.22.153	192.168.1.26	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
31	0.029052	10.34.255.241	192.168.1.26	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
32	0.030303	10.34.255.241	192.168.1.26	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
33	0.030325	10.34.255.241	192.168.1.26	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
34	0.033160	10.34.255.81	192.168.1.26	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
35	0.033663	10.34.255.81	192.168.1.26	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
36	0.033687	10.34.255.81	192.168.1.26	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
37	0.037812	10.34.255.86	192.168.1.26	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)

Figure 3: Screenshot of Wireshark capture of 'traceroute metu.edu.tr -I' command.

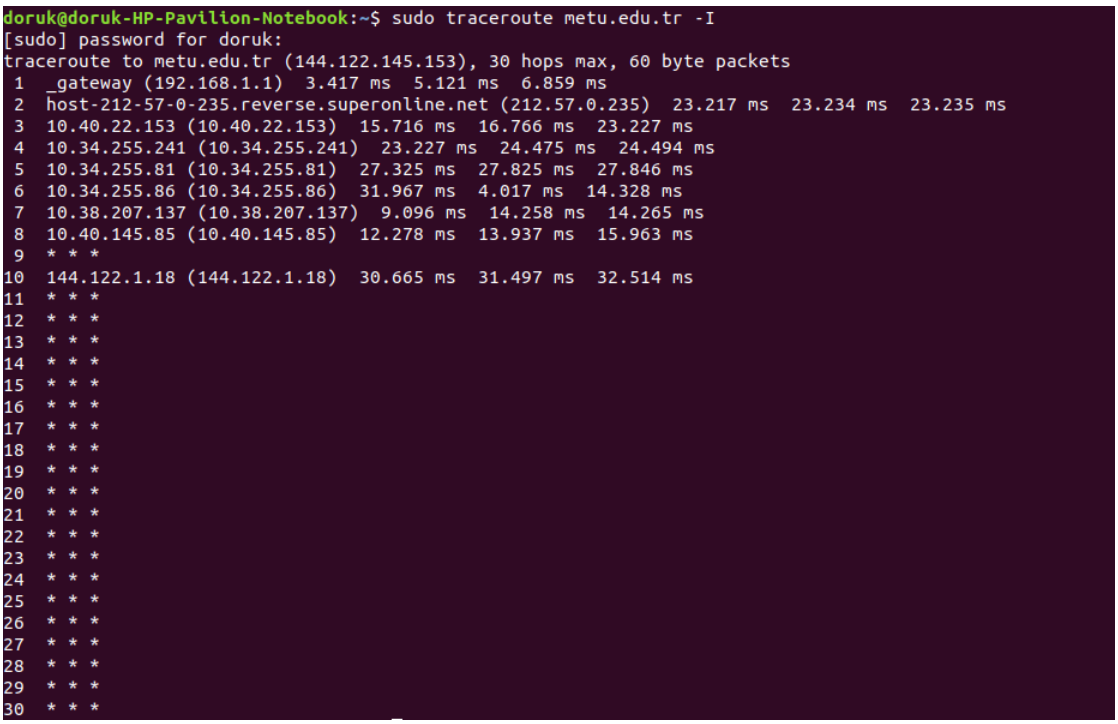


Figure 4: Screenshot of 'traceroute metu.edu.tr -I' command from the terminal.

Question 4

I chose 'National University of Tres de Febrero' from Argentina and 'Sultan Idris Education University' from Malaysia. Traceroute program can directly reach the servers of the universities' websites. The IP address of the 'National University of Tres de Febrero's website server (untref.edu.ar) is '192.169.226.87' (Figure 5) and the IP address of the 'Sultan Idris Education University's website server (upsi.edu.my) is '103.228.53.122' (Figure 6).

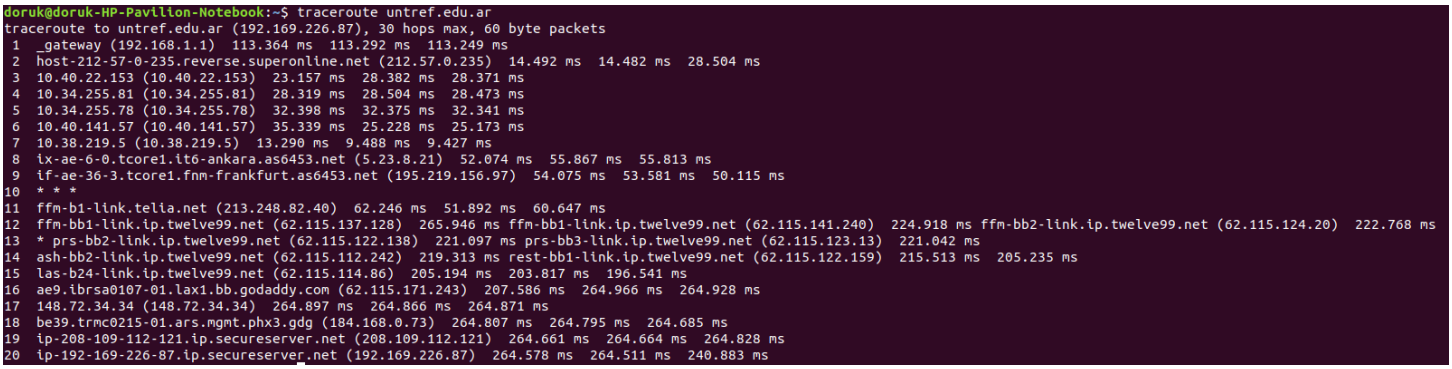


Figure 5: Screenshot of 'traceroute untref.edu.ar' command from the terminal.

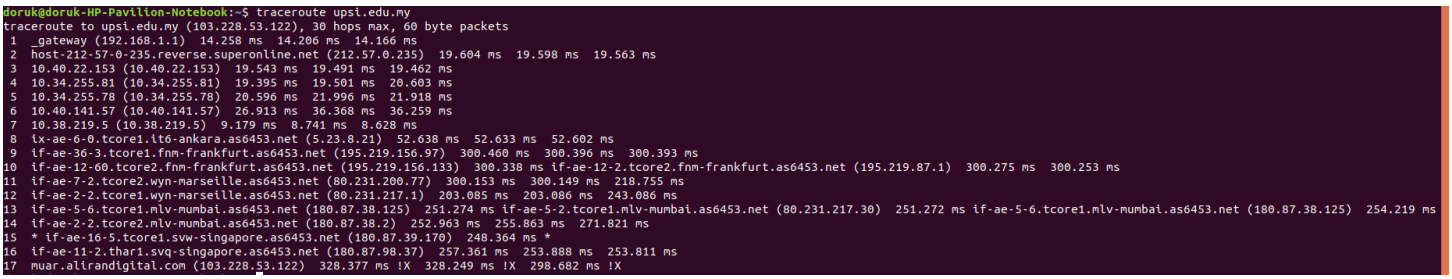


Figure 6: Screenshot of 'traceroute upsi.edu.my' command from the terminal.

Question 4 (Bonus Part)

I tried to find path to website of National University of the West ('uno.edu.ar'). Without using any flags I couldn't reach the website server (Figure 7). This is because the default method of traceroute is sending UDP packets to unlikely destination ports, therefore they may be filtered by the firewalls. In order to bypass these firewalls I used '-T' flag. By using this flag, traceroute program sends 'TCP SYN' as a probe, therefore the destination ports of the web server can't understand whether this packet was a probe or not (therefore bypasses the firewall). Also I used another option '-p 80'. This option changes the destination port of the probe packets. I looked at the website of the university and noticed that it uses a HTTP web server (noticed from the URL of the website). Therefore in order to reach this website, I needed to send probes to the HTTP port of this server, therefore I changed the destination of the probes to port 80 (HTTP uses port 80 by default). By using these options, I reached the webserver of the university (Figure 8).

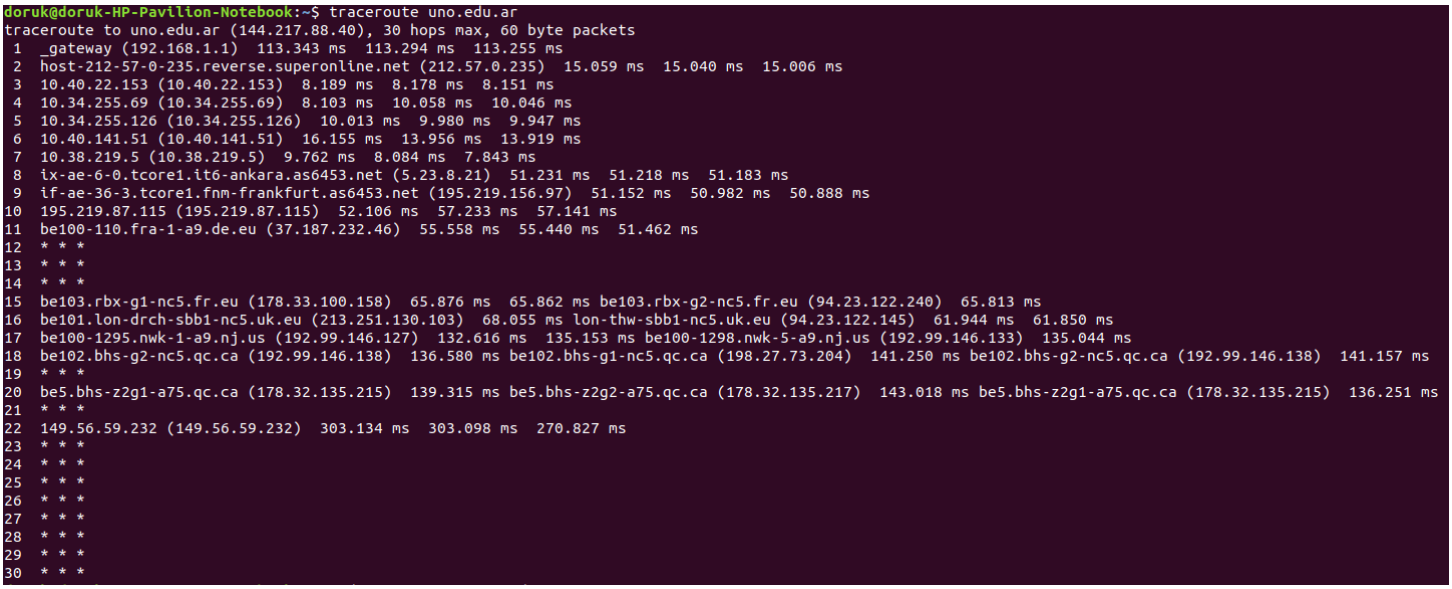


Figure 7: Screenshot of 'traceroute uno.edu.ar' command from the terminal.

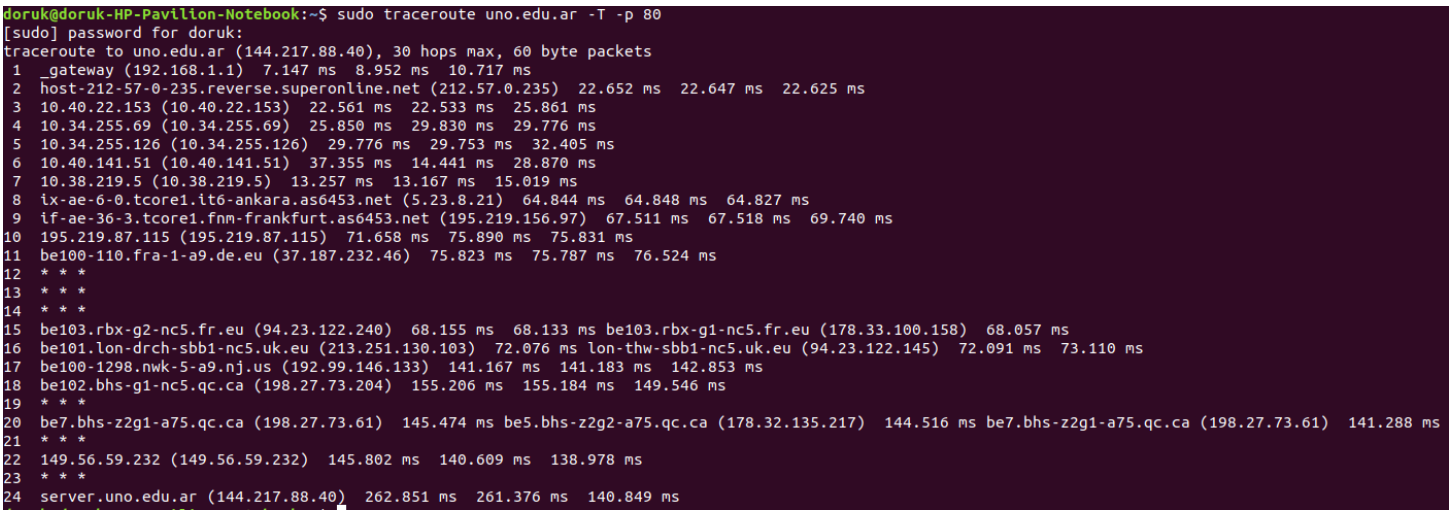


Figure 8: Screenshot of 'traceroute uno.edu.ar -T -p 80' command from the terminal.

Question 5

The value of the IPv4 protocol field is ICMP(1) (Figure 9). This makes sense as we know that traceroute program sends ICMP probe packages with '-I' flag.

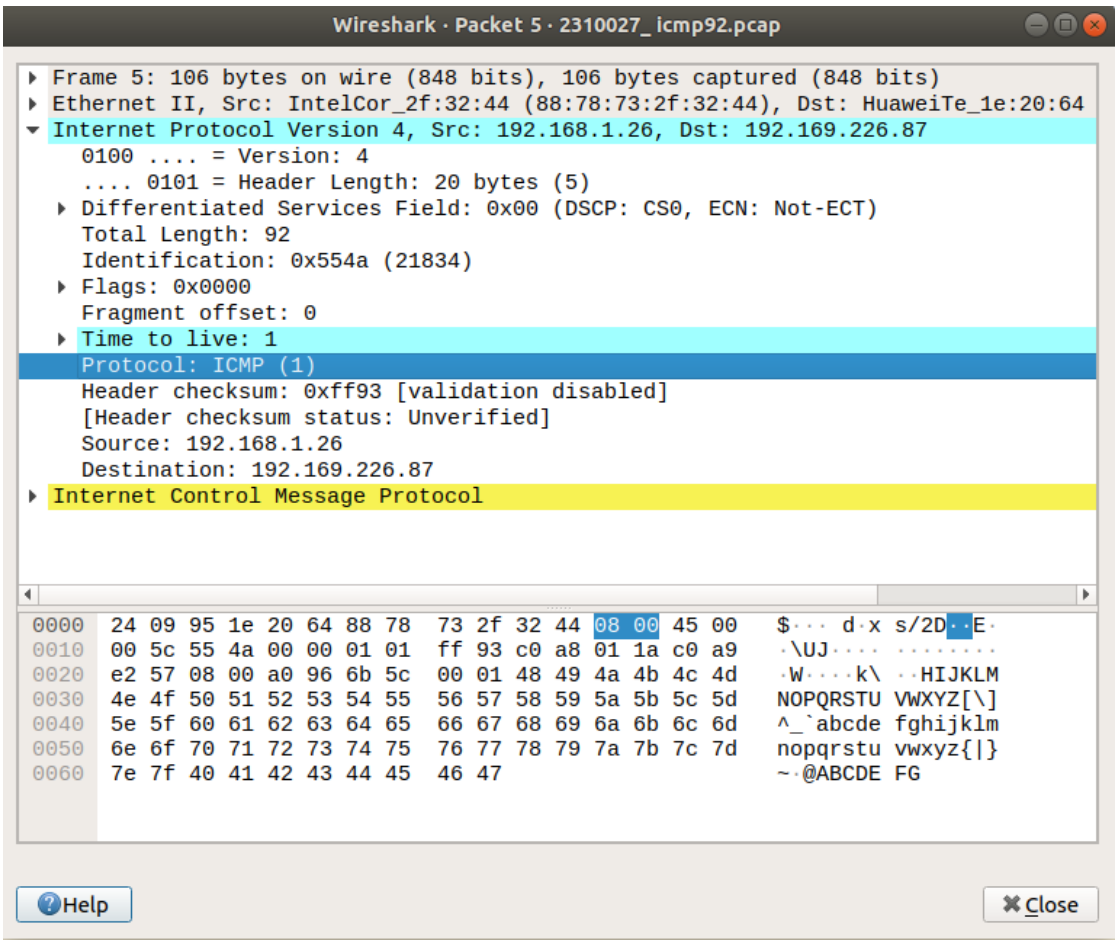


Figure 9: Screenshot of the first ICMP package sent from my computer (Protocol field highlighted).

Question 6

By looking at the IP header we can learn the length of the IP header ('Header Length' field of IP header). Also we can learn the total length of IP header and payload together ('Total Length' field of IP header). From the Figure 10, we can observe that the length of the IP header is 20 bytes, and the total length of IP header and payload together is 92 bytes. Therefore length of the payload is 72 bytes (92 bytes - 20 bytes).

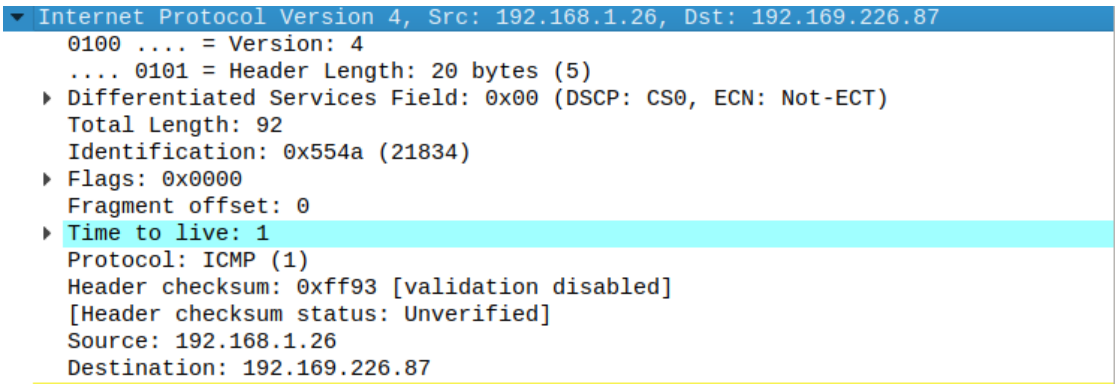


Figure 10: Screenshot of the first ICMP package sent from my computer (Only the IP header).

Question 7

I analyzed the TTL exceeded packet (Figure 11) with the lowest number (no. 21). The value in the Identification field is 58125 and the value in the TTL field is 64. Also it is noticeable that there are three consecutive TTL exceeded packets sent from the same host (this is because traceroute program sends three probe packets to each hop). Among the TTL exceeded packets, TTL is constant between the packets which are sent from the same host (packets have the same source address). But these TTL values are different between the packets which are sent from the different hosts. Identification fields are incremented by one between the packets which are sent from the same host (for example, Identification field of the first TTL exceeded packet sent from the source '192.168.1.1' is 58125 and the following packet's Identification field is 58126), but value of this field between different sources are independent from each other.

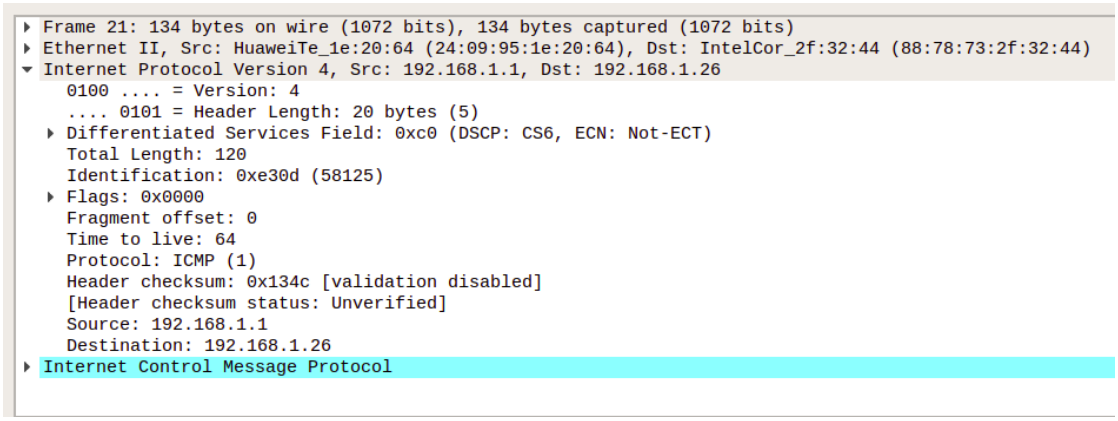


Figure 11: Screenshot of the TTL-Exceeded packet with the lowest number.

Question 8

The first ICMP Echo Request (fragemented to Frame 5, Frame 6 and Frame 7) fragment is Frame 5. When we look to the 'Flags' field from IP Header of Frame 5,

we can see that 'More Fragments' bit is set (Figure 12). Therefore we can observe that this datagram is fragmented.

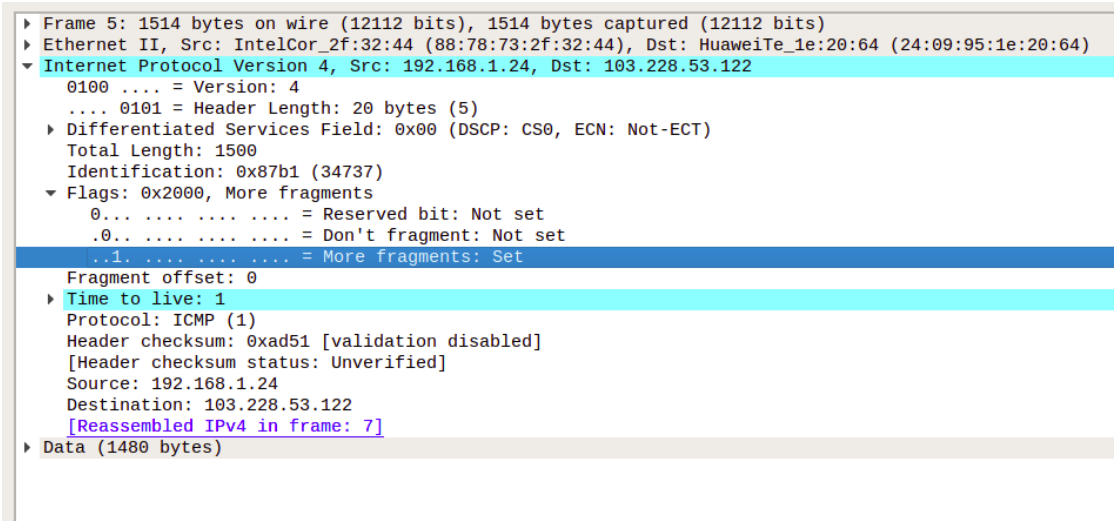


Figure 12: First fragmented datagram of the first ICMP Echo Request ('More fragments' bit in the 'Flags' field is highlighted).

Question 9

By looking at the IP header of the first datagram (first fragmented packet), we can't tell how many fragments are created, we can just tell whether the packet is fragmented or not (notice that Wireshark also writes that the fragments are reassembled in frame: 7 in Figure 12). We can only learn how many fragments are created by looking at the last datagram of the ICMP Echo request. For example, Frame 7 is the last datagram (fragmented) of the first ICMP Echo request, and when we check it's IP header field, we can observe that there is a new options field added which indicates that there are three IPv4 fragments (Figure 13). Also number of bytes that each frame contains, is also included in this field.

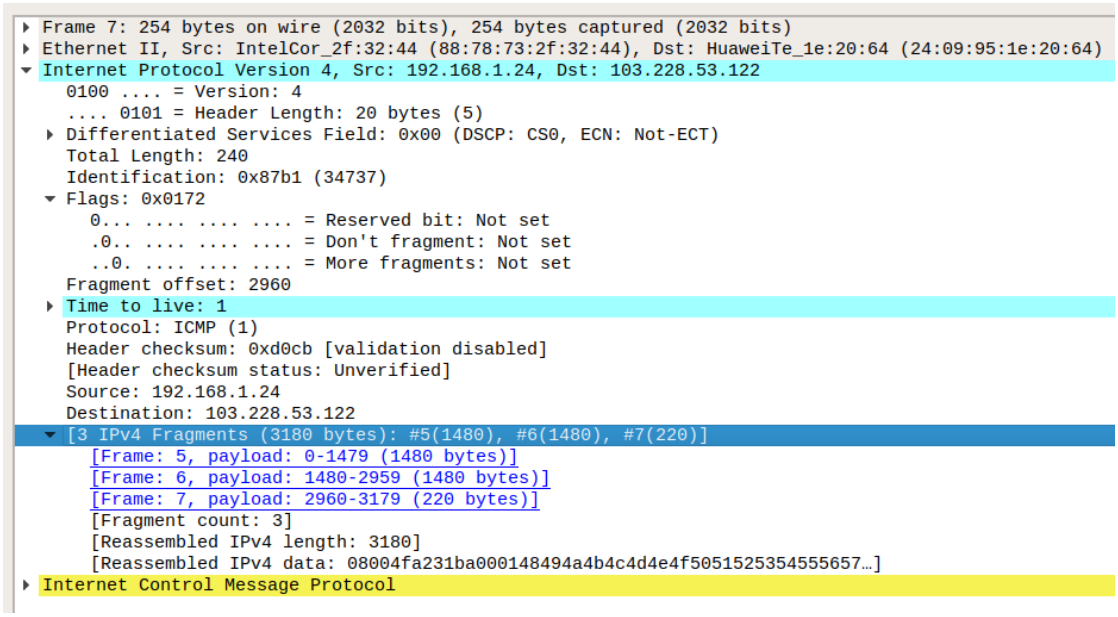


Figure 13: Last fragmented datagram of the first ICMP Echo Request (Number of fragmentation is highlighted).

Question 10

We know that these datagrams are fragmented and they include different parts (sequentially) of the first ICMP Echo request. Therefore all of these fragmented datagrams have different fragment offsets. When we look at the IP header of these datagrams, we can observe that 'Fragment offset' fields of these datagrams are different (change can be observed in Figures 14, 15 and 16). Also another difference is in the 'Flags' field. The 'More fragments' bit of the fragmented datagrams except the last datagram are set to 1, but this bit is not set (is 0) in the last fragmented datagram. Moreover the 'Header checksum' fields of each package is different.

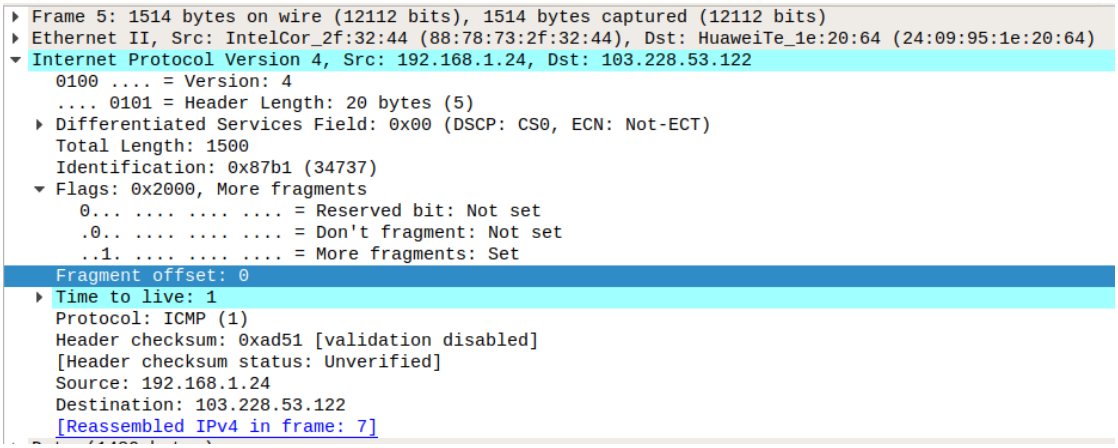


Figure 14: First fragmented datagram of the first ICMP Echo Request ('Fragment offset' field is highlighted).

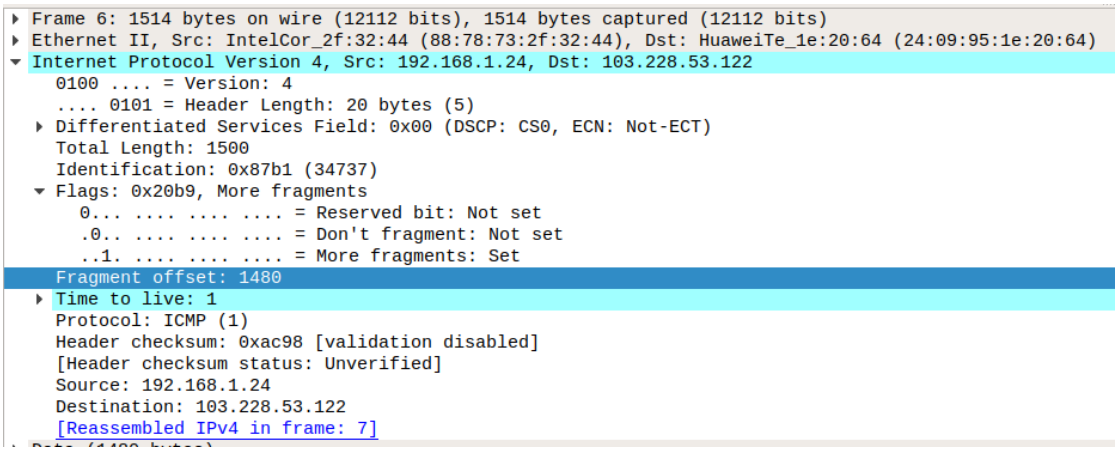


Figure 15: Second fragmented datagram of the first ICMP Echo Request ('Fragment offset' field is highlighted).

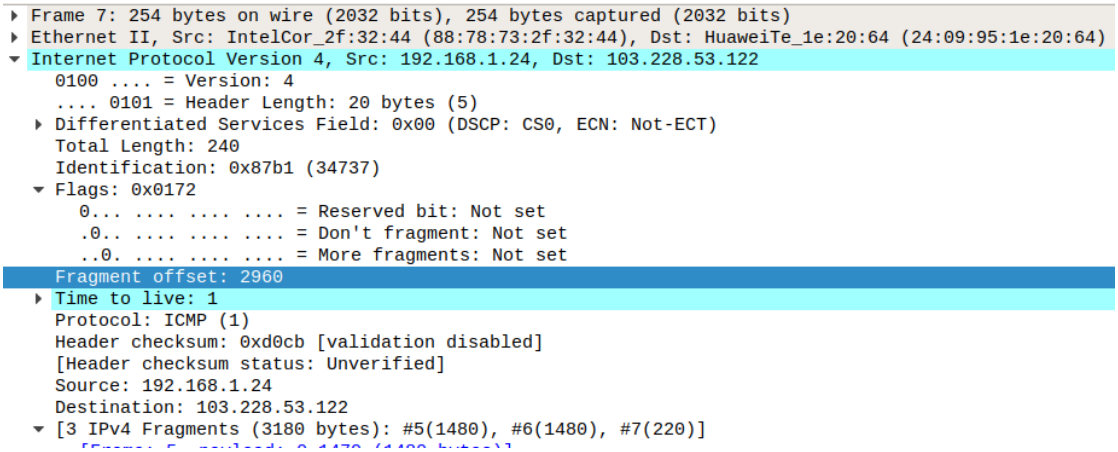


Figure 16: Last fragmented datagram of the first ICMP Echo Request ('Fragment offset' field is highlighted).