Assignment 1: Networking Tools and Wireshark

Part1: Networking Tools

1. IP address of your machine, subnet mask, and network ID

   `ifconfig` – configure network interface parameters

```
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=6460<TSO4,TSO6,CHANNEL_IO,PARTIAL_CSUM,ZEROINVERT_CSUM>
        ether 4a:e7:63:10:51:4a
        inet6 fe80::14f6:ed4:24a6:bab0%en0 prefixlen 64 secured scopeid 0xb
        inet 10.145.69.12 netmask 0xffff8000 broadcast 10.145.127.255
        nd6 options=201<PERFORMNUD,DAD>
        media: autoselect
        status: active
```

IP Address: Listed as `inet` (**`192.168.1.100`**).
Subnet Mask: Listed as `netmask` (**`255.255.255.0`**).
Network ID: Bitwise AND operation between the *IP address* and the *subnet mask* (**`192.168.1.0`**).

With `ifconfig` on OS X, we found :

```
inet 192.168.1.2 netmask 0xffffff00 broadcast 192.168.1.255
```

netmask `0xffffff00` means `255.255.255.0` [source: stack exchange]

2. IP address associated with and **`www.google.com and www.facebook.com`** using `nslookup`

   `nslookup` - query Internet name servers interactively

```
~ > nslookup www.google.com

Server:         172.16.1.166
Address:        172.16.1.166#53

Non-authoritative answer:
Name:   www.google.com
Address: 142.250.77.36

~ > nslookup www.facebook.com

Server:         172.16.1.166
Address:        172.16.1.166#53

Non-authoritative answer:
www.facebook.com        canonical name = star-mini.c10r.facebook.com.
Name:   star-mini.c10r.facebook.com
Address: 31.13.79.35
```

On running the basic `nslookup` command we got the IP Addresses using our system's default DNS server:

- `www.google.com`: `142.250.77.36`
- `www.facebook.com`: `31.13.79.35`

Changing the DNS server address in the `nslookup` command:

```
~ > nslookup www.google.com 172.16.1.164

Server:         172.16.1.164
Address:        172.16.1.164#53

Non-authoritative answer:
Name:   www.google.com
Address: 142.250.66.4

~ > nslookup www.google.com 172.16.1.180

Server:         172.16.1.180
Address:        172.16.1.180#53

Non-authoritative answer:
Name:   www.google.com
Address: 142.250.192.196

~ > nslookup www.google.com 172.16.1.165

Server:         172.16.1.165
Address:        172.16.1.165#53

Non-authoritative answer:
Name:   www.google.com
Address: 142.250.192.132

~ > nslookup www.google.com 172.16.1.166

Server:         172.16.1.166
Address:        172.16.1.166#53

Non-authoritative answer:
Name:   www.google.com
Address: 142.250.77.36

~ >
```

On using the `server` option, we got a different IP address of `www.google.com` for different DNS servers.
One of the reason behind this could be:

- Since Google is a large service and many devices may try to reach the same domain, so different DNS server may return different IP addresses to distribute the traffic on its server, i.e. **Load Balancing**.

3. Ping the IP address of one of our friend's machine IP

   `ping` – send ICMP ECHO_REQUEST packets to network hosts

   ICMP: Internet Control Message Protocol

Packet size: 64 bytes ⇒

```
~ > ping -s 64 -W 100 10.5.16.210 -c 5
PING 10.5.16.210 (10.5.16.210): 64 data bytes
72 bytes from 10.5.16.210: icmp_seq=0 ttl=61 time=13.074 ms
72 bytes from 10.5.16.210: icmp_seq=1 ttl=61 time=19.941 ms
72 bytes from 10.5.16.210: icmp_seq=2 ttl=61 time=4.924 ms
72 bytes from 10.5.16.210: icmp_seq=3 ttl=61 time=12.442 ms
72 bytes from 10.5.16.210: icmp_seq=4 ttl=61 time=6.853 ms

--- 10.5.16.210 ping statistics ---
5 packets transmitted, 5 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 4.924/11.447/19.941/5.281 ms
~ >
```

Packet size: 128 bytes ⇒

```
~ > ping -s 128 -W 100 10.5.16.210 -c 5
PING 10.5.16.210 (10.5.16.210): 128 data bytes
136 bytes from 10.5.16.210: icmp_seq=0 ttl=61 time=16.298 ms
136 bytes from 10.5.16.210: icmp_seq=1 ttl=61 time=63.483 ms
136 bytes from 10.5.16.210: icmp_seq=2 ttl=61 time=19.113 ms
136 bytes from 10.5.16.210: icmp_seq=3 ttl=61 time=17.392 ms
136 bytes from 10.5.16.210: icmp_seq=4 ttl=61 time=21.998 ms

--- 10.5.16.210 ping statistics ---
5 packets transmitted, 5 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 16.298/27.657/63.483/18.016 ms
~ >
```

Packet size: 512 bytes ⇒

```
~ > ping -s 512 -W 100 10.5.16.210 -c 5
PING 10.5.16.210 (10.5.16.210): 512 data bytes
520 bytes from 10.5.16.210: icmp_seq=0 ttl=61 time=6.548 ms
520 bytes from 10.5.16.210: icmp_seq=1 ttl=61 time=10.422 ms
520 bytes from 10.5.16.210: icmp_seq=2 ttl=61 time=5.855 ms
520 bytes from 10.5.16.210: icmp_seq=3 ttl=61 time=11.959 ms
520 bytes from 10.5.16.210: icmp_seq=4 ttl=61 time=12.767 ms

--- 10.5.16.210 ping statistics ---
5 packets transmitted, 5 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 5.855/9.510/12.767/2.813 ms
~ >
```

	64 bytes	128 bytes	512 bytes
Packet loss (%)	0.0	0.0	0.0
Min (ms)	4.924	16.298	5.855
Avg (ms)	11.447	27.657	9.510
Max (ms)	19.941	63.483	12.767
Stddev (ms)	5.281	18.016	2.813

4. Summary of `placeroute` to `www.google.com`

   `traceroute` – print the route packets take to network host

```
~ > traceroute www.google.com

traceroute to www.google.com (142.250.207.228), 64 hops max, 40 byte packets
 1  10.105.52.2 (10.105.52.2)  1.133 ms  0.559 ms  0.517 ms
 2  10.120.1.5 (10.120.1.5)  0.541 ms  0.726 ms  0.523 ms
 3  10.255.1.3 (10.255.1.3)  4.060 ms  3.038 ms  3.011 ms
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  72.14.204.62 (72.14.204.62)  54.716 ms  59.207 ms
    142.250.172.80 (142.250.172.80)  48.649 ms
 9  * * *
10  192.178.86.202 (192.178.86.202)  57.159 ms
    108.170.234.156 (108.170.234.156)  37.830 ms
    192.178.86.200 (192.178.86.200)  51.344 ms
11  142.250.226.134 (142.250.226.134)  46.582 ms
    192.178.111.60 (192.178.111.60)  48.541 ms
    192.178.110.198 (192.178.110.198)  47.278 ms
12  172.253.68.121 (172.253.68.121)  60.167 ms  56.518 ms
    192.178.251.219 (192.178.251.219)  68.085 ms
13  172.253.51.137 (172.253.51.137)  78.558 ms
    108.170.232.203 (108.170.232.203)  71.100 ms
    216.239.62.219 (216.239.62.219)  74.689 ms
14  192.178.83.221 (192.178.83.221)  59.041 ms
    192.178.83.227 (192.178.83.227)  60.045 ms
    192.178.83.225 (192.178.83.225)  78.367 ms
15  192.178.83.215 (192.178.83.215)  71.869 ms
    192.178.83.221 (192.178.83.221)  67.173 ms
    del12s11-in-f4.1e100.net (142.250.207.228)  60.907 ms
~ >
```

> **Target:**
- Destination: `www.google.com` (`142.250.207.228`)
- Max Hops: 64
- Packet Size: 40 bytes

> RTTs range from **0.517 ms** (at hop 1) to **78.558 ms** (at hop 13).
> At several points, multiple IPs responded at the same hop. This suggests **load balancing**, where traffic is distributed across multiple routers.

> The final hop (15) successfully reached the target:

```
del12s11-in-f4.1e100.net (Google's server) at 142.250.207.228.
```

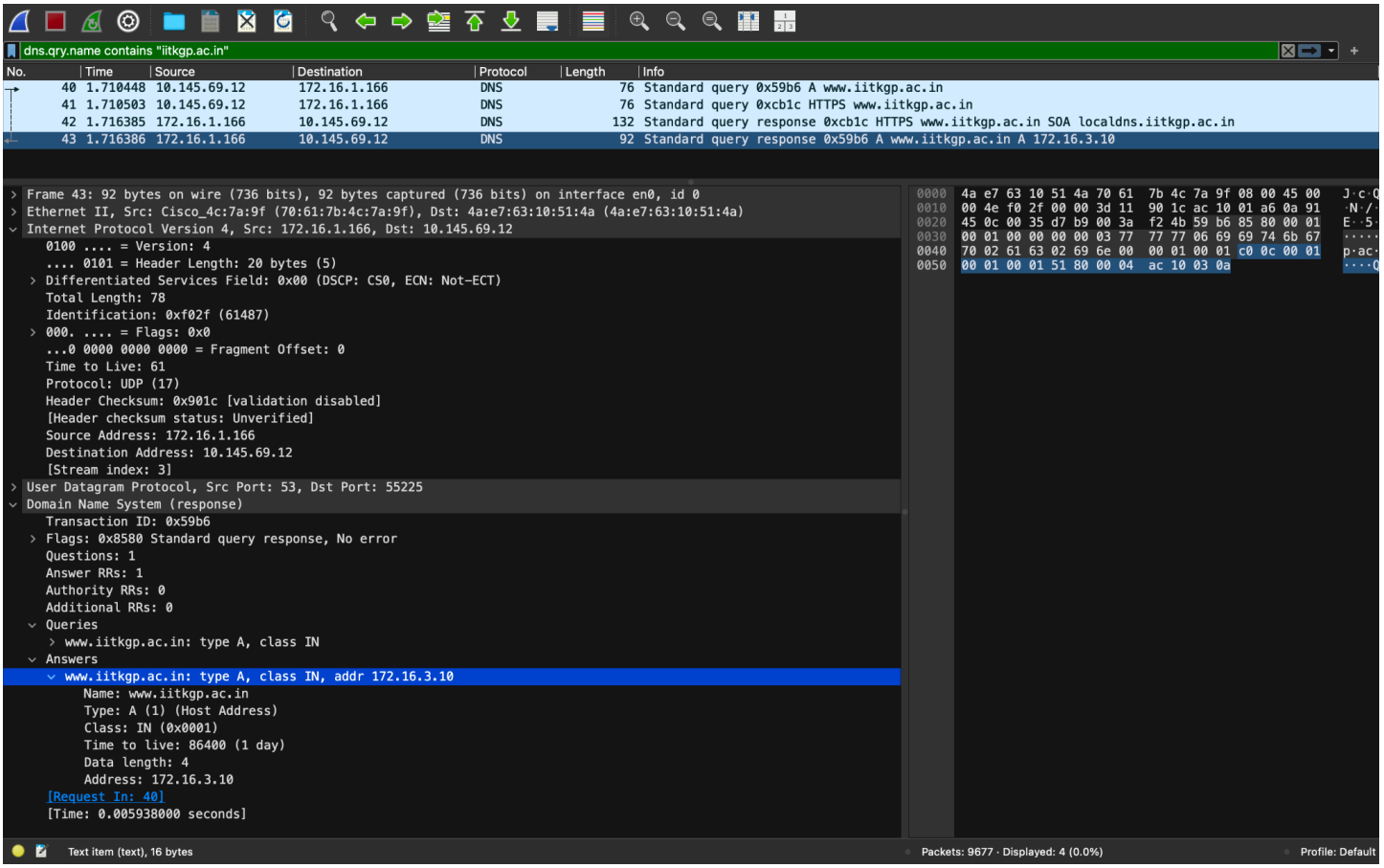
RTT to the destination: **60.907 ms**

Count of Responding Hosts: **10 hosts** (hops: 1, 2, 3, 8, 10, 11, 12, 13, 14, 15).

“ `* * *` ”: Indicates that the routers/devices at these hops did not respond to the `traceroute` probes.
Some intermediate routers or devices are configured to **drop ICMP packets** (used by `traceroute`) which could be due to security settings, firewalls, or ICMP restrictions.

Part 2: Packet Analysis

1. Analysis of DNS Packets: Structure and its Traffic while visiting https://www.iitkgp.ac.in



a) DNS is using **UDP** (User Datagram Protocol) in the observed packet as highlighted above.

b) The IP addresses are visible in the top section as well as in the Internet Protocol the details pane.
- Source IP address of the DNS query: `10.145.69.12`
- Destination IP address of the DNS query: `172.16.1.166`

c) **2** DNS query packets were sent

```
40      1.71044810.145.69.12    172.16.1.166    DNS     76      Standard query 0x59b6 A www.iitkgp.ac.in
41      1.71050310.145.69.12    172.16.1.166    DNS     76      Standard query 0xcb1c HTTPS www.iitkgp.ac.in
```

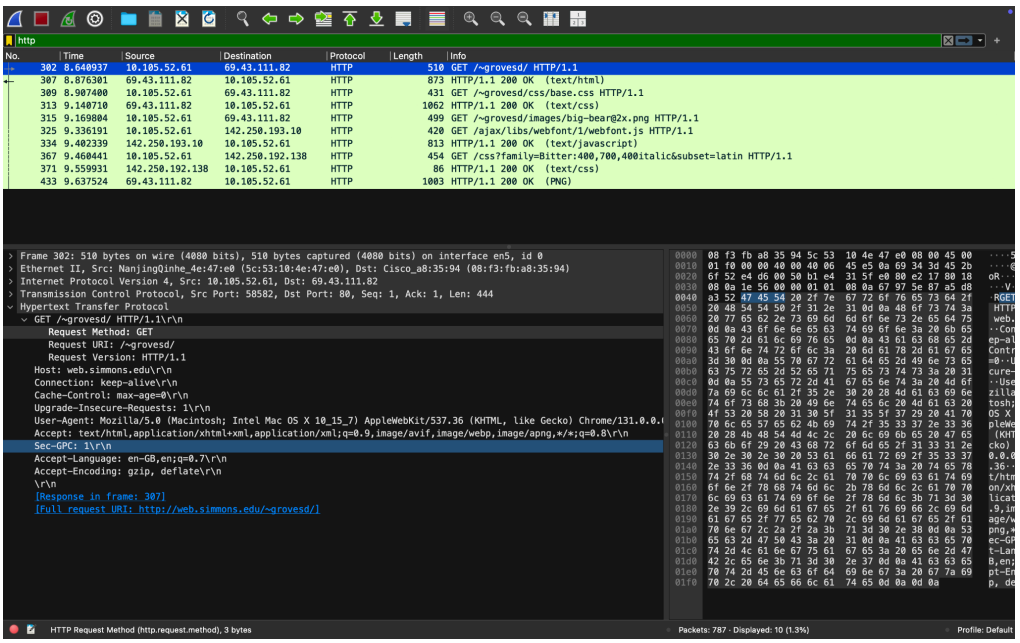
d) `172.16.1.166` is the DNS server that gives the query response with the actual IP address.

e) Only a single DNS server (`172.16.1.166`) is involved here, and it gives responses to the two corresponding queries.

f) The resource records involved in resolving the site's IP address in the complete resolving process of this DNS conversation :

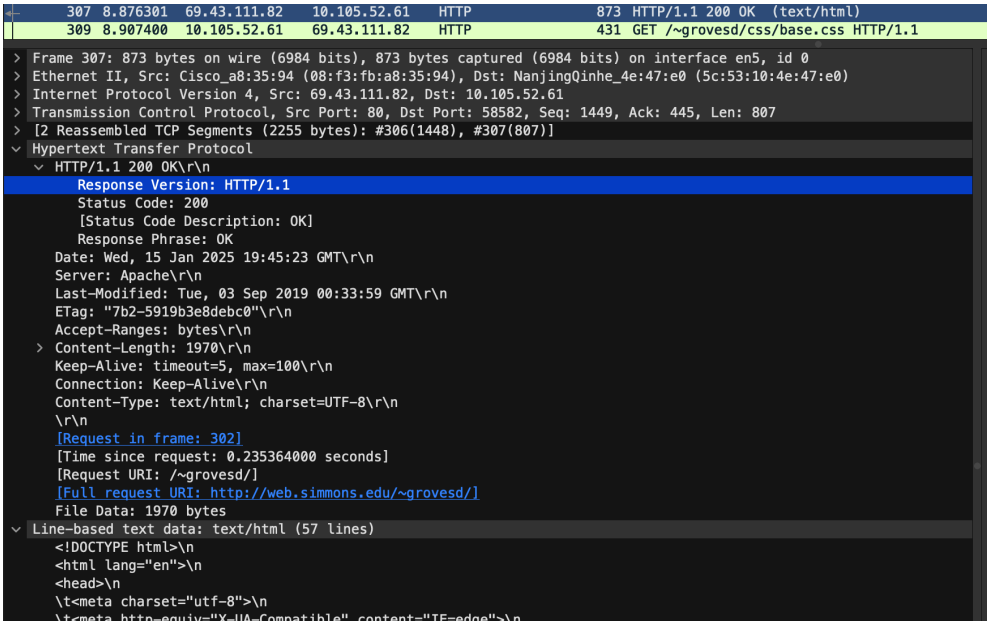
```
Name: www.iitkgp.ac.in
Type: A (1) (Host Address)
Class: IN (0x0001)
Time to live: 86400 (1 day)
Data length: 4
Address: 172.16.3.10
```

2. Web Traffic (HTTP) for the web server- http://web.simmons.edu/~grovesd/



a) We can filter the HTTP packets between the client and web server using the 'http' filter. Observations:
   - We can see HTTP requests (`GET`) and responses (200 OK)
   - Our client requests for resources to load the page from the server, such as HTML, CSS, images, scripts and fonts.
   - The server responds with status code or corresponding files.

b) To identify the HTTP request and response:
   - We can see the header and find the request/response in **info.**
   - In the packet details pane, we can expand Hypertext Transfer Protocol to identify HTTP request and response with their methods, version, status code and other details.



c) The client send 5 GET requests packets, and the server also responds with those corresponding request and send 5 response packets to load the complete web page
   - So, in total **10** HTTP packets are exchanged between client and server to load an entire web page (http://web.simmons.edu/~grovesd/)

3. ICMP Traffic (Ping/Traceroute)

a) Inspect & crosscheck the Source and Destination IP address of captured ICMP packets.

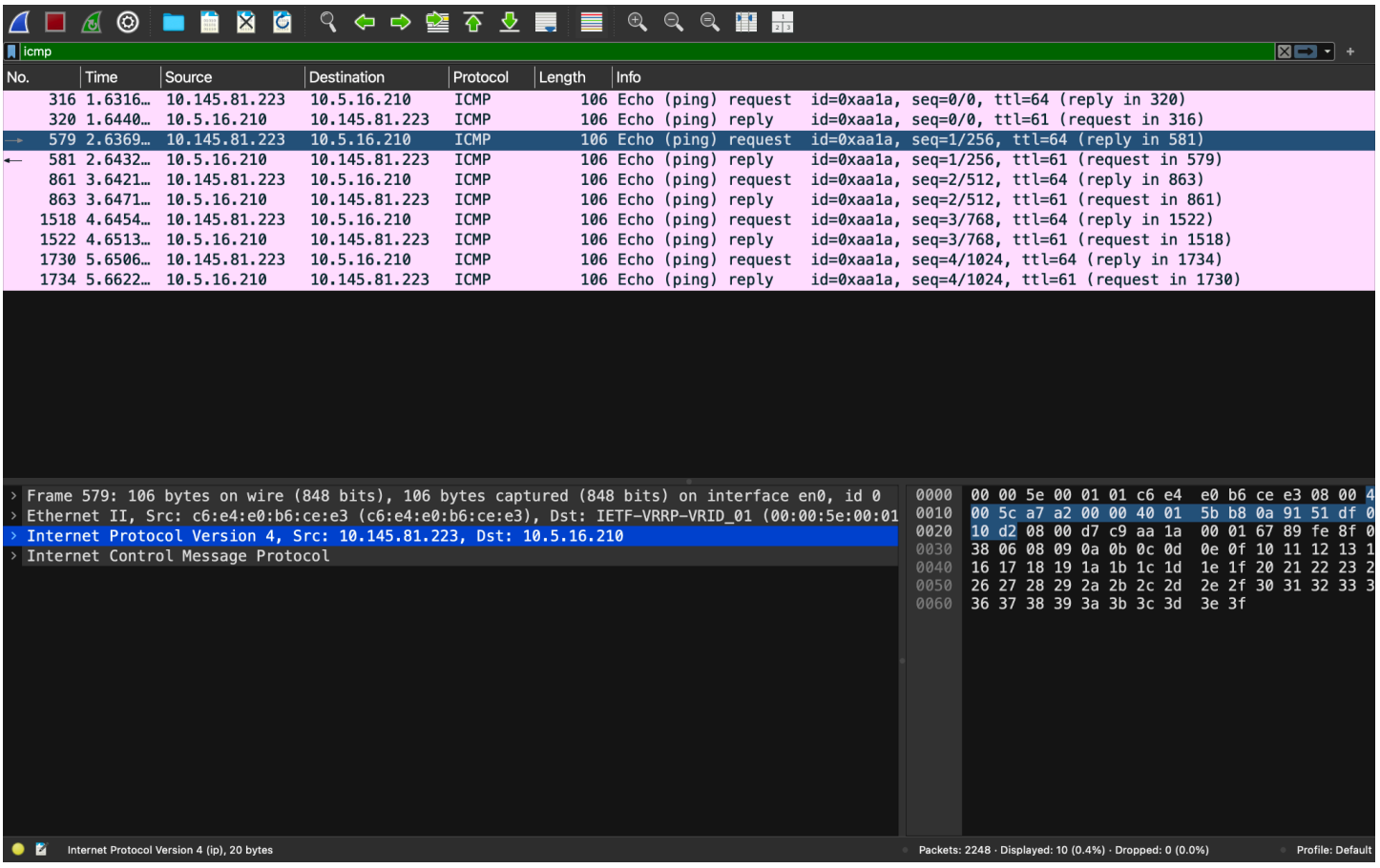
i) Running `ping` command to initiate ICMP traffic for our friend's machine and capturing it.

```
~ > ping -s 64 -W 100 10.5.16.210 -c 5
PING 10.5.16.210 (10.5.16.210): 64 data bytes
72 bytes from 10.5.16.210: icmp_seq=0 ttl=61 time=13.074 ms
72 bytes from 10.5.16.210: icmp_seq=1 ttl=61 time=19.941 ms
72 bytes from 10.5.16.210: icmp_seq=2 ttl=61 time=4.924 ms
72 bytes from 10.5.16.210: icmp_seq=3 ttl=61 time=12.442 ms
72 bytes from 10.5.16.210: icmp_seq=4 ttl=61 time=6.853 ms

--- 10.5.16.210 ping statistics ---
5 packets transmitted, 5 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 4.924/11.447/19.941/5.281 ms
~ >
```

In the image below, we can cross check the source and destination IP addresses:

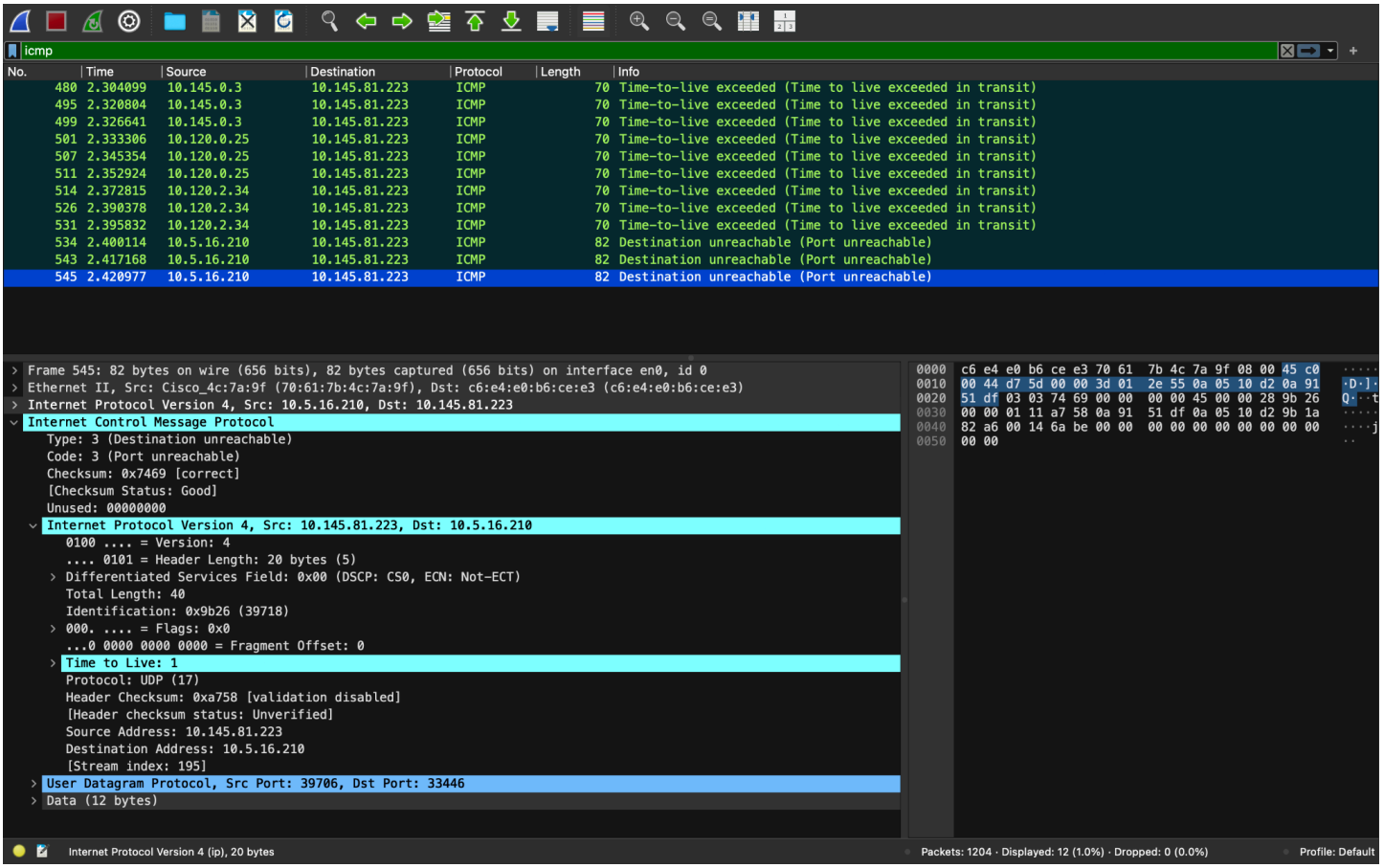
- Source IP address: `10.145.81.223`
- Destination Address: `10.5.16.210`



ii) Running `traceroute` command to initiate ICMP traffic for our friend's machine and capturing it.

```
~ > traceroute 10.5.16.210
traceroute to 10.5.16.210 (10.5.16.210), 64 hops max, 40 byte packets
 1  10.145.0.3 (10.145.0.3)  5.503 ms  3.620 ms  5.822 ms
 2  10.120.0.25 (10.120.0.25)  6.244 ms  4.448 ms  9.524 ms
 3  10.120.2.34 (10.120.2.34)  19.870 ms  5.698 ms  5.405 ms
 4  10.5.16.210 (10.5.16.210)  4.264 ms  6.784 ms  3.773 ms
```

This traceroute output shows a successful traversal of **4 hops** within a campus network to the destination `10.5.16.210` .
As we can see below, the destination of each ICMP packet is `10.145.81.223` (i.e. our device) and the final destination(`10.5.16.210`) is the source of the last packet that reached and responded with port unreachable.



`traceroute` works as follow:

1. Sends a UDP datagram with TTL as 1 to the destination host.The router reads the datagram, decrements the TTL and sends back an ICMP time exceeded message.
2. Traceroute receives the above message and sends another UDP datagram with TTL as 2. Routers on the internet read this datagram, each decrement the TTL, and finally send back the ICMP time exceeded message.
3. The above steps continue and finally, with TTL as N, the UDP datagram has reached the destination host. Then, what should the host return? It cannot send back ICMP time exceeded messages as before -- TTL is not exceeded.

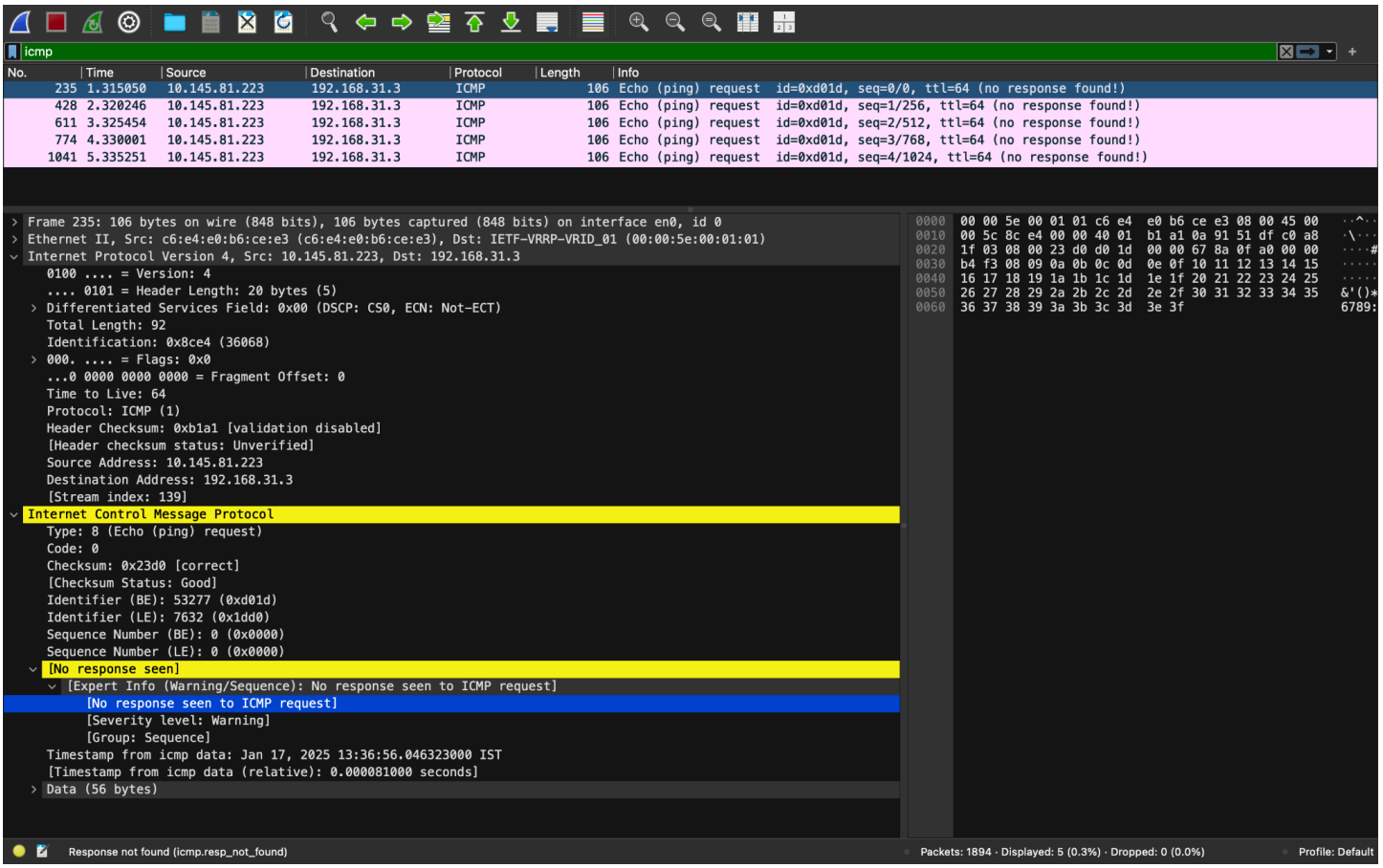
Traceroute design to send the UDP datagram to a port of host, and it is almost impossible that the port is listened to (`33446` for example). The destination host receives the UDP datagram, finds the datagram's target port is not listened to, and then returns a "Destination Unreachable" message -- more accurately, "Port Unreachable".
That's why traceroute expects a "Destination Unreachable" message at the final hop to determine that the UDP datagram has already reached the destination. [source: Stack overflow].

b) `ping` to an unreachable host (a host with IP `192.168.31.3` does not exist in the IIT KGP network)

```
~ > ping -s 64 -W 100 192.168.31.3 -c 5
PING 192.168.31.3 (192.168.31.3): 64 data bytes
Request timeout for icmp_seq 0
Request timeout for icmp_seq 1
Request timeout for icmp_seq 2
Request timeout for icmp_seq 3

--- 192.168.31.3 ping statistics ---
5 packets transmitted, 0 packets received, 100.0% packet loss
~ >
```

From the `ping` statistics above, we can observe that no packets were received from the unreachable host.



In the details pane, we can observe that **No response seen to ICMP request**.

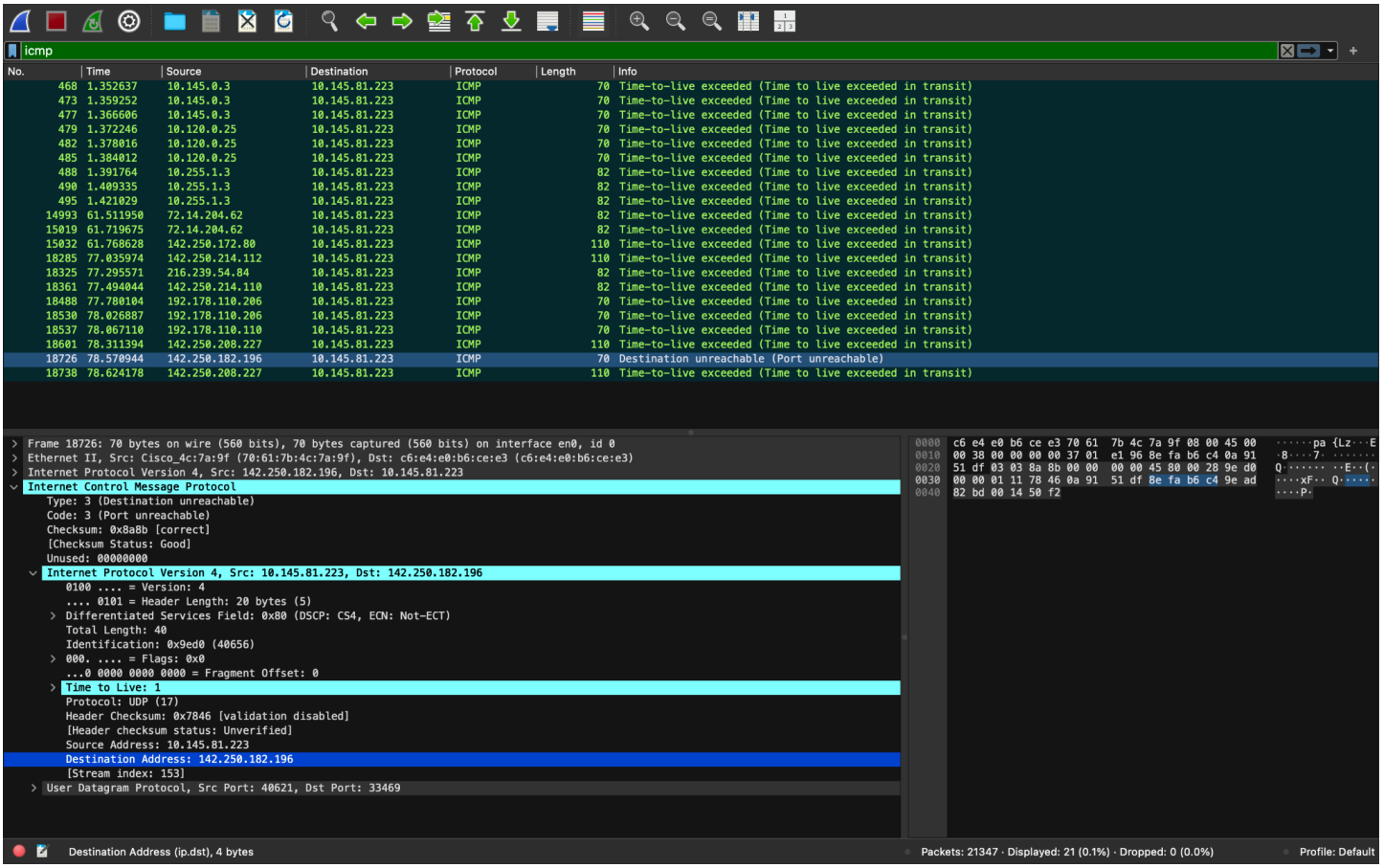
```
[No response seen]
    [Expert Info (Warning/Sequence): No response seen to ICMP request]
        [No response seen to ICMP request]
        [Severity level: Warning]
        [Group: Sequence]
```

c) Perform a `'traceroute'` operation

i) Reachable host (www.google.com)

```
~ > traceroute www.google.com
traceroute to www.google.com (142.250.182.196), 64 hops max, 40 byte packets
 1  10.145.0.3 (10.145.0.3)  10.054 ms  4.799 ms  7.372 ms
 2  10.120.0.25 (10.120.0.25)  5.563 ms  4.777 ms  5.964 ms
 3  10.255.1.3 (10.255.1.3)  7.721 ms  16.657 ms  11.643 ms
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  72.14.204.62 (72.14.204.62)  47.043 ms  40.639 ms
    142.250.172.80 (142.250.172.80)  47.689 ms
 9  * * *
10  142.250.214.112 (142.250.214.112)  48.347 ms
    216.239.54.84 (216.239.54.84)  57.377 ms
    142.250.214.110 (142.250.214.110)  47.898 ms
11  192.178.110.206 (192.178.110.206)  68.988 ms  47.105 ms
    192.178.110.110 (192.178.110.110)  39.448 ms
12  142.250.208.227 (142.250.208.227)  41.500 ms
    bom07s28-in-f4.1e100.net (142.250.182.196)  47.531 ms
    142.250.208.227 (142.250.208.227)  44.869 ms
~ >
```

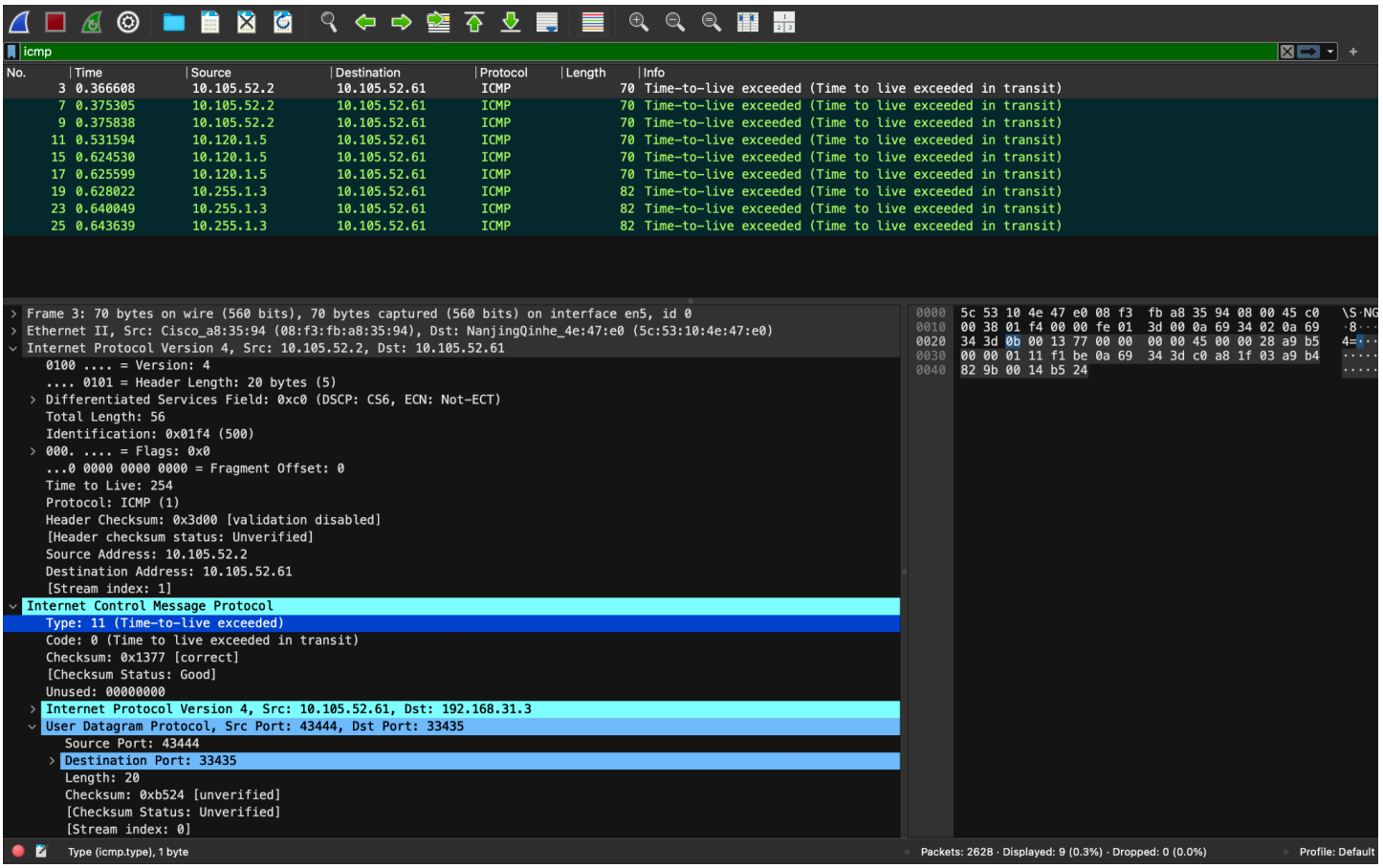
We can observe a **total of 12 hops** here, out of which 7 are responding hosts. At 12th hop, we successfully reached the target (www.google.com).




```
Time to Live: 1
    [Expert Info (Note/Sequence): "Time To Live" only 1]
        ["Time To Live" only 1]
        [Severity level: Note]
        [Group: Sequence]
```

ii) Unreachable host (`192.168.31.3`)

```
~ > traceroute -w 1 -m 30 192.168.31.3
traceroute to 192.168.31.3 (192.168.31.3), 30 hops max, 40 byte packets
 1  10.105.52.2 (10.105.52.2)  1.109 ms  0.647 ms  0.463 ms
 2  10.120.1.5 (10.120.1.5)  156.004 ms  84.723 ms  0.832 ms
 3  10.255.1.3 (10.255.1.3)  2.491 ms  3.868 ms  3.487 ms
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  * * *
 9  * * *
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *
~ >
```



On running `traceroute` for an unreachable host (`192.168.31.3`), we get a few TTL exceeded ICMP packets till we can find a router to route the IP, but after a few reachable hosts, all the hosts are unresponsive because our destination IP address is unreachable.

- We used `-w` and `-m` flags to set the timeout to 1 sec for each probe (default is 5 sec), and the max hops for traceroute to 30.
- We can also set a custom number of probes per hop using the -q option (3 by default).