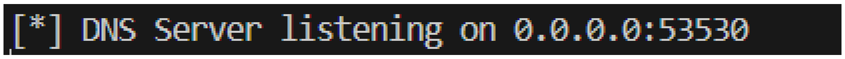
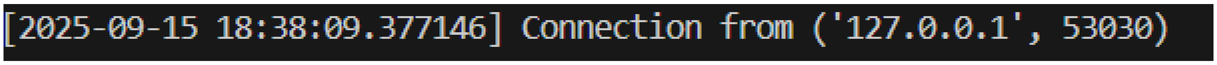
Task - 1



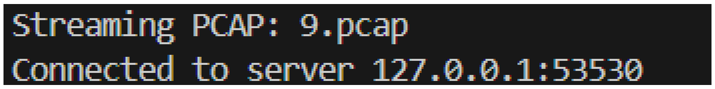
```
[*] DNS Server listening on 0.0.0.0:53530
```

The server has started and is waiting for packets to be received.



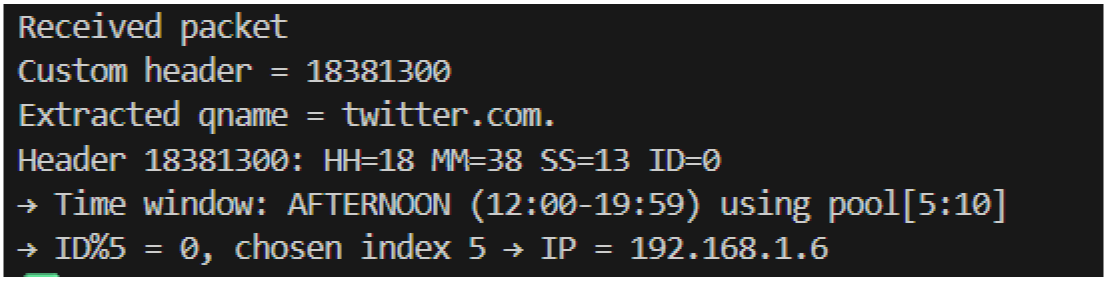
```
[2025-09-15 18:38:09.377146] Connection from ('127.0.0.1', 53030)
```

The server receives a connection from the client.

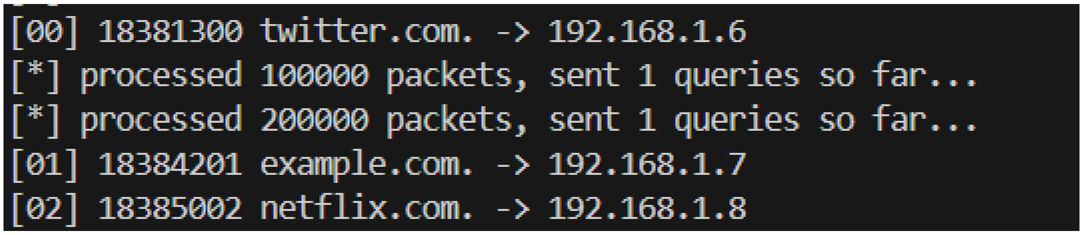


```
Streaming PCAP: 9.pcap
Connected to server 127.0.0.1:53530
```

The packet is sent to the server and is processed as shown in the image. The custom header is generated using the current time. Since HH = 18, it is the afternoon section as mentioned in the predefined rules file, so we use the IP pool from 5. Using hash mod = 5 on ID, we get the IP as shown in the image.



```
Received packet
Custom header = 18381300
Extracted qname = twitter.com.
Header 18381300: HH=18 MM=38 SS=13 ID=0
→ Time window: AFTERNOON (12:00-19:59) using pool[5:10]
→ ID%5 = 0, chosen index 5 → IP = 192.168.1.6
```



```
[00] 18381300 twitter.com. -> 192.168.1.6
[*] processed 100000 packets, sent 1 queries so far...
[*] processed 200000 packets, sent 1 queries so far...
[01] 18384201 example.com. -> 192.168.1.7
[02] 18385002 netflix.com. -> 192.168.1.8
```

Since the PCAP file contained both system and internet queries, we processed it in both ways, and the data can be seen in report.csv and report_final.csv.

The file report.csv contains both system(.local) and internet queries, while report_final.csv contains only internet queries.

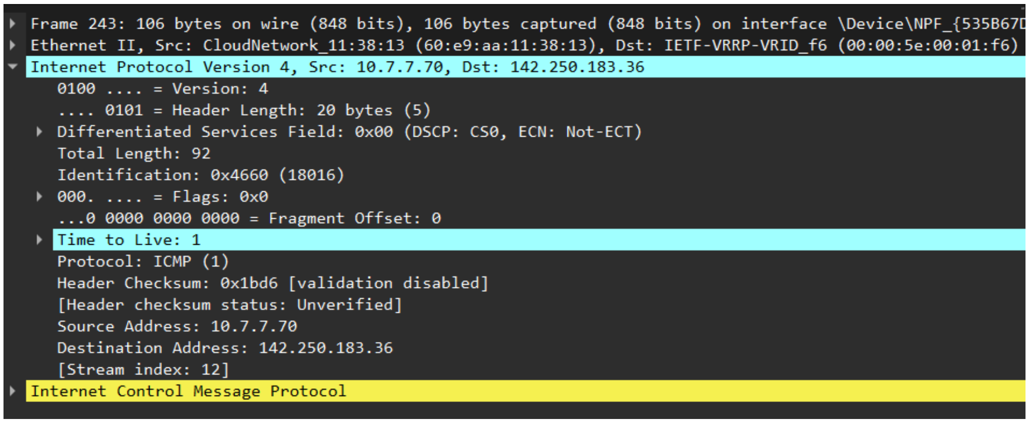
The following table contains data from report_final.csv.

Custom Header (HHMMSSID)	Domain	Resolved IP
18381300	twitter.com.	192.168.1.6
18384201	example.com.	192.168.1.7
18385002	netflix.com.	192.168.1.8
18391103	linkedin.com.	192.168.1.9
18393904	reddit.com.	192.168.1.10
18400105	openai.com.	192.168.1.6

Task - 2

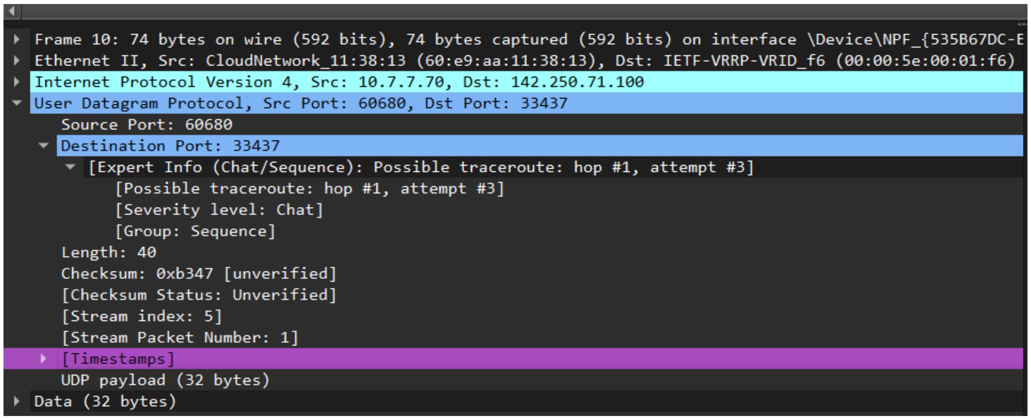
1. What protocol does Windows tracert use by default, and what protocol does Linux traceroute use by default?

Tracert in Windows uses ICMP echo request as the default protocol, while Traceroute in Linux uses UDP datagrams at high, unlikely to be used ports and receives ICMP replies from the router.



```
Frame 243: 106 bytes on wire (848 bits), 106 bytes captured (848 bits) on interface \Device\NPF_{535B67D
Ethernet II, Src: CloudNetwork_11:38:13 (60:e9:aa:11:38:13), Dst: IETF-VRRP-VRID_f6 (00:00:5e:00:01:f6)
Internet Protocol Version 4, Src: 10.7.7.70, Dst: 142.250.183.36
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 92
  Identification: 0x4660 (18016)
  000. .... = Flags: 0x0
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 1
  Protocol: ICMP (1)
  Header Checksum: 0x1bd6 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 10.7.7.70
  Destination Address: 142.250.183.36
  [Stream index: 12]
Internet Control Message Protocol
```

Packet details for Windows

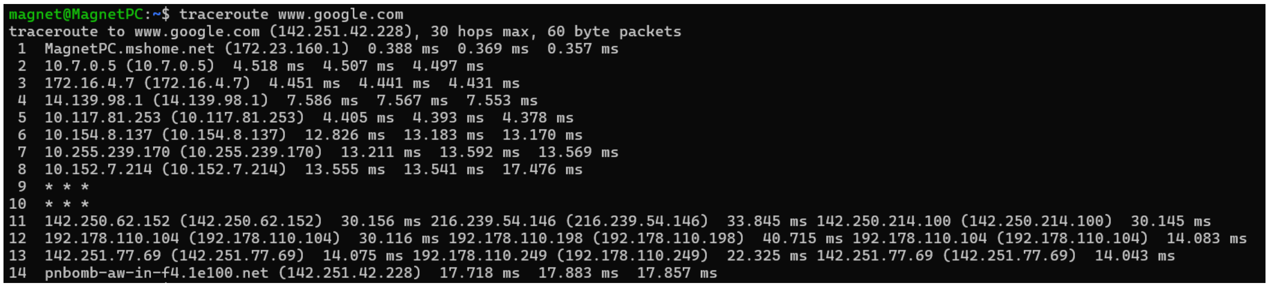


```
▸ Frame 10: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{535B67DC-E
▸ Ethernet II, Src: CloudNetwork_11:38:13 (60:e9:aa:11:38:13), Dst: IETF-VRRP-VRID_f6 (00:00:5e:00:01:f6)
▸ Internet Protocol Version 4, Src: 10.7.7.70, Dst: 142.250.71.100
▾ User Datagram Protocol, Src Port: 60680, Dst Port: 33437
    Source Port: 60680
  ▾ Destination Port: 33437
    ▾ [Expert Info (Chat/Sequence): Possible traceroute: hop #1, attempt #3]
        [Possible traceroute: hop #1, attempt #3]
        [Severity level: Chat]
        [Group: Sequence]
    Length: 40
    Checksum: 0xb347 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 5]
    [Stream Packet Number: 1]
  ▸ [Timestamps]
    UDP payload (32 bytes)
▸ Data (32 bytes)
```

Packet details for Linux

2. Some hops in your traceroute output may show *. Provide at least two reasons why a router might not reply.**

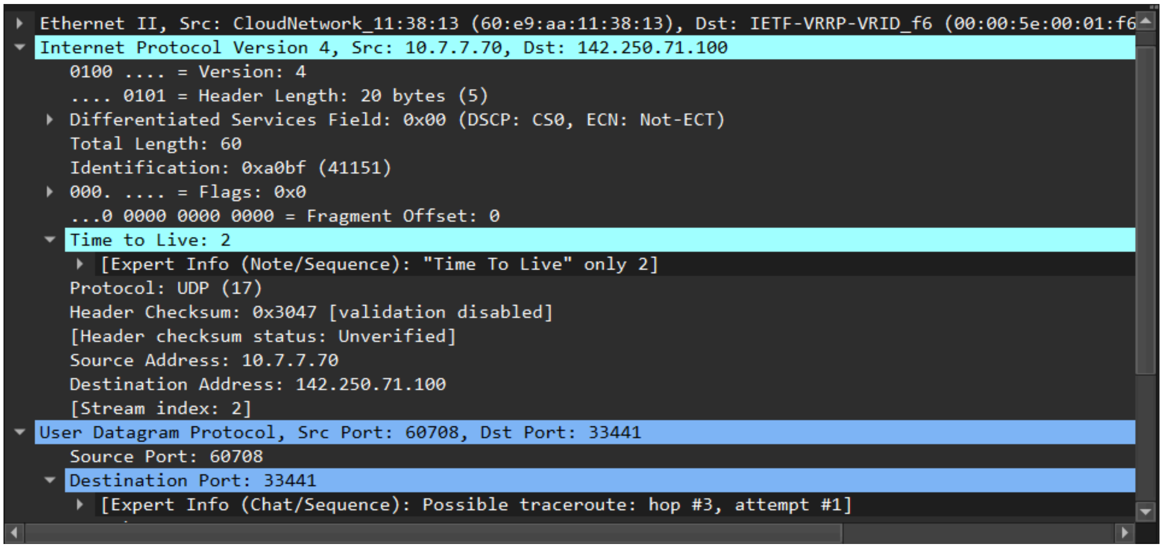
1. The router may block or filter ICMP responses (for security).
2. The router may implement ICMP rate limiting and simply not reply within the timeout.



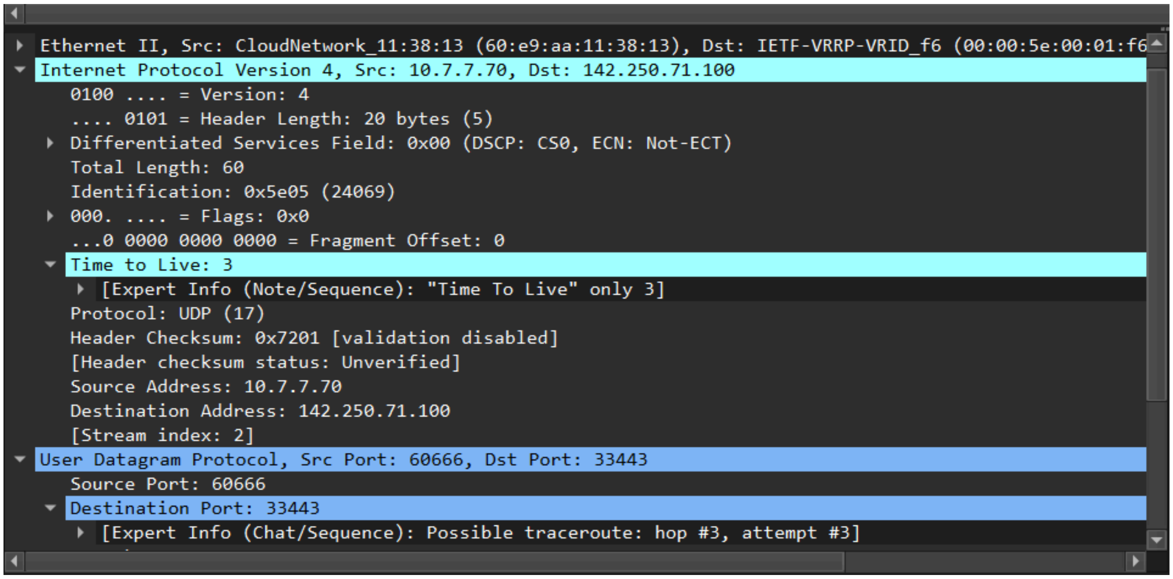
```
magnet@MagnetPC:~$ traceroute www.google.com
traceroute to www.google.com (142.251.42.228), 30 hops max, 60 byte packets
 1  MagnetPC.mshome.net (172.23.160.1)  0.388 ms  0.369 ms  0.357 ms
 2  10.7.0.5 (10.7.0.5)  4.518 ms  4.507 ms  4.497 ms
 3  172.16.4.7 (172.16.4.7)  4.451 ms  4.441 ms  4.431 ms
 4  14.139.98.1 (14.139.98.1)  7.586 ms  7.567 ms  7.553 ms
 5  10.117.81.253 (10.117.81.253)  4.405 ms  4.393 ms  4.378 ms
 6  10.154.8.137 (10.154.8.137)  12.826 ms  13.183 ms  13.170 ms
 7  10.255.239.170 (10.255.239.170)  13.211 ms  13.592 ms  13.569 ms
 8  10.152.7.214 (10.152.7.214)  13.555 ms  13.541 ms  17.476 ms
 9  * * *
10  * * *
11  142.250.62.152 (142.250.62.152)  30.156 ms 216.239.54.146 (216.239.54.146)  33.845 ms 142.250.214.100 (142.250.214.100)  30.145 ms
12  192.178.110.104 (192.178.110.104)  30.116 ms 192.178.110.198 (192.178.110.198)  40.715 ms 192.178.110.104 (192.178.110.104)  14.083 ms
13  142.251.77.69 (142.251.77.69)  14.075 ms 192.178.110.249 (192.178.110.249)  22.325 ms 142.251.77.69 (142.251.77.69)  14.043 ms
14  pnbomb-aw-in-f4.1e100.net (142.251.42.228)  17.718 ms  17.883 ms  17.857 ms
```

3. In Linux traceroute, which field in the probe packets changes between successive probes sent to the destination?

In **Linux traceroute**, the field that changes between successive probes is the **Time To Live (TTL)** field in the IP header. Traceroute starts by sending probe packets (by default, UDP on Linux) with **TTL = 1**. The first router decrements the TTL to 0, drops the packet, and sends back an **ICMP Time Exceeded** message. Traceroute then increases the TTL to 2, 3, 4… until the packet finally reaches the destination. At the destination, instead of Time Exceeded, you get an **ICMP Port Unreachable** (because traceroute uses high-numbered UDP ports).



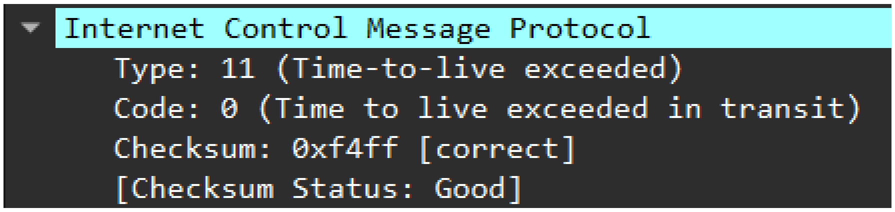
```
▸ Ethernet II, Src: CloudNetwork_11:38:13 (60:e9:aa:11:38:13), Dst: IETF-VRRP-VRID_f6 (00:00:5e:00:01:f6
▾ Internet Protocol Version 4, Src: 10.7.7.70, Dst: 142.250.71.100
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0xa0bf (41151)
  ▸ 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
  ▾ Time to Live: 2
    ▸ [Expert Info (Note/Sequence): "Time To Live" only 2]
    Protocol: UDP (17)
    Header Checksum: 0x3047 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.7.7.70
    Destination Address: 142.250.71.100
    [Stream index: 2]
▾ User Datagram Protocol, Src Port: 60708, Dst Port: 33441
    Source Port: 60708
  ▾ Destination Port: 33441
    ▸ [Expert Info (Chat/Sequence): Possible traceroute: hop #3, attempt #1]
```



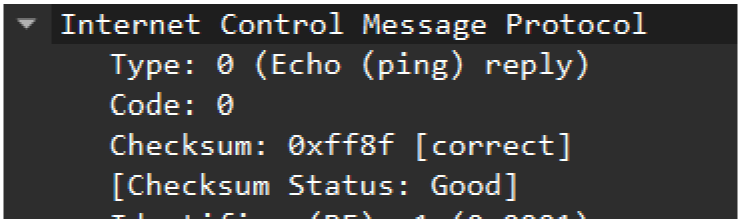
```
▸ Ethernet II, Src: CloudNetwork_11:38:13 (60:e9:aa:11:38:13), Dst: IETF-VRRP-VRID_f6 (00:00:5e:00:01:f6
▾ Internet Protocol Version 4, Src: 10.7.7.70, Dst: 142.250.71.100
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0x5e05 (24069)
  ▸ 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
  ▾ Time to Live: 3
    ▸ [Expert Info (Note/Sequence): "Time To Live" only 3]
    Protocol: UDP (17)
    Header Checksum: 0x7201 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.7.7.70
    Destination Address: 142.250.71.100
    [Stream index: 2]
▾ User Datagram Protocol, Src Port: 60666, Dst Port: 33443
    Source Port: 60666
  ▾ Destination Port: 33443
    ▸ [Expert Info (Chat/Sequence): Possible traceroute: hop #3, attempt #3]
```

4. At the final hop, how is the response different compared to the intermediate hop?

At the final hop, the response we see is the **Destination Unreachable(Port Unreachable) in Linux and ICMP (Echo reply) in Windows,** as compared to the response of **Time to Live exceeded** in case of intermediate hops. At the destination, instead of Time Exceeded, you get an ICMP Port Unreachable (because traceroute uses high-numbered UDP ports).

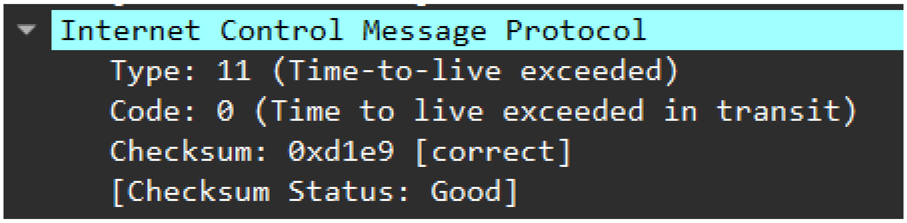


```
▾ Internet Control Message Protocol
    Type: 11 (Time-to-live exceeded)
    Code: 0 (Time to live exceeded in transit)
    Checksum: 0xf4ff [correct]
    [Checksum Status: Good]
```

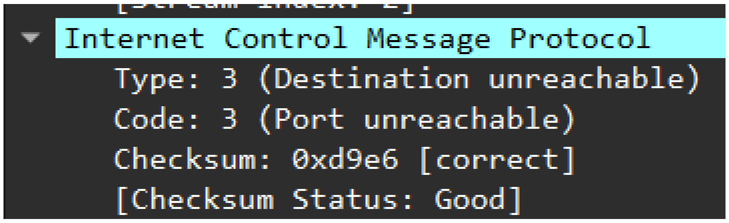


```
▾ Internet Control Message Protocol
    Type: 0 (Echo (ping) reply)
    Code: 0
    Checksum: 0xff8f [correct]
    [Checksum Status: Good]
```

Windows



```
▾ Internet Control Message Protocol
    Type: 11 (Time-to-live exceeded)
    Code: 0 (Time to live exceeded in transit)
    Checksum: 0xd1e9 [correct]
    [Checksum Status: Good]
```



```
▾ Internet Control Message Protocol
    Type: 3 (Destination unreachable)
    Code: 3 (Port unreachable)
    Checksum: 0xd9e6 [correct]
    [Checksum Status: Good]
```

Linux

5. Suppose a firewall blocks UDP traffic but allows ICMP — how would this affect the results of Linux traceroute vs. Windows tracert?

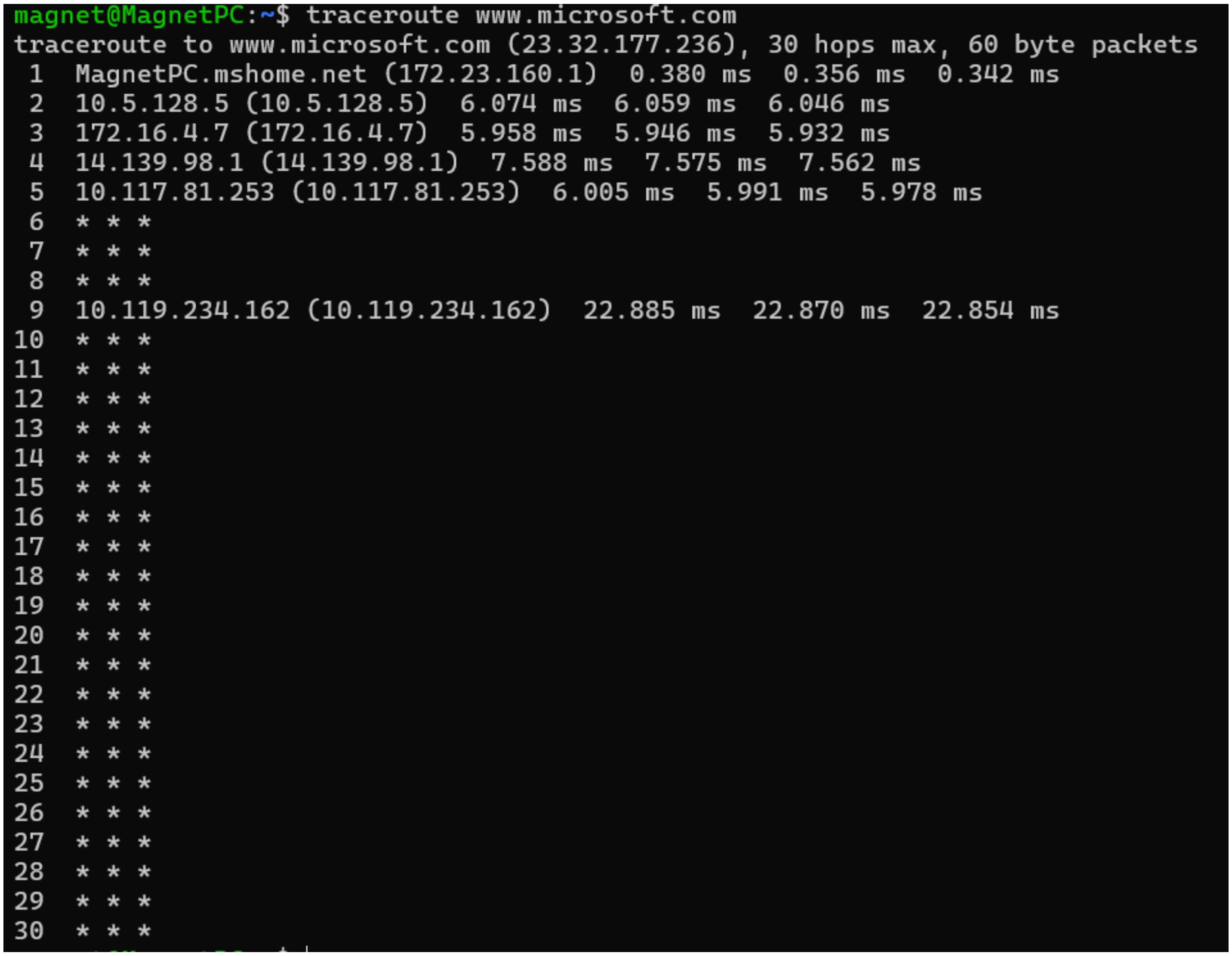
Since Linux uses UDP datagrams by default, blocking them through a firewall would result in -

a. **The probes never reach the routers/destination** (or get dropped silently).
b. You will **not see ICMP Time Exceeded** messages, because the UDP never gets forwarded.
c. The traceroute output will just show `(***)` for many hops and eventually fail.

If the firewall allows ICMP, then:

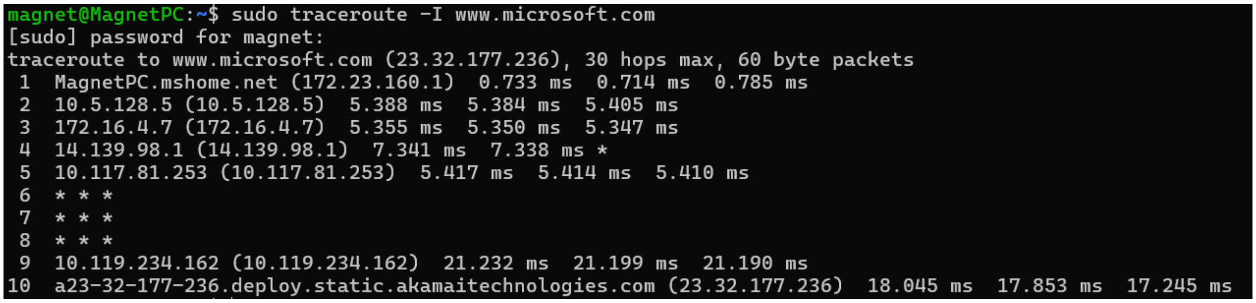
a. Probes go through normally.
b. Intermediate routers still reply with **ICMP Time Exceeded**.
c. Final destination replies with **ICMP Echo Reply**.

Thus, when the **Firewall blocks UDP but allows ICMP,** Linux `traceroute` fails (shows `***`), but Windows `tracert` works normally.



```
magnet@MagnetPC:~$ traceroute www.microsoft.com
traceroute to www.microsoft.com (23.32.177.236), 30 hops max, 60 byte packets
 1  MagnetPC.mshome.net (172.23.160.1)  0.380 ms  0.356 ms  0.342 ms
 2  10.5.128.5 (10.5.128.5)  6.074 ms  6.059 ms  6.046 ms
 3  172.16.4.7 (172.16.4.7)  5.958 ms  5.946 ms  5.932 ms
 4  14.139.98.1 (14.139.98.1)  7.588 ms  7.575 ms  7.562 ms
 5  10.117.81.253 (10.117.81.253)  6.005 ms  5.991 ms  5.978 ms
 6  * * *
 7  * * *
 8  * * *
 9  10.119.234.162 (10.119.234.162)  22.885 ms  22.870 ms  22.854 ms
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *
```

When using default UDP packets on Linux Traceroute, we can see that our requests have been blocked, and we cannot reach the Microsoft server.



```
magnet@MagnetPC:~$ sudo traceroute -I www.microsoft.com
[sudo] password for magnet:
traceroute to www.microsoft.com (23.32.177.236), 30 hops max, 60 byte packets
 1  MagnetPC.mshome.net (172.23.160.1)  0.733 ms  0.714 ms  0.785 ms
 2  10.5.128.5 (10.5.128.5)  5.388 ms  5.384 ms  5.405 ms
 3  172.16.4.7 (172.16.4.7)  5.355 ms  5.350 ms  5.347 ms
 4  14.139.98.1 (14.139.98.1)  7.341 ms  7.338 ms *
 5  10.117.81.253 (10.117.81.253)  5.417 ms  5.414 ms  5.410 ms
 6  * * *
 7  * * *
 8  * * *
 9  10.119.234.162 (10.119.234.162)  21.232 ms  21.199 ms  21.190 ms
10  a23-32-177-236.deploy.static.akamaitechnologies.com (23.32.177.236)  18.045 ms  17.853 ms  17.245 ms
```

When using ICMP packets on Linux Traceroute, we can see that our requests were able to reach the Microsoft server.