

Choosing the pcap file:

V: 23110363

N: 23110224

$$(363+224) \% 10 = 7$$

Task 1

The following table has been extracted after updating the DNS query packets with the custom headers and then sending them to the server for DNS resolution.

This table contains the query name, the Custom header value and its resolved IP address.

The predefined DNS rules helped map the time to the resolved IP address.

1	Custom header value (HHMMSSID)	Domain name	Resolved IP address
2	01003700	wikipedia.org.	192.168.1.11
3	01003701	reddit.com.	192.168.1.12
4	01003702	apple.com.	192.168.1.13
5	01003703	twitter.com.	192.168.1.14
6	01003704	yahoo.com.	192.168.1.15
7	01003705	linkedin.com.	192.168.1.11

Task 2

In this task, we understood how the traceroute utility works in different operating systems, in our case, Windows and Linux.

‘tracert’ on Windows and ‘traceroute’ on Linux are network diagnostic tools. They show the path a packet takes when transmitted from your computer to a host.

Running ‘tracert’ on Windows Command Prompt

tracert www.google.com

```
C:\Users\nupoo>tracert www.google.com

Tracing route to www.google.com [142.251.42.68]
over a maximum of 30 hops:

  1  28 ms    2 ms      1 ms     10.7.0.5
  2  19 ms    3 ms      3 ms     172.16.4.7
  3   7 ms    3 ms      5 ms     14.139.98.1
  4   6 ms    2 ms      2 ms     10.117.81.253
  5  28 ms    59 ms     10 ms     10.154.8.137
  6  12 ms    22 ms     14 ms     10.255.239.170
  7  12 ms    10 ms      9 ms     10.152.7.214
  8  17 ms    18 ms     13 ms     72.14.204.62
  9  22 ms    15 ms     12 ms     72.14.239.103
 10  13 ms    13 ms     13 ms     142.251.69.105
 11  29 ms    68 ms     67 ms     bom12s21-in-f4.1e100.net [142.251.42.68]

Trace complete.
```

Running ‘traceroute’ on Linux Shell

Traceroute isn't installed by default

sudo apt install traceroute

traceroute www.google.com

```
nupoor.ka@NKAssudani:~$ traceroute www.google.com
traceroute to www.google.com (142.250.71.100), 30 hops max, 60 byte packets
 1 NKAssudani.mshome.net (172.18.96.1)  1.203 ms  1.169 ms  1.148 ms
 2 10.7.0.5 (10.7.0.5)  3.596 ms  3.575 ms  3.891 ms
 3 172.16.4.7 (172.16.4.7)  3.640 ms  3.616 ms  3.590 ms
 4 14.139.98.1 (14.139.98.1)  5.839 ms  5.133 ms  5.799 ms
 5 10.117.81.253 (10.117.81.253)  3.708 ms  3.689 ms  3.669 ms
 6 10.154.8.137 (10.154.8.137)  13.974 ms  15.279 ms  14.346 ms
 7 10.255.239.170 (10.255.239.170)  15.305 ms  23.557 ms  23.530 ms
 8 10.152.7.214 (10.152.7.214)  23.412 ms  23.402 ms  22.055 ms
 9 72.14.204.62 (72.14.204.62)  23.488 ms  23.478 ms *
10 * * *
11 192.178.86.244 (192.178.86.244)  24.019 ms  142.250.238.196 (142.250.238.196)  13.900 ms  216.239.58.18 (216.239.58.18)  5.216 ms
12 192.178.110.208 (192.178.110.208)  17.397 ms  192.178.86.247 (192.178.86.247)  14.850 ms  192.178.110.208 (192.178.110.208)  12.973 ms
13 192.178.110.249 (192.178.110.249)  14.059 ms  142.250.226.135 (142.250.226.135)  13.363 ms  pnbomb-ad-in-f4.1e100.net (142.250.71.100)  21.921 ms
```

Network Traffic

The image shows a Wireshark network traffic capture window titled "linux_trcr_capture.pcapng". The filter bar at the top shows "udp && udp.dstport>33433". The packet list on the left shows 35 packets. The packet details pane on the right shows the selected packet (No. 31) expanded, showing the Ethernet II, Internet Protocol Version 4, Internet Control Message Protocol (ICMP), and User Datagram Protocol (UDP) layers.

No.	Time	Source	Destination	Protocol	Length	Info
21	2025-09-13 18:09:12.207904	10.7.60.96	142.251.42.228	UDP	74	58019 → 33452 Len=32
22	2025-09-13 18:09:12.207951	10.7.60.96	142.251.42.228	UDP	74	58038 → 33450 Len=32
23	2025-09-13 18:09:12.251065	10.7.0.5	10.7.60.96	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
24	2025-09-13 18:09:12.251065	172.16.4.7	10.7.60.96	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
25	2025-09-13 18:09:12.251065	172.16.4.7	10.7.60.96	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
26	2025-09-13 18:09:12.251065	172.16.4.7	10.7.60.96	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
27	2025-09-13 18:09:12.252102	10.7.0.5	10.7.60.96	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
28	2025-09-13 18:09:12.252102	10.7.0.5	10.7.60.96	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
29	2025-09-13 18:09:12.252102	10.117.81.253	10.7.60.96	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
30	2025-09-13 18:09:12.252102	10.117.81.253	10.7.60.96	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
31	2025-09-13 18:09:12.252102	10.117.81.253	10.7.60.96	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
32	2025-09-13 18:09:12.252891	10.7.60.96	142.251.42.228	UDP	74	58051 → 33456 Len=32
33	2025-09-13 18:09:12.252891	10.7.60.96	142.251.42.228	UDP	74	58078 → 33454 Len=32
34	2025-09-13 18:09:12.253011	10.7.60.96	142.251.42.228	UDP	74	58021 → 33453 Len=32
35	2025-09-13 18:09:12.253011	10.7.60.96	142.251.42.228	UDP	74	58047 → 33455 Len=32

Packet 31 details:

- Ethernet II, Src: Cisco_6c:2d:7f (88:1d:fc:6c:2d:7f), Dst: Intel_b7:3f:ae (f4:6d:3f)
- Internet Protocol Version 4, Src: 10.7.0.5, Dst: 10.7.60.96
- Internet Control Message Protocol
 - Type: 11 (Time-to-live exceeded)
 - Code: 0 (Time to live exceeded in transit)
 - Checksum: 0xea84 [correct]
 - [Checksum Status: Good]
 - Unused: 00000000
- Internet Protocol Version 4, Src: 10.7.60.96, Dst: 142.251.42.228
- User Datagram Protocol, Src Port: 58071, Dst Port: 33438
 - Source Port: 58071
 - Destination Port: 33438
 - Length: 40
 - Checksum: 0xa4dc [unverified]
 - [Checksum Status: Unverified]
 - [Stream index: 2]

Packet 31 hex dump:

```
0000  f4 6d 3f b7 3f ae 88 1d fc 6c 2d 7f 08 00 45 00  .m??...-1---E-
0010  00 38 da aa 00 00 ff 01 90 a7 0a 07 00 05 0a 07  .8.....
0020  3c 60 0b 00 ea 84 00 00 00 00 45 00 00 3c da aa  <.....E-<-
0030  00 00 01 11 de c0 0a 07 3c 60 8e fb 2a e4 e2 d7  .....<^-*...
0040  82 9e 00 28 a4 dc  ...()
```

Questions

Question 1

In Linux, traceroute uses the TTL (time to live) field in the IP header to send UDP requests to routers in the path to the domain servers to get their IP addresses through ICMP messages.

```

LIST OF AVAILABLE METHODS
In general, a particular traceroute method may have to be chosen by -M name, but most of the methods have their simple
cmdline switches (you can see them after the method name, if present).

default
The traditional, ancient method of tracerouting. Used by default.

Probe packets are udp datagrams with so-called "unlikely" destination ports. The "unlikely" port of the first probe is
33434, then for each next probe it is incremented by one. Since the ports are expected to be unused, the destination host
normally returns "icmp unreachable" as a final response. (Nobody knows what happens when some application listens for
such ports, though).

This method is allowed for unprivileged users.

icmp      -I
Most usual method for now, which uses icmp echo packets for probes.
If you can ping(8) the destination host, icmp tracerouting is applicable as well.

This method may be allowed for unprivileged users since the kernel 3.0 (IPv4, for IPv6 since 3.11), which supports new
dgram icmp (or "ping") sockets. To allow such sockets, sysadmin should provide net/ipv4/ping_group_range sysctl range to
match any group of the user.
Options:

raw       Use only raw sockets (the traditional way).
          This way is tried first by default (for compatibility reasons), then new dgram icmp sockets as fallback.

dgram     Use only dgram icmp sockets.

tcp       -T
Well-known modern method, intended to bypass firewalls.
Uses the constant destination port (default is 80, http).

If some filters are present in the network path, then most probably any "unlikely" udp ports (as for default method) or
even icmp echoes (as for icmp) are filtered, and whole tracerouting will just stop at such a firewall. To bypass a net-
work filter, we have to use only allowed protocol/port combinations. If we trace for some, say, mailserver, then more
Manual page traceroute(1) line 214 (press h for help or q to quit)

```

The default section of the above screenshot shows that the Linux traceroute uses ICMP.

In Windows, tracert uses the ICMP echo protocol, sending ICMP Echo Requests with some times in the TTL field in the IP header to trace the routers in the path between the current device and the server of the domain being looked for.

Question 2

The entry at hop 10 shows ‘* * *’ indicating the router did not reply.

The reasons for this could include that

1. the last router reached at that hop had a firewall or security setup that prevents it from being seen or mapped
2. ICMP messages were dropped at that router due to the rate limit.

```

-z sendwait, --sendwait=sendwait
Minimal time interval between probes (default 0). If the value is more than 10, then it specifies a number in mil-
liseconds, else it is a number of seconds (float point values allowed too). Useful when some routers use rate-
limit for ICMP messages.

```

This program attempts to trace the route an IP packet would follow to some internet host by launching probe packets with a small ttl (time to live) then listening for an ICMP "time exceeded" reply from a gateway. We start our probes with a ttl of one and increase by one until we get an ICMP "port unreachable" (or TCP reset), which means we got to the "host", or hit a max (which defaults to 30 hops). Three probes (by default) are sent at each ttl setting and a line is printed showing the ttl, address of the gateway and round trip time of each probe. The address can be followed by additional information when requested. If the probe answers come from different gateways, the address of each responding system will be printed. If there is no response within a certain timeout, an "*" (asterisk) is printed for that probe.

Question 3

The UDP source port, usually starting from 33434, is incremented by 1 with every change in the probe packet.

```
default
The traditional, ancient method of tracerouting. Used by default.

Probe packets are udp datagrams with so-called "unlikely" destination ports. The "unlikely" port of the first probe is 33434, then for each next probe it is incremented by one. Since the ports are expected to be unused, the destination host normally returns "icmp unreachable port" as a final response. (Nobody knows what happens when some application listens for such ports, though).

This method is allowed for unprivileged users.
```

Question 4

In the case of tracert in Windows, an ICMP Time Exceeded message is received if the TTL has become zero and the destination hasn't been reached. When a probe reaches the destination, and the probe here is an ICMP Echo Request, our device receives an ICMP Echo Reply, and stops probing, having found the steps to the destination.

In the case of traceroute, used in Linux, intermediate hops receive ICMP Time Exceeded messages, whereas the final hop, one that has reached the destination with TTL>0, receives an ICMP Port Unreachable message, as traceroute uses UDP packets with high, almost certainly unused destination ports as probes.

Question 5

If a firewall blocks UDP but allows ICMP, Windows tracert would work but Linux traceroute wouldn't.

This is because Windows uses ICMP Echo, which is allowed by the firewall, while Linux uses UDP, which has been blocked, so the traceroute would fail, leading to [***] for each hop.

The above screenshots show that the traceroute has the default set as UDP.