Assignment 1

Name: Naveen Pal - 23220216
Lakshya Kesarwani 23110187 - 23110187

Task-1: DNS Resolver

NS Query Filtering (filterdns.py)

This script filters DNS query packets from a PCAP file:
Reads the original dns.pcap capture file Extracts only DNS query packets (QR=0). Saves the filtered packets to dns_queries.pcap

Custom Header Addition (modify_query.py)

This script adds custom headers to DNS queries:

Reads the filtered DNS queries. pcap adds an 8-byte custom header in "HHMMSSID" format, where HH is the hour in 24-hour format and MM is the minute. SS: Second ID: Sequence number. Prepend this header to each DNS query. Saves modified packets to dns_custom_header.pcap

Client Implementation (client.py)

The client handles sending queries and processing responses:

Reads the modified packets from dns_custom_header.pcap. Creates a UDP socket to communicate with the server. For each packet: Extracts the payload (custom header + DNS query). Sends it to the server, receives and parses the response, extracts the domain name and resolved IP, Collects results and saves them to dns_results.csv

Server Implementation (server.py)

The server processes requests and applies resolution rules:

Listens for UDP packets on port 5359. For each received packet: Extracts the custom header and DNS query Parses the DNS query to get the domain name. Applies time-based rules to determine which IP to return Builds and sends a DNS response with the resolved IP Logs the transaction

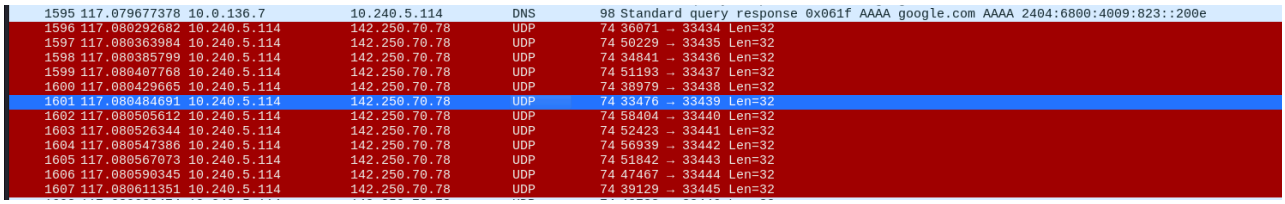
Custom Header	Domain Name	Resolved IP
17210200	_apple-mobdev._tcp.local.	192.168.1.6
17210201	_apple-mobdev._tcp.local.	192.168.1.7
17210202	netflix.com.	192.168.1.8
17210203	Brother MFC-7860DW._pdl-datastream._tcp.local.	192.168.1.9
17210204	Brother MFC-7860DW._pdl-datastream._tcp.local.	192.168.1.10
17210205	linkedin.com.	192.168.1.6
17210206	Brother MFC-7860DW._pdl-datastream._tcp.local.	192.168.1.7
17210207	Brother MFC-7860DW._pdl-datastream._tcp.local.	192.168.1.8
17210208	example.com.	192.168.1.9
17210209	google.com.	192.168.1.10
17210210	_apple-mobdev._tcp.local.	192.168.1.6
17210211	Brother MFC-7860DW._pdl-datastream._tcp.local.	192.168.1.7
17210212	Brother MFC-7860DW._pdl-datastream._tcp.local.	192.168.1.8
17210213	facebook.com.	192.168.1.9
17210214	Brother MFC-7860DW._pdl-datastream._tcp.local.	192.168.1.10
17210215	Brother MFC-7860DW._pdl-datastream._tcp.local.	192.168.1.6
17210216	_apple-mobdev._tcp.local.	192.168.1.7
17210217	_apple-mobdev._tcp.local.	192.168.1.8
17210218	Brother MFC-7860DW._pdl-datastream._tcp.local.	192.168.1.9
17210219	Brother MFC-7860DW._pdl-datastream._tcp.local.	192.168.1.10
17210220	Brother MFC-7860DW._pdl-datastream._tcp.local.	192.168.1.6
17210221	Brother MFC-7860DW._pdl-datastream._tcp.local.	192.168.1.7
17210222	amazon.com.	192.168.1.8

Task-2:

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

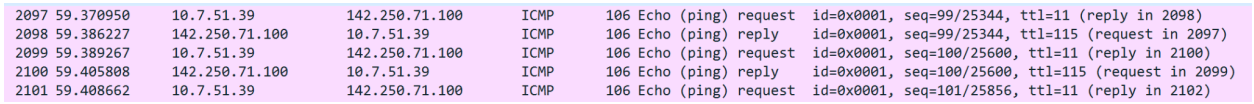
→ **Windows tracert** uses **ICMP Echo Request** packets by default. **Linux traceroute** traditionally uses **UDP probes** to high-numbered ports by default. (can use ICMP with options)

Linux



1595	117.079677378	10.0.136.7	10.240.5.114	DNS	98 Standard query response 0x061f AAAA google.com AAAA 2404:6800:4009:823::200e
1596	117.080292682	10.240.5.114	142.250.70.78	UDP	74 36071 → 33434 Len=32
1597	117.080363984	10.240.5.114	142.250.70.78	UDP	74 50229 → 33435 Len=32
1598	117.080385799	10.240.5.114	142.250.70.78	UDP	74 34841 → 33436 Len=32
1599	117.080407768	10.240.5.114	142.250.70.78	UDP	74 51193 → 33437 Len=32
1600	117.080429665	10.240.5.114	142.250.70.78	UDP	74 38979 → 33438 Len=32
1601	117.080484691	10.240.5.114	142.250.70.78	UDP	74 33476 → 33439 Len=32
1602	117.080505612	10.240.5.114	142.250.70.78	UDP	74 58404 → 33440 Len=32
1603	117.080526344	10.240.5.114	142.250.70.78	UDP	74 52423 → 33441 Len=32
1604	117.080547386	10.240.5.114	142.250.70.78	UDP	74 56939 → 33442 Len=32
1605	117.080567073	10.240.5.114	142.250.70.78	UDP	74 51842 → 33443 Len=32
1606	117.080590345	10.240.5.114	142.250.70.78	UDP	74 47467 → 33444 Len=32
1607	117.080611351	10.240.5.114	142.250.70.78	UDP	74 39129 → 33445 Len=32

Windows



2097	59.370950	10.7.51.39	142.250.71.100	ICMP	106 Echo (ping) request id=0x0001, seq=99/25344, ttl=11 (reply in 2098)
2098	59.386227	142.250.71.100	10.7.51.39	ICMP	106 Echo (ping) reply id=0x0001, seq=99/25344, ttl=115 (request in 2097)
2099	59.389267	10.7.51.39	142.250.71.100	ICMP	106 Echo (ping) request id=0x0001, seq=100/25600, ttl=11 (reply in 2100)
2100	59.405808	142.250.71.100	10.7.51.39	ICMP	106 Echo (ping) reply id=0x0001, seq=100/25600, ttl=115 (request in 2099)
2101	59.408662	10.7.51.39	142.250.71.100	ICMP	106 Echo (ping) request id=0x0001, seq=101/25856, ttl=11 (reply in 2102)

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

→ 2 reasons for the reasons are:

1. The router is overloaded/drops low-priority ICMP.
2. For security many router configured not to send TTL Exceed signal

```
 5  10.154.8.137 (10.154.8.137)  14.394 ms  13.264 ms  11.995 ms
 6  10.255.239.170 (10.255.239.170)  10.936 ms  12.320 ms  10.494 ms
 7  10.152.7.214 (10.152.7.214)  11.727 ms  11.710 ms  12.622 ms
 8  72.14.204.62 (72.14.204.62)  12.603 ms  13.857 ms *
 9  * * *
10  192.178.86.202 (192.178.86.202)  15.944 ms 142.250.228.46 (142.250.228.46)  18.655 ms 17
11  192.178.110.208 (192.178.110.208)  21.564 ms 192.178.110.104 (192.178.110.104)  14.490 m
12  192.178.110.249 (192.178.110.249)  17.037 ms 142.251.77.69 (142.251.77.69)  11.499 ms 14
13  pnbomb-bd-in-f14.1e100.net (142.251.220.78)  20.743 ms 142.250.214.105 (142.250.214.105)
```

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

→ The **TTL (Time-To-Live)** field changes between successive probes.

```
dnsm.iitgn.ac.in.domain > subh.34630: 5202 NXDomain 0/0/1 (55)
02:28:58.044280 IP (tos 0x0, ttl 6, id 22931, offset 0, flags [none], proto UDP (17), length 60)
    subh.39867 > pnbomb-bp-in-f14.1e100.net.33451: UDP, length 32
02:28:58.044361 IP (tos 0x0, ttl 7, id 2273, offset 0, flags [none], proto UDP (17), length 60)
    subh.57910 > pnbomb-bp-in-f14.1e100.net.33452: UDP, length 32
02:28:58.044411 IP (tos 0x0, ttl 7, id 1334, offset 0, flags [none], proto UDP (17), length 60)
    subh.46543 > pnbomb-bp-in-f14.1e100.net.33453: UDP, length 32
02:28:58.044459 IP (tos 0x0, ttl 7, id 58788, offset 0, flags [none], proto UDP (17), length 60)
    subh.59477 > pnbomb-bp-in-f14.1e100.net.33454: UDP, length 32
02:28:58.044504 IP (tos 0x0, ttl 8, id 58093, offset 0, flags [none], proto UDP (17), length 60)
    subh.36561 > pnbomb-bp-in-f14.1e100.net.33455: UDP, length 32
02:28:58.044550 IP (tos 0x0, ttl 8, id 59579, offset 0, flags [none], proto UDP (17), length 60)
    subh.42208 > pnbomb-bp-in-f14.1e100.net.33456: UDP, length 32
02:28:58.044596 IP (tos 0x0, ttl 8, id 10764, offset 0, flags [none], proto UDP (17), length 60)
    subh.47659 > pnbomb-bp-in-f14.1e100.net.33457: UDP, length 32
02:28:58.044642 IP (tos 0x0, ttl 9, id 43257, offset 0, flags [none], proto UDP (17), length 60)
    subh.39090 > pnbomb-bp-in-f14.1e100.net.33458: UDP, length 32
02:28:58.044686 IP (tos 0x0, ttl 9, id 63236, offset 0, flags [none], proto UDP (17), length 60)
    subh.41643 > pnbomb-bp-in-f14.1e100.net.33459: UDP, length 32
```

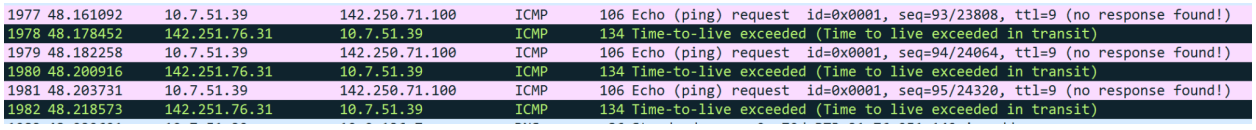
4. At the final hop, how is the response different compared to the intermediate hop?

→ **Intermediate hops** send back **ICMP Time Exceeded** messages (because TTL expired).

The **final destination** sends back an **ICMP Port Unreachable** (for Linux UDP-based traceroute) or an **ICMP Echo Reply** (for Windows tracert).

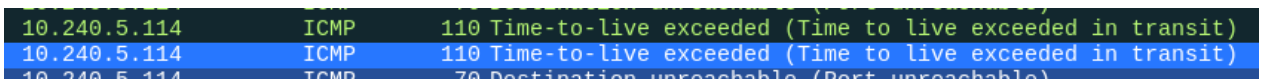
Intermediate:

Windows:



1977 48.161092	10.7.51.39	142.250.71.100	ICMP	106 Echo (ping) request id=0x0001, seq=93/23808, ttl=9 (no response found!)
1978 48.178452	142.251.76.31	10.7.51.39	ICMP	134 Time-to-live exceeded (Time to live exceeded in transit)
1979 48.182258	10.7.51.39	142.250.71.100	ICMP	106 Echo (ping) request id=0x0001, seq=94/24064, ttl=9 (no response found!)
1980 48.200916	142.251.76.31	10.7.51.39	ICMP	134 Time-to-live exceeded (Time to live exceeded in transit)
1981 48.203731	10.7.51.39	142.250.71.100	ICMP	106 Echo (ping) request id=0x0001, seq=95/24320, ttl=9 (no response found!)
1982 48.218573	142.251.76.31	10.7.51.39	ICMP	134 Time-to-live exceeded (Time to live exceeded in transit)

Linux:



10.240.5.114	ICMP	110 Time-to-live exceeded (Time to live exceeded in transit)
10.240.5.114	ICMP	110 Time-to-live exceeded (Time to live exceeded in transit)

Final Hop:

Windows:

```
2097 59.370950    10.7.51.39          142.250.71.100      ICMP     106 Echo (ping) request  id=0x0001, seq=99/25344, ttl=11 (reply in 2098)
2098 59.386227    142.250.71.100      10.7.51.39          ICMP     106 Echo (ping) reply    id=0x0001, seq=99/25344, ttl=115 (request in 2097)
2099 59.389267    10.7.51.39          142.250.71.100      ICMP     106 Echo (ping) request  id=0x0001, seq=100/25600, ttl=11 (reply in 2100)
2100 59.405808    142.250.71.100      10.7.51.39          ICMP     106 Echo (ping) reply    id=0x0001, seq=100/25600, ttl=115 (request in 2099)
2101 59.408662    10.7.51.39          142.250.71.100      ICMP     106 Echo (ping) request  id=0x0001, seq=101/25856, ttl=11 (reply in 2102)
```

Linux:

```
17:12:55.015505 IP pnbomb-ab-in-f14.1e100.net > 10.240.5.114: ICMP pnbomb-ab-in-f14.1e100.net udp port 33481 unreachable, length 36
```

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

→ **Linux traceroute** (default UDP) would fail and show *** at/after the firewall, because UDP probes never get responses. **Windows tracert** (ICMP) would still work, since it uses ICMP Echo Requests and receives ICMP replies.

```
moba@subh:~/Documents/courseprep/CN/Assignment1$ traceroute google.com
traceroute to google.com (142.251.220.78), 30 hops max, 60 byte packets
 1  10.7.0.5 (10.7.0.5)  1.927 ms  2.870 ms  3.181 ms
 2  172.16.4.7 (172.16.4.7)  5.030 ms  5.004 ms  4.980 ms
 3  14.139.98.1 (14.139.98.1)  6.520 ms  8.662 ms  7.507 ms
 4  10.117.81.253 (10.117.81.253)  5.445 ms  4.871 ms  5.709 ms
 5  10.154.8.137 (10.154.8.137)  14.394 ms  13.264 ms  11.995 ms
 6  10.255.239.170 (10.255.239.170)  10.936 ms  12.320 ms  10.494 ms
 7  10.152.7.214 (10.152.7.214)  11.727 ms  11.710 ms  12.622 ms
 8  72.14.204.62 (72.14.204.62)  12.603 ms  13.857 ms *
 9  * * *
10  192.178.86.202 (192.178.86.202)  15.944 ms 142.250.228.46 (142.250.228.46)  18.655 ms 172.253.7
11  192.178.110.208 (192.178.110.208)  21.564 ms 192.178.110.104 (192.178.110.104)  14.490 ms 192.1
12  192.178.110.249 (192.178.110.249)  17.037 ms 142.251.77.69 (142.251.77.69)  11.499 ms 142.250.2
13  pnbomb-bd-in-f14.1e100.net (142.251.220.78)  20.743 ms 142.250.214.105 (142.250.214.105)  13.80
```

```
PS C:\Users\LAKSHYA> tracert www.google.com

Tracing route to www.google.com [142.250.71.100]
over a maximum of 30 hops:

  1     4 ms     2 ms     2 ms  10.7.0.5
  2     5 ms     5 ms     5 ms  172.16.4.7
  3     5 ms     4 ms     4 ms  14.139.98.1
  4     5 ms     4 ms     5 ms  10.117.81.253
  5    13 ms    10 ms    10 ms  10.154.8.137
  6    10 ms    10 ms    10 ms  10.255.239.170
  7    10 ms    10 ms    10 ms  10.152.7.214
  8    15 ms    15 ms    15 ms  142.250.172.80
  9    18 ms    19 ms    15 ms  142.251.76.31
 10    13 ms    12 ms    12 ms  192.178.86.247
 11    16 ms    17 ms    15 ms  pnbomb-ad-in-f4.1e100.net
0.71.100]

Trace complete.
```