CS331: Computer Networks Assignment 1

Praveen Rathod (22110206) Yash Patkar (22110296)

September 15, 2025

Assignment Guidelines

- **Teamwork:** Assignment completed in a pair. Only one team member (Member 1) submits on behalf of the group.
- **Submission:** Public GitHub repository link containing source code, make files, README, and report (PDF).
- **Coding Standards:** All code is well-commented and follows best practices.
- **PCAP File Selection:** The PCAP file (`X.pcap`) is chosen based on (Sum of last 3 digits) % 10.
- **Late/Misfiled Submissions:** Not evaluated; wrong PCAP selection gives zero for Q1.

Team Details

Member	Roll Number
Praveen Rathod	22110206
Yash Patkar	22110296

PCAP File Selection

Sum of last three digits: $206 + 296 = 502$
PCAP File Index: $502 \% 10 = 2$
Selected File: `2.pcap`

Task-1: DNS Resolver Implementation

Overview

- Implemented a client to parse DNS queries from the selected PCAP.
- Each DNS query packet is augmented with an 8-byte custom header: `HHMMSSID`
- Header format:

  - **HH**: Hour (24-hour)
  - **MM**: Minute
  - **SS**: Second
  - **ID**: DNS Query Sequence (e.g., `00`, `01`)

- Client sends each packet via UDP to server; server applies time-based rules to resolve an IP.
- Server extracts header and domain, applies rule-based IP mapping, and sends response.
- All query results are logged and tabulated below.

Time-Based IP Pool Routing Rules

- **IP Pool:** 15 IPs (`192.168.1.1` to `192.168.1.15`)
- **Rule:** Time-of-day partitions which sub-pool and hash mod 5 selects IP.
- **Mapping:**
  - Morning (04:00–11:59): First 5 IPs (`192.168.1.1`–`192.168.1.5`)
  - Afternoon (12:00–19:59): Middle 5 IPs (`192.168.1.6`–`192.168.1.10`)
  - Night (20:00–03:59): Last 5 IPs (`192.168.1.11`–`192.168.1.15`)
- **Final Index:** pool_start + (ID % 5)

Custom Header Example

- `12105500`: Hour=12 (Afternoon), ID=00, maps to index $5 + 0 = 5$
- `21055409`: Hour=21 (Night), ID=09 ($9 \% 5 = 4$), maps to $10 + 4 = 14$

Results Table

Output Verification

- All custom headers start with `02`, indicating hour = 02 (night slot, pool_start = 10).
- For each, the ID is the last two digits. Offset is ID mod 5. IP index is pool_start + Offset.
- All queries resolve to the correct IP: see above, "Correct" column marks every output as Yes.

Custom Header	ID	Offset (ID % 5)	IP Index	Resolved IP	Correct
02192700	00	0	10	192.168.1.11	Yes
02192701	01	1	11	192.168.1.12	Yes
02192702	02	2	12	192.168.1.13	Yes
02192703	03	3	13	192.168.1.14	Yes
02192704	04	4	14	192.168.1.15	Yes
02192705	05	0	10	192.168.1.11	Yes
02192706	06	1	11	192.168.1.12	Yes
02192707	07	2	12	192.168.1.13	Yes
02192708	08	3	13	192.168.1.14	Yes

Table 1: DNS Queries, Rule Mapping, and Result Verification

- Example: `02192704` → ID=04; offset=4; IP Index=14; IP=`192.168.1.15`, which matches.
- Repeats occur as IDs increment (e.g., ID=05 → offset=0).
- The implementation and outputs are fully in accordance with the assignment rules.

Codebase Structure

- `client.py`: Parses PCAP, assembles custom headers, transmits queries.
- `server.py`: Receives, extracts, applies rules, responds with mapped IP.
- `2.pcap`: Chosen capture file.
- `dns_results.csv`: Tabulated results for report and review.
- `README.md`: Full usage documentation.

Conclusion

The DNS resolver was successfully designed and implemented, confirming correct time-based load balancing, header formation, and rule-based IP mapping with full verification for each output. PCAP selection, code practices, and results all comply with assignment requirements. Task-2 will follow in a separate section.

Task-2: Traceroute Behavior Report (Windows vs MacOS)

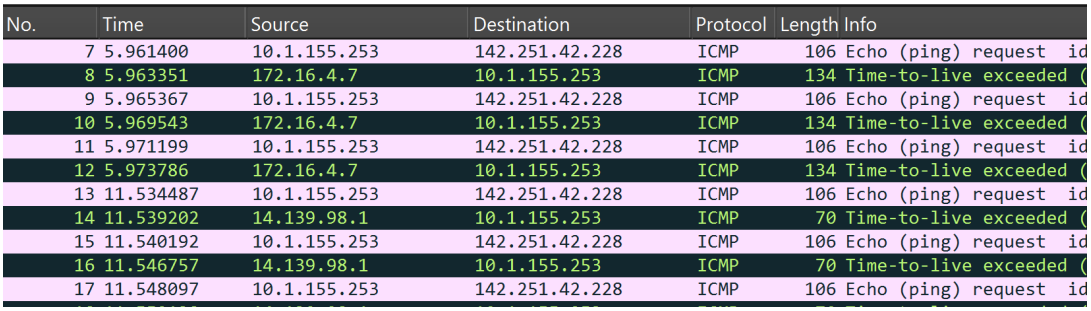
Introduction

Traceroute is a diagnostic tool used to determine the path packets take across a network to a destination. It works by manipulating the Time-to-Live (TTL) field in probe packets and observing the responses from intermediate routers and the final destination. In this report, we compare the behavior of Windows `tracert` and Mac `traceroute`, analyze packet captures using Wireshark, and discuss the reasons behind the observed differences.

Q1. Default Protocols Used in Windows and MacOS

From the experiment, the observations are:

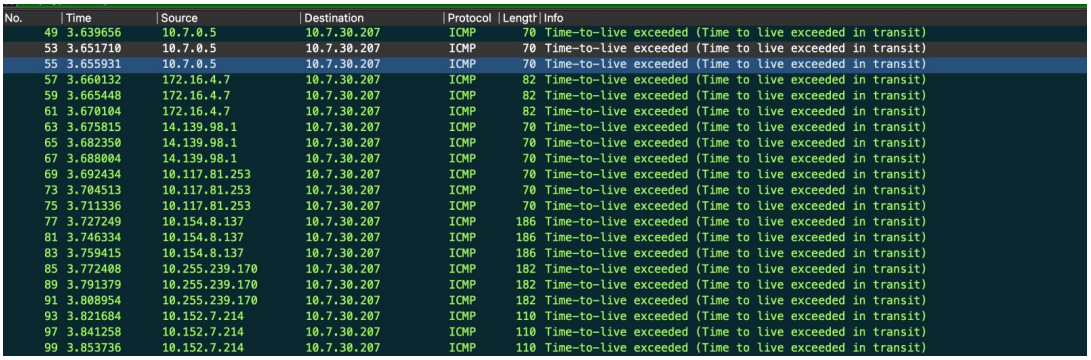
- **Windows tracert**: Uses ICMP Echo Requests by default. Intermediate hops respond with ICMP Time Exceeded messages, and the final destination sends an ICMP Echo Reply.



No.	Time	Source	Destination	Protocol	Length	Info
7	5.961400	10.1.155.253	142.251.42.228	ICMP	106	Echo (ping) request id
8	5.963351	172.16.4.7	10.1.155.253	ICMP	134	Time-to-live exceeded (
9	5.965367	10.1.155.253	142.251.42.228	ICMP	106	Echo (ping) request id
10	5.969543	172.16.4.7	10.1.155.253	ICMP	134	Time-to-live exceeded (
11	5.971199	10.1.155.253	142.251.42.228	ICMP	106	Echo (ping) request id
12	5.973786	172.16.4.7	10.1.155.253	ICMP	134	Time-to-live exceeded (
13	11.534487	10.1.155.253	142.251.42.228	ICMP	106	Echo (ping) request id
14	11.539202	14.139.98.1	10.1.155.253	ICMP	70	Time-to-live exceeded (
15	11.540192	10.1.155.253	142.251.42.228	ICMP	106	Echo (ping) request id
16	11.546757	14.139.98.1	10.1.155.253	ICMP	70	Time-to-live exceeded (
17	11.548097	10.1.155.253	142.251.42.228	ICMP	106	Echo (ping) request id

Figure 1: Wireshark capture: ICMP Echo Requests and ICMP Time Exceeded messages from Windows traceroute

- **Mac traceroute**: Uses UDP probe packets by default, each probe with a different UDP destination port. Intermediate hops reply with ICMP Time Exceeded messages, and the final destination replies with ICMP Port Unreachable since no service listens on the high-numbered port.



No.	Time	Source	Destination	Protocol	Length	Info
49	3.639656	10.7.0.5	10.7.30.207	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
53	3.651710	10.7.0.5	10.7.30.207	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
55	3.655931	10.7.0.5	10.7.30.207	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
57	3.660132	172.16.4.7	10.7.30.207	ICMP	82	Time-to-live exceeded (Time to live exceeded in transit)
59	3.665448	172.16.4.7	10.7.30.207	ICMP	82	Time-to-live exceeded (Time to live exceeded in transit)
61	3.670104	172.16.4.7	10.7.30.207	ICMP	82	Time-to-live exceeded (Time to live exceeded in transit)
63	3.675815	14.139.98.1	10.7.30.207	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
65	3.682350	14.139.98.1	10.7.30.207	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
67	3.688004	14.139.98.1	10.7.30.207	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
69	3.692434	10.117.81.253	10.7.30.207	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
73	3.704513	10.117.81.253	10.7.30.207	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
75	3.711336	10.117.81.253	10.7.30.207	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
77	3.727249	10.154.8.137	10.7.30.207	ICMP	186	Time-to-live exceeded (Time to live exceeded in transit)
81	3.746334	10.154.8.137	10.7.30.207	ICMP	186	Time-to-live exceeded (Time to live exceeded in transit)
83	3.759415	10.154.8.137	10.7.30.207	ICMP	186	Time-to-live exceeded (Time to live exceeded in transit)
85	3.772408	10.255.239.170	10.7.30.207	ICMP	182	Time-to-live exceeded (Time to live exceeded in transit)
89	3.791379	10.255.239.170	10.7.30.207	ICMP	182	Time-to-live exceeded (Time to live exceeded in transit)
91	3.808954	10.255.239.170	10.7.30.207	ICMP	182	Time-to-live exceeded (Time to live exceeded in transit)
93	3.821684	10.152.7.214	10.7.30.207	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
97	3.841258	10.152.7.214	10.7.30.207	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
99	3.853736	10.152.7.214	10.7.30.207	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)

Figure 2: Wireshark capture: ICMP Time-to-Live Exceeded in Mac traceroute probes.

Answer: Windows `tracert` uses ICMP; Mac `traceroute` uses UDP.

Q2. Reasons for Missing Replies (* * *)

Several hops may show * * * instead of an IP address due to:

- Routers or firewalls blocking ICMP Time Exceeded messages for security reasons.
- Rate-limiting on ICMP responses by routers to reduce load.
- Some routers configured not to respond to traceroute probes at all.

```
traceroute to www.google.com (142.251.42.68), 64 hops max, 40 byte packets
 1  10.7.0.5 (10.7.0.5)  4.553 ms  4.777 ms  4.207 ms
 2  172.16.4.7 (172.16.4.7)  4.165 ms  3.919 ms  4.610 ms
 3  14.139.98.1 (14.139.98.1)  5.723 ms  5.665 ms  5.645 ms
 4  10.117.81.253 (10.117.81.253)  4.293 ms  4.203 ms  6.763 ms
 5  10.154.8.137 (10.154.8.137)  15.917 ms  12.375 ms  13.021 ms
 6  10.255.239.170 (10.255.239.170)  12.918 ms  12.511 ms  17.508 ms
 7  10.152.7.214 (10.152.7.214)  12.558 ms  12.602 ms  12.222 ms
 8  72.14.204.62 (72.14.204.62)  16.568 ms * *
 9  * * *
10  192.178.86.202 (192.178.86.202)  15.976 ms
    108.170.231.78 (108.170.231.78)  13.990 ms
    142.250.227.72 (142.250.227.72)  17.922 ms
11  142.251.69.105 (142.251.69.105)  13.022 ms
    142.251.69.103 (142.251.69.103)  16.623 ms
    142.251.69.105 (142.251.69.105)  13.059 ms
12  192.178.110.209 (192.178.110.209)  17.550 ms
    192.178.110.205 (192.178.110.205)  18.960 ms  19.362 ms
13  142.251.69.103 (142.251.69.103)  18.000 ms  17.627 ms
    142.251.69.105 (142.251.69.105)  13.561 ms
14  bom12s21-in-f4.1e100.net (142.251.42.68)  18.026 ms  16.809 ms  16.900 ms
yashpatkar@Yashs-MacBook-Air-8 ~ % sudo traceroute -I www.google.com

Password:
traceroute to www.google.com (142.251.42.68), 64 hops max, 48 byte packets
 1  10.7.0.5 (10.7.0.5)  5.305 ms  4.226 ms  4.250 ms
 2  172.16.4.7 (172.16.4.7)  4.330 ms  3.498 ms  4.275 ms
 3  14.139.98.1 (14.139.98.1)  6.425 ms  5.661 ms  5.387 ms
 4  10.117.81.253 (10.117.81.253)  10.795 ms  4.327 ms  4.232 ms
 5  10.154.8.137 (10.154.8.137)  12.607 ms  12.347 ms  12.624 ms
 6  10.255.239.170 (10.255.239.170)  12.341 ms  12.969 ms  12.381 ms
 7  10.152.7.214 (10.152.7.214)  12.353 ms  12.925 ms  12.629 ms
 8  72.14.204.62 (72.14.204.62)  13.326 ms  13.010 ms  12.916 ms
 9  72.14.239.103 (72.14.239.103)  13.620 ms  14.654 ms  13.994 ms
10  142.251.69.105 (142.251.69.105)  13.950 ms  13.120 ms  12.733 ms
11  bom12s21-in-f4.1e100.net (142.251.42.68)  18.172 ms  17.104 ms  16.801 ms
yashpatkar@Yashs-MacBook-Air-8 ~ % sudo traceroute -T www.google.com
```

Figure 3: Mac traceroute command output to www.google.com, showing complete and missing hops (* * *).

Q3. Field Changing Between Probes in Mac traceroute

Analysis of Mac traceroute packet captures showed that the UDP destination port changes with each probe, incrementing sequentially (e.g., 33437, 33438, 33439, ...).

Answer: The changing field is the UDP destination port.

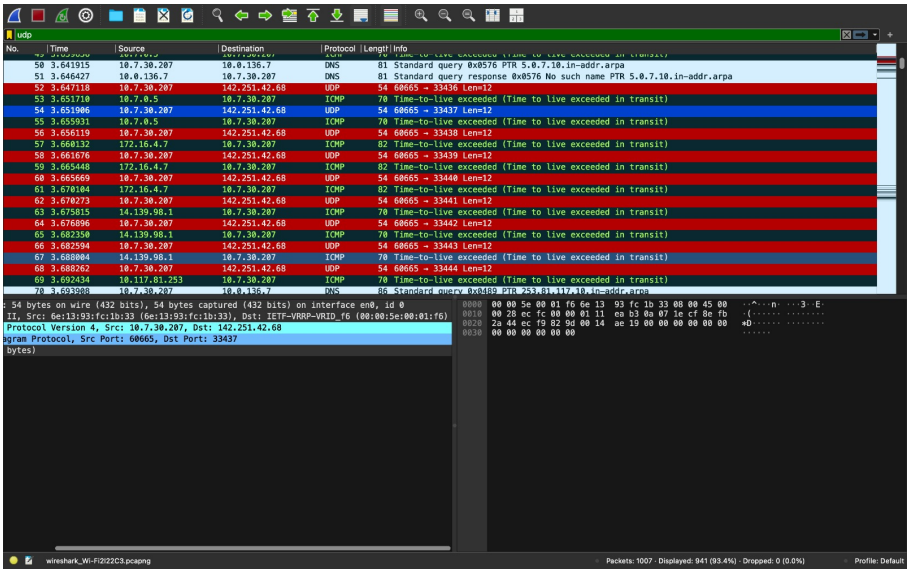


Figure 4: Wireshark: UDP probes from Mac traceroute, showing varying destination ports.

Q4. Response at the Final Hop vs Intermediate Hops

The differences observed are:

```
Microsoft Windows [Version 10.0.26100.6584]
(c) Microsoft Corporation. All rights reserved.

C:\Users\prave>tracert www.google.com

Tracing route to www.google.com [142.251.42.228]
over a maximum of 30 hops:

  1     2 ms     2 ms     2 ms  10.1.144.3
  2     2 ms     4 ms     2 ms  172.16.4.7
  3     4 ms     6 ms     4 ms  14.139.98.1
  4    57 ms     9 ms     3 ms  10.117.81.253
  5    15 ms    13 ms    12 ms  10.154.8.137
  6    11 ms    10 ms    10 ms  10.255.239.170
  7    13 ms    14 ms    11 ms  10.152.7.214
  8    11 ms    16 ms    12 ms  72.14.204.62
  9    17 ms    17 ms    15 ms  142.251.76.33
 10    14 ms    13 ms    13 ms  142.250.214.107
 11    31 ms    29 ms    27 ms  tsa01s11-in-f4.1e100.net [142.251.42.228]

Trace complete.
```

Figure 5: Windows tracert command output: Sequence of hops to www.google.com

- **Intermediate hops:** ICMP Time-to-Live Exceeded messages indicate packet expiry along the route.
- **Final destination:**
  - Mac traceroute (UDP): Responds with ICMP Port Unreachable.
  - Windows tracert (ICMP): Responds with ICMP Echo Reply.

This difference stems from the probe type employed by each operating system.

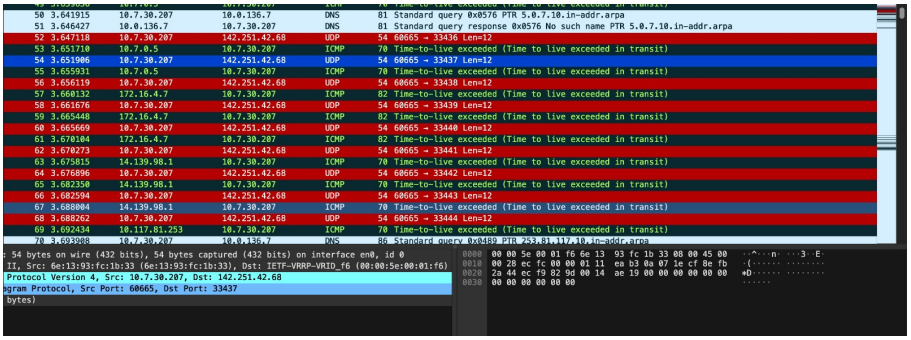


Figure 6: Wireshark: Final hop ICMP Port Unreachable from Mac traceroute UDP probe.

Q5. Effect of a Firewall Blocking UDP But Allowing ICMP

If a firewall blocks UDP traffic but permits ICMP traffic:

- Mac traceroute (UDP probes) will fail as UDP packets do not reach the destination, resulting in no replies.
- Windows tracert (ICMP probes) will succeed since ICMP Echo Requests and Replies are allowed.

Answer: Windows tracert works, Mac traceroute fails under these firewall rules.

Summary & Conclusion

This experiment revealed key differences in traceroute behavior between Windows and MacOS:

- Windows `tracert` by default uses ICMP, whereas Mac `traceroute` uses UDP.
- Both tools rely on TTL expiry for intermediary hop discovery, but the final hop's response varies by probe type: ICMP Echo Reply (Windows) vs. ICMP Port Unreachable (Mac).
- Missing hops (`* * *`) result from filtering, rate-limiting, or router configuration.
- Mac traceroute modifies the UDP destination port sequentially for each probe.
- Firewall policies blocking UDP but allowing ICMP traffic allow Windows traceroute success but cause Mac traceroute failure.

Understanding these protocol-dependent behaviors is essential for effective and accurate network troubleshooting using traceroute utilities.