Capture and Analyze Network Traffic Using Wireshark

Step 1 Install Wireshark (and Npcap) on your Windows laptop

1. On your Windows machine, download the **latest stable Windows installer** of Wireshark from the official site (use the Windows 64-bit installer if your OS is 64-bit).

2. Right-click the installer and **Run as administrator**. (Capturing packets requires elevated rights; installing Npcap will ask for admin consent.)

3. During the installer steps:

    - **Allow** installation of **Npcap** (this is the packet capture driver).

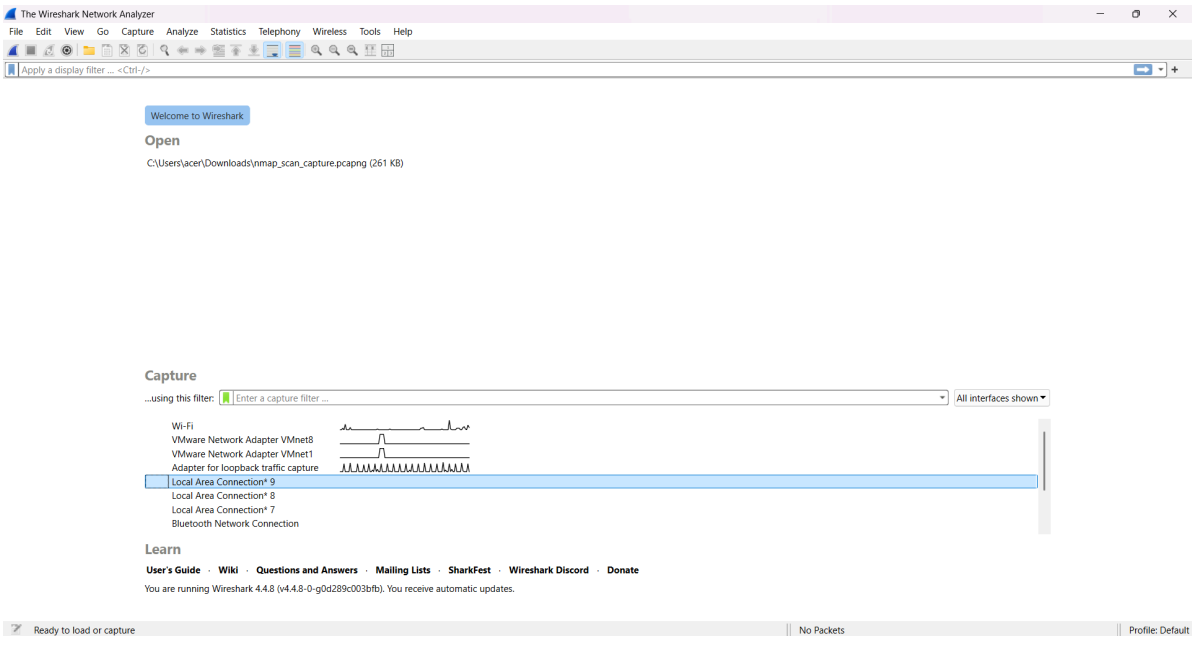
    - If prompted, **enable the WinPcap API-compatibility** option (helps compatibility with some tools).

    - Leave default options unless you know you need monitor mode (don't enable advanced 802.11/raw options unless you understand them).

4. Finish installation and **reboot** if the installer asks.

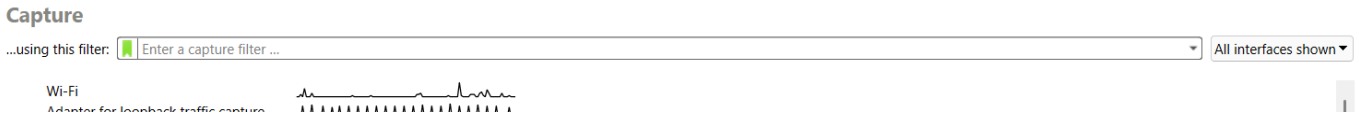
5. After reboot, open Wireshark (you don't need to "Run as admin" every time, but you may if you get permission errors).

6. Verify installation: from the Wireshark menu choose **Help > About Wireshark** (note the version) and then **Capture > Options** you should see a list of network interfaces (Wi-Fi / Ethernet).

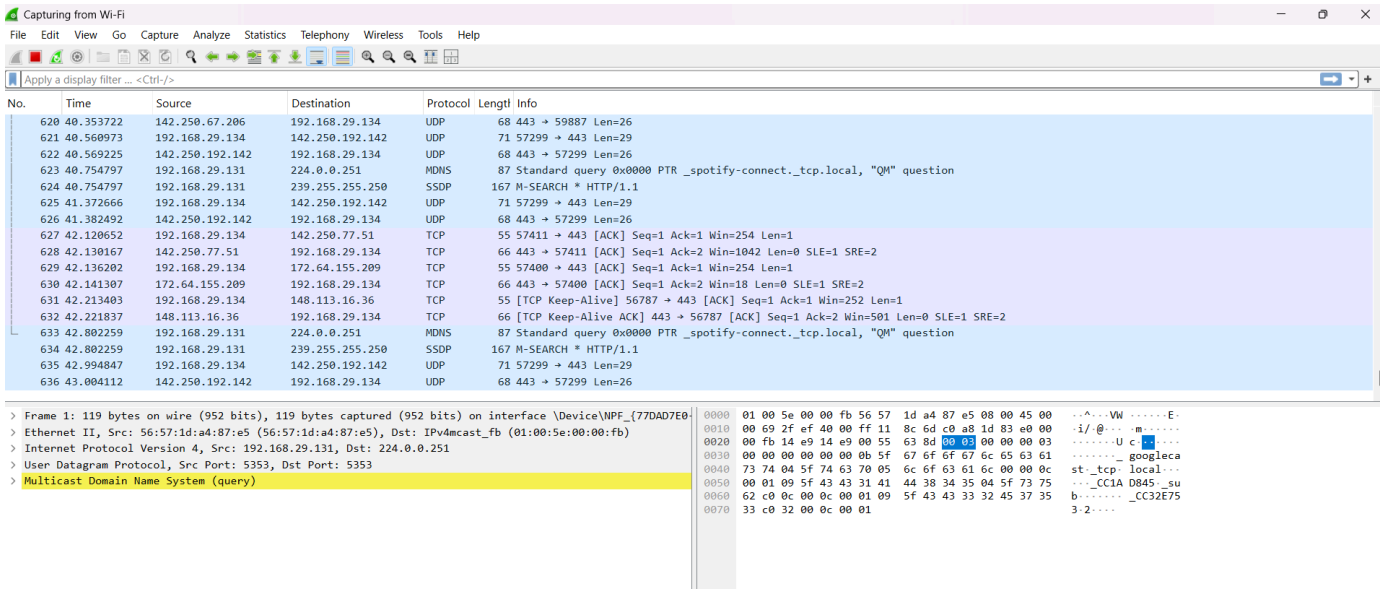


Step 2 Start a Packet Capture

1. **Open Wireshark**.
2. In the main window, under **“Capture”**, locate your active network interface:
    - If you’re on Wi-Fi, it will usually be named something like Wi-Fi or WLAN.
    - If you’re on Ethernet, it will be Ethernet.
3. **Click once** on your active interface to highlight it.



4. In the toolbar, click the blue shark fin icon (or press Ctrl + E) to start capturing packets.
5. You will now see packets scrolling in real time. This is your **raw packet capture**.



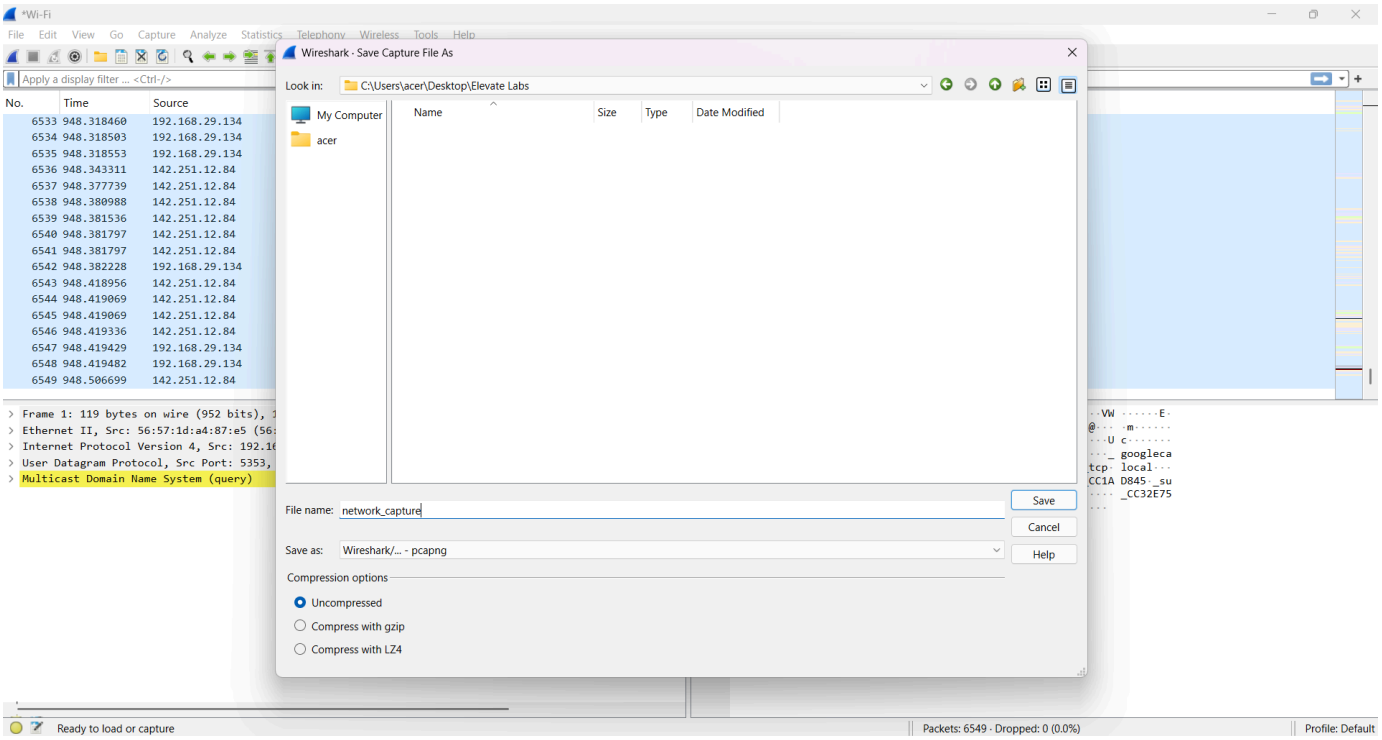
6. Let it run for **about 1–2 minutes** while you perform normal internet activity (e.g., open a few websites, check email). This will ensure you capture different protocols for later analysis.

Step 3 Save and Name the Capture

1. In Wireshark, click the Stop button beside the blue shark fin icon and then click File > Save As (or press Ctrl + S).

2. Choose a folder where you'll keep all internship-related files.
3. Give the file a descriptive name:
   Network_capture.pcapng

(".pcapng" is the default Wireshark format and keeps all capture details.)

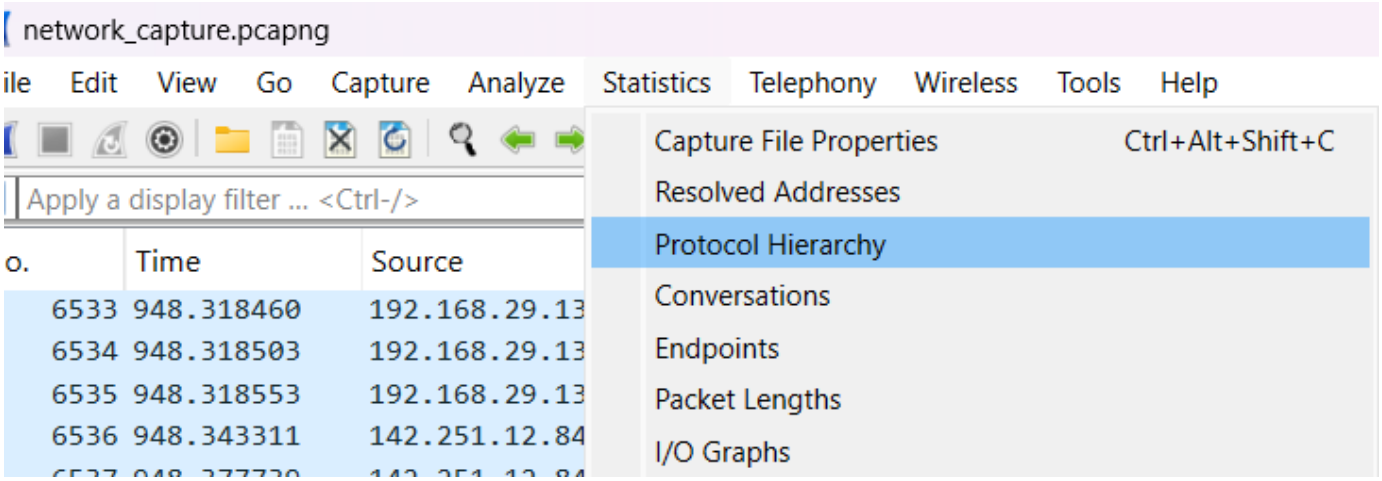


4. Click **Save**.

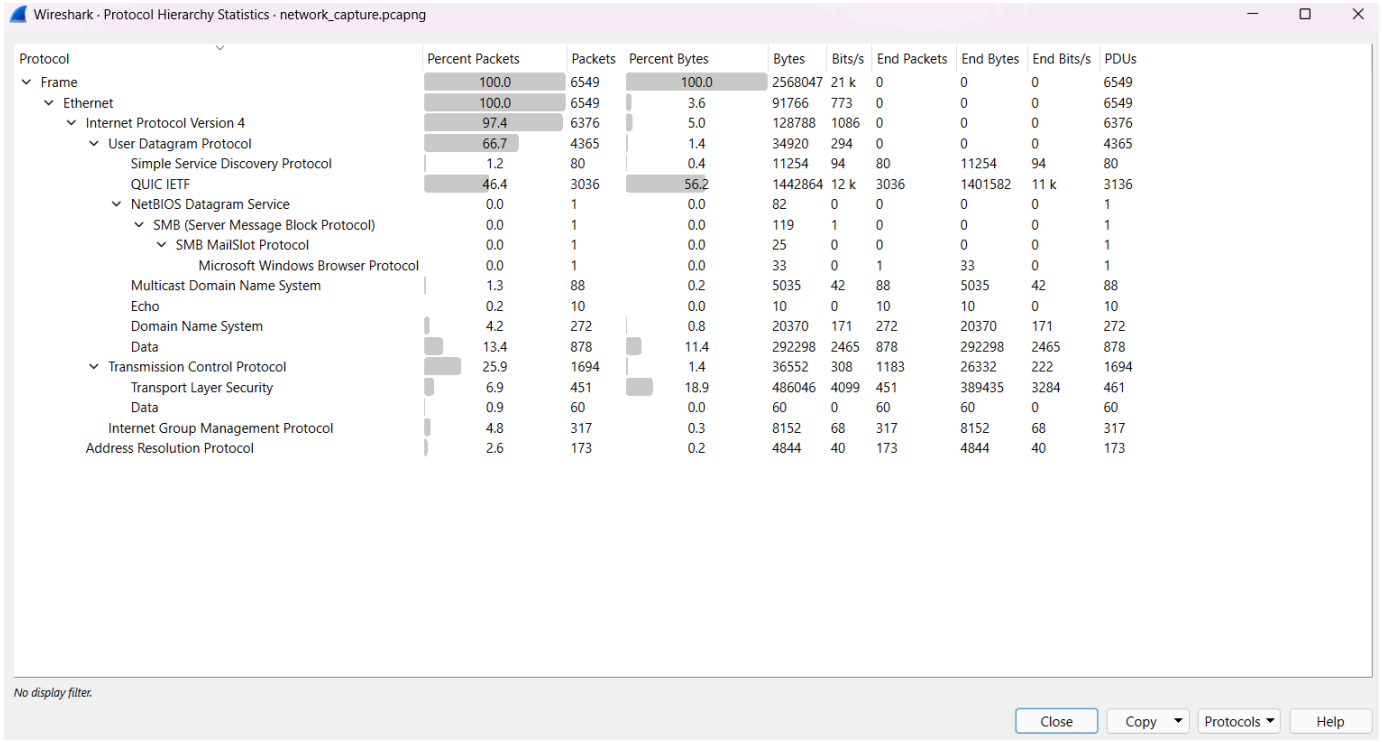
This saved file will be our **base dataset** for protocol analysis, TCP/IP inspection, troubleshooting, and filtering in the next steps.

Step 4 Protocol Identification & TCP/IP Analysis

1. With your capture file still open in Wireshark, go to **Statistics > Protocol Hierarchy**.



2. This will show you a breakdown of all protocols seen in the capture (e.g., Ethernet, IPv4, TCP, UDP, HTTP, DNS, TLS).



Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s	PDUs
Frame	100.0	6549	100.0	2568047	21 k	0	0	0	6549
Ethernet	100.0	6549	3.6	91766	773	0	0	0	6549
Internet Protocol Version 4	97.4	6376	5.0	128788	1086	0	0	0	6376
User Datagram Protocol	66.7	4365	1.4	34920	294	0	0	0	4365
Simple Service Discovery Protocol	1.2	80	0.4	11254	94	80	11254	94	80
QUIC IETF	46.4	3036	56.2	1442864	12 k	3036	1401582	11 k	3136
NetBIOS Datagram Service	0.0	1	0.0	82	0	0	0	0	1
SMB (Server Message Block Protocol)	0.0	1	0.0	119	1	0	0	0	1
SMB MailSlot Protocol	0.0	1	0.0	25	0	0	0	0	1
Microsoft Windows Browser Protocol	0.0	1	0.0	33	0	1	33	0	1
Multicast Domain Name System	1.3	88	0.2	5035	42	88	5035	42	88
Echo	0.2	10	0.0	10	0	10	10	0	10
Domain Name System	4.2	272	0.8	20370	171	272	20370	171	272
Data	13.4	878	11.4	292298	2465	878	292298	2465	878
Transmission Control Protocol	25.9	1694	1.4	36552	308	1183	26332	222	1694
Transport Layer Security	6.9	451	18.9	486046	4099	451	389435	3284	461
Data	0.9	60	0.0	60	0	60	60	0	60
Internet Group Management Protocol	4.8	317	0.3	8152	68	317	8152	68	317
Address Resolution Protocol	2.6	173	0.2	4844	40	173	4844	40	173

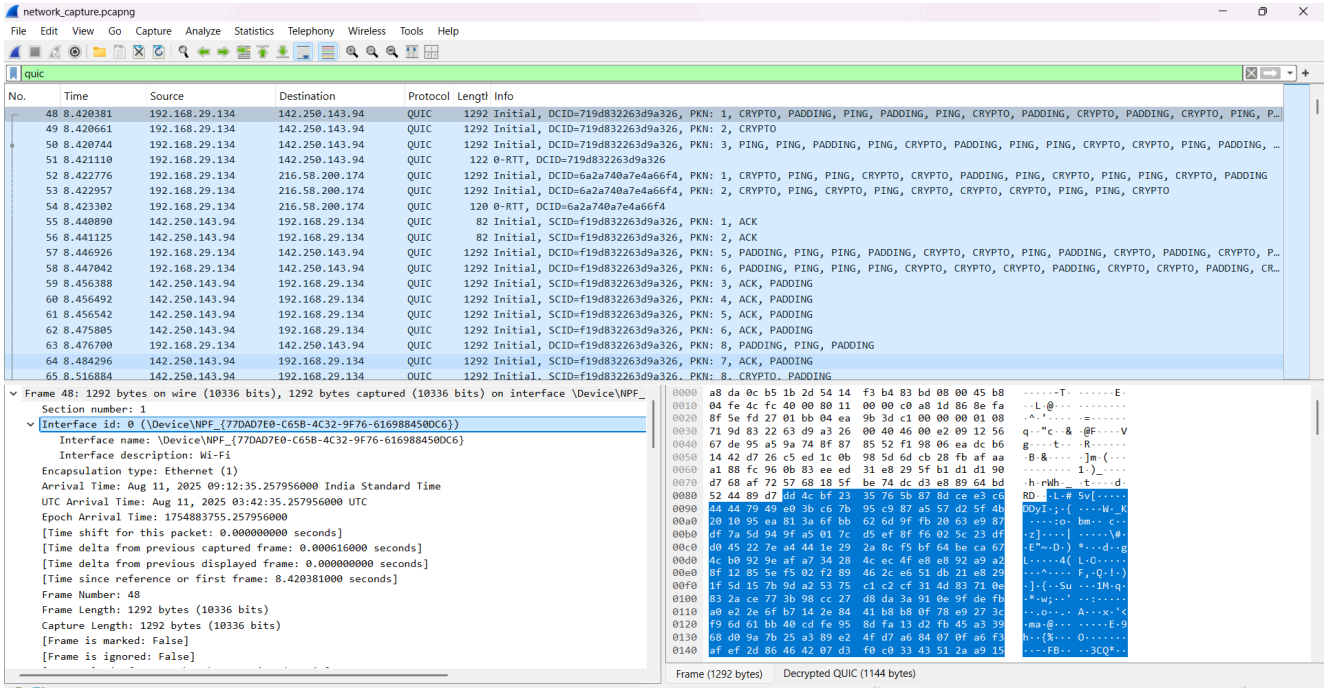
3. Close the Protocol Hierarchy window.

4. From Protocol Hierarchy the capture of 6,549 packets:

Protocol	% Packets	% Bytes	Notes
Ethernet	100%	3.57%	Link-layer framing for all traffic
IPv4	97.36%	5.01%	Dominant network-layer protocol used
UDP	66.65%	1.36%	Transport protocol used for QUIC, DNS, mDNS
QUIC IETF	46.35%	56.18%	HTTP/3 traffic over UDP (port 443)
TCP	25.87%	1.42%	Small portion of connections still use TCP
TLS	6.88%	18.93%	Encrypted HTTPS over TCP
DNS	4.15%	0.79%	Domain resolution queries/responses
IGMP / ARP	4.84% / 2.64%	0.31% / 0.18%	Multicast group management and address resolution

5. QUIC Packet Layer Analysis

- In the filter bar, type: quic
- Press Enter, now you'll see only QUIC packets.
- Click one QUIC packet to select it.



QUIC Packet Breakdown (Layer Analysis)

- **Frame Information**:
  - Arrival Time: Aug 11, 2025, 09:12:35 IST
  - Frame Length: 1292 bytes (10336 bits)
  - Capture Interface: Wi-Fi (Npcap driver)

- **Ethernet II**:
  - Source MAC: Intel_b4:83:bd (54:14:f3:b4:83:bd)
  - Destination MAC: ServercomPri_b5:1b:2d (a8:da:0c:b5:1b:2d)
  - Type: IPv4 (0x0800)

- **Internet Protocol Version 4 (IPv4)**:
  - Source IP: 192.168.29.134 (local device)
  - Destination IP: 142.250.143.94 (Google server)
  - TTL: 128
  - Total Length: 1278 bytes

- **User Datagram Protocol (UDP)**:
  - Source Port: 64807
  - Destination Port: 443 (HTTPS over QUIC)
  - Length: 1258 bytes

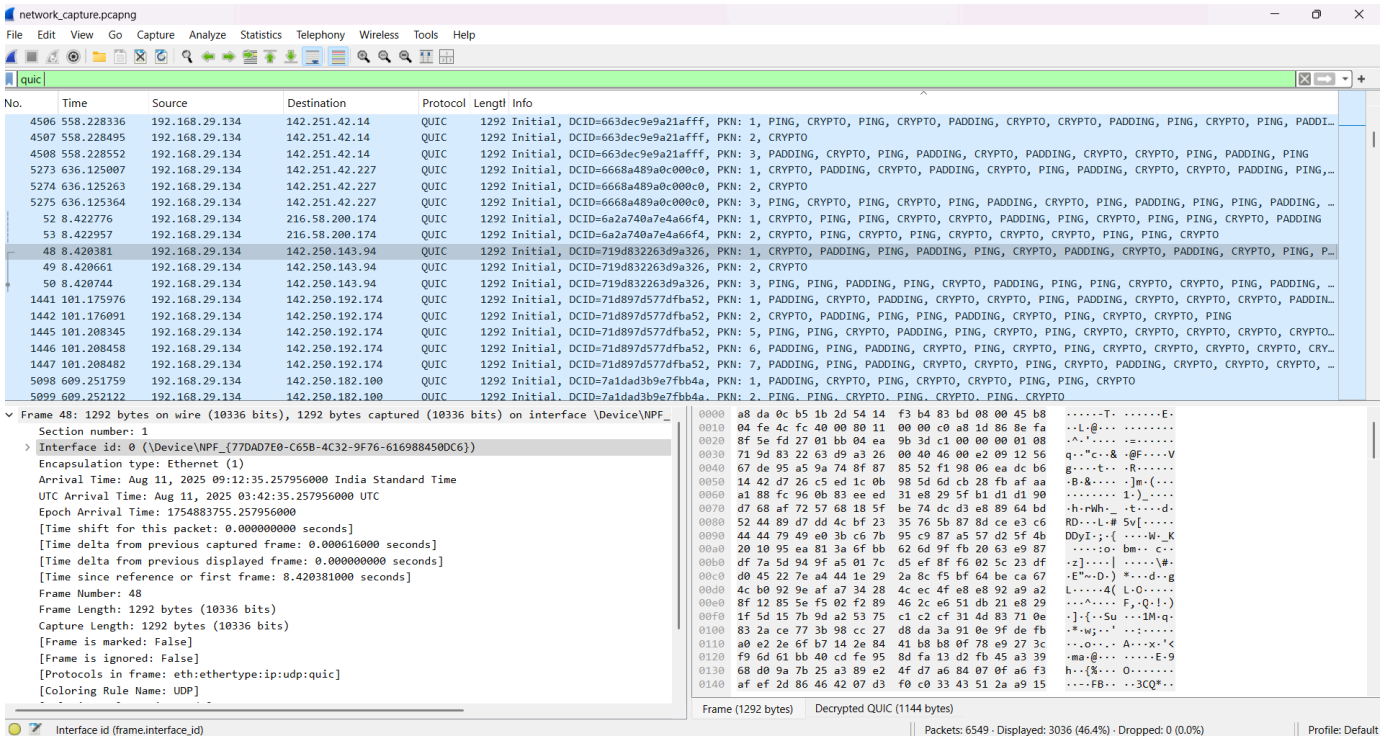
- **QUIC IETF**:
  - Packet Type: **Initial** (used for connection setup)
  - Version: 1 (0x00000001) — HTTP/3 standard version
  - Destination Connection ID Length: 8
  - Destination Connection ID: 719d832263d9a326
  - Contains: CRYPTO, PADDING, possibly handshake messages

Step 5 Filtering & Network Troubleshooting

We'll use filters in Wireshark to isolate and investigate different traffic types from your capture.

1. QUIC Traffic Analysis

   Filter Used: quic

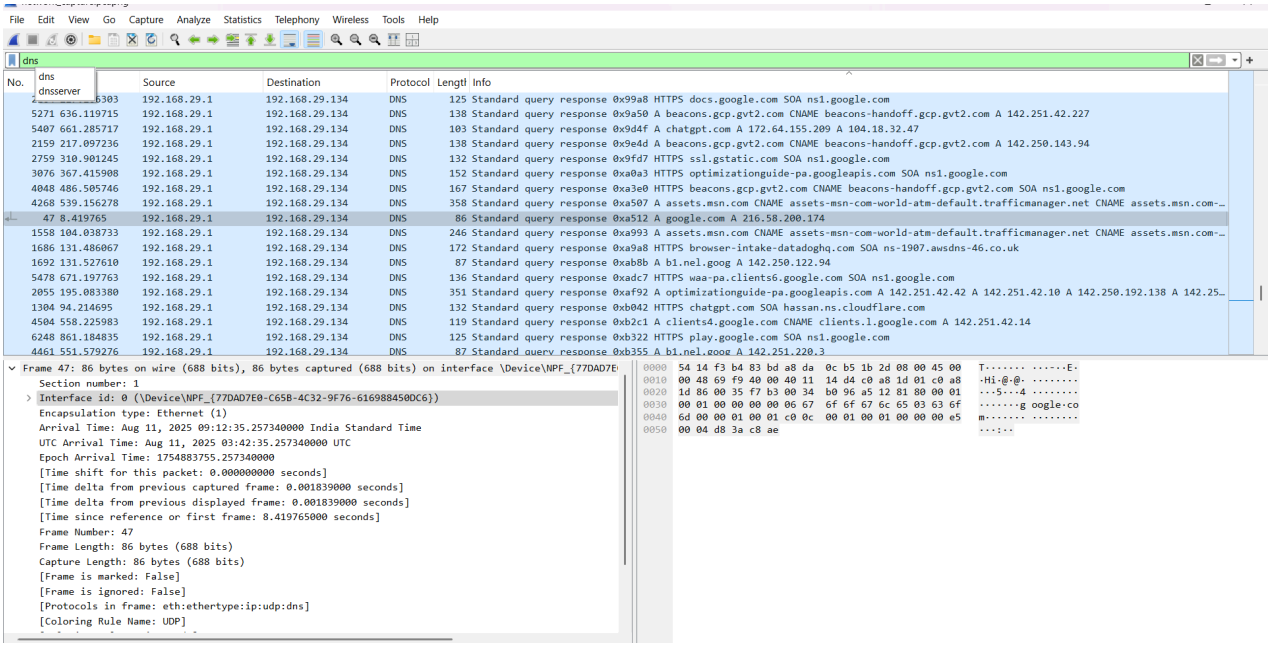


Observations:

- Total QUIC packets: 3036
- Primary Destination IPs: 49.44.118.8
- Destination Ports: Mostly 443 (HTTPS over QUIC).
- Example Server: 142.250.143.94 (Google)
- Packet Types Seen: Initial, 0-RTT, Handshake, ACK.

2. DNS Traffic Analysis

   Filter Used: dns

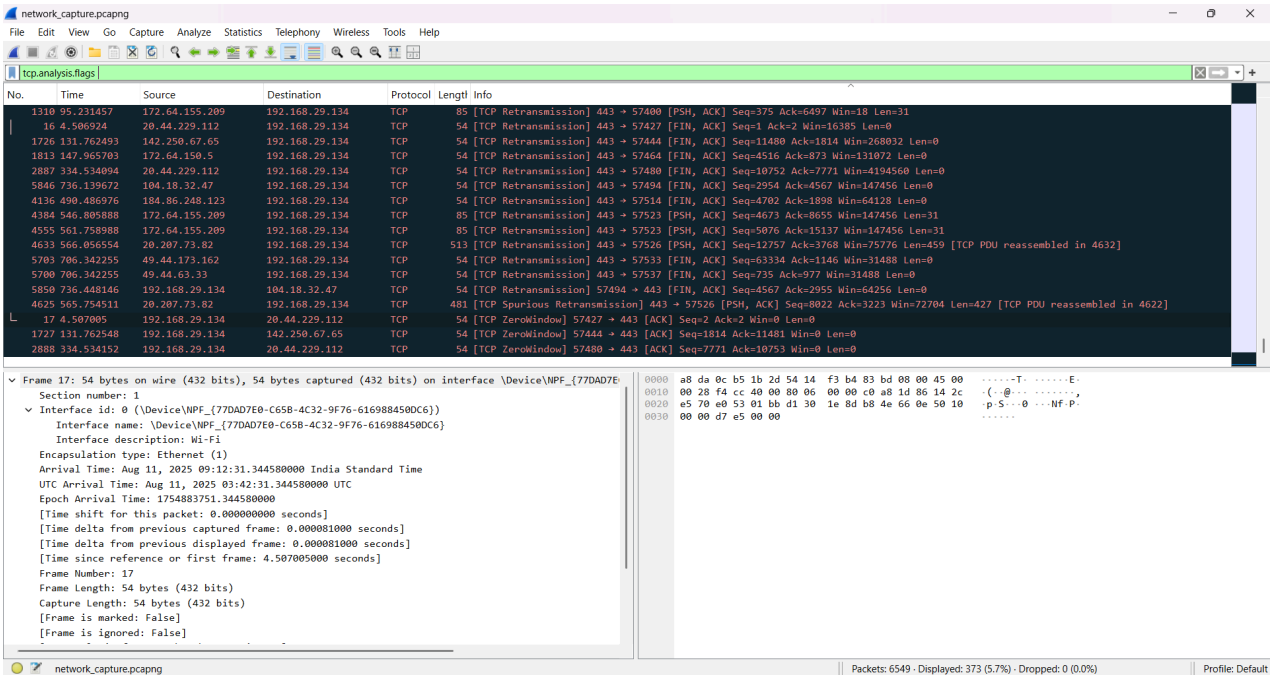


Observations:

- Total DNS packets: 272
- Common Queried Domains:
  - google.com
- DNS Response Codes: Mostly No error, no failed lookups found.

3. TCP Error Analysis

   Filter Used: tcp.analysis.flags

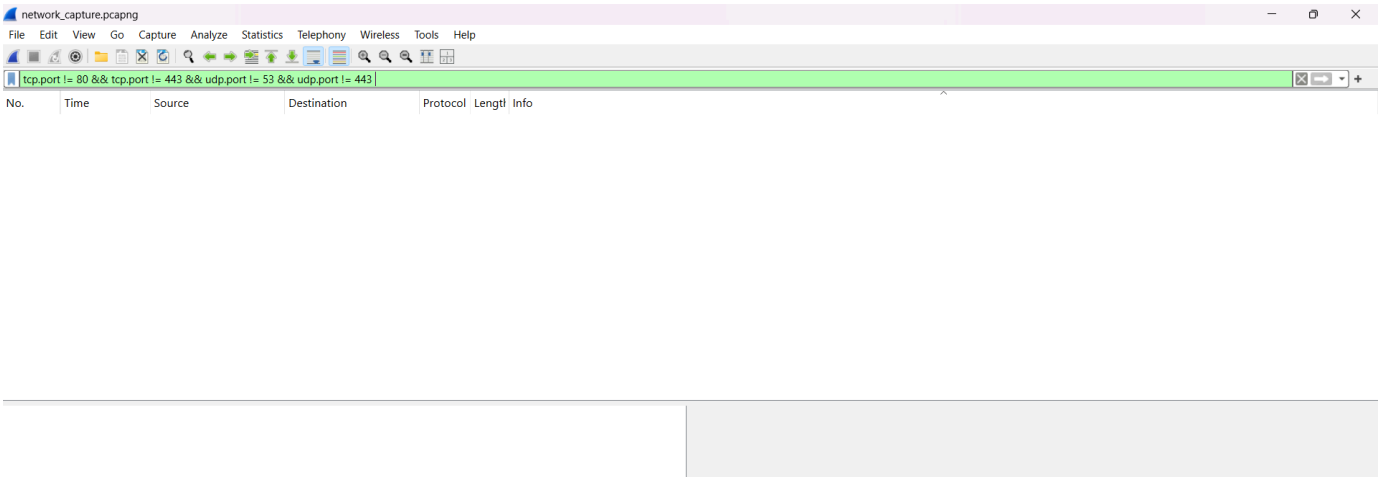


Observations:

- Total TCP anomaly packets: 373
- Error Types Found: Duplicate ACKs, Out-of-order packets.
- Possible Causes: Network congestion, Wi-Fi interference.

4. Non-Standard Port Analysis

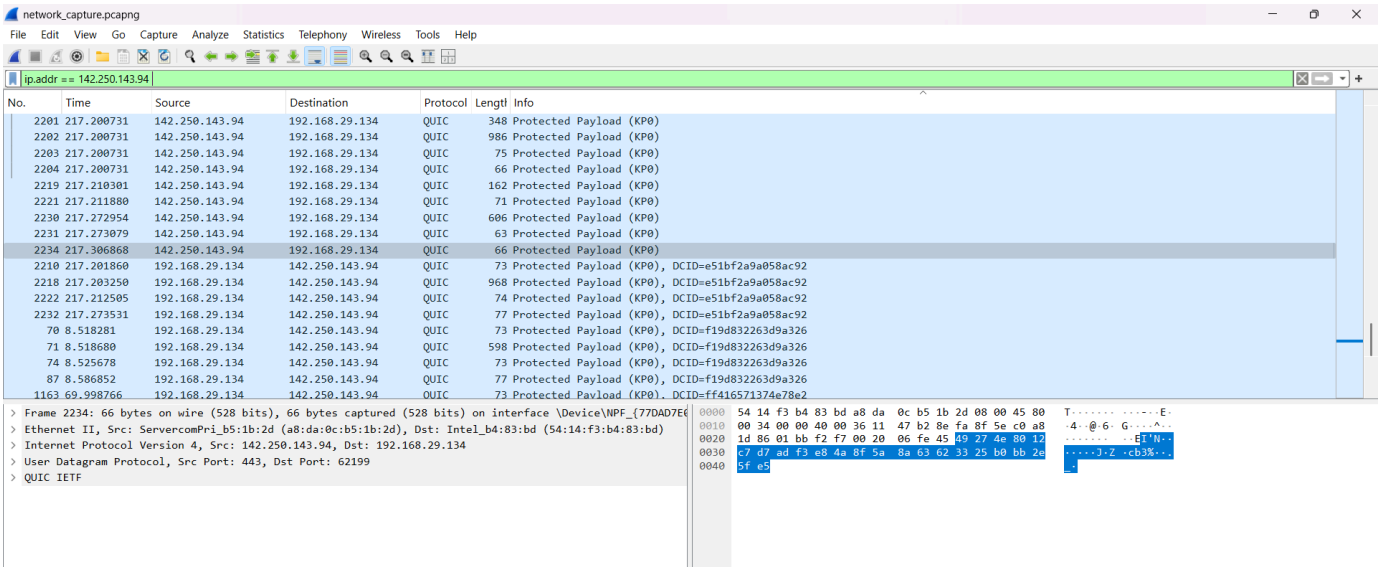
   Filter Used: tcp.port != 80 && tcp.port != 443 && udp.port != 53 && udp.port != 443



   “No traffic detected on non-standard ports. All communications occurred over standard web (443) and DNS (53) ports.”

5. IP-Specific Traffic Analysis

Filter Used: ip.addr == 142.250.143.94



Observations:

- All traffic to/from Google server over QUIC.
- Packet Types: Initial, ACK, CRYPTO frames.
- No retransmissions detected.

Step 6 Outcome & Final Report

The network traffic capture and analysis using Wireshark successfully met the objectives of the internship task.

Key Achievements:

- Installed and configured Wireshark with Npcap for packet capture on Windows.
- Captured **6,549 packets** over a live 1–2 minute session, covering multiple protocols.
- Identified the top protocols by packet volume, including **QUIC (46.35%)**, **UDP (66.65%)**, and **TCP (25.87%)**.
- Performed **layer-by-layer analysis** of QUIC packets to understand connection setup and handshake details.
- Applied targeted filters (quic, dns, tcp.analysis.flags) to isolate protocol-specific traffic and detect anomalies.
- Found minor TCP retransmission and duplicate ACKs, possibly due to network congestion or Wi-Fi interference.
- Confirmed that all communications occurred over standard secure ports (**443 for HTTPS** and **53 for DNS**), with no suspicious non-standard traffic detected.

Conclusion:
The captured data confirms that the monitored system primarily uses secure, encrypted channels (QUIC/HTTPS) with normal DNS resolution patterns. No signs of malicious activity or unusual port usage were found during this analysis. The filtering process proved effective in narrowing down protocol-specific data, highlighting Wireshark's value for both troubleshooting and security monitoring.