Report Submission

To

Task 5: Wireshark Network Traffic Analysis

Name	Yash Javiya
Submission to	Elevated Labs

Task 5th: Wireshark Network Traffic Analysis

Wireshark Network Traffic Analysis

Objective:

The goal of this task is to familiarize yourself with real-time network traffic analysis. By capturing live packets using Wireshark, you will:

- Understand how data travels across the network.
- Identify different types of network protocols.
- Gain practical skills in filtering, decoding, and interpreting packet data.
- Learn the importance of protocol structures in cybersecurity and troubleshooting.

Tools and Environment:

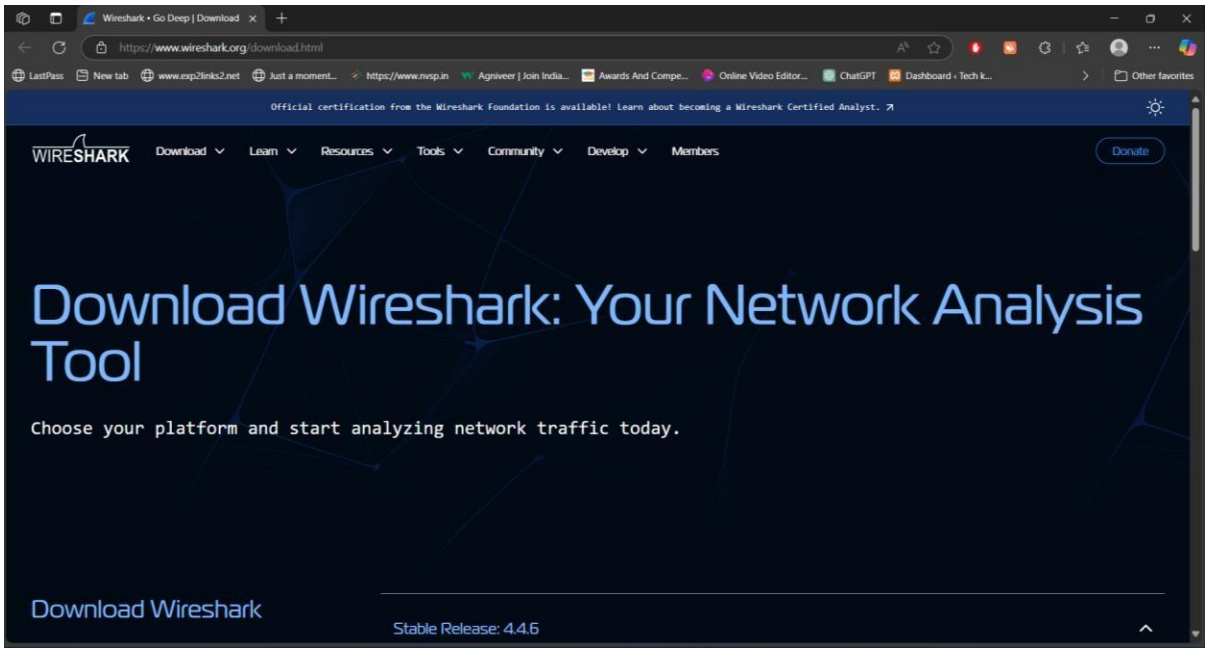
Component	Description
Wireshark	Open-source network packet analyzer
Operating System	Windows 10 (64-bit)
Traffic Generator	Web browsers (Google Chrome), ping command via CMD
Network	Wi-Fi connection (private network)

Task 5th: Wireshark Network Traffic Analysis

Step-by-Step Task Execution

Step 1: Install Wireshark

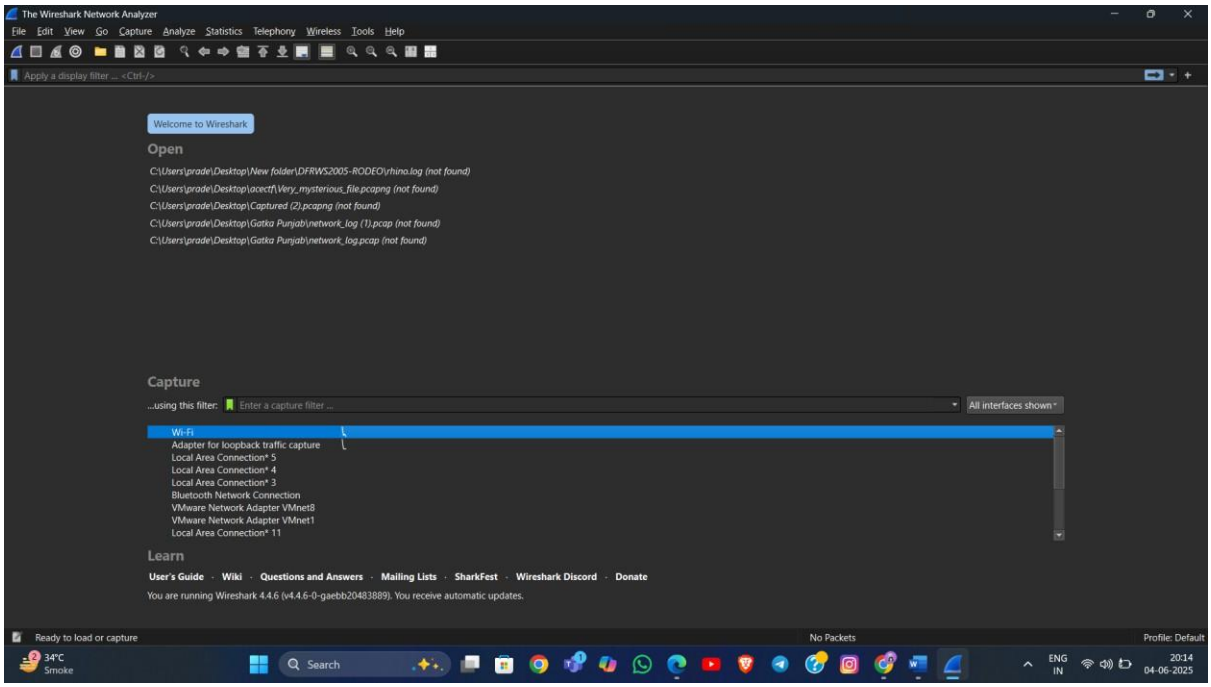
- Downloaded from: https://www.wireshark.org/download.html



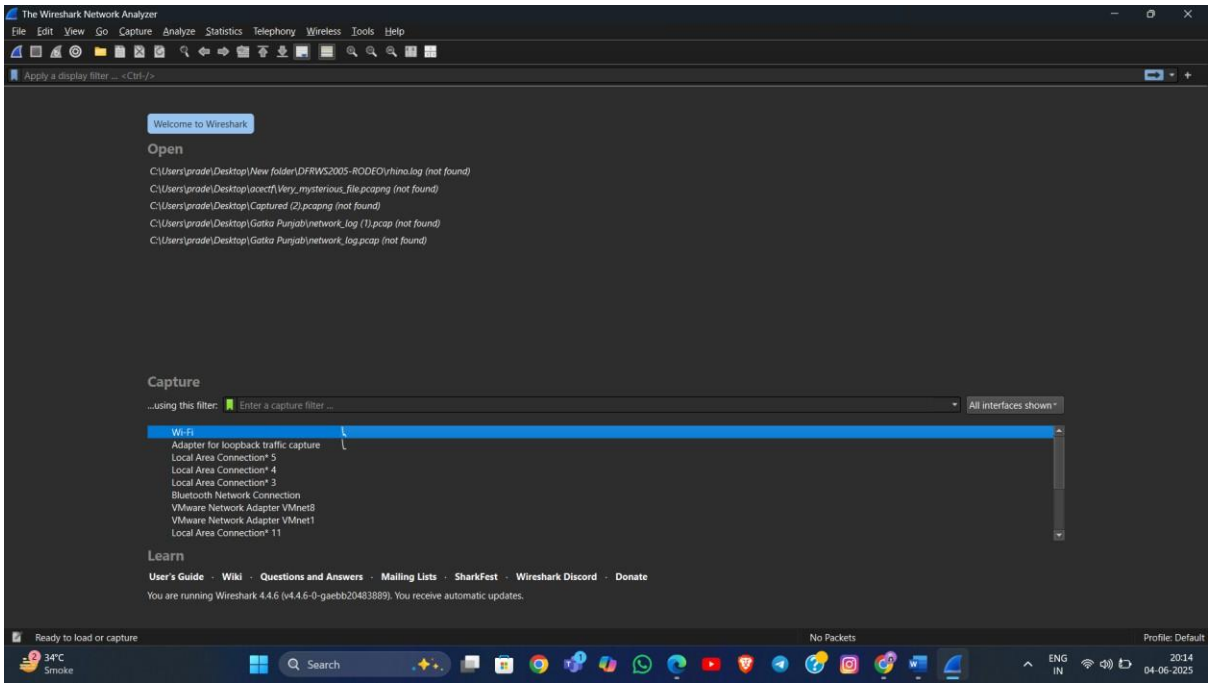
- Chose **Npcap** (WinPcap alternative) during installation – allows packet capture on Windows.
- Launched Wireshark after successful installation.

Step 2: Start Packet Capture

- Opened Wireshark GUI.
- Selected the **Wi-Fi interface** (typically the active one for most laptops).



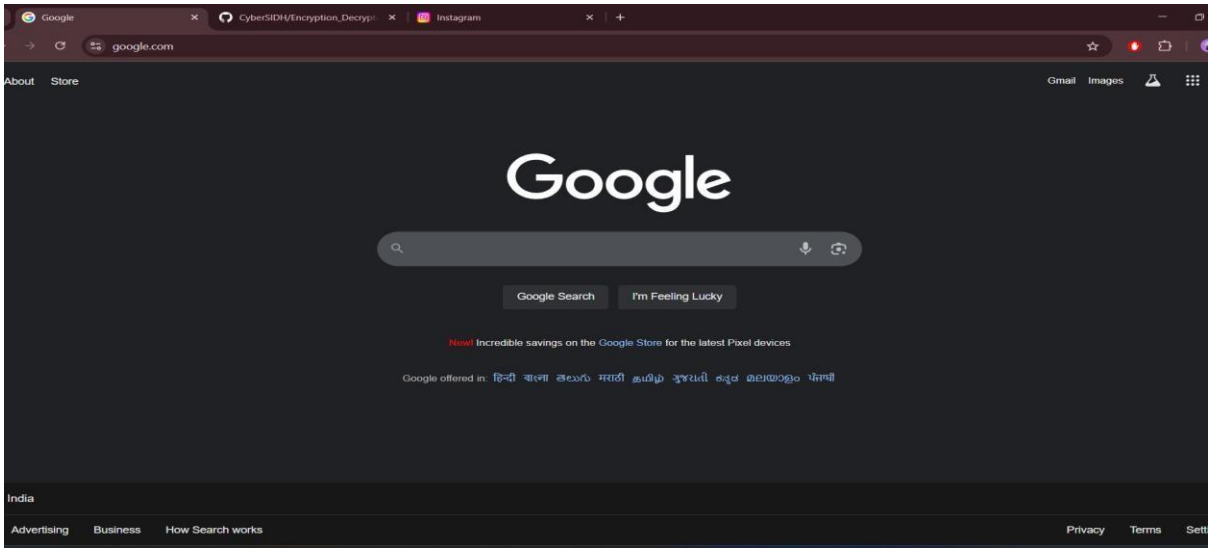
- Pressed the blue **shark fin icon** to start live capture.
- Observed real-time packet inflow in the packet list pane.



Step 3: Generate Network Traffic

To ensure meaningful traffic during capture:

- Opened various websites:
  - https://www.google.com
  - https://www.github.com
  - https://www.example.com



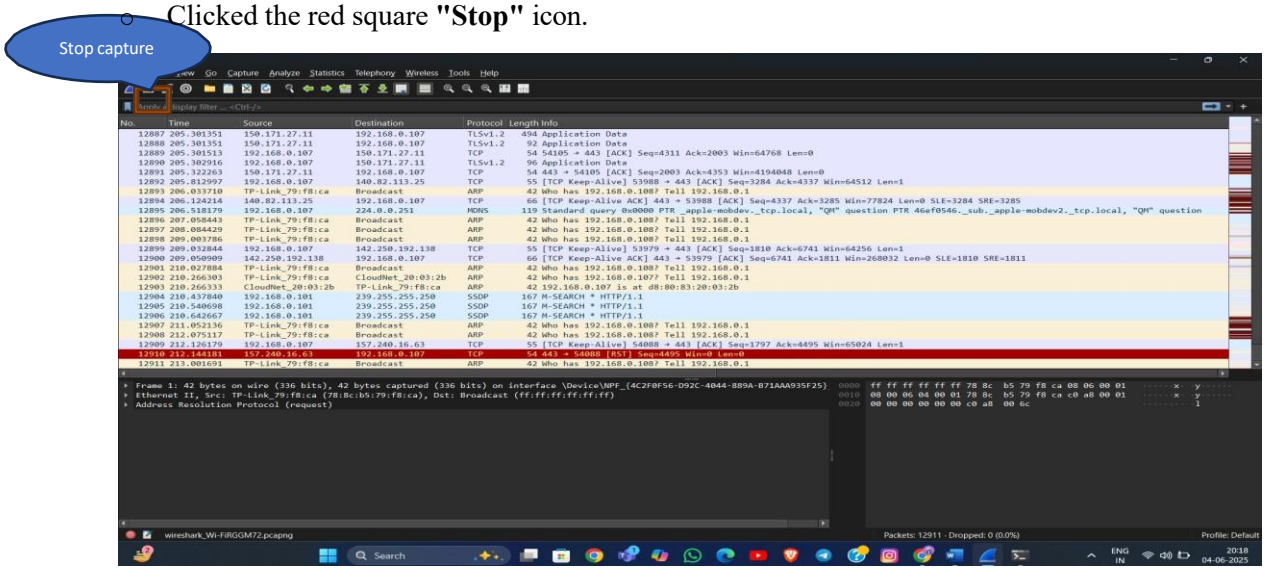
- Open terminal and Run command:
- ping 8.8.8.8 -n 5

```
C:\Users\prade>ping 8.8.8.8 -n 5

Pinging 8.8.8.8 with 32 bytes of data:
Reply from 8.8.8.8: bytes=32 time=22ms TTL=118
Reply from 8.8.8.8: bytes=32 time=17ms TTL=118
Reply from 8.8.8.8: bytes=32 time=17ms TTL=118
Reply from 8.8.8.8: bytes=32 time=18ms TTL=118
Reply from 8.8.8.8: bytes=32 time=21ms TTL=118

Ping statistics for 8.8.8.8:
    Packets: Sent = 5, Received = 5, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 17ms, Maximum = 22ms, Average = 19ms
```

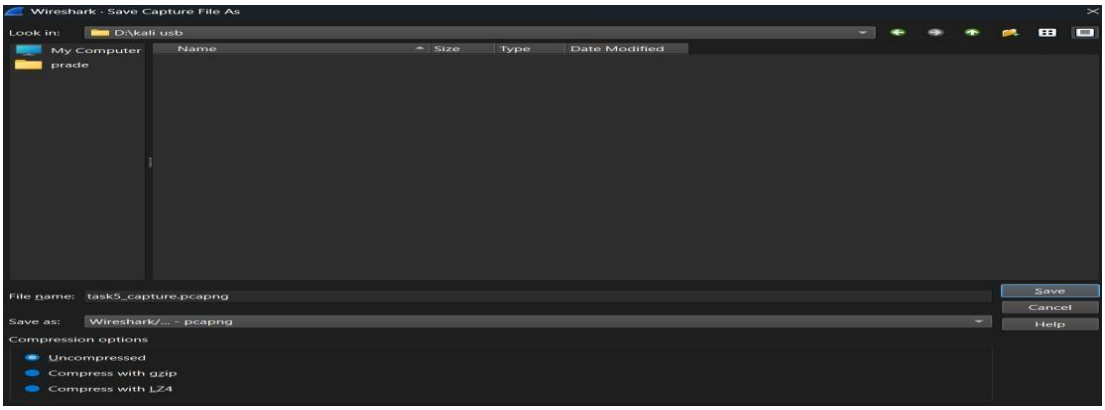
This sends ICMP Echo Requests to Google DNS and triggers ICMP packet activity.

Step 4: Stop Capture

- After ~1 minute of active browsing and pinging:
  - Clicked the red square **"Stop"** icon.



  - Saved the capture file as: task5_capture.pcapng via **File > Save As**.



Task 5th: Wireshark Network Traffic Analysis

Step 5: Filter and Analyze Network Traffic

Purpose of Filtering

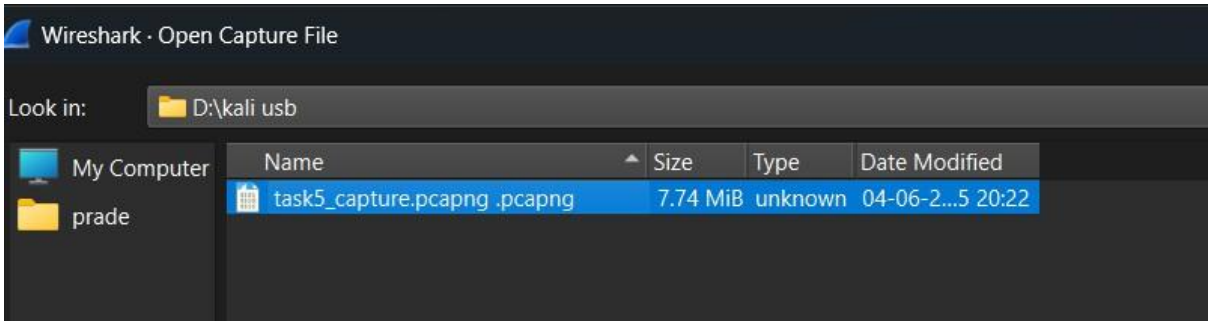
Wireshark captures **thousands of packets**, making filtering essential to narrow down data relevant to analysis.

How to Use Filters in Wireshark

In the **Display Filter** bar (top of the screen), type a filter command and press **Enter**.

Now, Open the task5_capture.pcapng in Wireshark

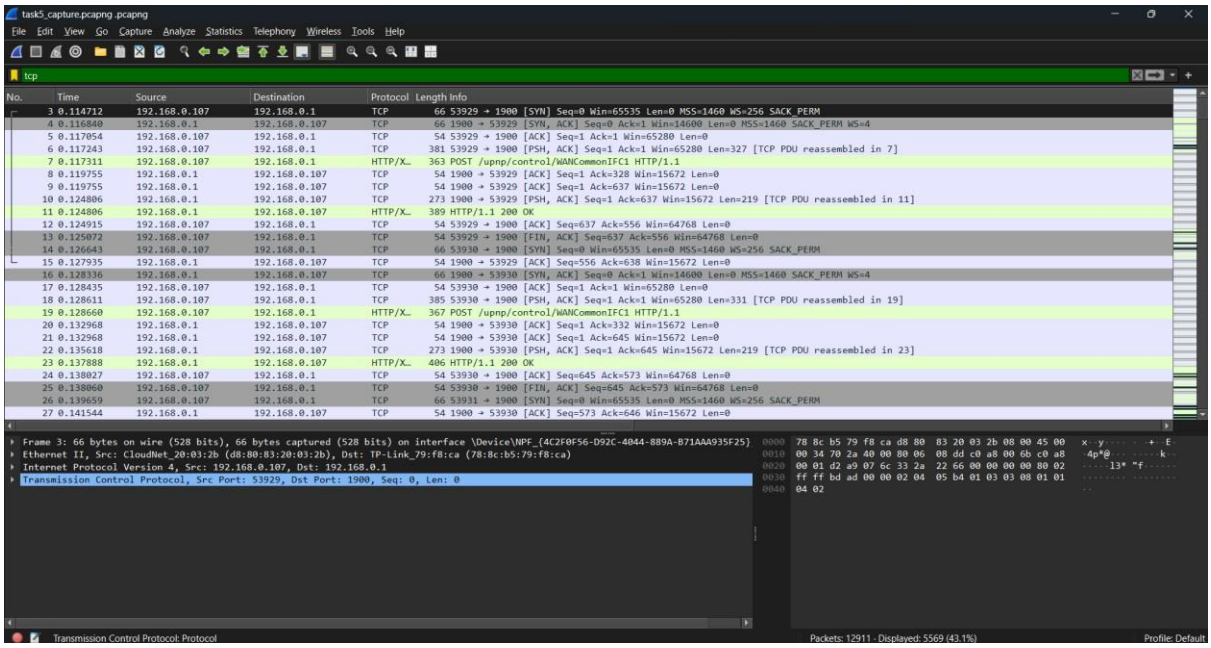
1. **Launch Wireshark.**
2. **Go to File > Open.**
3. **Browse to the file path (My file name is task5_capture.pcapng) and open it.**



Step 5.1: Filter Captured Packets by Protocol

Use these filters one at a time in the display filter bar (top of Wireshark window):

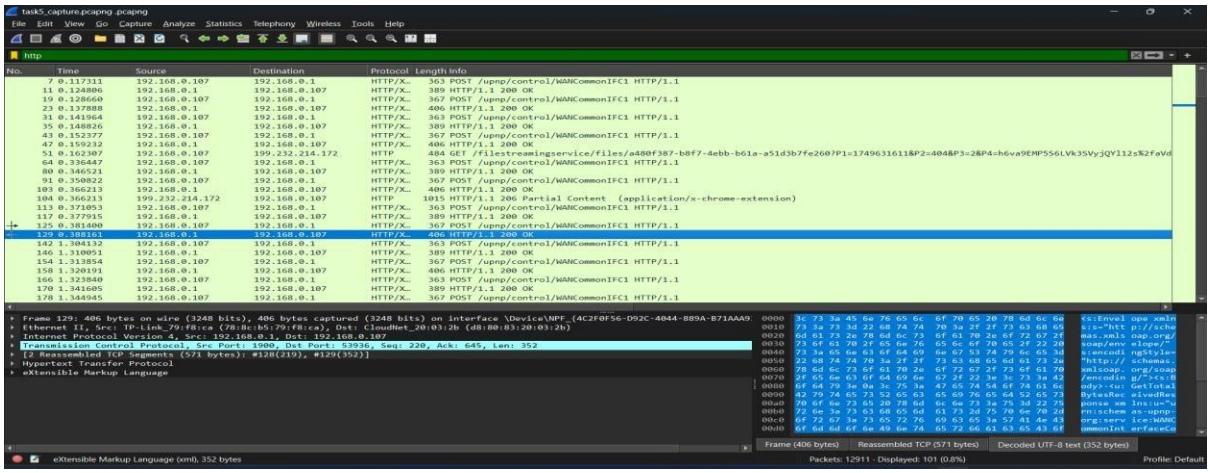
- **TCP Traffic:**
- **Tcp**



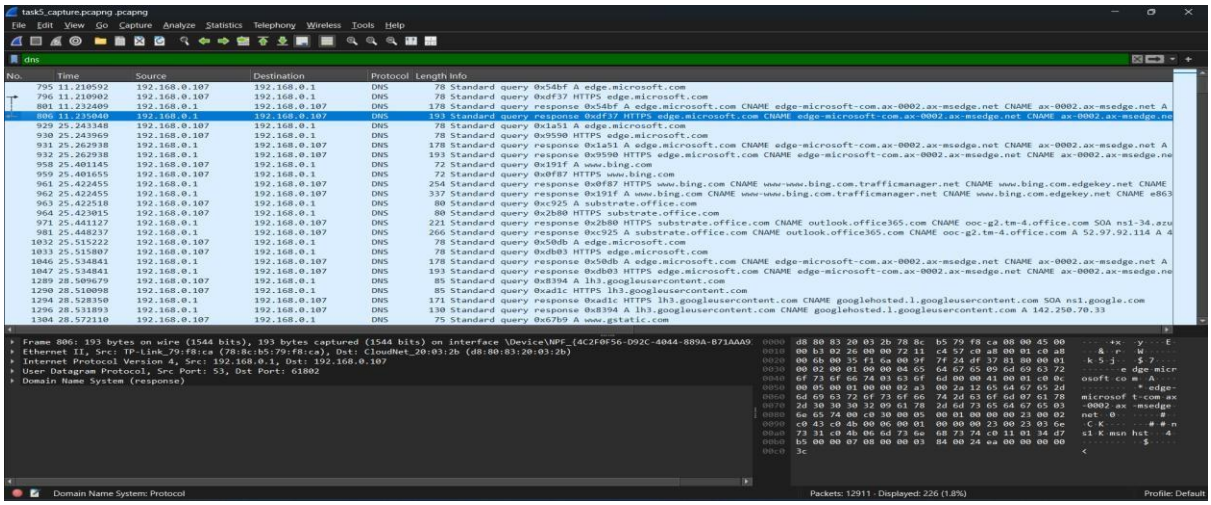
- **HTTP Traffic:**

Task 5th: Wireshark Network Traffic Analysis

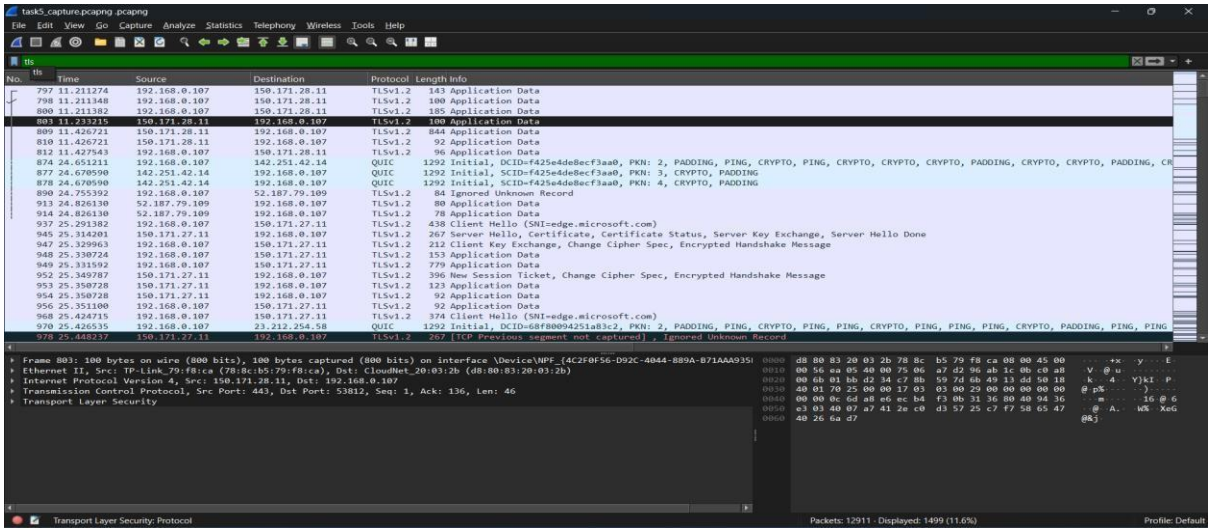
- **http**



- **DNS Traffic:**
- **Dns**



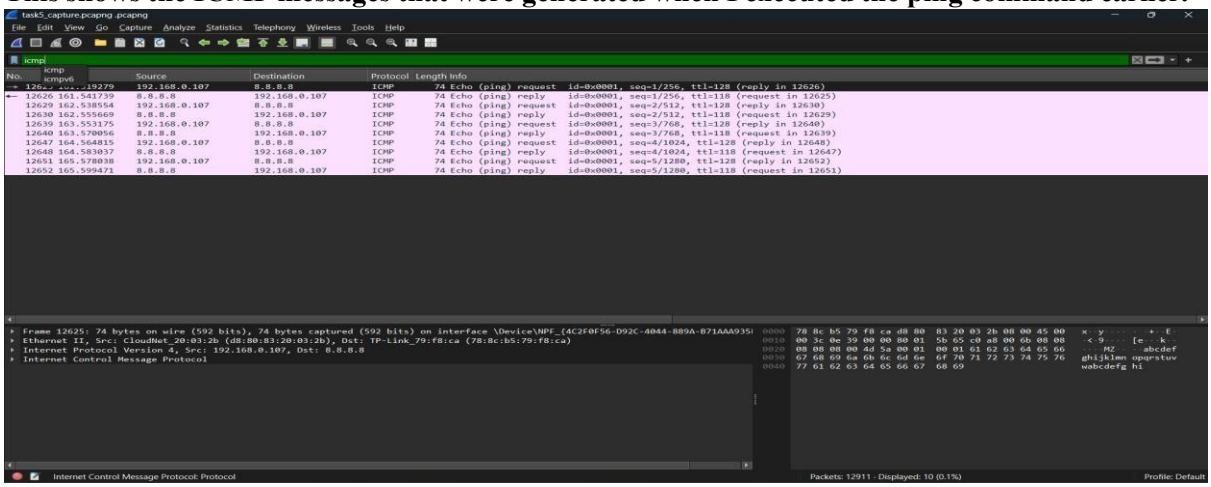
- **Another ports :**
- **tls**



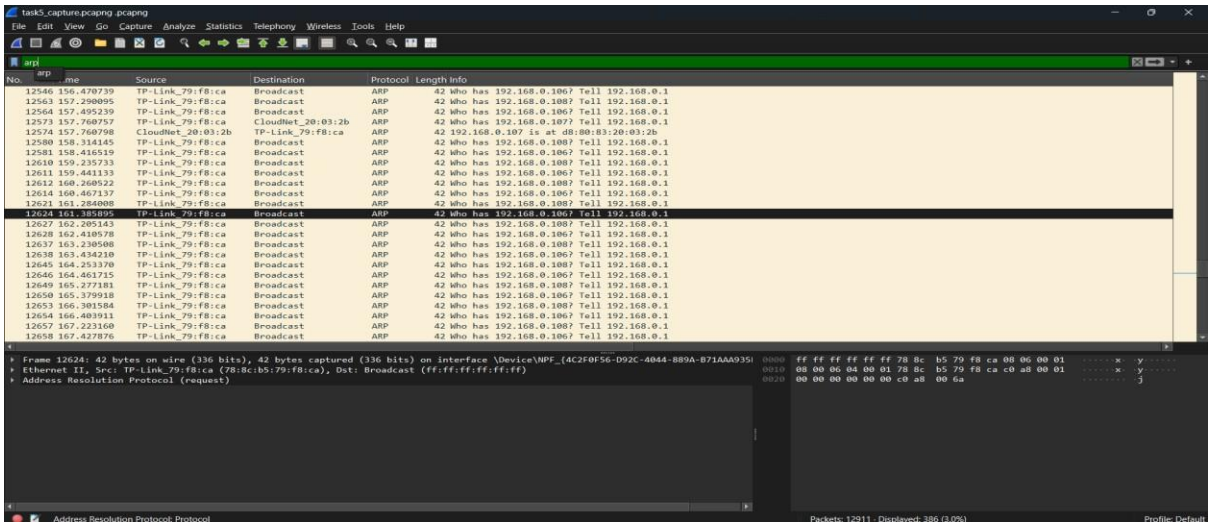
Task 5th: Wireshark Network Traffic Analysis

- **Icmp**

  This shows the ICMP messages that were generated when I executed the ping command earlier.

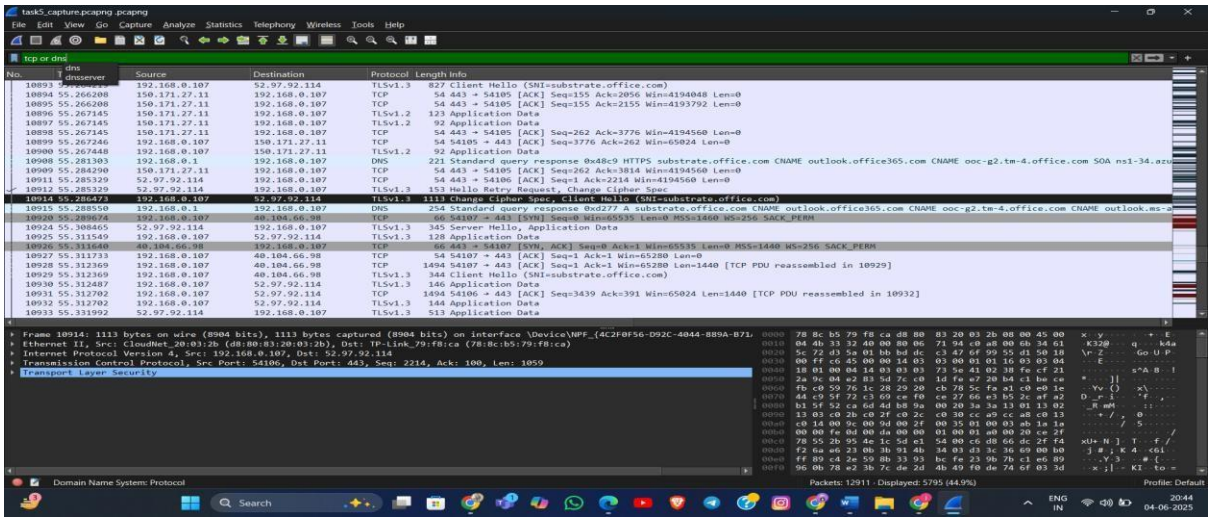


- **Arp**



Tip: You can apply a combined filter like:

tcp or dns



Task 5th: Wireshark Network Traffic Analysis

Use to see the multiple filtered port in one view.

Useful Display Filters and Their Purpose

Filter	Description
http	Shows only HTTP packets (used in web browsing over port 80)
tcp	Displays all TCP traffic (reliable connections)
udp	Displays all UDP traffic (used in DNS, video streaming)
icmp	Displays ping requests and replies
dns	Shows DNS queries and responses
ip.addr == 8.8.8.8	Filters all traffic to/from Google DNS
tcp.port == 443	Filters HTTPS (encrypted) packets
frame contains "example"	Finds packets that include specific strings

What We Observed Using Filters

- http: Plain GET requests and responses (before redirection to HTTPS)
- dns: Hostname-to-IP resolution process
- icmp: Echo request and reply for ping to 8.8.8.8
- tcp: SYN, ACK, FIN handshake and session details

Protocols Identified and Analysis

Protocol	Layer	Role in Networking	Detailed Observation
HTTP	Application	Web browsing (port 80)	Saw plain-text GET requests before HTTPS redirection
HTTPS/TLS	Application	Secure web browsing	TLS handshake observed; payload encrypted
DNS	Application	Resolving hostnames	DNS queries for google.com, example.com
ICMP	Network	Diagnostic (ping)	Echo request/reply packets with TTL and response times
TCP	Transport	Reliable communication	Observed TCP three-way handshake (SYN → SYN/ACK → ACK)
UDP	Transport	Lightweight, connectionless	Used in DNS queries/responses
ARP	Link	Address Resolution	Maps IP to MAC for devices on local network

Outcome and Skills Gained

- Mastered the use of Wireshark for live packet capture and analysis.

Task 5th: Wireshark Network Traffic Analysis

- Learned to identify multiple protocol layers and packet types.
- Applied practical filters to streamline analysis.
- Gained foundational skills in network troubleshooting using packet-level data.