

Task 5th: Wireshark Network Traffic Analysis

5th Task SUBMISSION REPORT OF ELEVATE LABS CYBERSECURITY INTERNSHIP

NAME	SANDEEP MEWARA
Submitted to:	Elevate Labs
Name of the Academic Institute	Ganpat University

REPORT SUBMITTED TO



As part of the Cyber Security Internship, I have completed "Task 5th: Wireshark Network Traffic Analysis " by following all steps as instructed. Below is a detailed summary of each step followed during the task:

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:

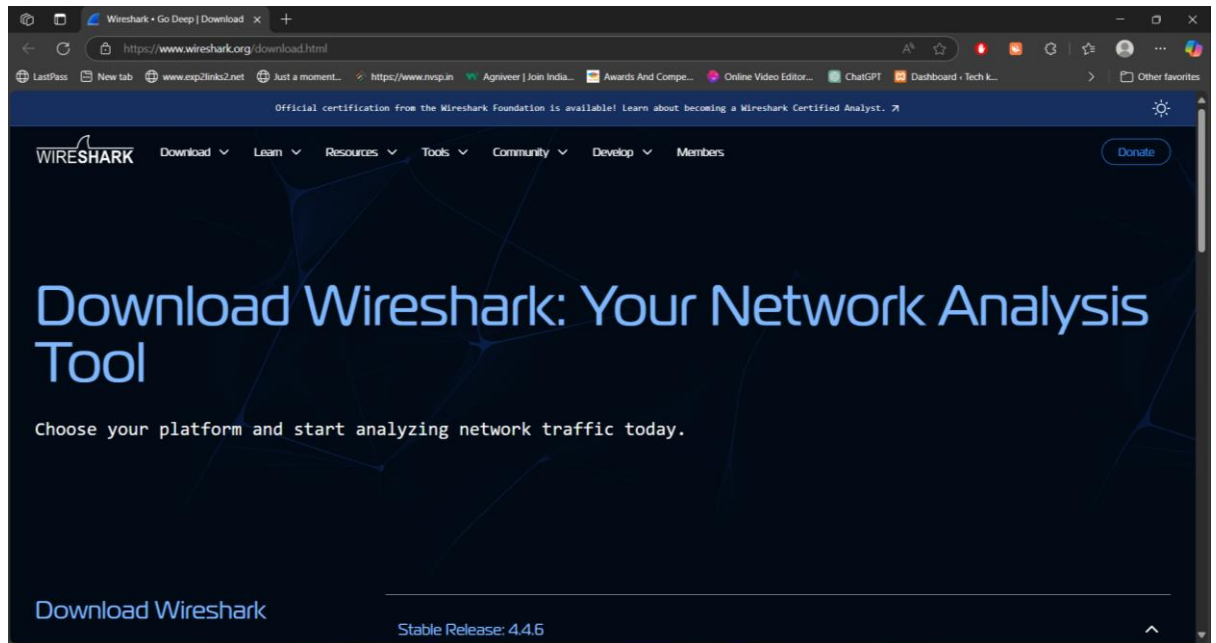
Component	Description
Wireshark	Open-source network packet analyzer
Operating System	Windows 10 (64-bit)
Traffic Generator	Web browsers (Google Chrome), ping command via CMD
Network	Wi-Fi connection (private network)

Task 5th: Wireshark Network Traffic Analysis

Step-by-Step Task Execution

Step 1: Install Wireshark

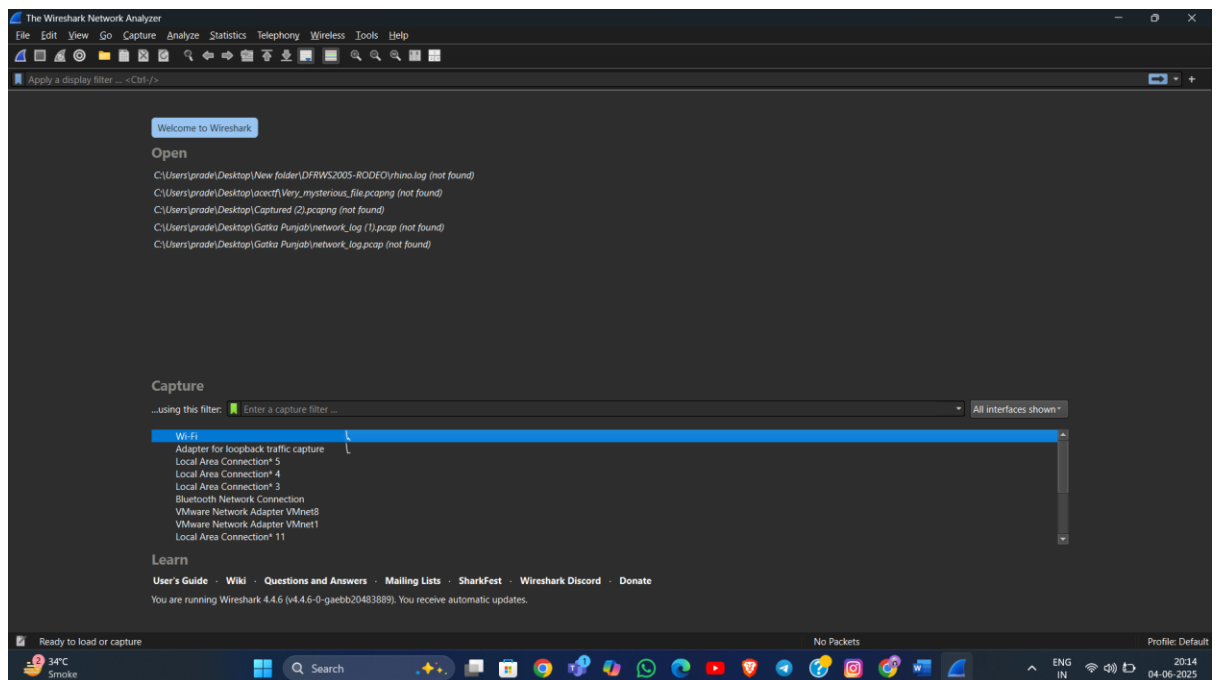
- Downloaded from: <https://www.wireshark.org/download.html>



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

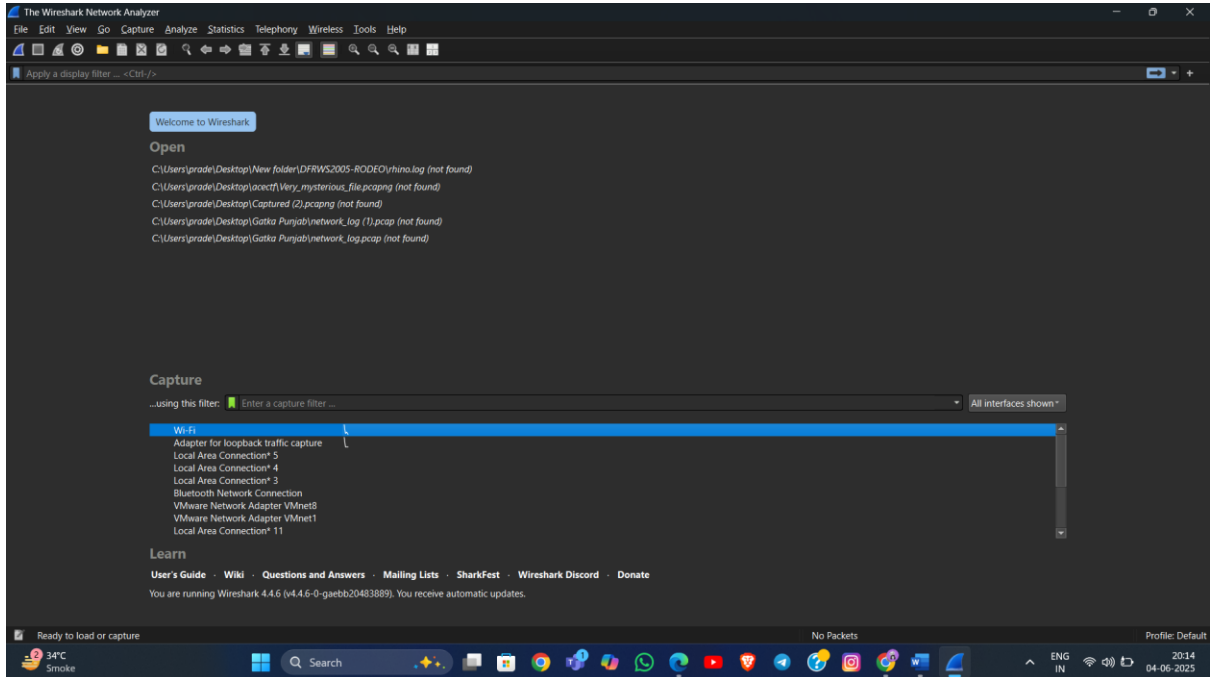
Step 2: Start Packet Capture

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



Task 5th: Wireshark Network Traffic Analysis

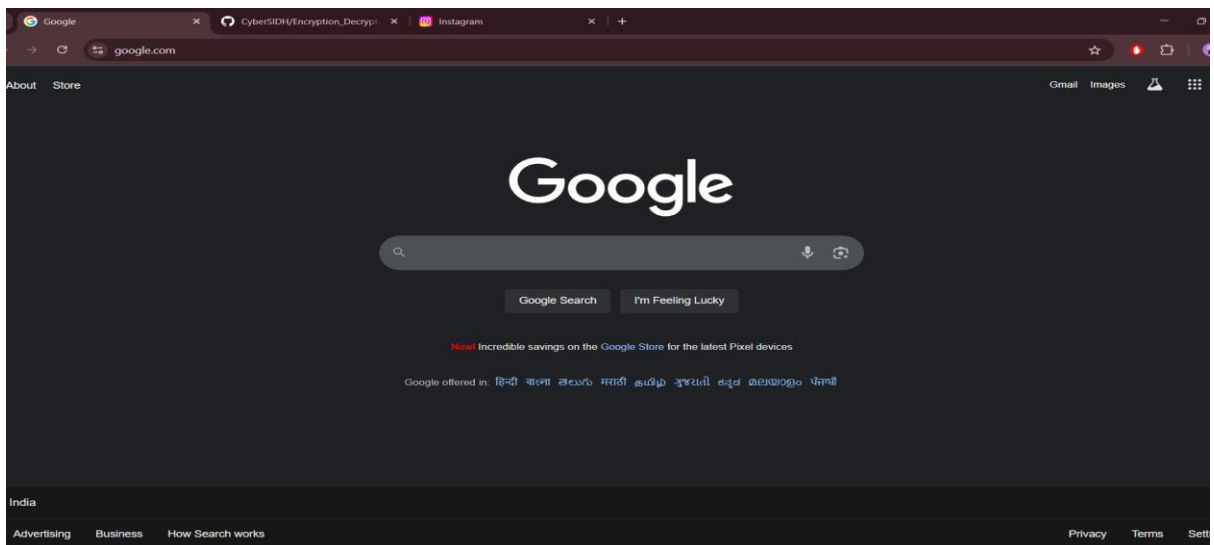
- Pressed the blue **shark fin icon** to start live capture.
- Observed real-time packet inflow in the packet list pane.



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`

Task 5th: Wireshark Network Traffic Analysis

```
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

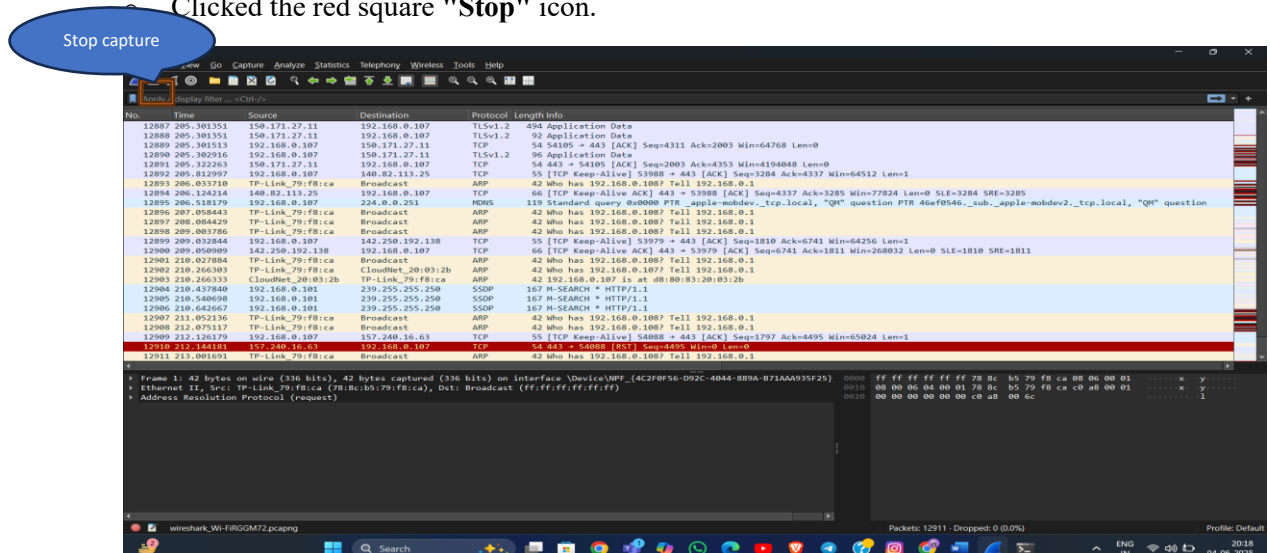
Ping statistics for 8.8.8.8:
    Packets: Sent = 5, Received = 5, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 17ms, Maximum = 22ms, Average = 19ms
```

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

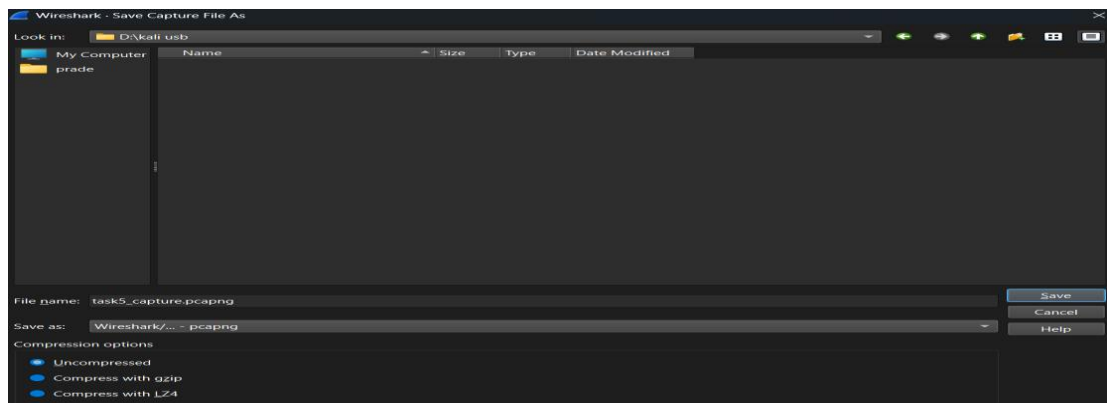
Step 4: Stop Capture

- 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

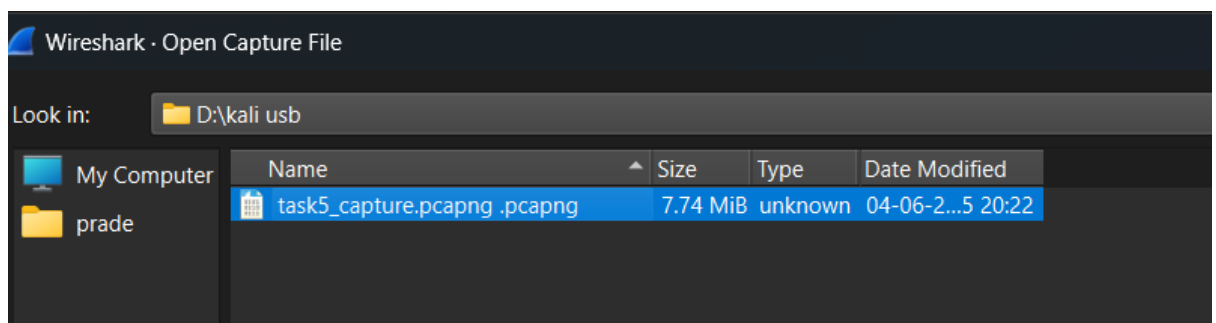
Wireshark captures **thousands of packets**, making filtering essential to narrow down data relevant to analysis.

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

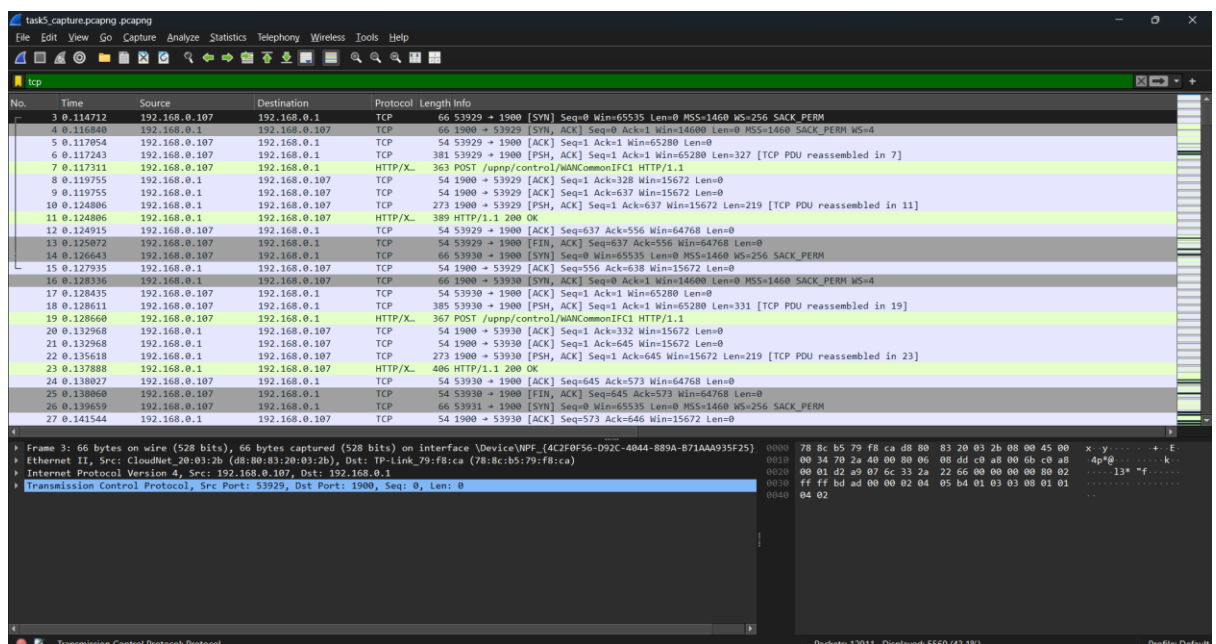
1. Launch Wireshark.
2. Go to File > Open.
3. Browse to the file path (My file name is task5_capture.pcapng) and open it.



Step 5.1: Filter Captured Packets by Protocol

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

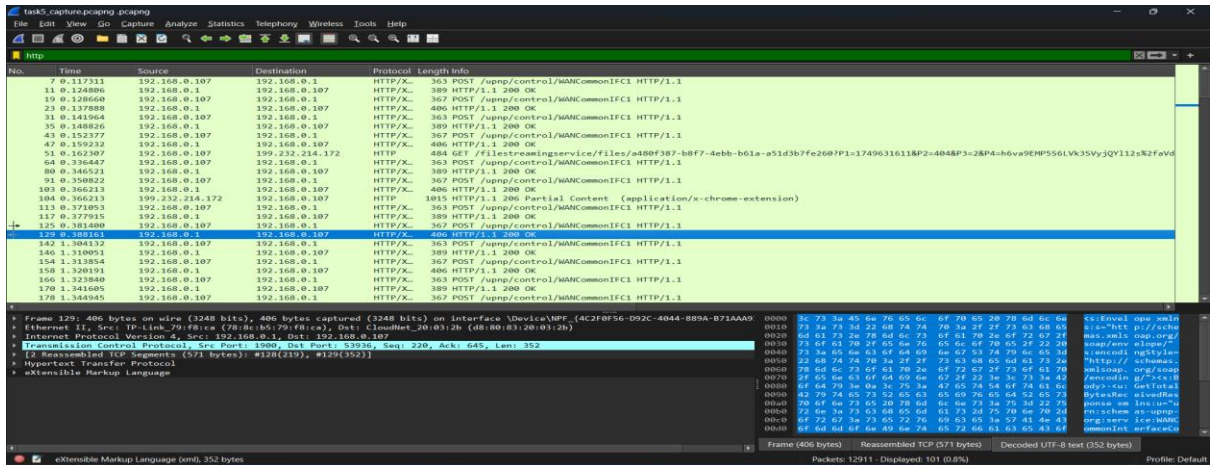
- TCP Traffic:
- Tcp



- HTTP Traffic:

Task 5th: Wireshark Network Traffic Analysis

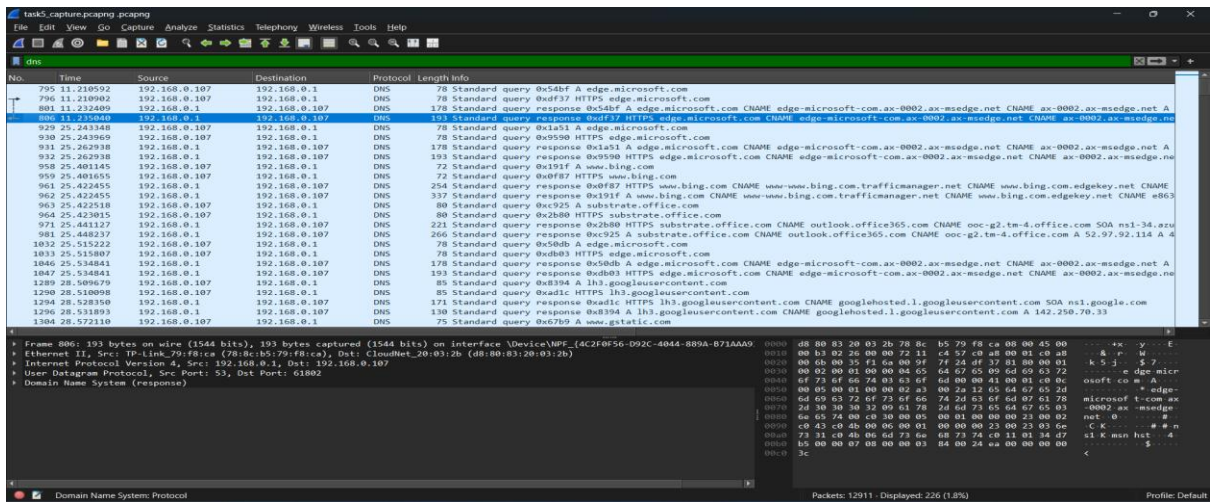
- http



The image shows a Wireshark capture of HTTP traffic. The packet list on the left shows several HTTP requests and responses. The selected packet (No. 129) is an HTTP POST request to /upnp/control/WANCommonIFC1. The packet details pane on the right shows the structure of the HTTP message, including the status bar, internet protocol version, Ethernet II, and transmission control protocol. The packet bytes pane on the right shows the raw data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
9	0.117311	192.168.0.107	192.168.0.1	HTTP/X.	363	POST /upnp/control/WANCommonIFC1 HTTP/1.1
11	0.124886	192.168.0.1	192.168.0.107	HTTP/X.	389	HTTP/1.1 200 OK
19	0.128660	192.168.0.107	192.168.0.1	HTTP/X.	363	POST /upnp/control/WANCommonIFC1 HTTP/1.1
23	0.137859	192.168.0.1	192.168.0.107	HTTP/X.	406	HTTP/1.1 200 OK
31	0.141964	192.168.0.107	192.168.0.1	HTTP/X.	363	POST /upnp/control/WANCommonIFC1 HTTP/1.1
35	0.148826	192.168.0.1	192.168.0.107	HTTP/X.	389	HTTP/1.1 200 OK
43	0.152377	192.168.0.107	192.168.0.1	HTTP/X.	367	POST /upnp/control/WANCommonIFC1 HTTP/1.1
47	0.159232	192.168.0.1	192.168.0.107	HTTP/X.	486	HTTP/1.1 200 OK
51	0.162307	192.168.0.107	192.168.0.1	HTTP/X.	484	GET /filestreaming/service/files/ab0f387-b8f7-4ebb-b61a-51d3b7fe2607?1=17496316118P2-4048P3-28P4-h6vaD6MP556LVk35VjQY112sR2fVdV
64	0.166447	192.168.0.107	192.168.0.1	HTTP/X.	363	POST /upnp/control/WANCommonIFC1 HTTP/1.1
80	0.166521	192.168.0.107	192.168.0.1	HTTP/X.	389	HTTP/1.1 200 OK
91	0.169822	192.168.0.107	192.168.0.1	HTTP/X.	367	POST /upnp/control/WANCommonIFC1 HTTP/1.1
103	0.166213	192.168.0.107	192.168.0.1	HTTP/X.	406	HTTP/1.1 200 OK
104	0.166213	192.168.0.107	192.168.0.1	HTTP/X.	1015	HTTP/1.1 200 Partial Content (application/x-chrome-extension)
113	0.171053	192.168.0.107	192.168.0.1	HTTP/X.	363	POST /upnp/control/WANCommonIFC1 HTTP/1.1
117	0.177915	192.168.0.1	192.168.0.107	HTTP/X.	389	HTTP/1.1 200 OK
125	0.181480	192.168.0.107	192.168.0.1	HTTP/X.	367	POST /upnp/control/WANCommonIFC1 HTTP/1.1
126	0.181481	192.168.0.107	192.168.0.1	HTTP/X.	406	HTTP/1.1 200 OK
142	1.104132	192.168.0.107	192.168.0.1	HTTP/X.	363	POST /upnp/control/WANCommonIFC1 HTTP/1.1
146	1.110051	192.168.0.1	192.168.0.107	HTTP/X.	389	HTTP/1.1 200 OK
154	1.113854	192.168.0.107	192.168.0.1	HTTP/X.	367	POST /upnp/control/WANCommonIFC1 HTTP/1.1
158	1.120191	192.168.0.1	192.168.0.107	HTTP/X.	406	HTTP/1.1 200 OK
168	1.123840	192.168.0.107	192.168.0.1	HTTP/X.	363	POST /upnp/control/WANCommonIFC1 HTTP/1.1
170	1.141605	192.168.0.1	192.168.0.107	HTTP/X.	389	HTTP/1.1 200 OK
179	1.148945	192.168.0.107	192.168.0.1	HTTP/X.	367	POST /upnp/control/WANCommonIFC1 HTTP/1.1

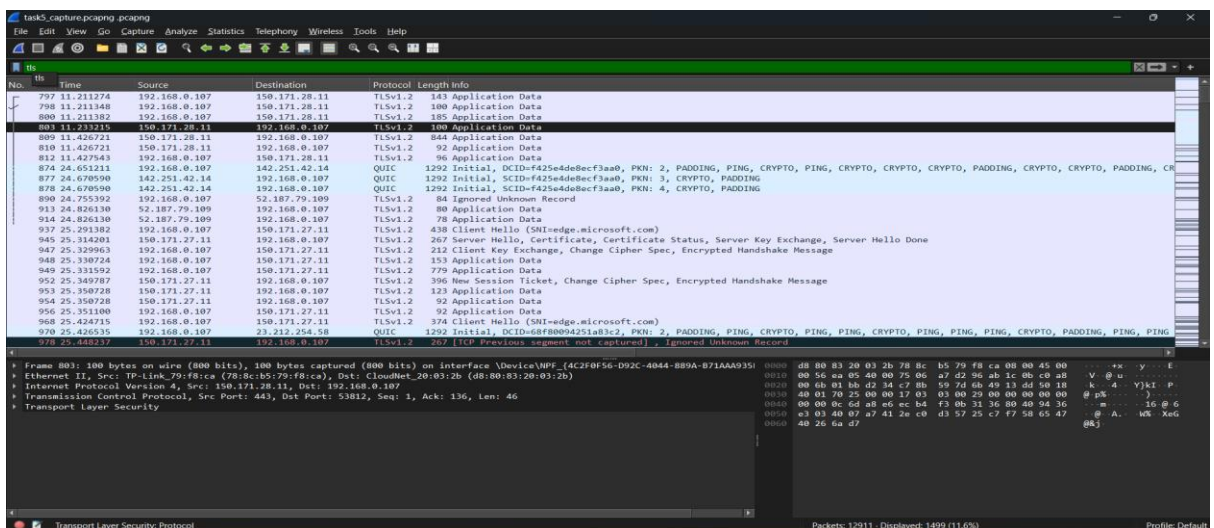
- DNS Traffic:
- Dns



The image shows a Wireshark capture of DNS traffic. The packet list on the left shows several DNS queries and responses. The selected packet (No. 886) is a DNS query for 192.168.0.107. The packet details pane on the right shows the structure of the DNS message, including the Ethernet II, internet protocol version, and domain name system. The packet bytes pane on the right shows the raw data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
795	11.210592	192.168.0.107	192.168.0.1	DNS	78	Standard query 0x54bf A edge.microsoft.com
796	11.210992	192.168.0.107	192.168.0.1	DNS	78	Standard query 0x54bf A edge.microsoft.com
801	11.232409	192.168.0.1	192.168.0.107	DNS	178	Standard query response 0x54bf A edge.microsoft.com CNAME edge-microsoft.com ax-0002.ax-msedge.net CNAME ax-0002.ax-msedge.net A
929	25.243348	192.168.0.107	192.168.0.1	DNS	178	Standard query 0x54bf A edge.microsoft.com
930	25.243969	192.168.0.107	192.168.0.1	DNS	78	Standard query 0x9590 HTTPS edge.microsoft.com
931	25.262918	192.168.0.107	192.168.0.1	DNS	178	Standard query response 0x9590 HTTPS edge.microsoft.com CNAME edge-microsoft.com ax-0002.ax-msedge.net CNAME ax-0002.ax-msedge.net A
932	25.262938	192.168.0.1	192.168.0.107	DNS	193	Standard query response 0x9590 HTTPS edge.microsoft.com CNAME edge-microsoft.com ax-0002.ax-msedge.net CNAME ax-0002.ax-msedge.net A
958	25.401145	192.168.0.107	192.168.0.1	DNS	72	Standard query 0x91af A www.bing.com
959	25.401655	192.168.0.107	192.168.0.1	DNS	72	Standard query 0x8f87 HTTPS www.bing.com
961	25.422455	192.168.0.1	192.168.0.107	DNS	254	Standard query response 0x8f87 HTTPS www.bing.com CNAME www.bing.com trafficmanager.net CNAME www.bing.com edgekey.net CNAME
962	25.422455	192.168.0.1	192.168.0.107	DNS	337	Standard query response 0x91af A www.bing.com CNAME www.bing.com trafficmanager.net CNAME www.bing.com edgekey.net CNAME
963	25.422518	192.168.0.107	192.168.0.1	DNS	80	Standard query 0xc925 A substrate.office.com
964	25.423015	192.168.0.1	192.168.0.107	DNS	80	Standard query 0x2b80 HTTPS substrate.office.com
971	25.441127	192.168.0.1	192.168.0.107	DNS	221	Standard query response 0x2b80 HTTPS substrate.office.com CNAME outlook.office365.com CNAME outlook-g2-tm-4.office.com SOA ns1-34.az
981	25.448237	192.168.0.1	192.168.0.107	DNS	266	Standard query response 0xc925 A substrate.office.com CNAME outlook.office365.com CNAME outlook-g2-tm-4.office.com 52.97.92.114 A
1032	25.515222	192.168.0.107	192.168.0.1	DNS	78	Standard query 0x5dbb A edge.microsoft.com
1033	25.515807	192.168.0.107	192.168.0.1	DNS	78	Standard query 0x8db3 HTTPS edge.microsoft.com
1046	25.534841	192.168.0.1	192.168.0.107	DNS	178	Standard query response 0x5dbb A edge.microsoft.com CNAME edge-microsoft.com ax-0002.ax-msedge.net CNAME ax-0002.ax-msedge.net A
1047	25.534861	192.168.0.1	192.168.0.107	DNS	31	Standard query 0x8db3 HTTPS edge.microsoft.com
1289	28.509679	192.168.0.107	192.168.0.1	DNS	85	Standard query 0x8394 A l3h3.googleusercontent.com
1290	28.510098	192.168.0.107	192.168.0.1	DNS	85	Standard query 0x8394 A l3h3.googleusercontent.com
1294	28.528356	192.168.0.1	192.168.0.107	DNS	171	Standard query response 0x8394 A l3h3.googleusercontent.com CNAME googlehosted.l.googleusercontent.com SOA ns1.google.c
1296	28.531893	192.168.0.1	192.168.0.107	DNS	130	Standard query response 0x8394 A l3h3.googleusercontent.com CNAME googlehosted.l.googleusercontent.com 142.250.70.33
1384	28.572118	192.168.0.107	192.168.0.1	DNS	75	Standard query 0x6789 A www.gstatic.com

- Another ports :
- tls



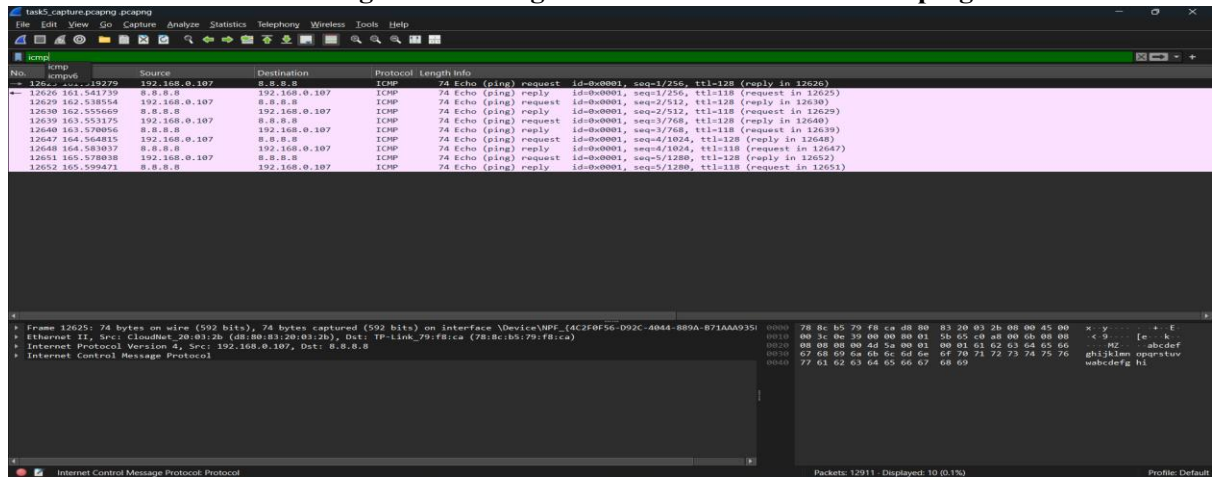
The image shows a Wireshark capture of TLS traffic. The packet list on the left shows several TLS messages. The selected packet (No. 886) is a TLS application data packet. The packet details pane on the right shows the structure of the TLS message, including the Ethernet II, internet protocol version, and transport layer security. The packet bytes pane on the right shows the raw data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
797	11.211274	192.168.0.107	150.171.28.11	TLSv1.2	143	Application Data
798	11.211348	192.168.0.107	150.171.28.11	TLSv1.2	100	Application Data
800	11.211382	192.168.0.107	150.171.28.11	TLSv1.2	185	Application Data
886	11.182325	150.171.28.11	192.168.0.107	TLSv1.2	140	Application Data
889	11.426721	150.171.28.11	192.168.0.107	TLSv1.2	144	Application Data
810	11.426721	150.171.28.11	192.168.0.107	TLSv1.2	92	Application Data
812	11.427943	192.168.0.107	150.171.28.11	TLSv1.2	96	Application Data
874	24.651211	192.168.0.107	142.251.42.14	QUIC	1292	Initial, DCID=f425d4de8ecf3a0b, PKN: 3, PADDING, PING, CRYPTO, PING, CRYPTO, CRYPTO, PADDING, CRYPTO, CRYPTO, PADDING, CRYPTO
877	24.678590	142.251.42.14	192.168.0.107	QUIC	1292	Initial, SCID=f425d4de8ecf3a0b, PKN: 3, CRYPTO, PADDING
878	24.678590	142.251.42.14	192.168.0.107	QUIC	1292	Initial, SCID=f425d4de8ecf3a0b, PKN: 4, CRYPTO, PADDING
890	24.755392	192.168.0.107	52.187.79.109	TLSv1.2	84	Ignored Unknown Record
913	24.826130	52.187.79.109	192.168.0.107	TLSv1.2	80	Application Data
914	24.826130	52.187.79.109	192.168.0.107	TLSv1.2	78	Application Data
937	25.291382	192.168.0.107	150.171.27.11	TLSv1.2	438	Client Hello (SHA256edge.microsoft.com)
945	25.314201	150.171.27.11	192.168.0.107	TLSv1.2	267	Server Hello, Certificate, Certificate Status, Server Key Exchange, Server Hello Done
947	25.329963	192.168.0.107	150.171.27.11	TLSv1.2	212	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
948	25.330724	192.168.0.107	150.171.27.11	TLSv1.2	153	Application Data
949	25.331592	192.168.0.107	150.171.27.11	TLSv1.2	779	Application Data
952	25.349787	150.171.27.11	192.168.0.107	TLSv1.2	396	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
953	25.350728	150.171.27.11	192.168.0.107	TLSv1.2	123	Application Data
954	25.350728	150.171.27.11	192.168.0.107	TLSv1.2	92	Application Data
956	25.351100	192.168.0.107	150.171.27.11	TLSv1.2	92	Application Data
968	25.424715	192.168.0.107	150.171.27.11	TLSv1.2	374	Client Hello (SHA256edge.microsoft.com)
970	23.425035	192.168.0.107	150.171.27.11	TLSv1.2	1292	Initial, DCID=f425d4de8ecf3a0b, PKN: 2, PADDING, PING, CRYPTO, PING, PING, CRYPTO, PING, PING, CRYPTO, PADDING, PING, PING
978	25.448237	150.171.27.11	192.168.0.107	TLSv1.2	267	[TCP Previous segment not captured], Ignored Unknown Record

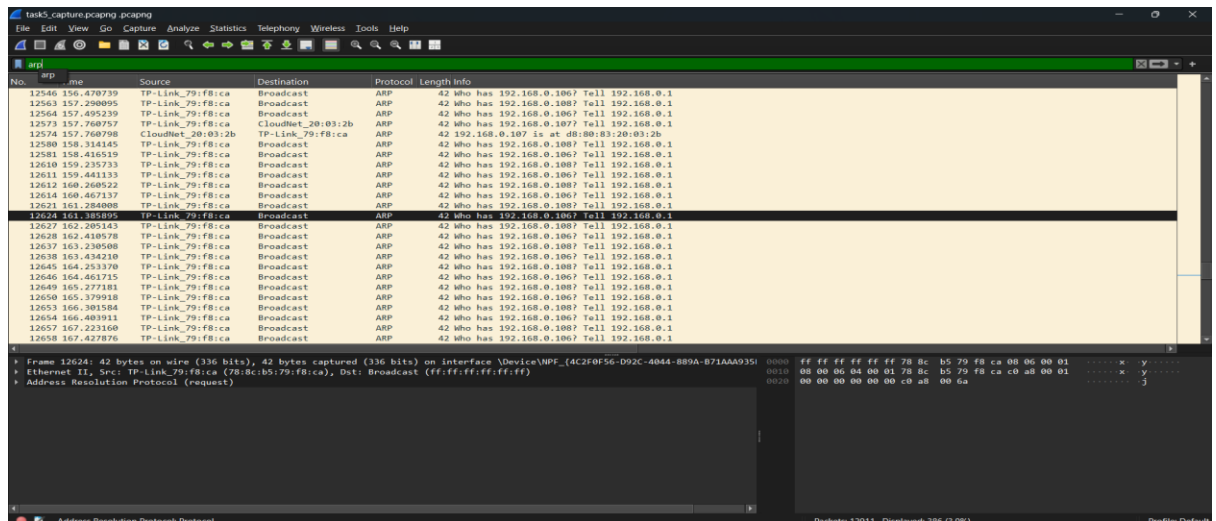
Task 5th: Wireshark Network Traffic Analysis

• Icmp

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

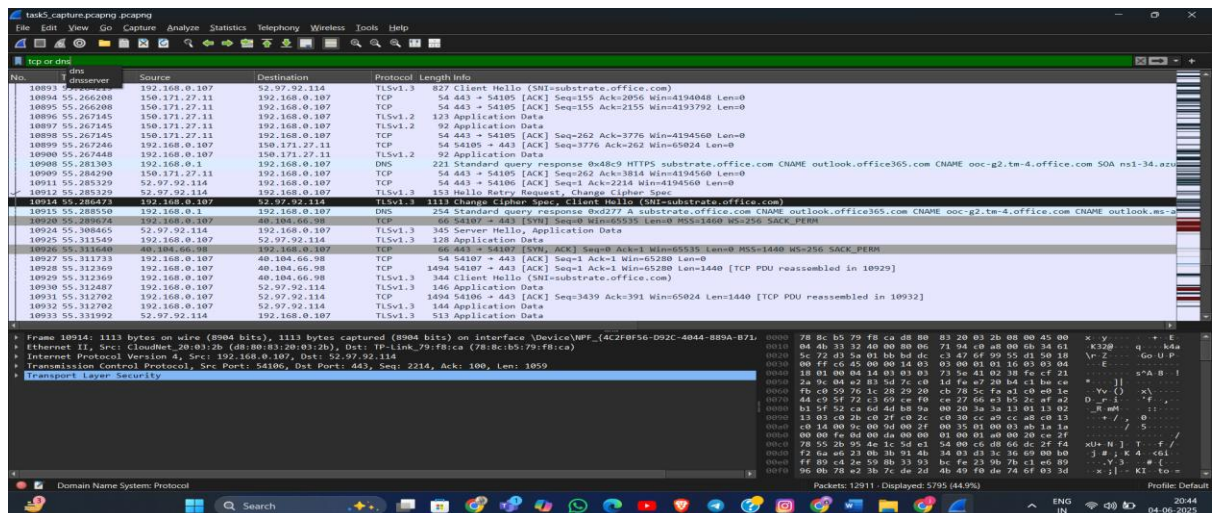


• Arp



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.