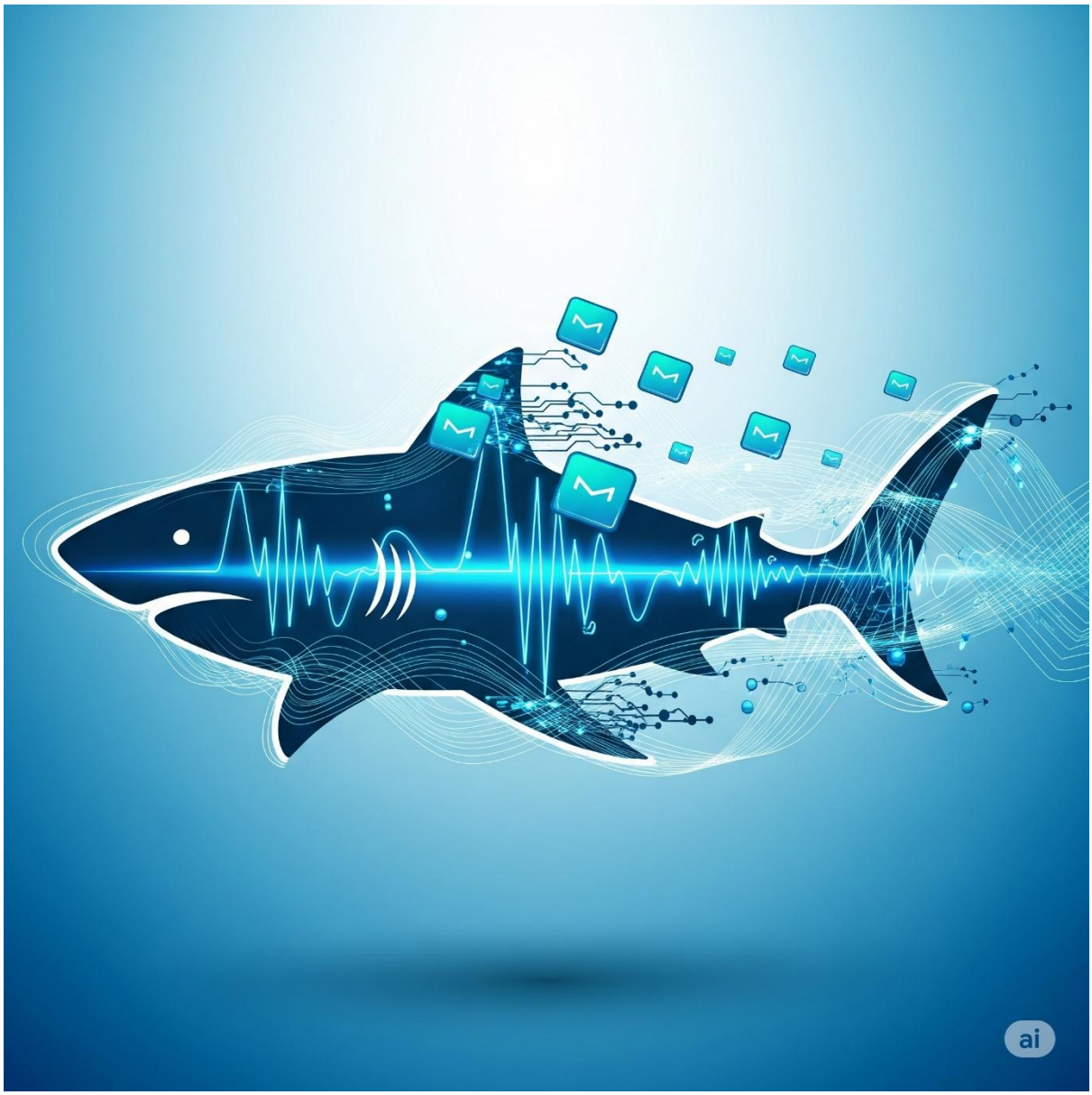
Day 6 of Learning Cyber Security

Platform: Kali Linux

Name: Ayesha Nadeem

Topic: Wireshark

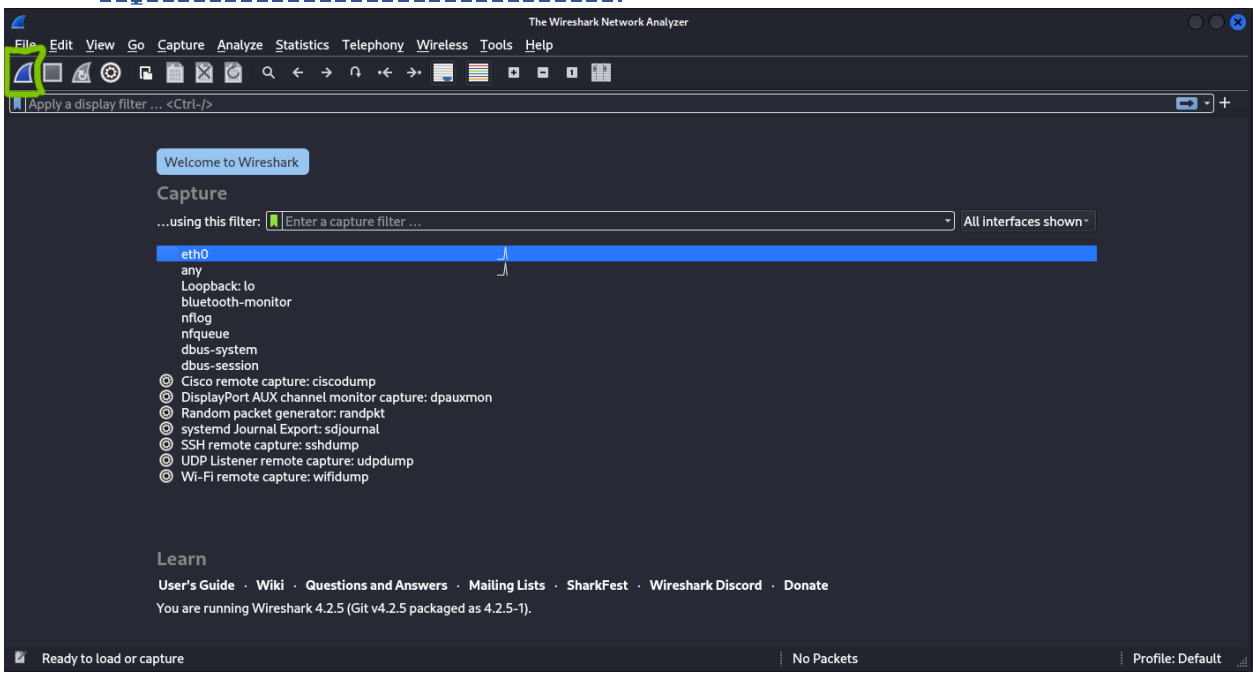


Contact Me: ayeshanm8@gmail.com

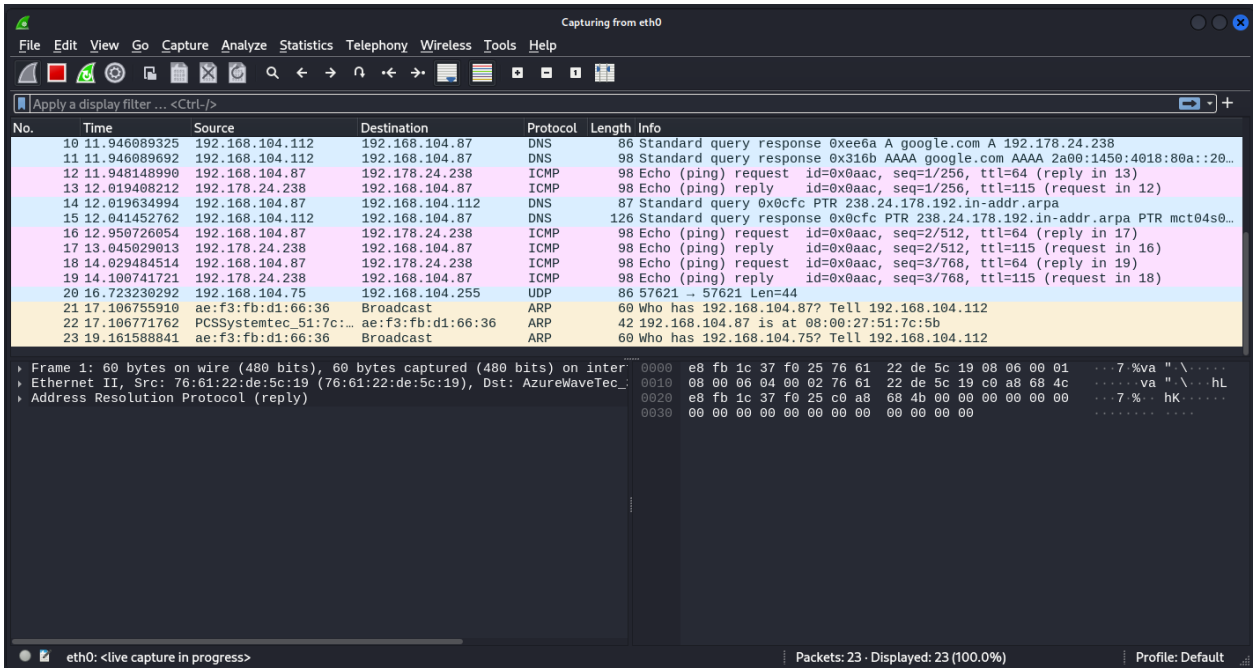
Date: 6th July, 2025

Wireshark: The Packet sniffer

1. Open Wireshark and Hit eth0

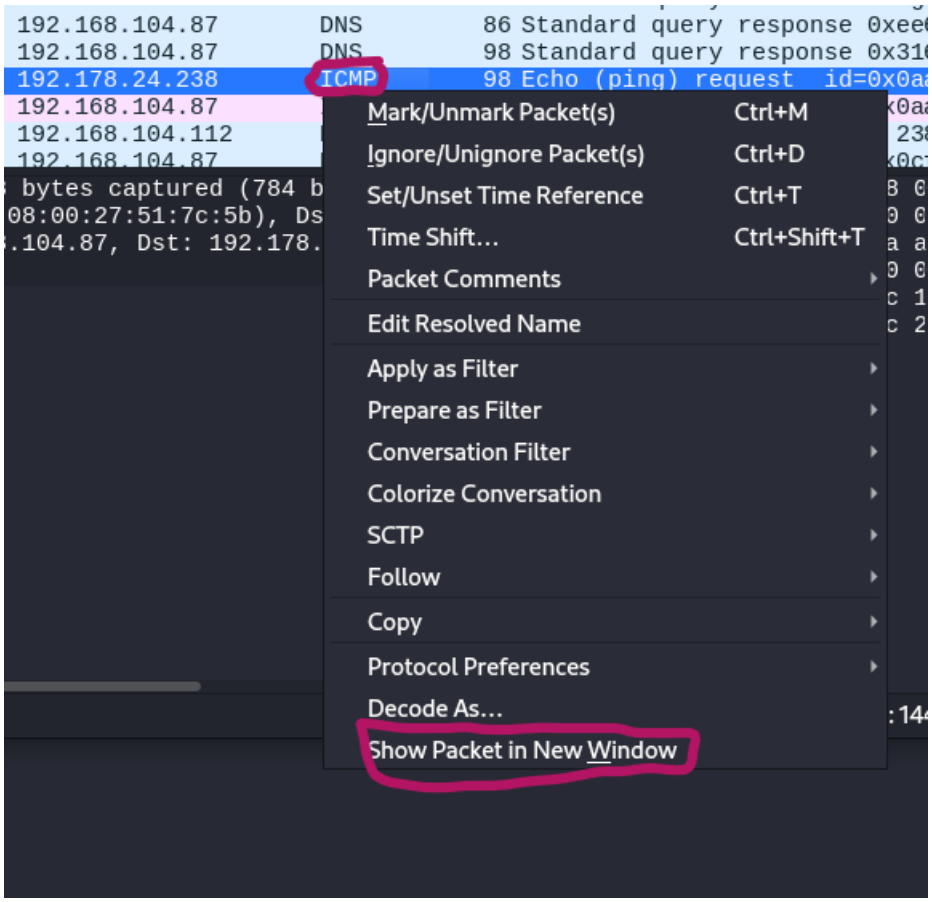


2. Packet Sniffed from eth0 network

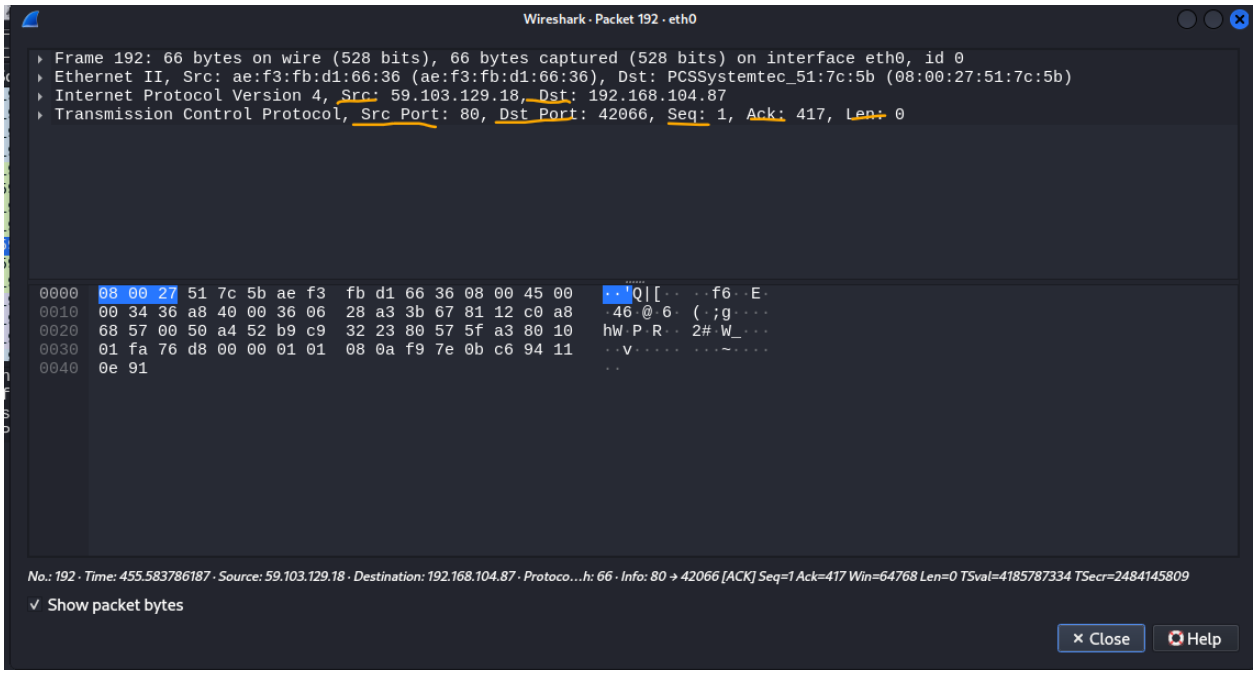


3. Inspect Packet Steps:

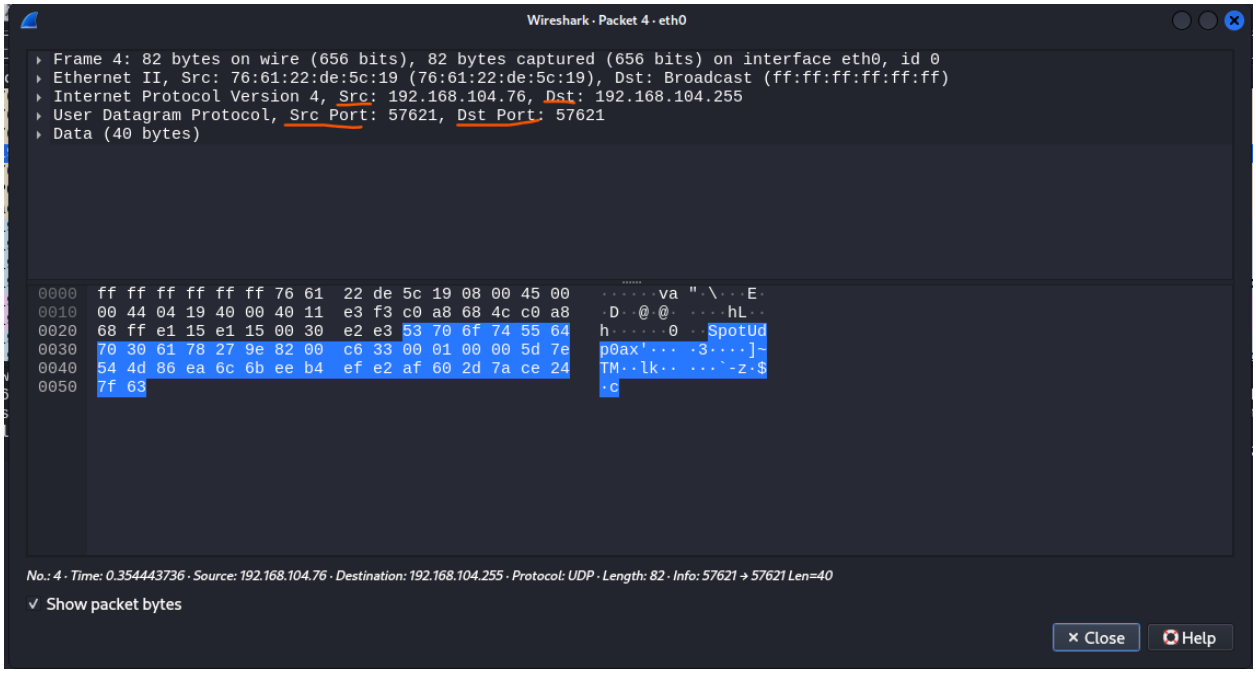
➔ Right click on any packet and select "Show packet in new Windows"



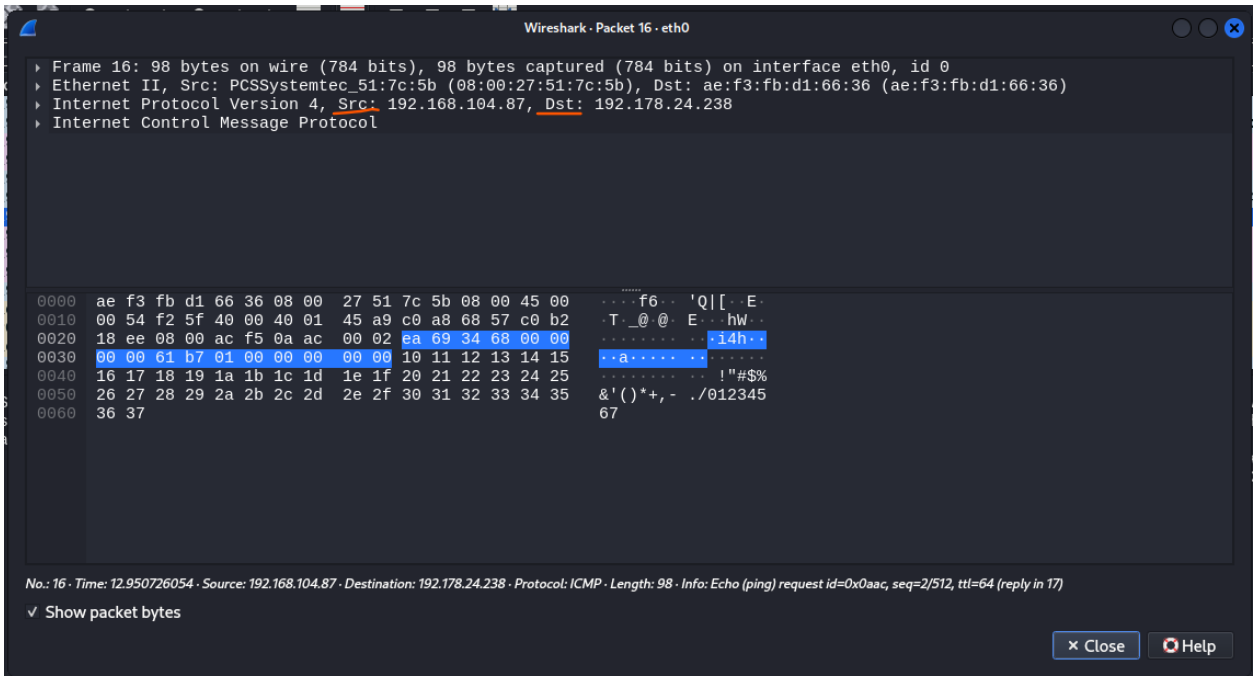
TCP Packet Details:



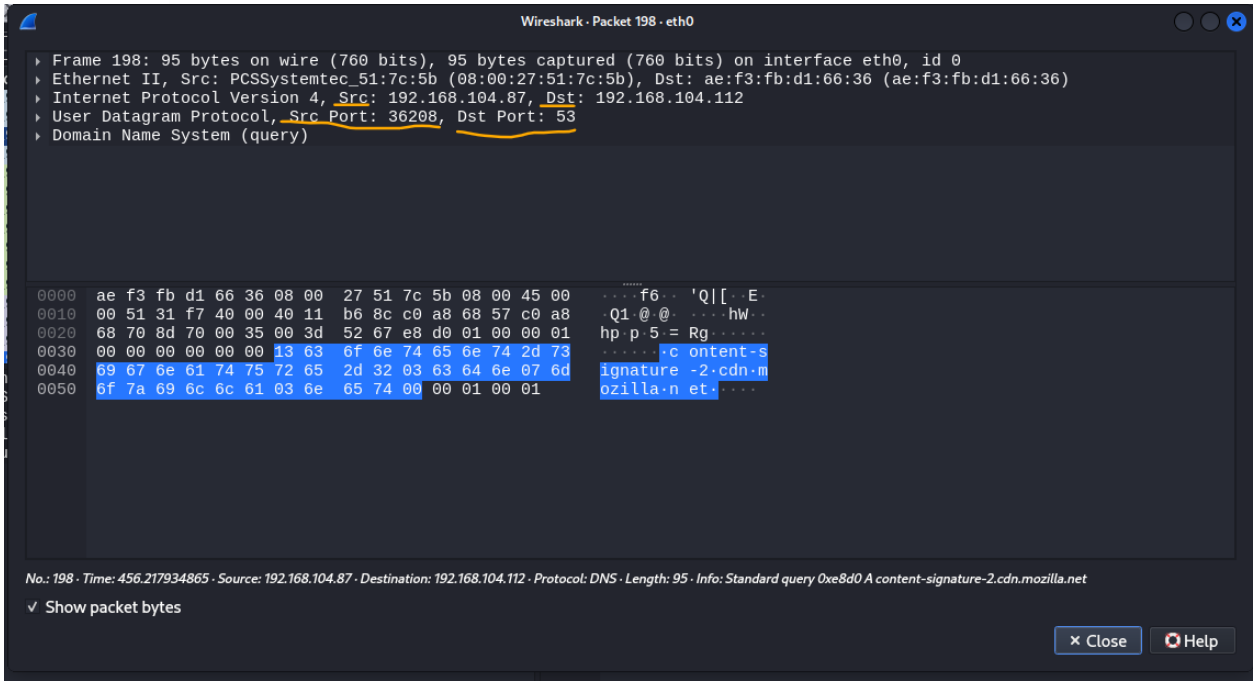
UDP Packet Details:



ICMP Packet Details:



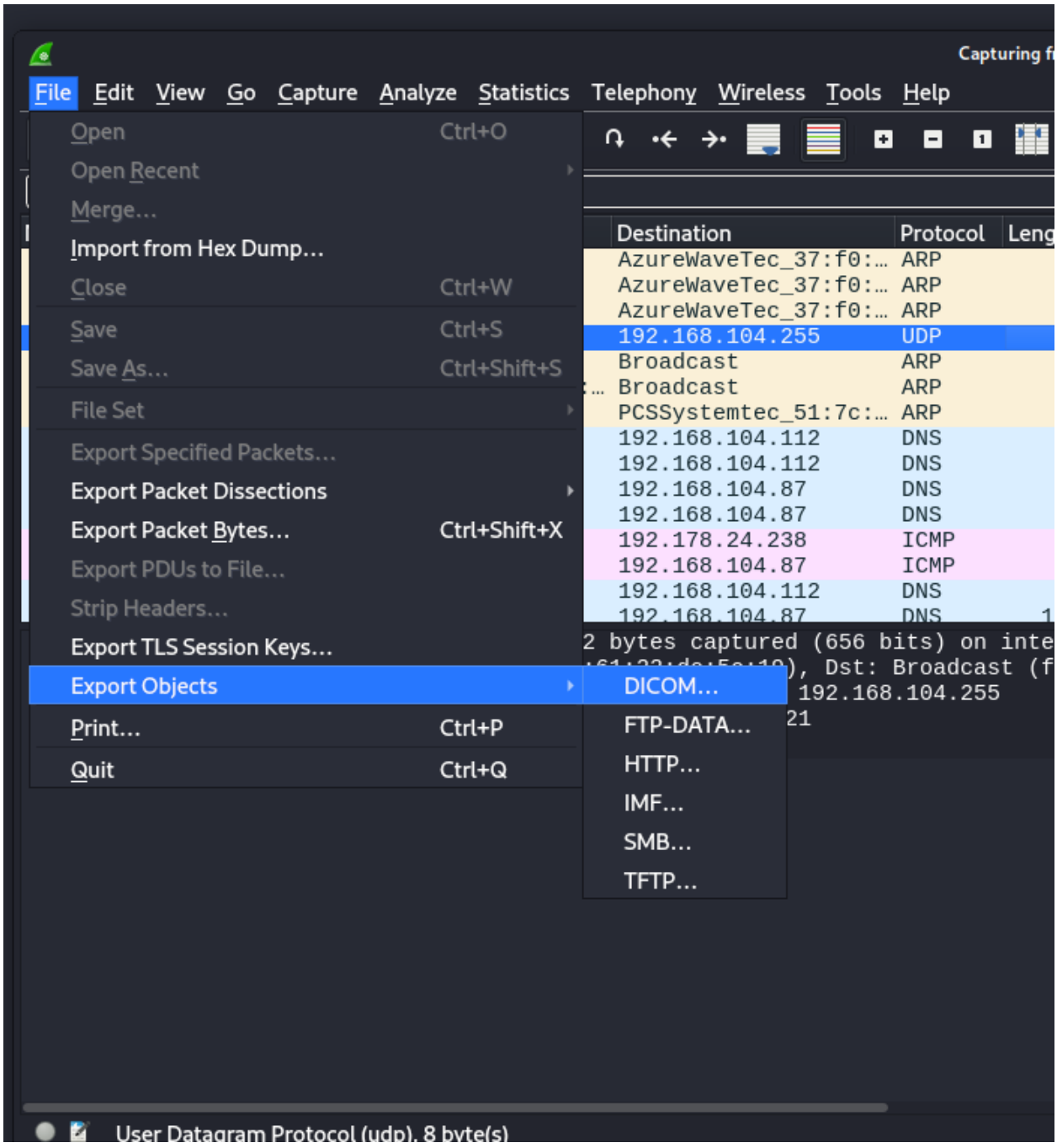
DNS Packets Details:

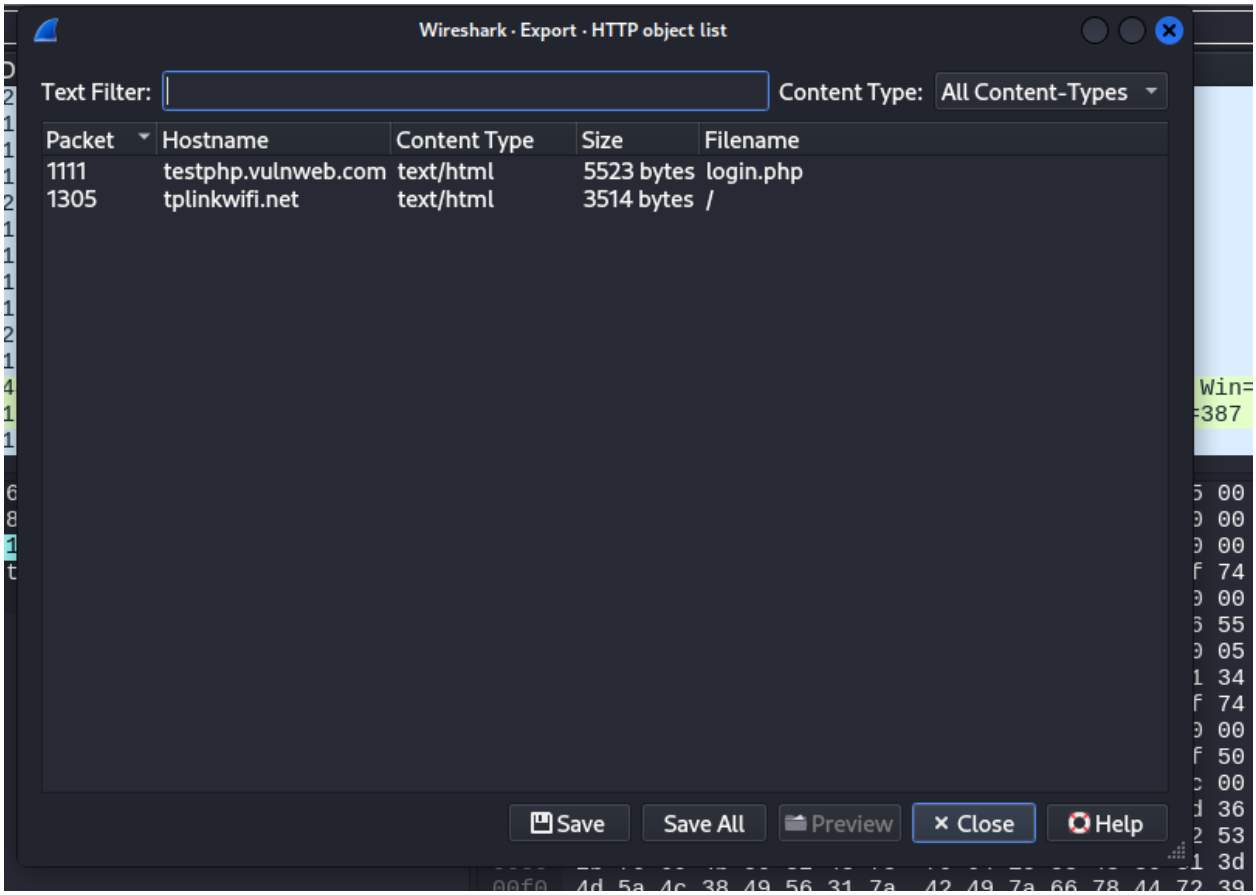


Similarly we can inspect any packet to get their IP's, port in which they communicates, sequence and acknowledgment numbers

4. Export Object:

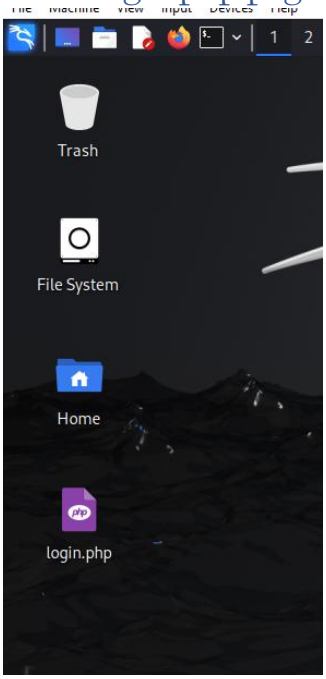
- Go to File
- Select Export Object
- Click on HTTP

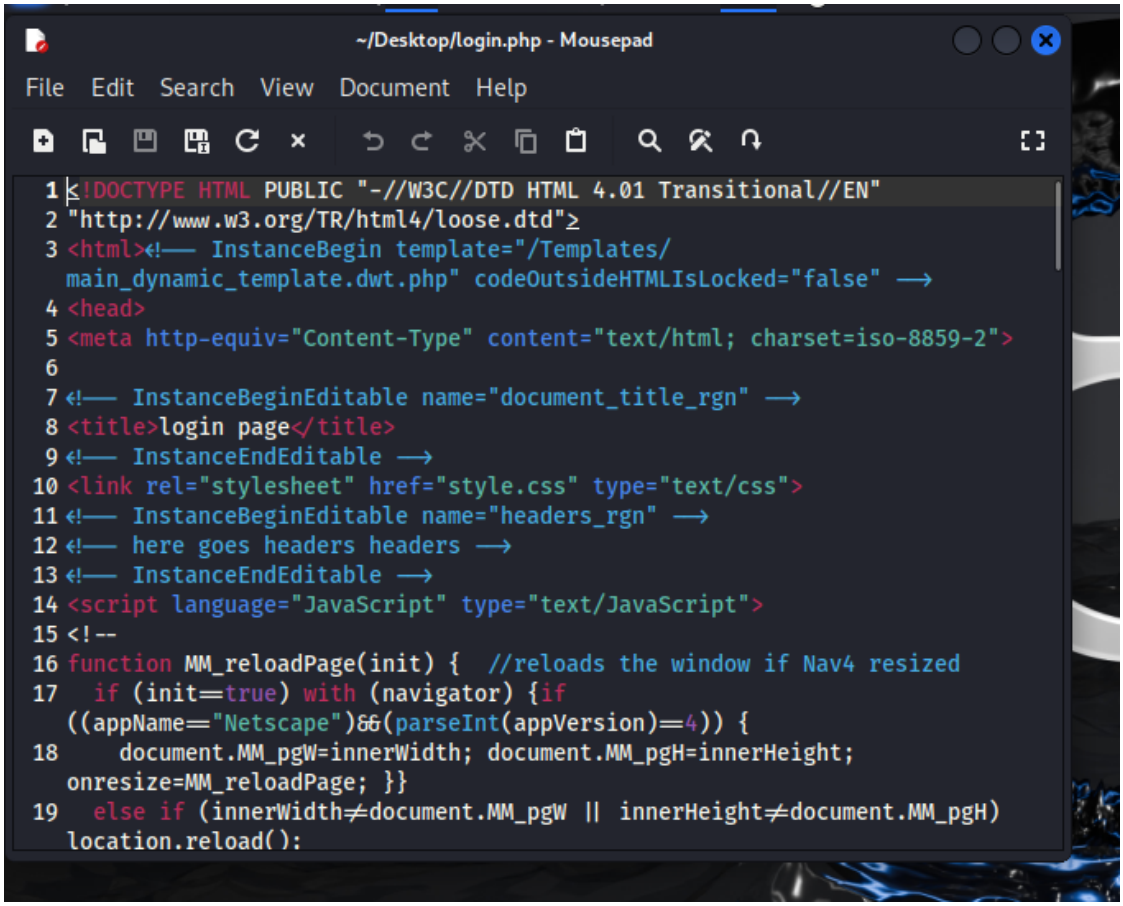




So it capture the site you opened in your firefox/chrome.

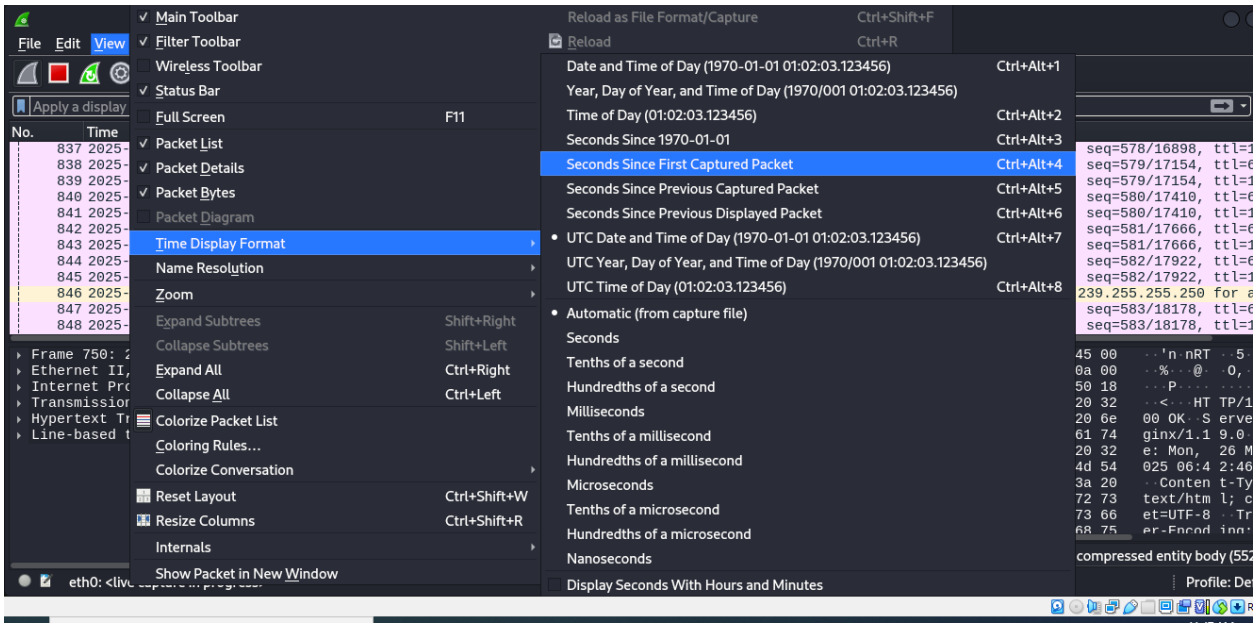
Save a login.php page to get its code:

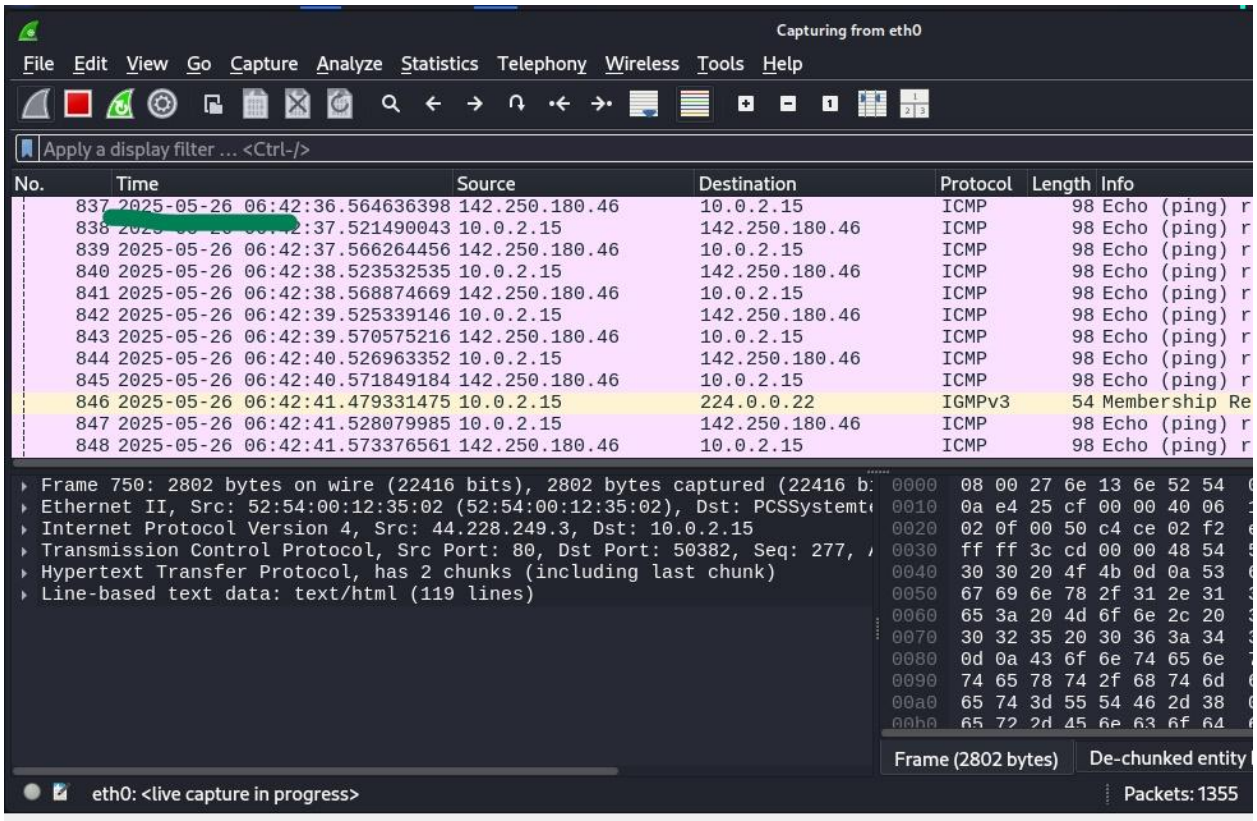




5. Setting data Format:

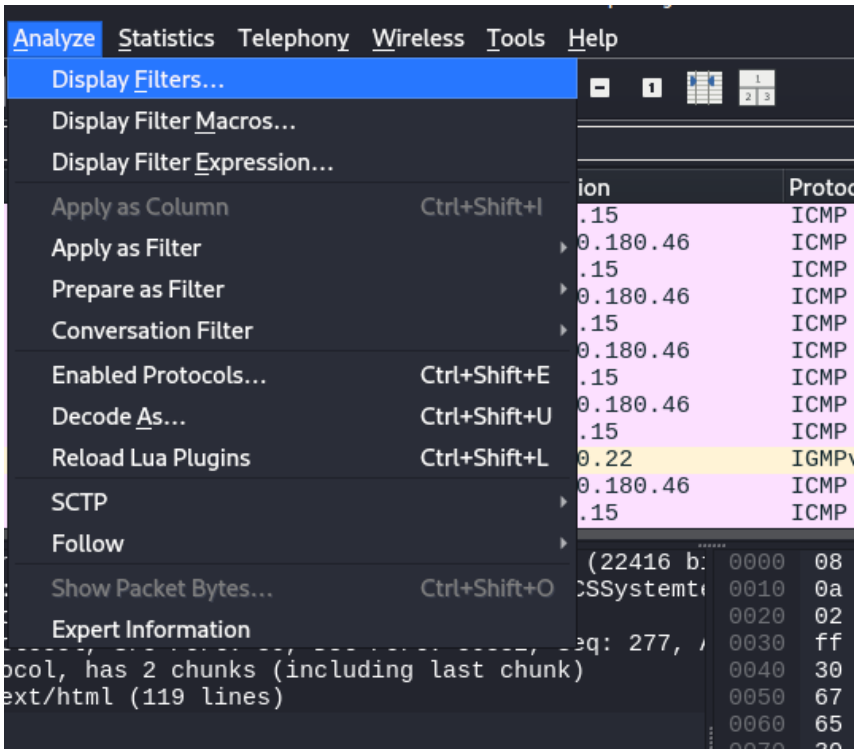
Go to view, select “Time Display Format” and click on any format you want.

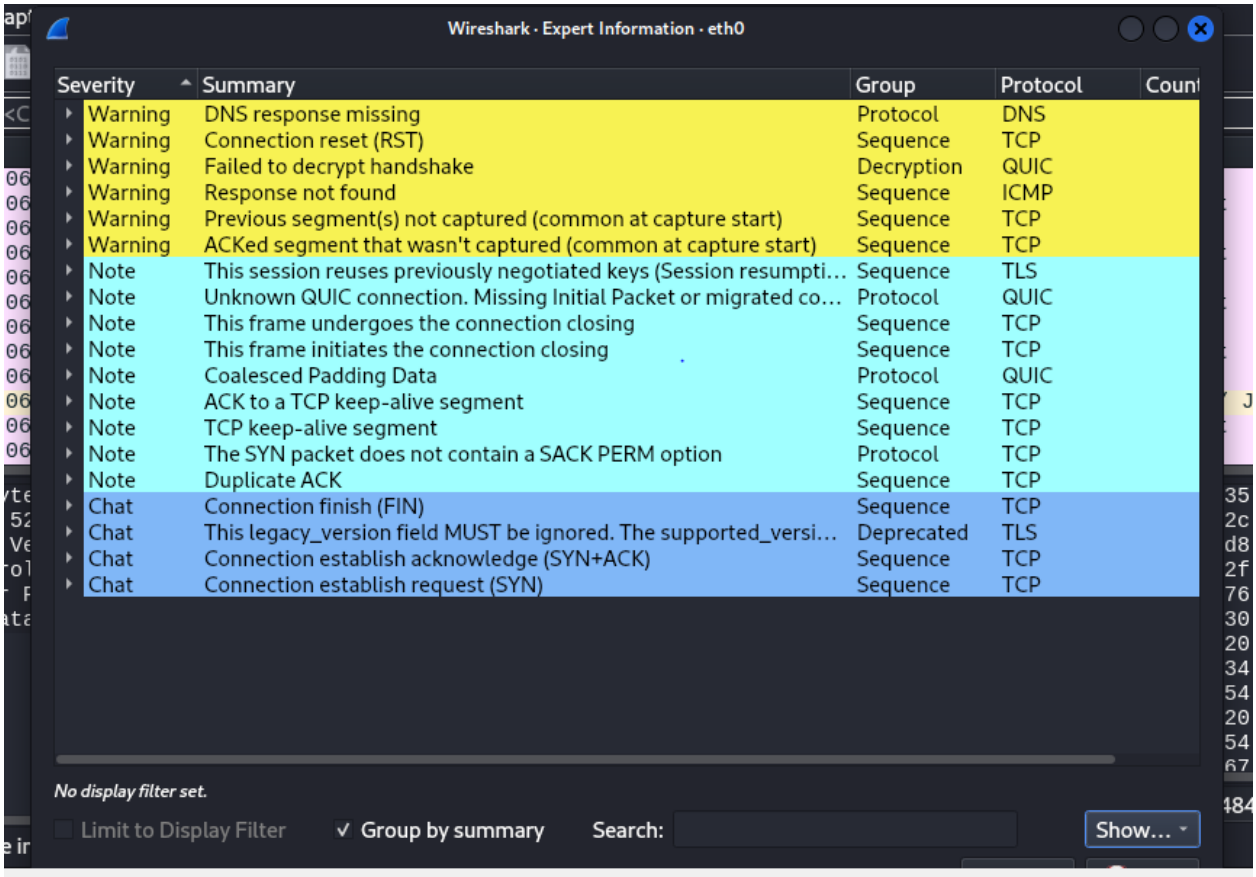




6. Analyzing packet with color strategy:

Go to analyze then go to Expert Information





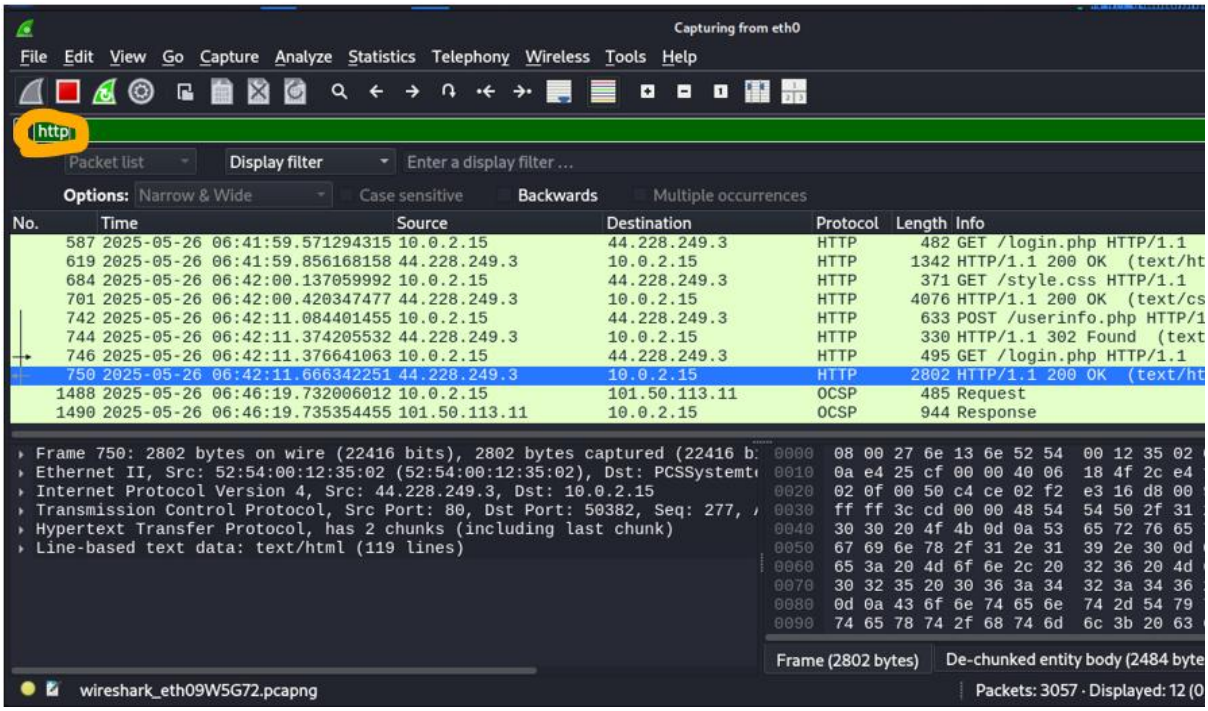
This show the network event happened with severity levels like:

- *Notes*: Informational messages, like TCP keep-alives.
- *Warnings*: Potential issues like missing responses.
- *Errors or Chats*: More serious anomalies, e.g. resets or failed handshakes

It helps in quick triage of issues without deep packet digging.

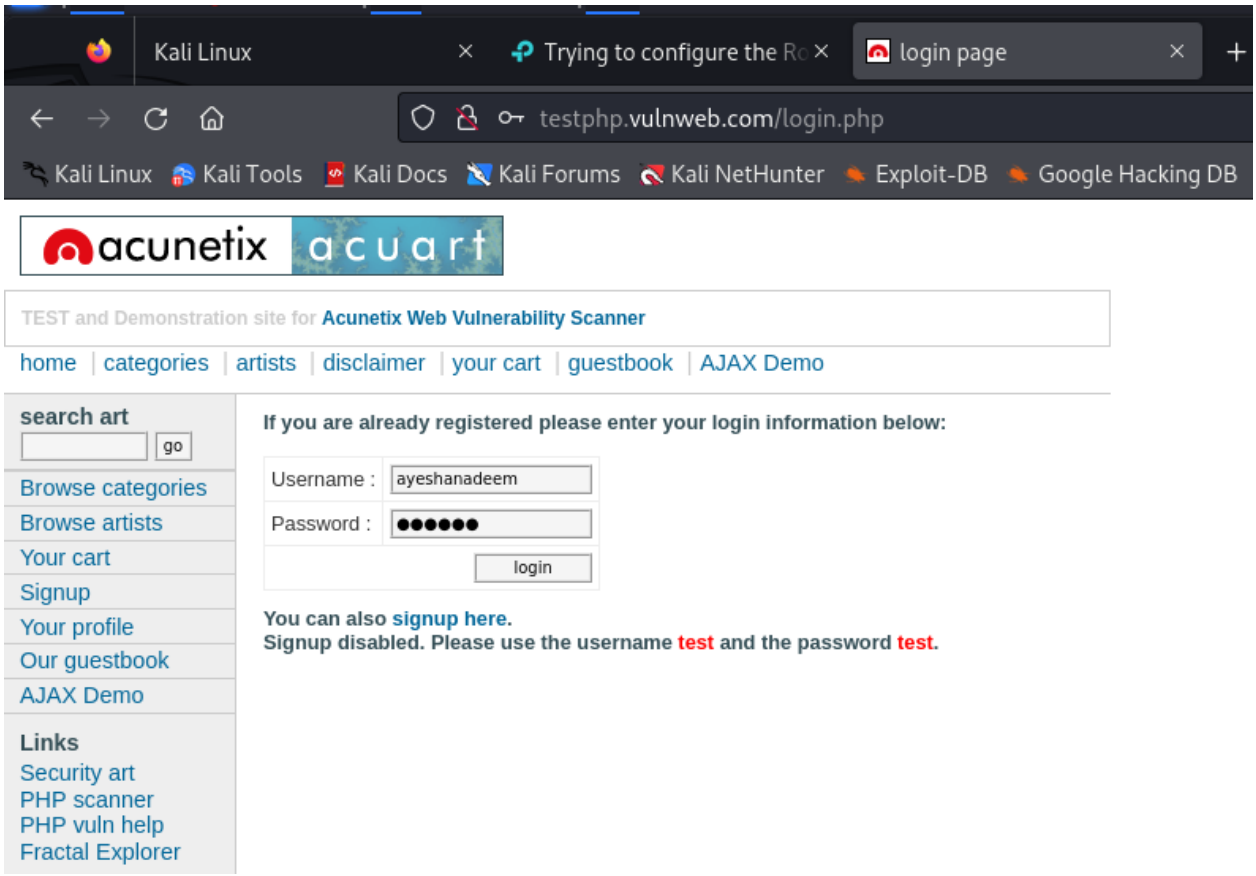
7. Filtering in Wireshark:

Applied filter of HTTP to just show http packets

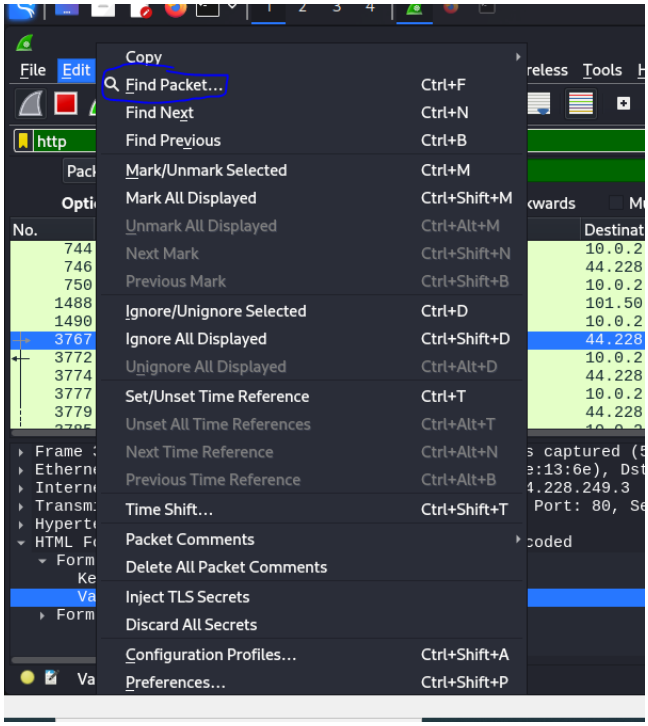


8. Extracting credentials using Wireshark packet filtering:

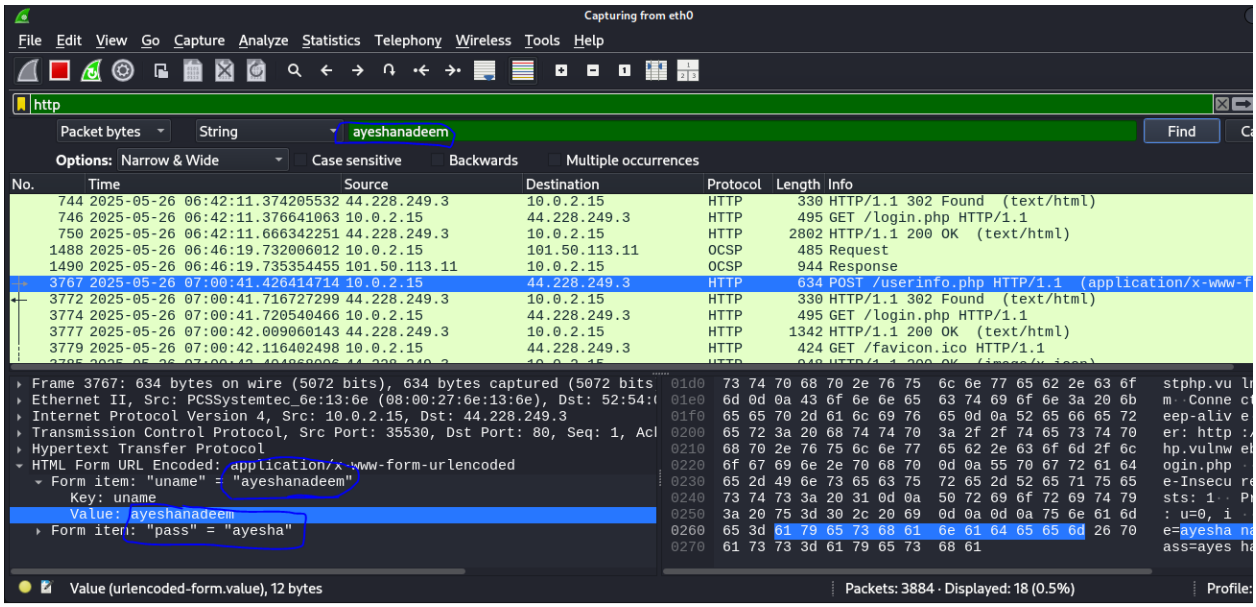
First go to http://testphp.vulnweb.com/login.php and enter fake credentials.



Come back to wireshark. Click on Edit on toolbar and select "Find Packet"

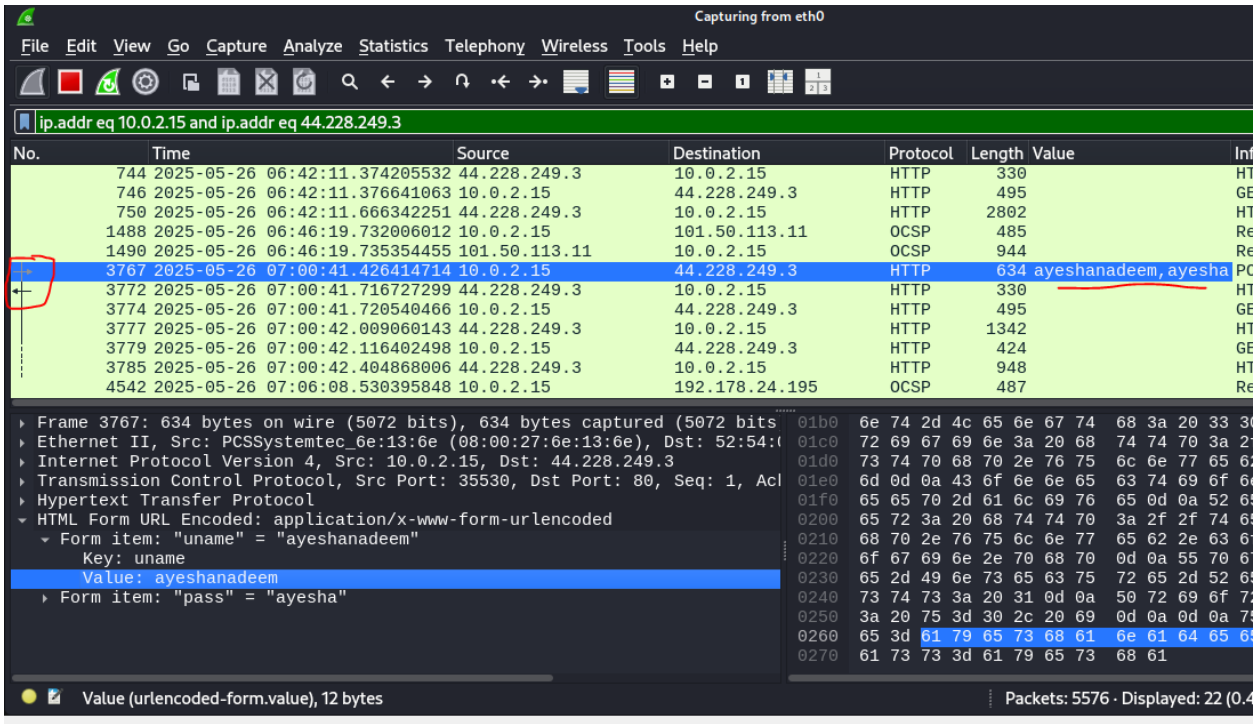


First set parameter like “stirng”, “packetbytes” and then type in search bar username.

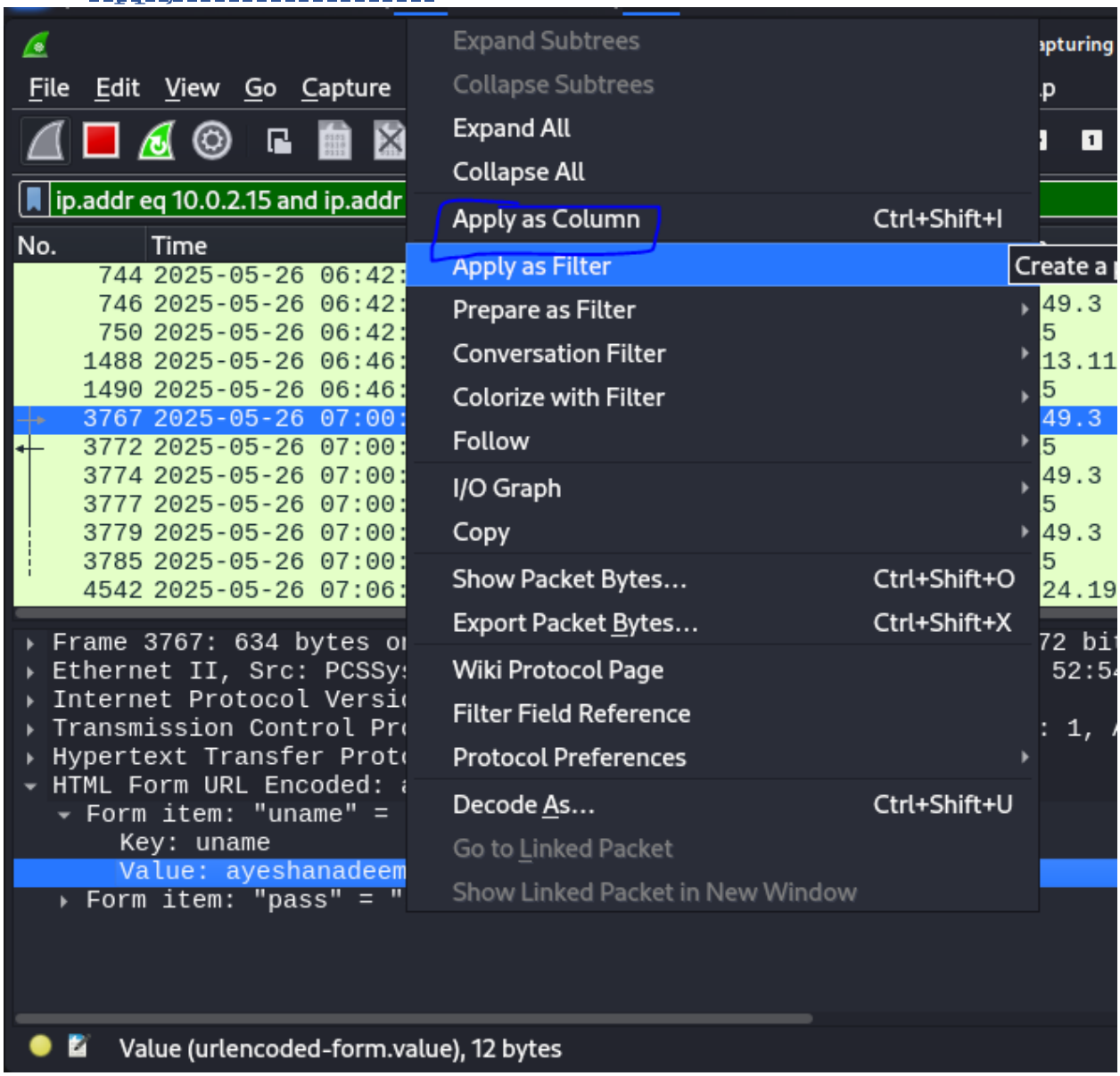


Forward arrow shows request packet

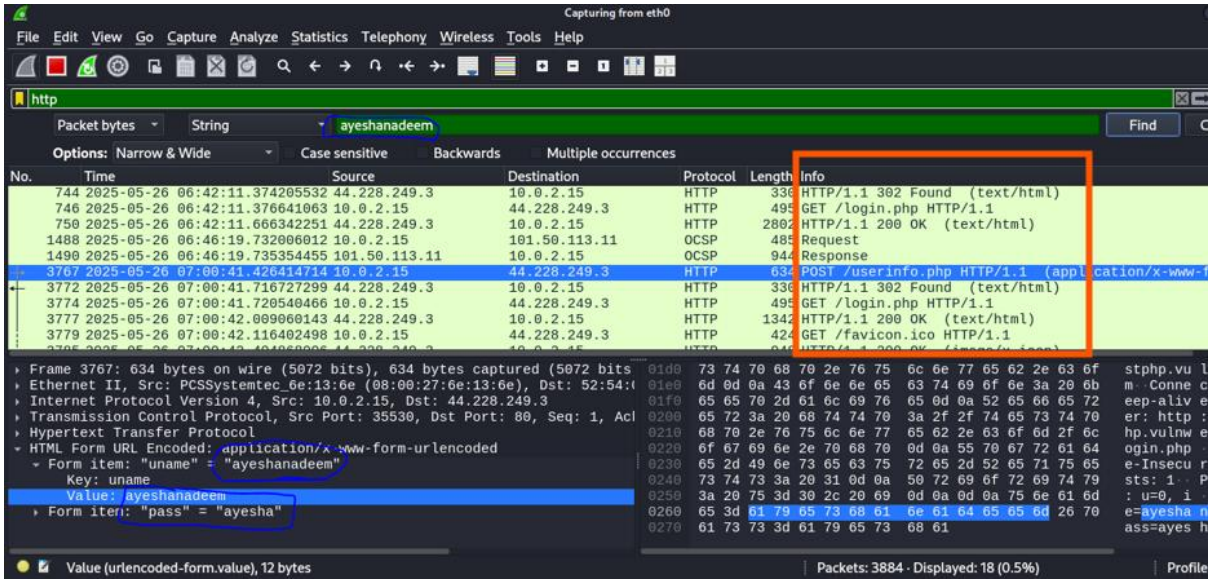
Backward arrow shows response packet



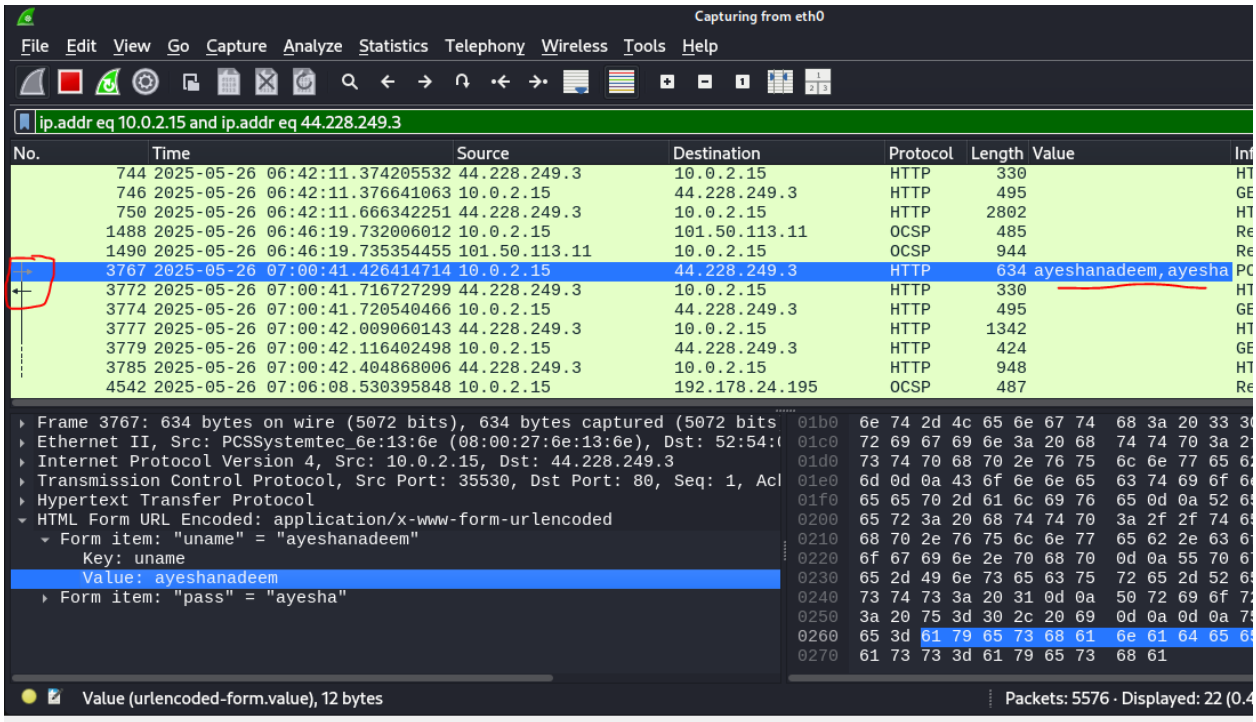
9. Apply filter as column:



Before

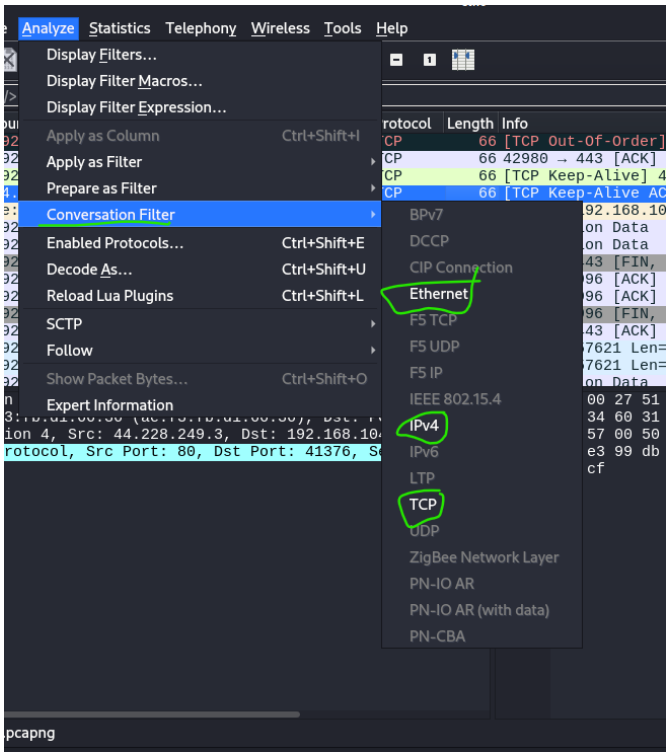


After:



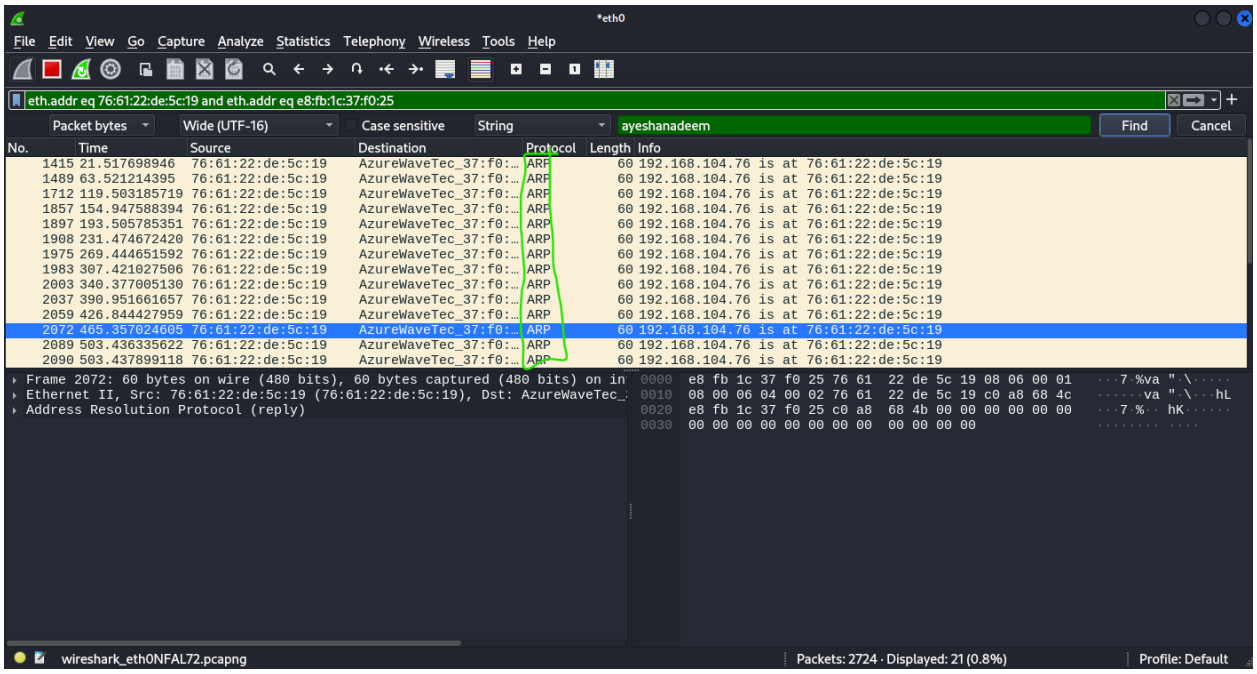
10. Conversational Filtering:

Go to analyze, select conversation filter.



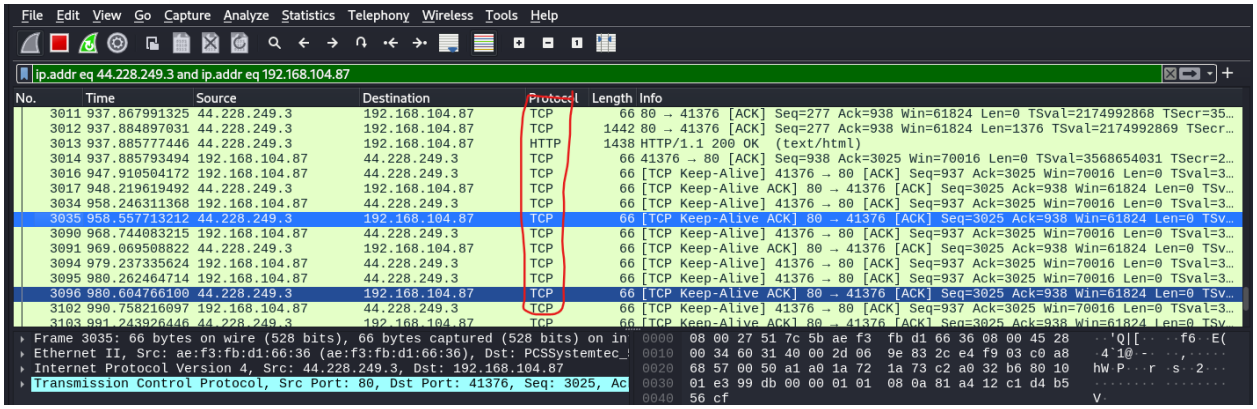
From here select any conversational filter for showing off specific traffic.

Ethernet filter



The conversation of ethernet follows ARP communication protocols.

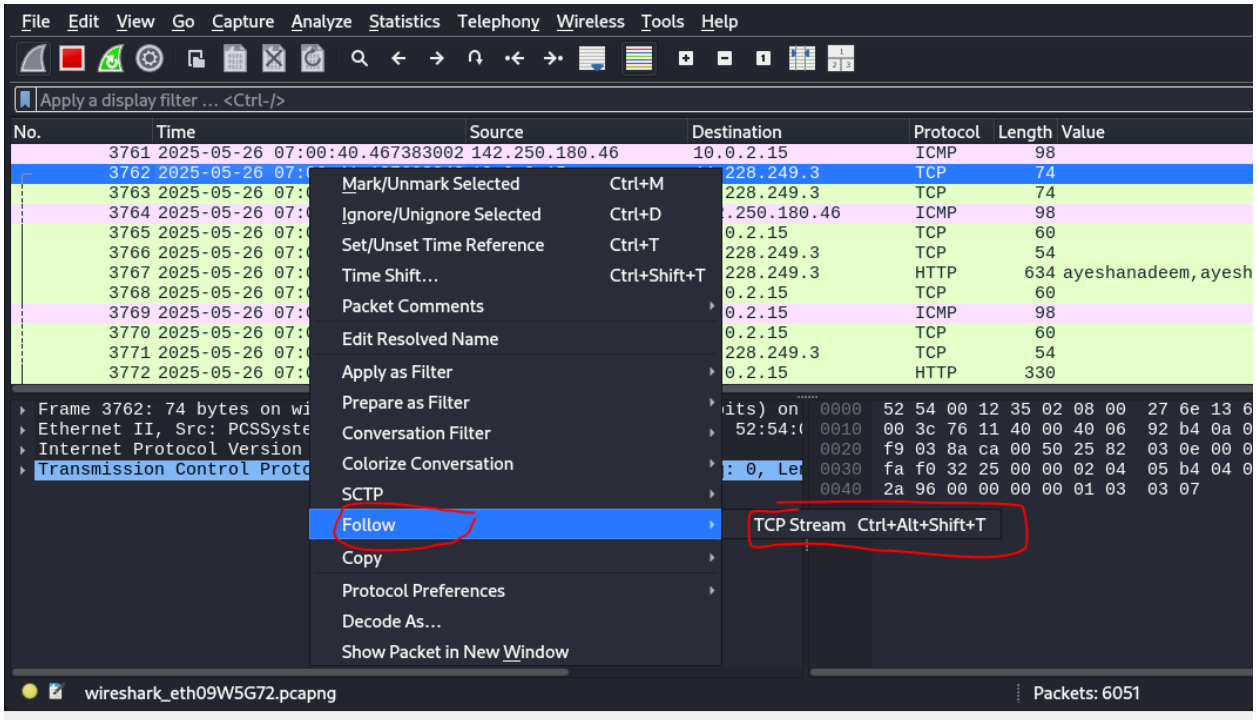
IPv4 filter

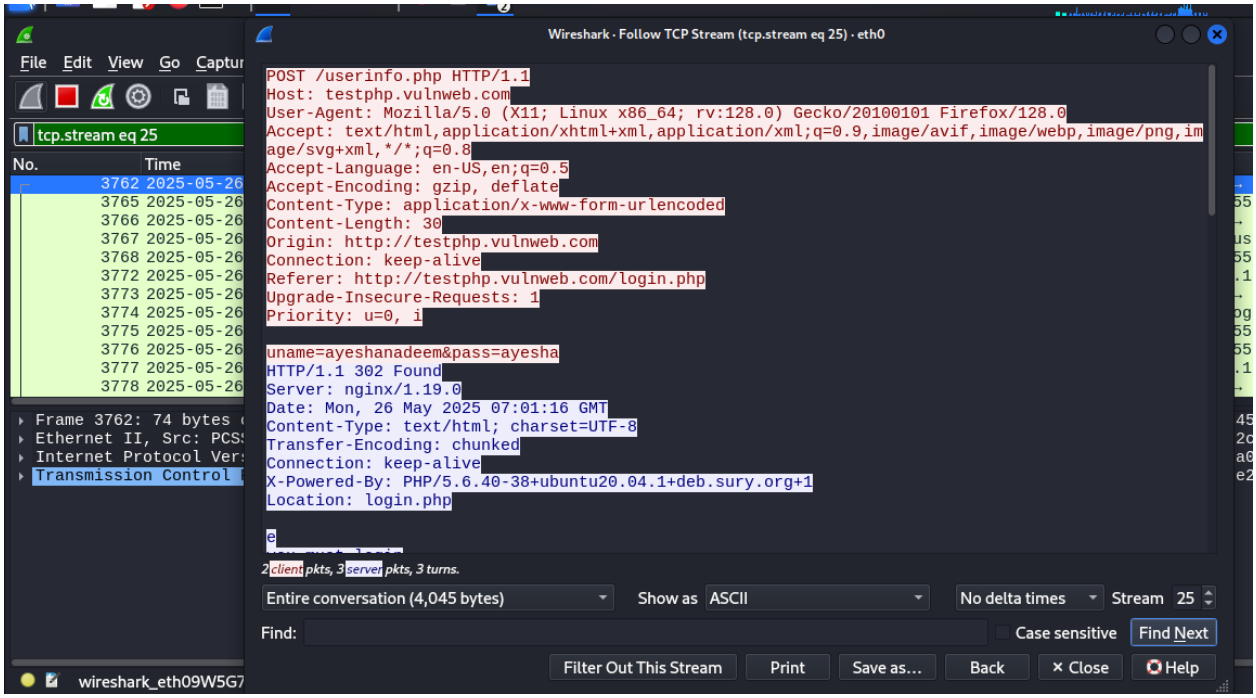


The conversation of IPv4 follows TCP, & HTTP communication protocols.

11. TCP Stream:

Right click on any **TCP packet**. Select **follow** and click on **TCP stream**

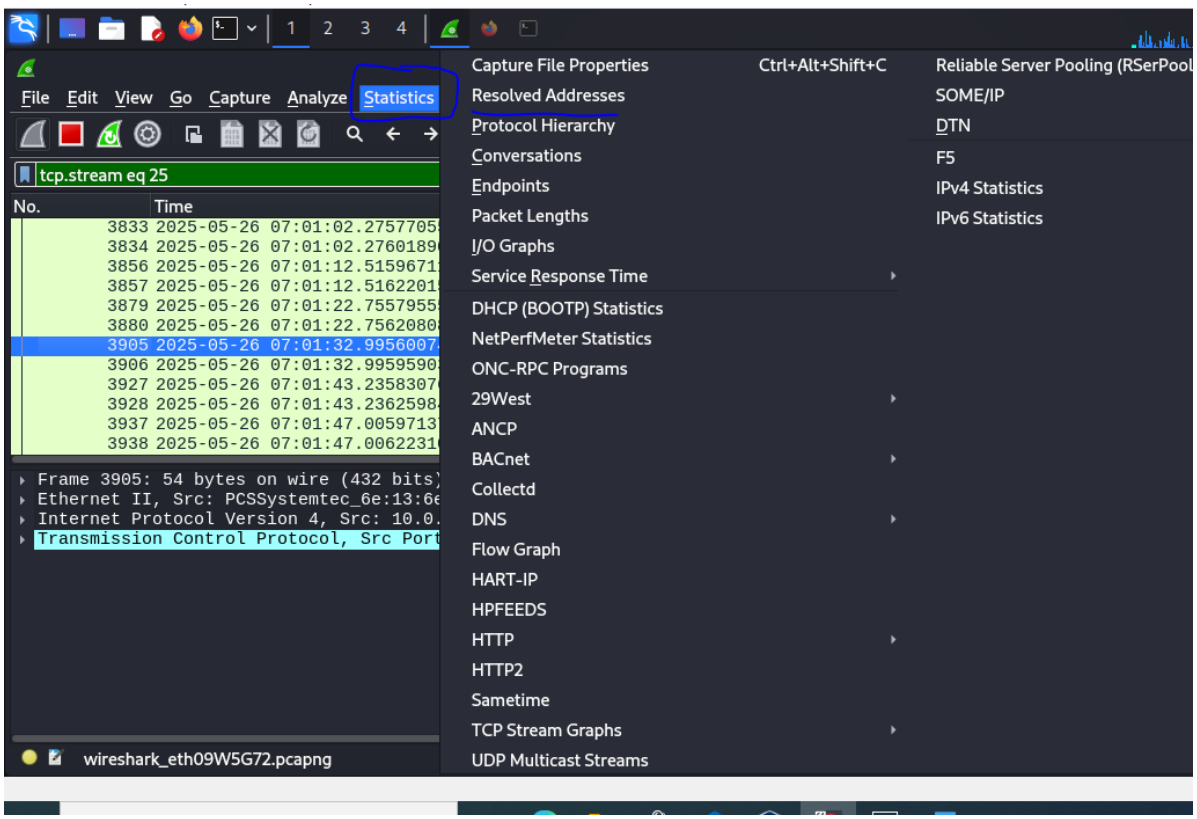


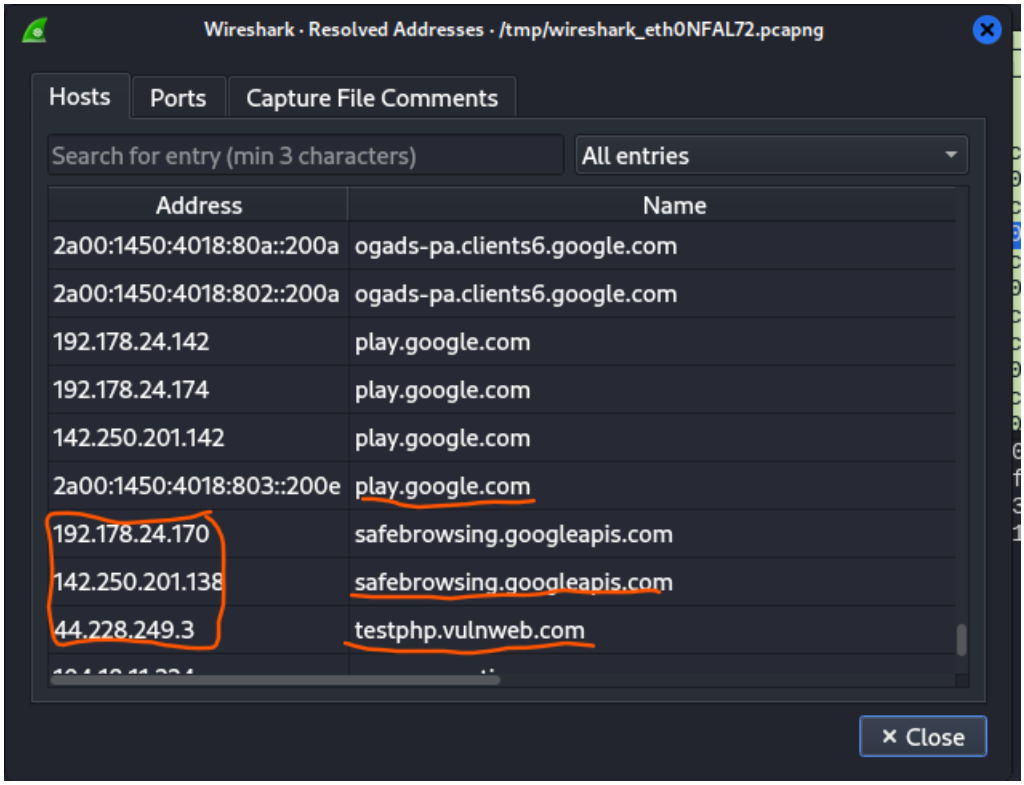


12. Statistic in Wireshark:

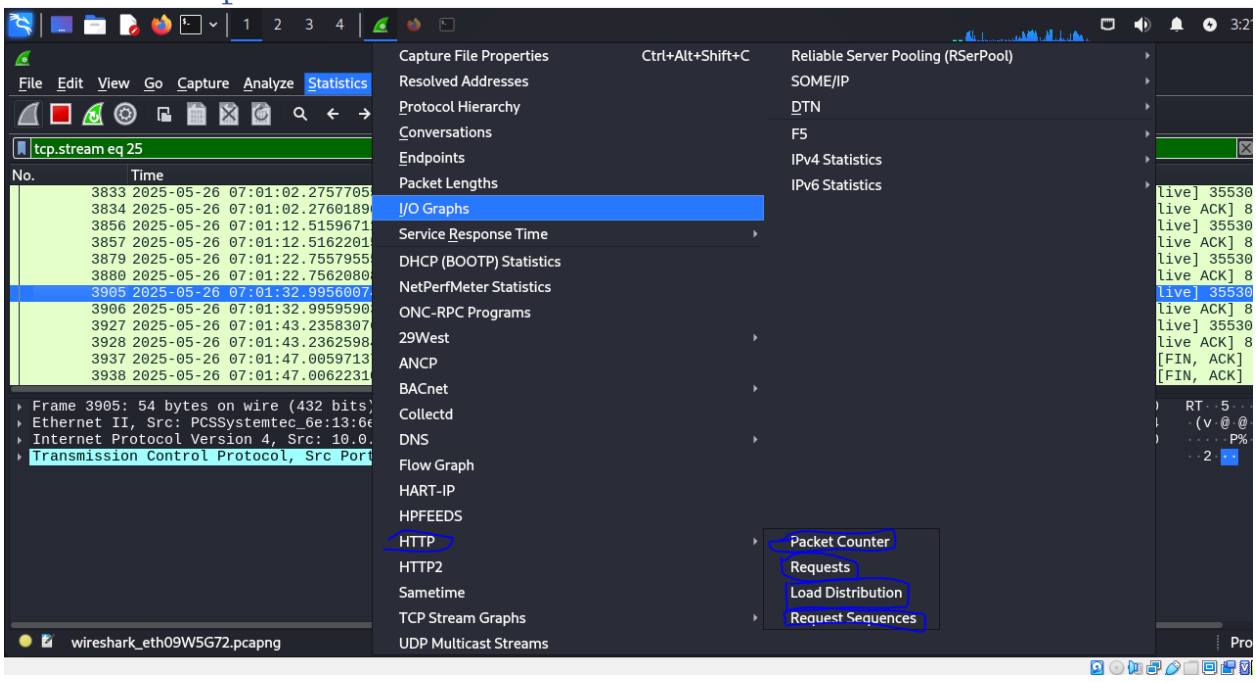
To see the list of IP's and DNS resolved:

➔ Go to statistic, select resolved addresses

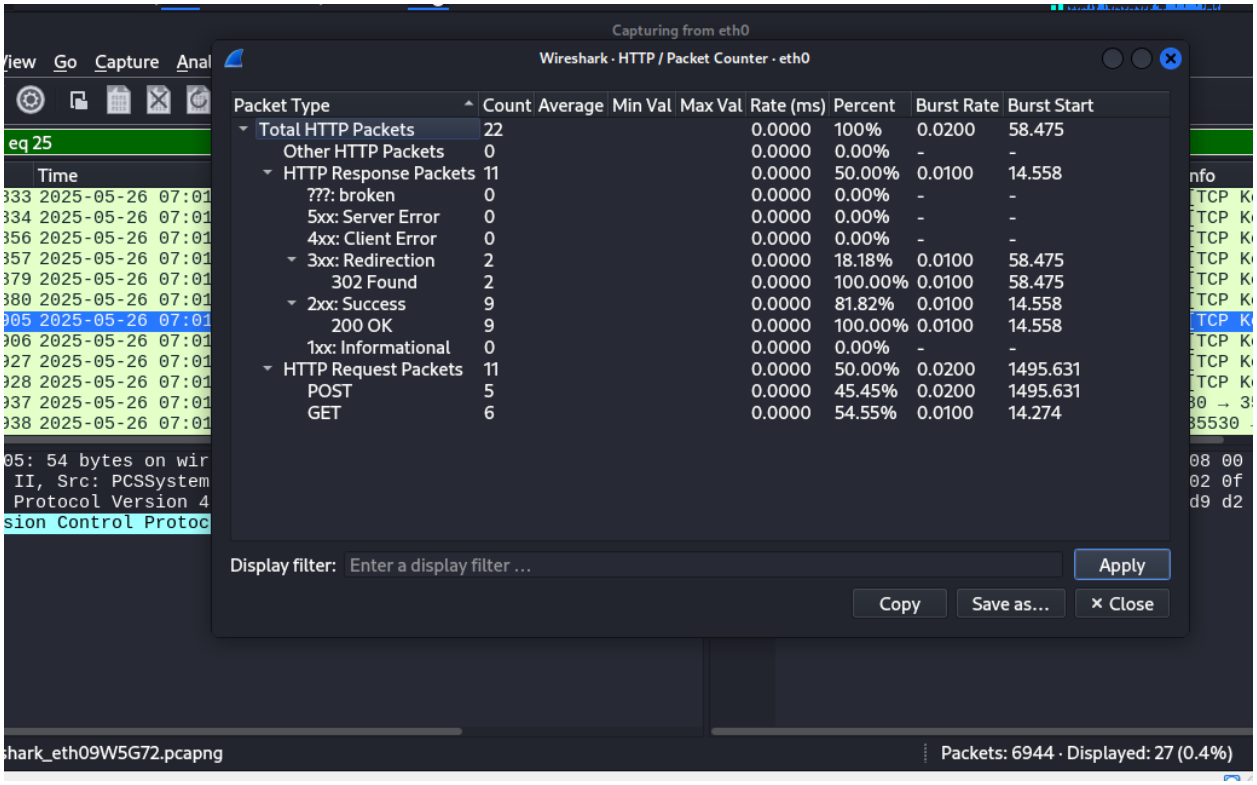




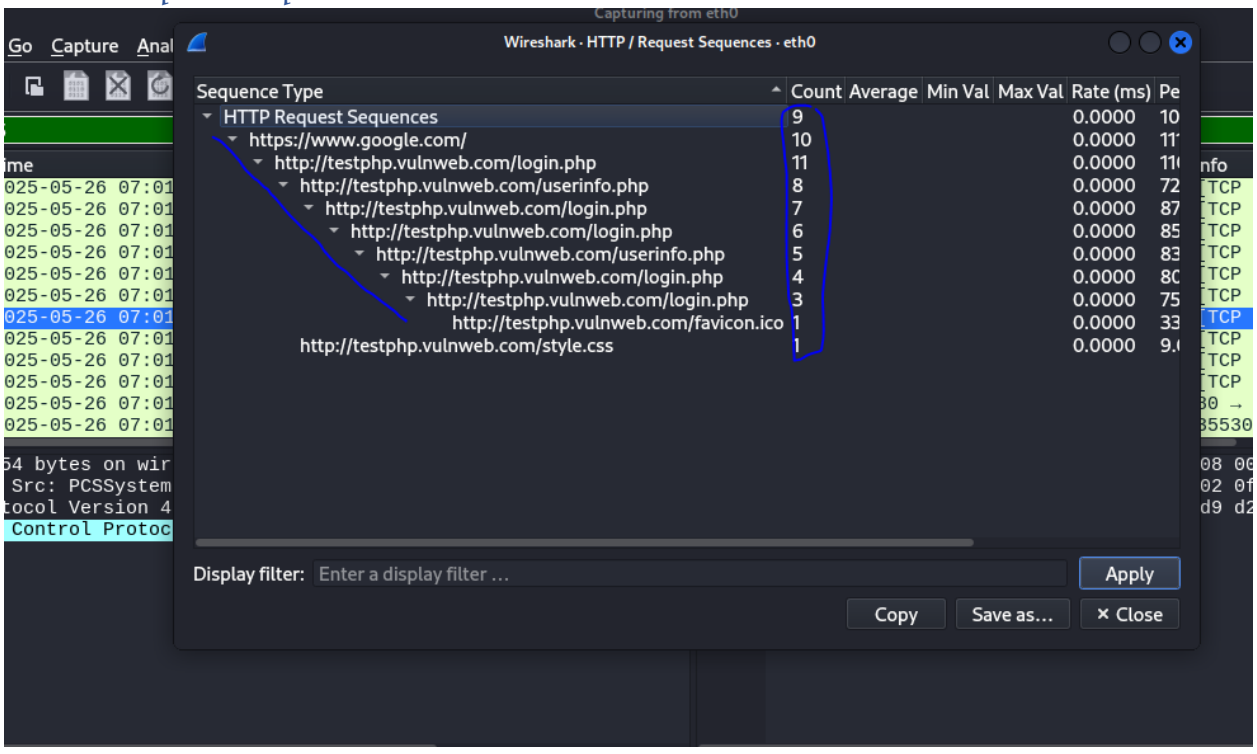
Statistic of http:



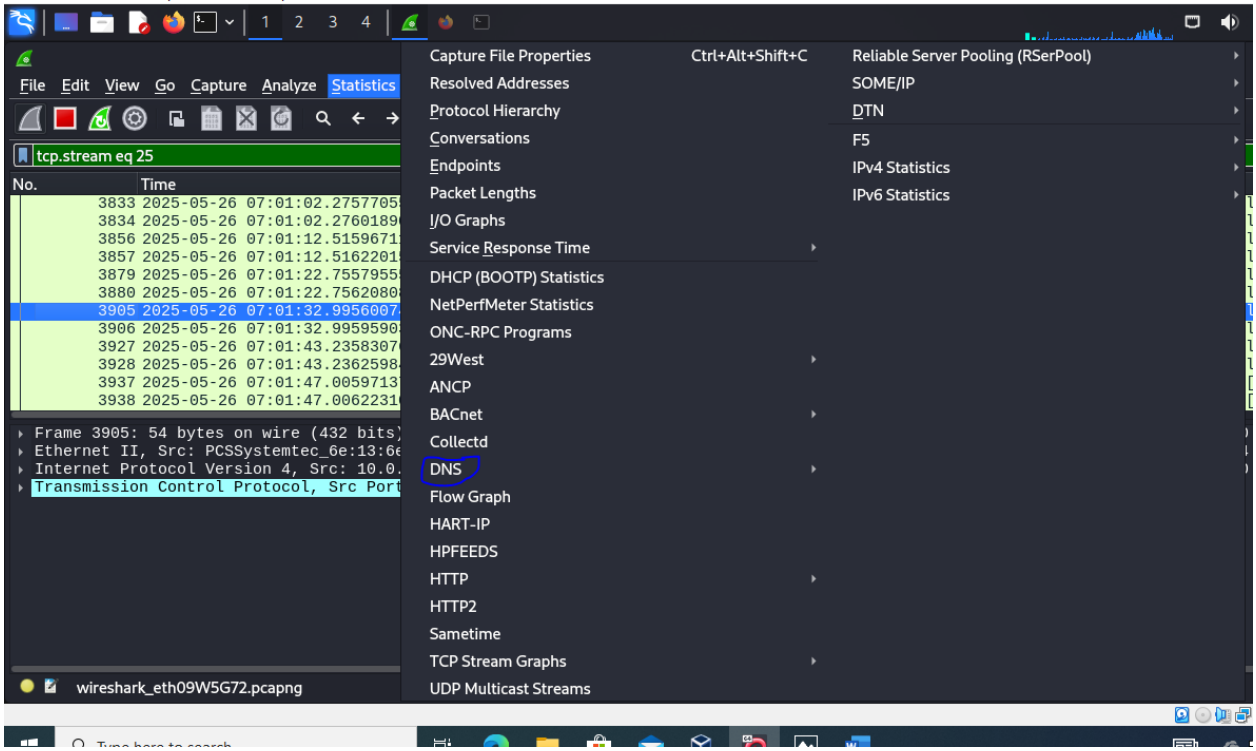
Packet counter:

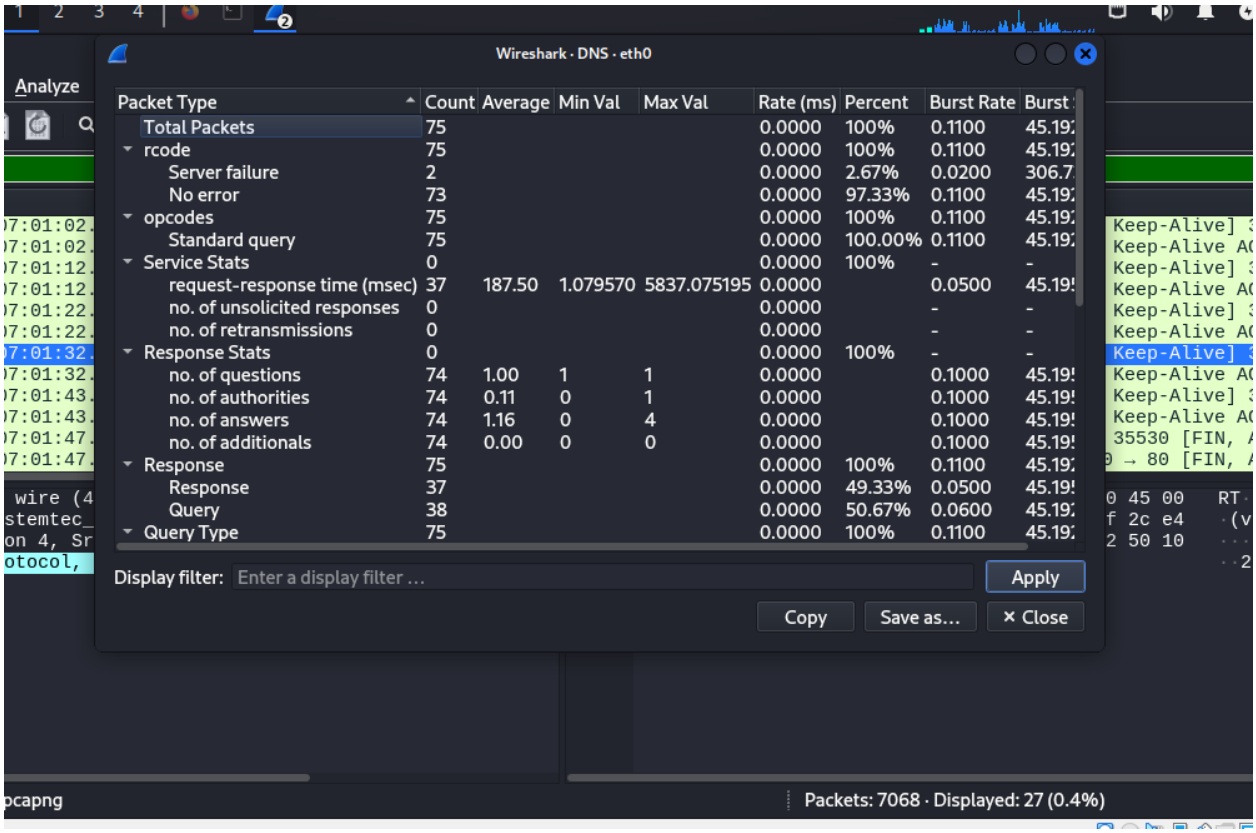


HTTP Request sequences:



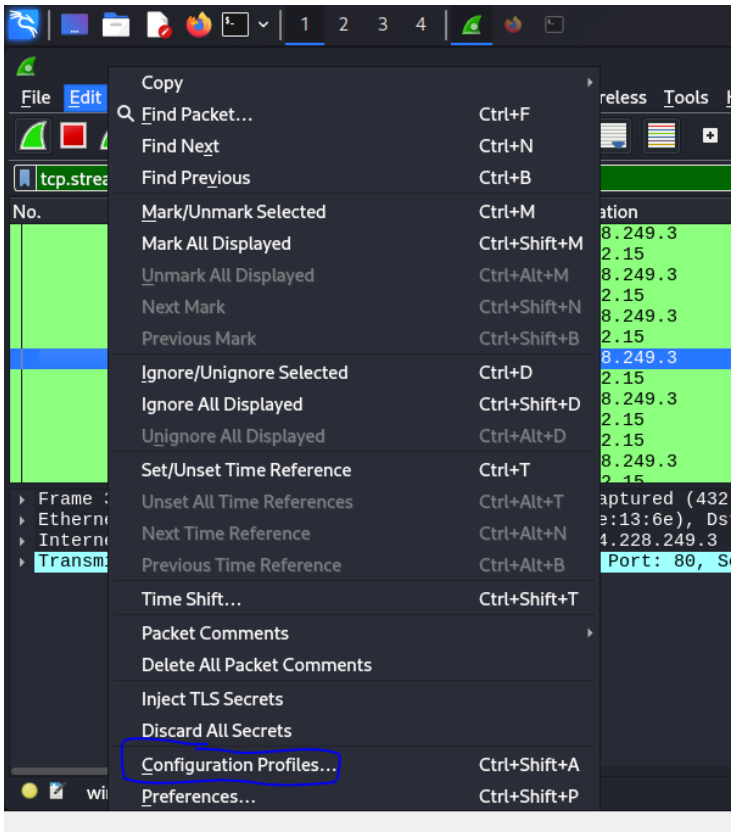
DNS Statistics:



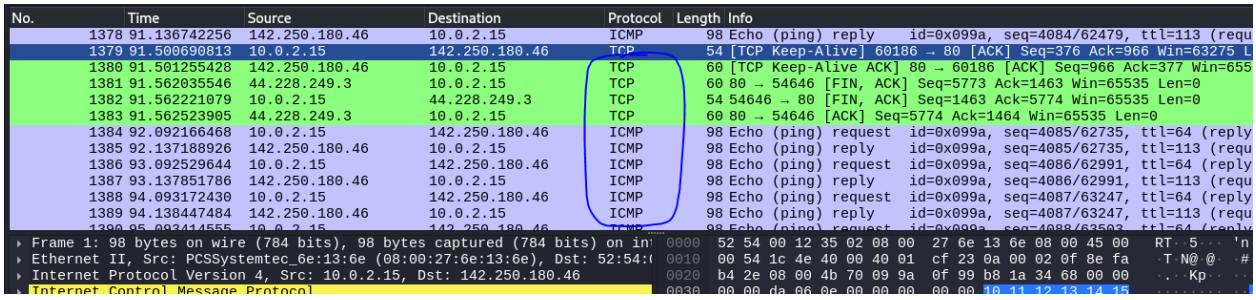


13. Configuration Profile:

➔ Go to Edit and Select Configuration file

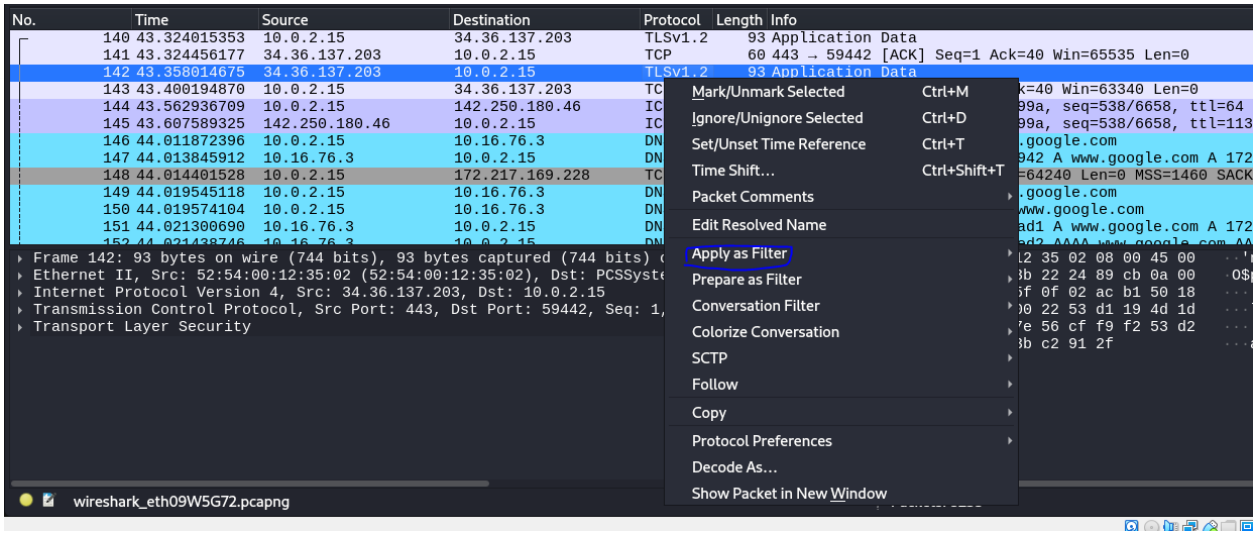


➔ Varriate packet into unique color to easy and quick analysis:



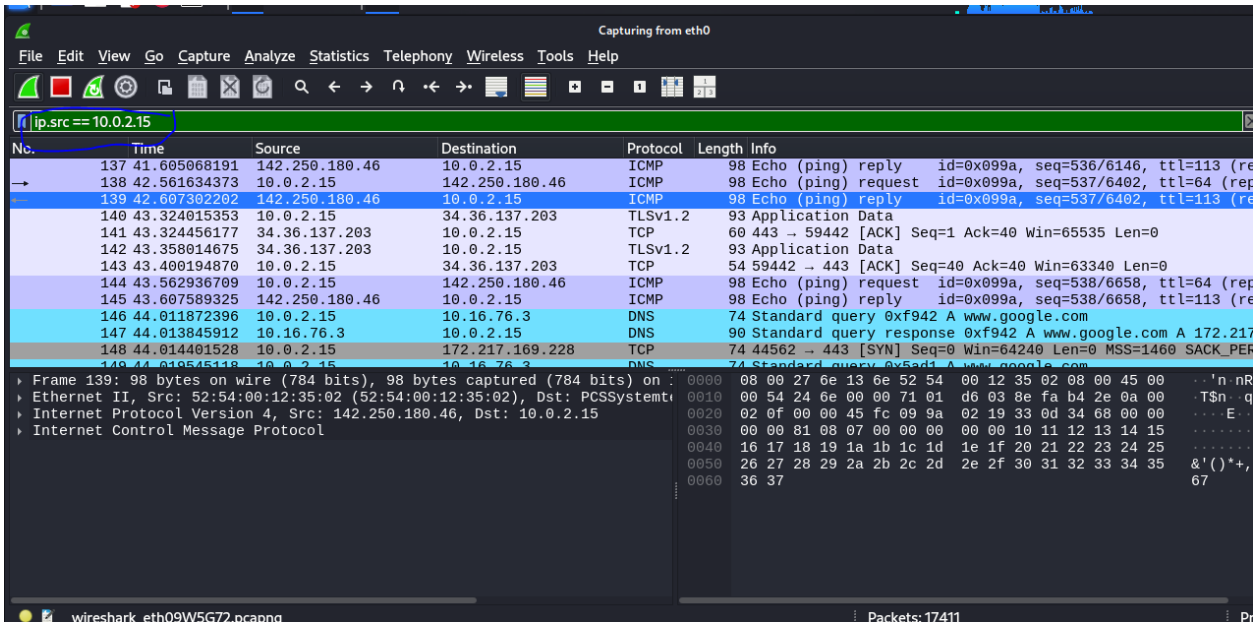
Target a specific IP:

➔ Right click on any packet and select apply as filter



So this is showing you the packets of 10.0.2.15

- ICMP packet in purple color
- DNS packet in blue color
- TLS packet in light purple color



A blend of Nmap & Wireshark:

What ping command actually do?

```
┌──(ayeshanadeem㉿ayeshanadeem)-[~]
└─$ ping cinesagerecommender.vercel.app
PING cinesagerecommender.vercel.app (216.198.79.1) 56(84) bytes of data.
64 bytes from 216.198.79.1: icmp_seq=7 ttl=245 time=93.7 ms
64 bytes from 216.198.79.1: icmp_seq=7 ttl=245 time=93.7 ms (DUP!)
64 bytes from 216.198.79.1: icmp_seq=7 ttl=245 time=119 ms (DUP!)
64 bytes from 216.198.79.1: icmp_seq=8 ttl=245 time=80.2 ms
64 bytes from 216.198.79.1: icmp_seq=9 ttl=245 time=71.8 ms
64 bytes from 216.198.79.1: icmp_seq=10 ttl=245 time=81.5 ms
64 bytes from 216.198.79.1: icmp_seq=11 ttl=245 time=122 ms
64 bytes from 216.198.79.1: icmp_seq=12 ttl=245 time=76.6 ms
```

The answer is, it only send an ICMP echo request just to check either host is up or down.

6224	2615.7609714…	192.168.104.87	216.198.79.1	ICMP	98	Echo (ping) request id=0x87fa, seq=115/29440, ttl=64 (reply in 6225)
6225	2615.8309966…	216.198.79.1	192.168.104.87	ICMP	98	Echo (ping) reply id=0x87fa, seq=115/29440, ttl=245 (request in 6224)
6226	2616.7622219…	192.168.104.87	216.198.79.1	ICMP	98	Echo (ping) request id=0x87fa, seq=116/29696, ttl=64 (reply in 6227)
6227	2616.8301203…	216.198.79.1	192.168.104.87	ICMP	98	Echo (ping) reply id=0x87fa, seq=116/29696, ttl=245 (request in 6226)
6228	2617.7638048…	192.168.104.87	216.198.79.1	ICMP	98	Echo (ping) request id=0x87fa, seq=117/29952, ttl=64 (reply in 6229)
6229	2617.8401835…	216.198.79.1	192.168.104.87	ICMP	98	Echo (ping) reply id=0x87fa, seq=117/29952, ttl=245 (request in 6228)
6230	2618.7654018…	192.168.104.87	216.198.79.1	ICMP	98	Echo (ping) request id=0x87fa, seq=118/30208, ttl=64 (reply in 6231)
6231	2618.8430260…	216.198.79.1	192.168.104.87	ICMP	98	Echo (ping) reply id=0x87fa, seq=118/30208, ttl=245 (request in 6230)
6232	2619.7676137…	192.168.104.87	216.198.79.1	ICMP	98	Echo (ping) request id=0x87fa, seq=119/30464, ttl=64 (reply in 6233)
6233	2619.8475248…	216.198.79.1	192.168.104.87	ICMP	98	Echo (ping) reply id=0x87fa, seq=119/30464, ttl=245 (request in 6232)

TCP three way handshake illustration using nmap command

```
┌──(ayeshanadeem㉿ayeshanadeem)-[~]
└─$ nmap cinesagerecommender.vercel.app
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-26 10:34 EDT
Nmap scan report for cinesagerecommender.vercel.app (216.198.79.65)
Host is up (0.086s latency).
Other addresses for cinesagerecommender.vercel.app (not scanned): 64.29.17.65
Not shown: 998 filtered tcp ports (no-response)
PORT    STATE SERVICE
80/tcp  open  http
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 8.44 seconds
```

Wireshark output:

28	3.146808270	192.168.104.87	216.198.79.193	ICMP	42	Echo (ping) request id=0xe2f0, seq=0/0, ttl=52 (no response found!)
29	3.147175181	192.168.104.87	216.198.79.193	TCP	58	58063 → 443 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
30	3.147372151	192.168.104.87	216.198.79.193	TCP	54	58063 → 80 [ACK] Seq=1 Ack=1 Win=1024 Len=0
31	3.147546414	192.168.104.87	216.198.79.193	ICMP	54	Timestamp request id=0x49a0, seq=0/0, ttl=46
32	3.229122753	216.198.79.193	192.168.104.87	TCP	60	443 → 58063 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1388
33	3.229186875	192.168.104.87	216.198.79.193	TCP	54	58063 → 443 [RST] Seq=1 Win=0 Len=0
34	3.267076103	192.168.104.87	192.168.104.112	DNS	87	Standard query 0x5c02 PTR 193.79.198.216.in-addr.arpa