

Task – 5 Capture and Analyze Network Traffic Using Wireshark.

- **Objective:** Capture live network packets and identify basic protocols and traffic types.
- **Tools:** Wireshark

Step-by-Step Procedure

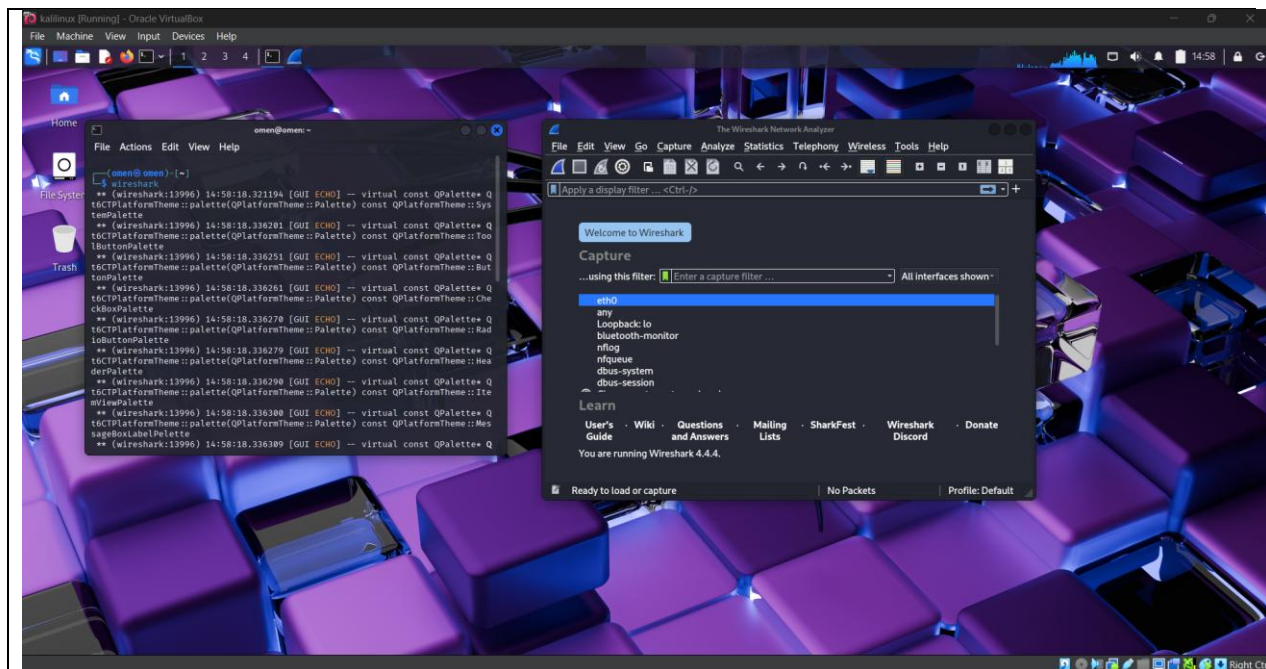
Note : I am performing this task on Kali Linux Virtual Machine.

1. Launch Wireshark

From Terminal:

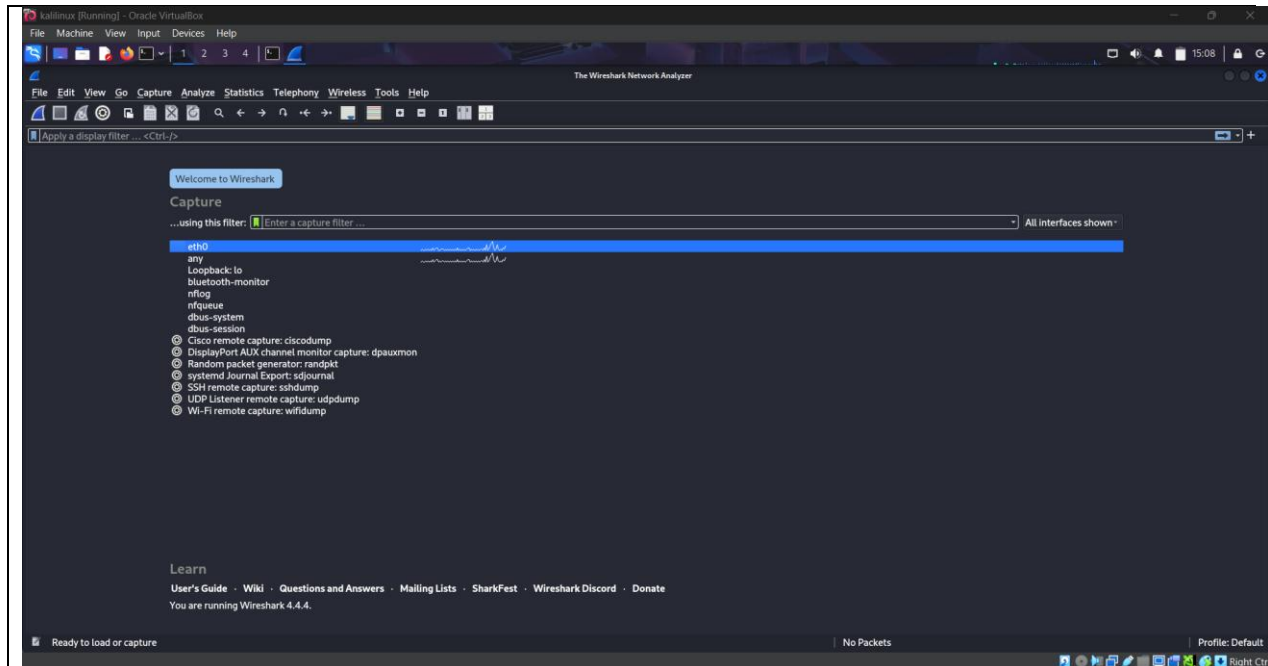
```
wireshark
```

Or navigate via GUI: **Applications > Sniffing & Spoofing > Wireshark**



2. Select the Active Network Interface

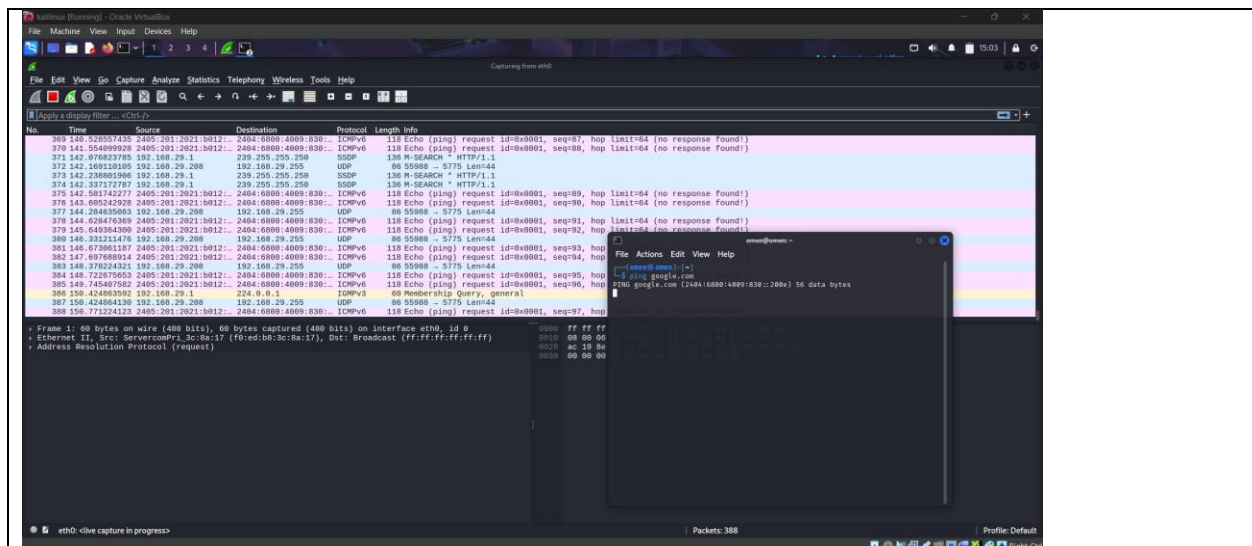
- Choose the active network interface (e.g., eth0, ens33, or wlan0).
- Click the blue **Start Capturing** button.



3. Generate Network Traffic

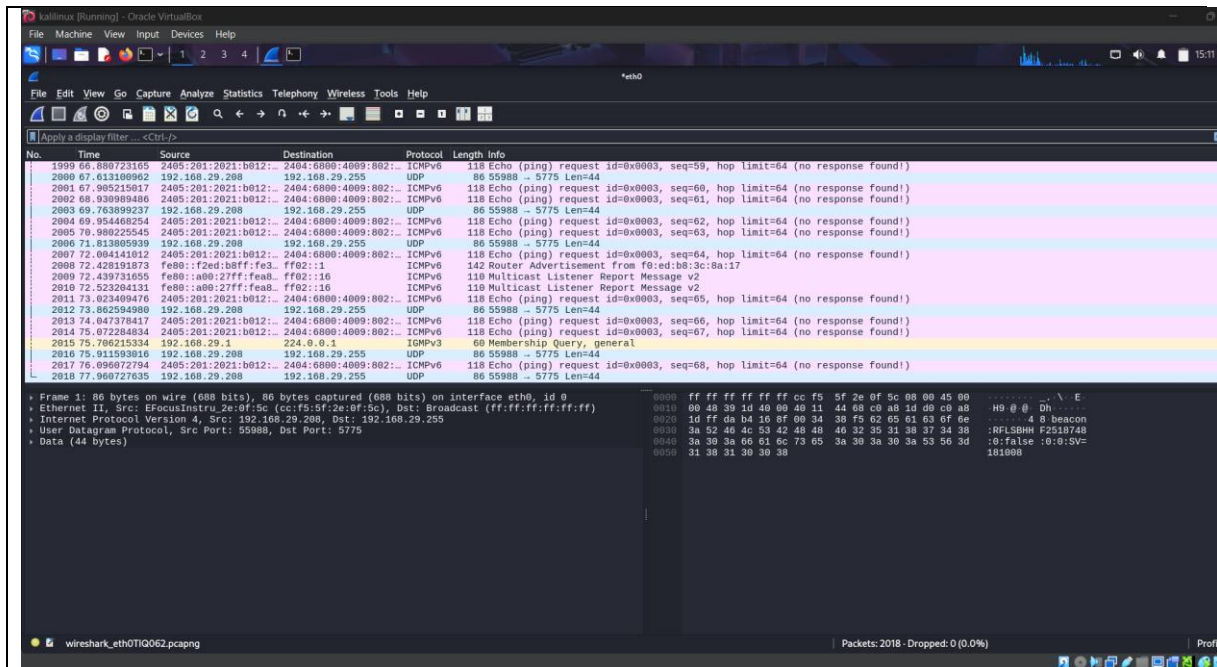
- Open a browser and visit a website (e.g., <http://example.com>).
- Or use terminal commands like:

ping google.com



4. Stop the Capture After One Minute

- Click the red **Stop** button in Wireshark.

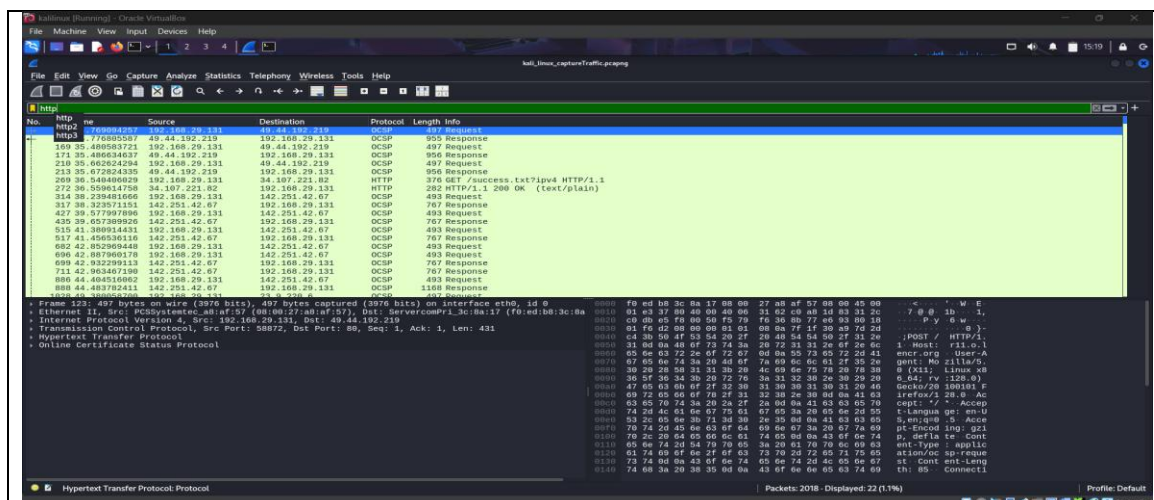


We have captured 2018 packets by ping google.com & visiting online websites.

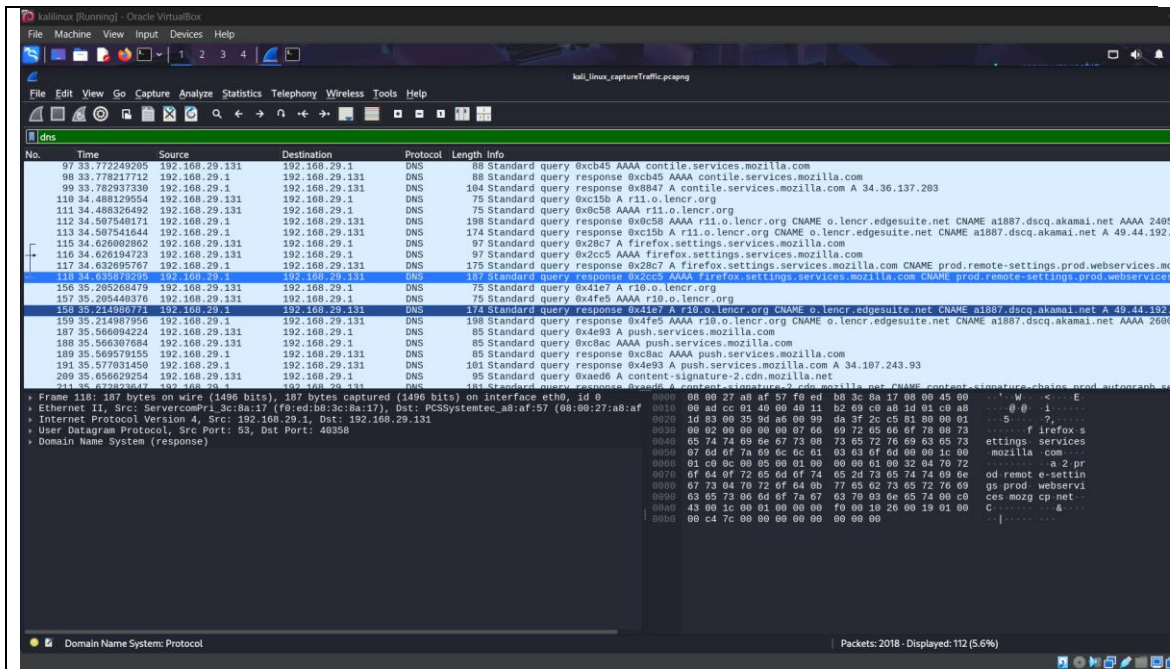
5. Apply Protocol Filters

Use the filter bar to view specific protocol traffic:

- http



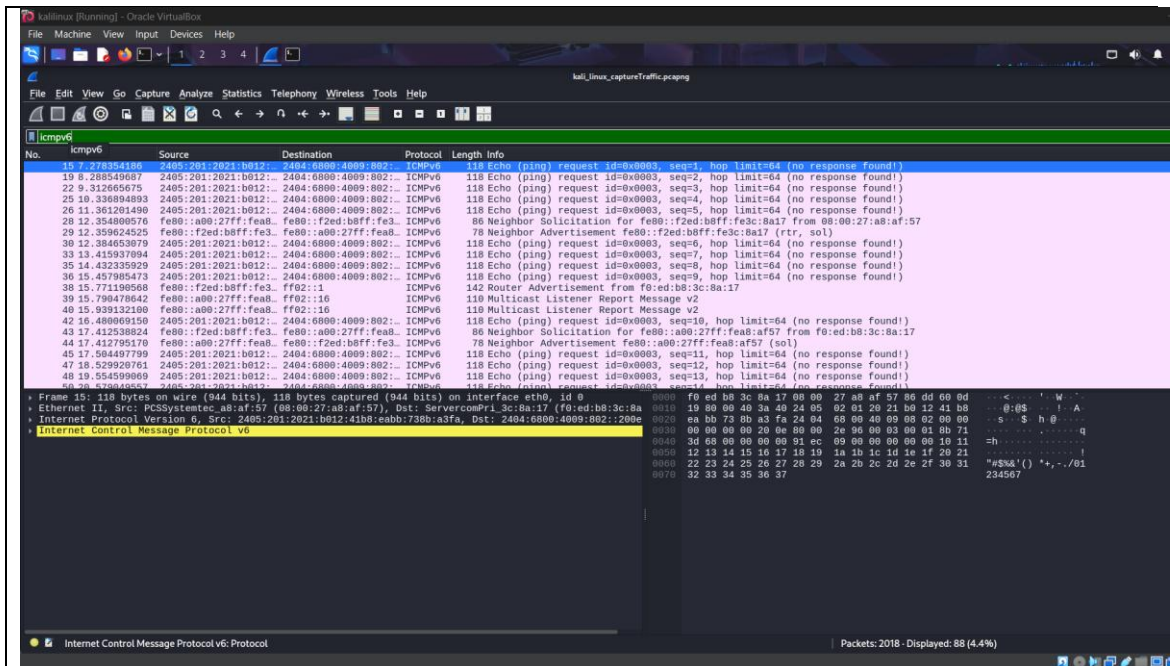
- dns



The image shows a Wireshark packet capture of DNS traffic. The top pane displays a list of 112 DNS packets. The middle pane shows the details of the selected packet (No. 112), including the Ethernet II header, Internet Protocol Version 4 header, and the Domain Name System (response) section. The bottom pane shows the raw packet data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
9	33.72248295	192.168.29.131	192.168.29.1	DNS	88	Standard query query 0xc045 AAAA contile.services.mozilla.com
98	33.78217112	192.168.29.1	192.168.29.131	DNS	88	Standard query response 0xc045 AAAA contile.services.mozilla.com
99	33.78297330	192.168.29.1	192.168.29.131	DNS	104	Standard query response 0x8847 A contile.services.mozilla.com A 34.36.137.203
110	34.488129554	192.168.29.131	192.168.29.1	DNS	75	Standard query 0xc15b A r11.o.lencr.org
111	34.488326492	192.168.29.131	192.168.29.1	DNS	75	Standard query 0xc058 AAAA r11.o.lencr.org
112	34.507548171	192.168.29.1	192.168.29.131	DNS	198	Standard query response 0xc058 AAAA r11.o.lencr.org CNAME o.lencr.edgesuite.net CNAME a1887.dscq.akamai.net AAAA 240f
113	34.507541644	192.168.29.1	192.168.29.131	DNS	174	Standard query response 0xc15b A r11.o.lencr.org CNAME o.lencr.edgesuite.net CNAME a1887.dscq.akamai.net A 49.44.192
115	34.626602962	192.168.29.131	192.168.29.1	DNS	97	Standard query 0x2b67 A firefox.settings.services.mozilla.com
116	34.626194723	192.168.29.131	192.168.29.1	DNS	97	Standard query 0x2c05 AAAA firefox.settings.services.mozilla.com
117	34.632695767	192.168.29.1	192.168.29.131	DNS	175	Standard query response 0x2b67 A firefox.settings.services.mozilla.com CNAME prod.remote-settings.prod.webservices.m
118	34.632695767	192.168.29.1	192.168.29.131	DNS	175	Standard query response 0x2c05 AAAA firefox.settings.services.mozilla.com CNAME prod.remote-settings.prod.webservices.m
150	35.295208479	192.168.29.131	192.168.29.1	DNS	75	Standard query 0x41e7 A r10.o.lencr.org
157	35.295440376	192.168.29.131	192.168.29.1	DNS	75	Standard query 0x4fe5 AAAA r10.o.lencr.org
158	35.295440376	192.168.29.131	192.168.29.1	DNS	174	Standard query response 0x41e7 A r10.o.lencr.org CNAME o.lencr.edgesuite.net CNAME a1887.dscq.akamai.net A 49.44.192
159	35.295440376	192.168.29.131	192.168.29.1	DNS	198	Standard query response 0x4fe5 AAAA r10.o.lencr.org CNAME o.lencr.edgesuite.net CNAME a1887.dscq.akamai.net AAAA 240f
187	35.566894224	192.168.29.131	192.168.29.1	DNS	85	Standard query 0x4e93 A push.services.mozilla.com
188	35.566894224	192.168.29.131	192.168.29.1	DNS	85	Standard query 0xc0ac AAAA push.services.mozilla.com
189	35.566894224	192.168.29.131	192.168.29.1	DNS	85	Standard query 0xc0ac AAAA push.services.mozilla.com
191	35.577631459	192.168.29.1	192.168.29.131	DNS	101	Standard query response 0x4e93 A push.services.mozilla.com A 34.107.243.93
209	35.656629254	192.168.29.131	192.168.29.1	DNS	95	Standard query 0xaed6 A content-signature-2.cdn.mozilla.net
211	35.672829247	192.168.29.1	192.168.29.131	DNS	181	Standard query response 0xaed6 A content-signature-2.cdn.mozilla.net CNAME content-signature-chains.prod.safesearch

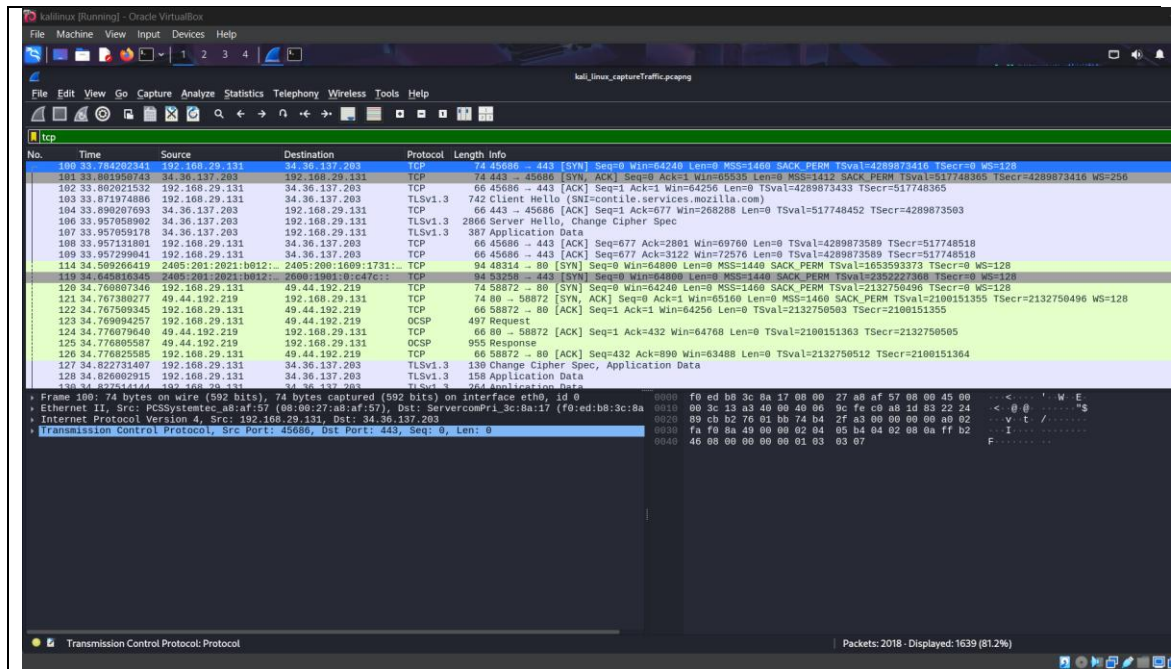
- icmp



The image shows a Wireshark packet capture of ICMPv6 traffic. The top pane displays a list of 88 ICMPv6 packets. The middle pane shows the details of the selected packet (No. 15), including the Ethernet II header, Internet Protocol Version 6 header, and the Internet Control Message Protocol v6 section. The bottom pane shows the raw packet data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
15	7.274364186	2404:6800:4009:802::	2404:6800:4009:802::	ICMPv6	118	Echo (ping) request id=0x0003, seq=1, hop limit=64 (no response found)
19	8.281540007	2404:6800:4009:802::	2404:6800:4009:802::	ICMPv6	118	Echo (ping) request id=0x0003, seq=2, hop limit=64 (no response found)
22	9.312665675	2404:6800:4009:802::	2404:6800:4009:802::	ICMPv6	118	Echo (ping) request id=0x0003, seq=3, hop limit=64 (no response found)
29	10.336894893	2404:6800:4009:802::	2404:6800:4009:802::	ICMPv6	118	Echo (ping) request id=0x0003, seq=4, hop limit=64 (no response found)
29	11.361261498	2404:6800:4009:802::	2404:6800:4009:802::	ICMPv6	118	Echo (ping) request id=0x0003, seq=5, hop limit=64 (no response found)
28	12.354808576	fe80::f2ed:b0ff:fe3c:8a17	fe80::f2ed:b0ff:fe3c:8a17	ICMPv6	86	Neighbor Solicitation for fe80::f2ed:b0ff:fe3c:8a17 from 08:00:27:a8:af:57
29	12.359624525	fe80::f2ed:b0ff:fe3c:8a17	fe80::f2ed:b0ff:fe3c:8a17	ICMPv6	78	Neighbor Advertisement fe80::f2ed:b0ff:fe3c:8a17 (rtr, sol)
30	12.386530879	2404:6800:4009:802::	2404:6800:4009:802::	ICMPv6	118	Echo (ping) request id=0x0003, seq=6, hop limit=64 (no response found)
33	13.415937894	2404:6800:4009:802::	2404:6800:4009:802::	ICMPv6	118	Echo (ping) request id=0x0003, seq=7, hop limit=64 (no response found)
35	14.432335929	2404:6800:4009:802::	2404:6800:4009:802::	ICMPv6	118	Echo (ping) request id=0x0003, seq=8, hop limit=64 (no response found)
36	15.457985473	2404:6800:4009:802::	2404:6800:4009:802::	ICMPv6	118	Echo (ping) request id=0x0003, seq=9, hop limit=64 (no response found)
38	15.771390568	fe80::f2ed:b0ff:fe3c:8a17	fe80::f2ed:b0ff:fe3c:8a17	ICMPv6	142	Router Advertisement from fe80::f2ed:b0ff:fe3c:8a17
39	15.790478842	fe80::f2ed:b0ff:fe3c:8a17	fe80::f2ed:b0ff:fe3c:8a17	ICMPv6	110	Multicast Listener Report Message v2
40	15.939132100	fe80::f2ed:b0ff:fe3c:8a17	fe80::f2ed:b0ff:fe3c:8a17	ICMPv6	118	Echo (ping) request id=0x0003, seq=10, hop limit=64 (no response found)
42	16.480609150	2404:6800:4009:802::	2404:6800:4009:802::	ICMPv6	118	Echo (ping) request id=0x0003, seq=11, hop limit=64 (no response found)
43	17.412538824	fe80::f2ed:b0ff:fe3c:8a17	fe80::f2ed:b0ff:fe3c:8a17	ICMPv6	86	Neighbor Solicitation for fe80::f2ed:b0ff:fe3c:8a17 from 08:00:27:a8:af:57
44	17.412795170	fe80::f2ed:b0ff:fe3c:8a17	fe80::f2ed:b0ff:fe3c:8a17	ICMPv6	78	Neighbor Advertisement fe80::f2ed:b0ff:fe3c:8a17 (sol)
45	17.504497199	2404:6800:4009:802::	2404:6800:4009:802::	ICMPv6	118	Echo (ping) request id=0x0003, seq=11, hop limit=64 (no response found)
47	18.529920761	2404:6800:4009:802::	2404:6800:4009:802::	ICMPv6	118	Echo (ping) request id=0x0003, seq=12, hop limit=64 (no response found)
48	18.554599869	2404:6800:4009:802::	2404:6800:4009:802::	ICMPv6	118	Echo (ping) request id=0x0003, seq=13, hop limit=64 (no response found)

- tcp



6. Identifying At Least 3 Protocols

- DNS – Domain resolution
- HTTP – Unencrypted web requests
- ICMP – Ping traffic

All the protocols have been identified and attached in above step.

7. Save the Packet Capture

Go to **File > Save As** and save the capture as a .pcap file.



8. Network Traffic Summary Report

• Protocols Identified:

1. **HTTP** – HyperText Transfer Protocol
2. **DNS** – Domain Name System
3. **TCP** – Transmission Control Protocol
4. **UDP** – User Datagram Protocol
5. **ICMPv6** – Internet Control Message Protocol for IPv6

• Brief Description of Packets:

- **HTTP:**
Used for web browsing. The packets captured indicate unencrypted web requests/responses, typically for non-secure sites or internal services.
- **DNS:**
Resolves human-readable domain names to IP addresses. Captured packets likely show queries to DNS servers (e.g., A or AAAA record lookups) and their responses.
- **TCP:**
A core transport protocol used for reliable communication. Observed in sessions involving protocols like HTTP and TLS, showing standard three-way handshakes and data transmission.
- **UDP:**
Lightweight, connectionless protocol used for faster communication. Seen in DNS requests and other services like QUIC or local network discovery.
- **ICMPv6:**
Used for diagnostics and control in IPv6 networks. Common packet types include echo requests/replies (similar to IPv4 ping), neighbor discovery, and router advertisement.

• Observations on Traffic Behavior:

- The presence of both **TCP and UDP** traffic shows a mix of reliable and fast, connectionless communication.
- **DNS traffic** indicates web activity, resolving hostnames before accessing websites.
- **HTTP traffic** confirms unencrypted browsing activity during the capture.
- **ICMPv6 packets** suggest IPv6 functionality is enabled and in use on the local network, with possible neighbor solicitation or ping events.
- The overall packet pattern reflects typical desktop or virtual machine internet usage, with background network services and user-initiated web activity.

Summary: How Wireshark Works and Its Importance

Wireshark is a powerful network protocol analyzer that captures and displays packets in real time. It provides deep visibility into network traffic, enabling users to inspect data at the microscopic level. Wireshark operates by putting a network interface into **promiscuous mode**, allowing it to intercept and log every packet that flows through the network interface.

How It Works:

- Captures packets from a selected network interface.
- Displays each packet with detailed information about protocol layers.
- Provides filtering, coloring, and decoding to make analysis easier.

Importance of Wireshark:

- **Network Troubleshooting:** Helps diagnose connectivity issues and latency problems.
- **Security Analysis:** Detects suspicious traffic, malware communications, or unauthorized connections.
- **Protocol Learning:** Valuable for understanding how protocols like TCP, HTTP, and DNS function.
- **Forensics:** Supports investigations by reconstructing and analyzing network activity.

In penetration testing environments like Kali Linux, Wireshark is an essential tool for analyzing both normal and malicious network behavior, making it invaluable for security professionals and network administrators.