Network Traffic Capture and Protocol Analysis using Wireshark

1. Introduction

The objective of this lab was to capture and analyse live network traffic using **Wireshark**, a network protocol analyser.
During the experiment, packets were captured while performing basic network activities such as browsing websites and running ping commands.
The captured data was then filtered and examined to identify different network protocols (like **ICMP**, **HTTP**, and **DNS**) and to understand how data travels across networks at the packet level.

This exercise helped in visualizing real-time communication between devices and provided practical exposure to protocol headers, fields, and flow of network communication.

2. Observations

- The captured packets contained multiple protocols such as **TCP**, **UDP**, **ICMP**, **HTTP**, **DNS**, and **ARP**.
- Using filters in Wireshark (icmp, http, dns), individual protocol packets were isolated and studied in detail.
- The **ICMP packets** corresponded to the ping request and reply messages to IP 8.8.8.8.
- The **DNS packets** showed name resolution requests for domains like wikipedia.org and example.com.
- The **HTTP packets** represented web communication between the browser and web servers.
- Under **Statistics → Protocol Hierarchy**, the most active protocol was **TCP**, followed by **DNS** and **ICMP**.
- No suspicious or abnormal traffic patterns were detected during the analysis.

3. Key Insights

- Each packet passes through multiple layers: Frame, Ethernet, IP, and transport/application layers (like ICMP, HTTP, DNS).
- The **TTL (Time to Live)** field helped understand how many hops a packet can travel before being discarded.
- The **custom filter** (ip.src == <your_IP> && (tcp.port == 80 || tcp.port == 443)) helped focus on traffic originating from the local system, mainly web traffic.
- Wireshark allows in-depth inspection of each packet, making it a powerful tool for **network troubleshooting, security analysis, and protocol study**.
- Understanding real network behaviour through packet captures gives better clarity on theoretical networking concepts like the **OSI and TCP/IP models**.

4. Conclusion

The Wireshark lab provided hands-on experience in monitoring and analysing live network traffic. By observing various protocol exchanges, we gained a practical understanding of how data moves across networks and how different protocols interact.
This experiment enhanced awareness of how tools like Wireshark can be used to **diagnose connectivity issues, analyse security threats, and study protocol performance**.

Overall, the activity effectively connected theoretical networking knowledge with real-world packet-level insights.