Network Traffic Capture and Protocol Analysis using Wireshark

Objective

The objective of this experiment is to capture and analyze live network traffic using Wireshark, identify various network protocols such as ICMP, DNS, and HTTP, and interpret important packet-level details like headers, fields, and communication patterns.

Tools Used

• Wireshark (for packet capture and analysis)
• Windows Command Prompt / Terminal (for ping and browsing activities)
• (Optional) tshark (for command-line extraction of HTTP requests)

Procedure

1. Opened Wireshark and selected the active Wi-Fi interface.
2. Started packet capture and simultaneously visited a few websites (example.com, wikipedia.org) and executed the command ping 8.8.8.8.
3. Stopped capture after about 2 minutes and saved the file as capture_lab1.pcap.
4. Applied protocol filters: icmp, http, and dns.
5. Selected one packet each from ICMP, HTTP, and DNS and noted details like Source IP, Destination IP, TTL, Packet Length, and Flags in a table.
6. Created a custom display filter to show only packets originating from my system using ports 80 or 443:
ip.src == <my_IP> && (tcp.port == 80 || tcp.port == 443)
7. Exported filtered packets as filtered_packets.pcap.

Observations

Protocol	Source IP	Destination IP	TTL	Packet Length	Flags
ICMP	192.168.x.x	8.8.8.8	64	98	Reply
DNS	192.168.x.x	8.8.8.8	128	74	Standard query
HTTP	192.168.x.x	142.250.xx.xx	64	590	ACK

Key Insights

• ICMP (Internet Control Message Protocol) is used for diagnostics such as the ping command.
• DNS (Domain Name System) translates human-readable domain names into IP addresses.
• HTTP (Hypertext Transfer Protocol) is used for web communication between browsers and servers.
• Most captured packets belonged to HTTP and DNS, indicating active web browsing during capture.
• No suspicious or abnormal traffic was observed.
• The analysis helped understand how multiple network layers interact in real-time communication.

Conclusion

This experiment provided practical exposure to packet capture and analysis using Wireshark. By applying filters, examining headers, and creating custom display filters, it became clear how different protocols operate and communicate within a network. Overall, this activity enhanced understanding of real-time data transmission and network behavior at the packet level.

(Optional: Bonus Task)

Using tshark, HTTP request URLs were extracted with the following command:

```
tshark -r capture_lab1.pcap -Y http.request -T fields -e http.host -e http.request.uri
```

This displayed the hostnames and requested URLs for each HTTP request.