# Wireshark Packet Capture Lab Report

## Introduction

This lab exercise demonstrates the use of Wireshark for capturing live network packets, filtering by protocols, and analyzing the traffic. The task follows a structured workflow from installation to exporting captured data as a .pcap file.

## 1. Install Wireshark

Wireshark was downloaded and installed from the official website (https://www.wireshark.org/). Default installation settings were used.

## 2. Start Capturing

Wireshark was launched, and the active network interface (Wi-Fi/Ethernet) was selected. Packet capturing was started by clicking on the shark fin icon.

## 3. Generate Traffic

To generate traffic, two activities were performed:  
- Opened a web browser and accessed 'example.com'.  
- Used the ping command (ping 8.8.8.8) from the terminal.

## 4. Stop Capture

After running the capture for approximately one minute, the process was stopped using the red stop button.

## 5. Filter Captured Packets

Protocol filters were applied in Wireshark to isolate specific traffic:  
- Filter: dns → Displays DNS queries and responses.  
- Filter: tcp → Displays TCP traffic including handshakes.  
- Filter: http/tls → Displays browsing-related traffic.

## 6. Identify Protocols

At least three different protocols were identified from the capture:  
1. DNS (Domain Name System) – used for hostname resolution.  
2. TCP (Transmission Control Protocol) – used for reliable transport.  
3. HTTPS – encrypted communication used for browsing.  
Additionally, ICMP packets were observed when using the ping command.

## 7. Export the Capture

The capture was exported as a .pcap file using the option File → Export Specified Packets. This file can be opened later in Wireshark for further analysis.

## 8. Findings and Analysis

During the analysis, the following observations were made:  
- DNS packets revealed queries for domain names such as example.com.  
- TCP packets showed the three-way handshake (SYN, SYN-ACK, ACK) between client and server.  
- HTTPS packets contained encrypted communication between the browser and the server.  
- ICMP packets displayed echo request and reply messages from the ping test.

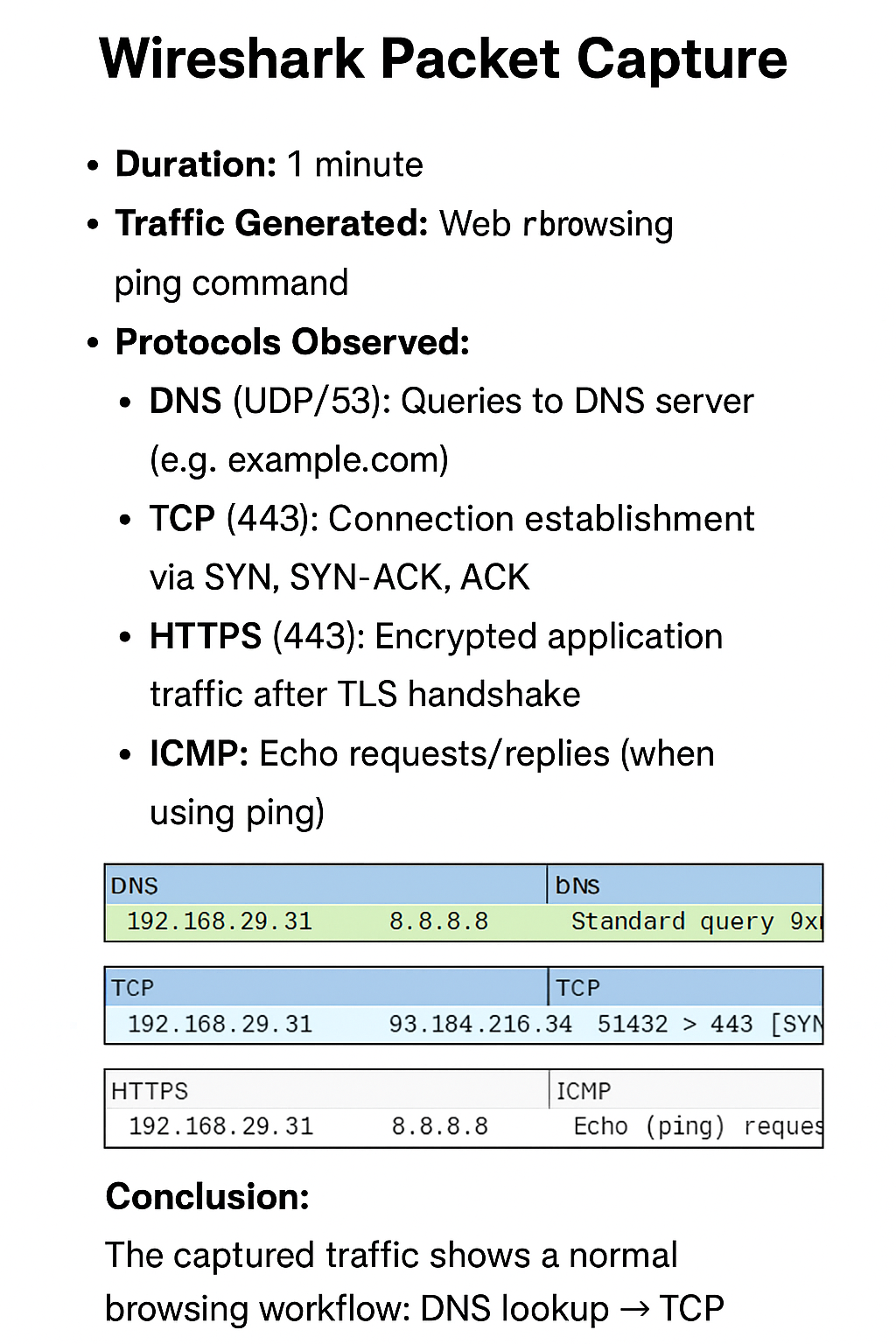
### Sample Packet Details

• DNS Request: Source = 192.168.29.31 → Destination = 8.8.8.8, Query = example.com

• TCP Handshake: Client Port = 51432 → Server Port = 443 (SYN, SYN-ACK, ACK)

• HTTPS Packet: Encrypted application data after TLS handshake

• ICMP Echo: Request and reply between client and server



## Conclusion

The Wireshark capture demonstrated the typical sequence of communication when accessing a website. It started with DNS resolution, followed by the TCP handshake, and then the exchange of encrypted HTTPS traffic. ICMP was observed from the ping test. These observations highlight how different protocols work together to enable secure and reliable communication over the Internet.