Karunya INSTITUTE OF TECHNOLOGY AND SCIENCES
(Declared as Deemed to be University under Sec.3 of the UGC Act, 1956)
MoE, UGC & AICTE Approved; NAAC Accredited A++
Karunya Nagar, Coimbatore - 641 114, Tamil Nadu, India.

# *A Project Report*

# *on*

# *Suspicious Network Traffic Monitoring and Alerting Using Wireshark*

Prepared by

Team Name: Cyber Champs

URK23DS3016 Mariya Beula A

Date of Submission: 31-01-2026

# Tactics and Techniques

**Tactics**

1. Packet Capturing Tactics

Wi-Fi adapter

Ethernet adapter
(depending on network used)

2.Use capture filters

Capture only http

Capture only tcp

Capture traffic from IP

**Techniques:**

**Display Filter Techniques (After Capturing)**

Used to analyze specific packets.

Common filters:

| **Purpose** | **Filter** |
|---|---|
| HTTP traffic | http |
| HTTPS | tls |
| DNS | dns |
| TCP only | tcp |
| UDP only | udp |
| Specific IP | ip.addr == 192.168.1.5 |
| Errors | tcp.analysis.flags |

**Procedure:**

This includes:

✓ Monitoring suspicious traffic
✓ Identifying DDoS patterns

✓ Detecting port scans
✓ Analyzing malware communication
✓ Understanding protocol misuse

**MITRE MAPPING:**

### 1. Reconnaissance – Network Scanning

**Wireshark Observation:**

• Many SYN packets
• One IP contacting many ports

**Filter:**

tcp.flags.syn == 1

### 2. Initial Access – Exploiting Services

**Wireshark Observation:**

• Unusual HTTP requests
• Repeated login attempts

**Filter:**

http

### 3. Command and Control (C2) Communication

**Wireshark Observation:**

• Unknown IP connections
• Repeated DNS queries to strange domains

**Filter:**

Dns

### 4. Impact – DDoS Attack

**Wireshark Observation:**

• High traffic spike
• Large number of packets from one IP

**Filter:**

ip.src == suspicious_ip

**5. Credential Access – Sniffing Data**

📜 **Wireshark Observation:**

• Plain text usernames/passwords in HTTP

📡 **Filter:** http.request.method == "POST"

**6. Exfiltration – Data Theft**

📜 Wireshark Observation:

• Large file transfers
• Unusual outbound traffic

📡 Filter:

ftp || http

**Simple MITRE Mapping Table**

| Wireshark Finding | MITRE Tactic | MITRE Technique |
|---|---|---|
| Port scanning | Reconnaissance | T1046 |
| Web exploit traffic | Initial Access | T1190 |
| Strange DNS/C2 | Command & Control | T1071 |
| Traffic flooding | Impact | T1498 |
| Packet sniffing | Credential Access | T1040 |
| File uploads | Exfiltration | T1048 |

# Project-based Assessment – Report

## *Introduction and Objective*

**Clear Description of Wireshark:**

- **Wireshark** is a free, open-source network protocol analyzer that allows users to see all the traffic on a network in real time. Originally known as "Ethereal," Wireshark has evolved into an essential tool for network administrators, security analysts, and digital forensic professionals.
- Its primary function is to capture data packets traveling across a network and display them in a highly detailed, human-readable format. With support for over 1,000 protocols, Wireshark is especially valuable for troubleshooting, analyzing, and securing networks.

**Purpose and Application in Network Security, Cybersecurity, and Digital Forensics:**

- **Network Security**: Wireshark enables network administrators to monitor traffic, detect anomalies, and troubleshoot network issues. By analyzing network packets, administrators can identify traffic that could indicate security threats, configuration issues, or performance bottlenecks.
- **Cybersecurity**: Cybersecurity analysts use Wireshark to detect malicious activities, such as unusual traffic patterns, unauthorized access attempts, or signatures of malware. Its filtering capabilities allow analysts to isolate suspicious packets and identify vulnerabilities within the network.
- **Digital Forensics**: In forensic investigations, Wireshark helps reconstruct network events by analyzing past packet captures. This can be crucial in understanding the timeline and methods of a security breach, identifying involved devices, and collecting evidence for incident response.

**Primary Functionalities of Wireshark**:

- **Packet Capture**: Wireshark captures live traffic over Ethernet, WiFi, and other network types, providing a real-time view of network activity. Capturing data packets is essential for analyzing network interactions and detecting anomalies.
- **Protocol Analysis**: Wireshark supports the decoding and analysis of numerous protocols (e.g., TCP, HTTP, DNS, SSL). It presents the structure and payload of each protocol layer, enabling users to investigate specific protocol interactions.
- **Filtering Capabilities**: Wireshark allows precise filtering of network traffic with capture and display filters. Capture filters restrict the data collected, while display filters narrow down visible data, making it easier to focus on relevant packets.
- **Traffic Decryption**: Wireshark supports decryption for certain protocols, such as SSL/TLS, given the required encryption keys. This is essential for analyzing encrypted

traffic in a secure network environment.

- **Statistical Analysis and Visualization**: Wireshark provides various statistical tools, including I/O graphs, endpoint analysis, and protocol hierarchy, allowing users to visualize traffic patterns and identify traffic types that dominate network usage.

**Applications in Different Scenarios**:

- **Network Troubleshooting**: Wireshark is invaluable for diagnosing network issues like packet loss, latency, and configuration errors. Administrators can identify the root cause of issues by analyzing packet flows and interactions.
- **Security Threat Detection**: By examining unusual traffic, Wireshark can help detect intrusions, DDoS attacks, and data exfiltration attempts. For example, identifying repeated access attempts from unknown IPs can indicate potential brute-force attacks.
- **Forensic Analysis of Data Breaches**: In forensic scenarios, Wireshark is used to trace back to the origin of data breaches, reconstruct compromised sessions, and identify affected assets.

## *Installation and Setup*

**Installation Process:**

**Step-by-Step Guide:**

- **Windows:** Download the installer from the Wireshark website, run the executable, and follow the setup wizard. The process includes optional installation of WinPcap/Npcap drivers for capturing live traffic.
- **macOS:** Download and install via Homebrew (brew install wireshark) or from the Wireshark website.
- **Linux:** Install using the package manager (e.g., sudo apt install wireshark for Debian/Ubuntu).

**Challenges Encountered:**

- **Permission Issues:** In Linux, capturing network packets typically requires root privileges or additional configuration to permit non-root users.
- **WiFi Adapter Compatibility:** For WiFi traffic analysis, an adapter supporting monitor mode may be necessary. This requirement can limit packet capture capabilities on devices without compatible adapters.

**Configuration and Environment Setup:**

- **Setting Up Monitor Mode for WiFi Capture:** On systems with compatible

wireless adapters, enable monitor mode to capture all WiFi traffic. This is done through commands like airmon-ng start wlan0 on Linux.

- **Configuring Capture Filters:** Specify capture filters (e.g., capturing only HTTP traffic) to limit traffic capture to relevant data.
- **Testing the Setup:** Ensure that the installation was successful by running sample captures and verifying the packets display correctly.

***Feature Exploration***

**Detailed Feature Analysis**:

- **Packet Capture**: Describe how Wireshark's packet capture functionality works, including configuring the capture interface, setting capture filters, and initiating/stopping captures.
- **Protocol Decoding**: Explain how Wireshark decodes and displays protocol information, breaking down packet headers, payloads, and fields, allowing for in-depth protocol analysis.
- **Filtering Capabilities**: Discuss Wireshark's display and capture filters, demonstrating their syntax (e.g., ip.addr == 192.168.1.1 to filter specific IP traffic).
- **Traffic Decryption**: Outline how to enable decryption for protocols like SSL/TLS, detailing how to add decryption keys and view decrypted packets.
- **Statistical Analysis Tools**: Describe statistical features, including I/O graphs, protocol hierarchy, and endpoint conversations, and how these are used to visualize traffic trends and detect anomalies.

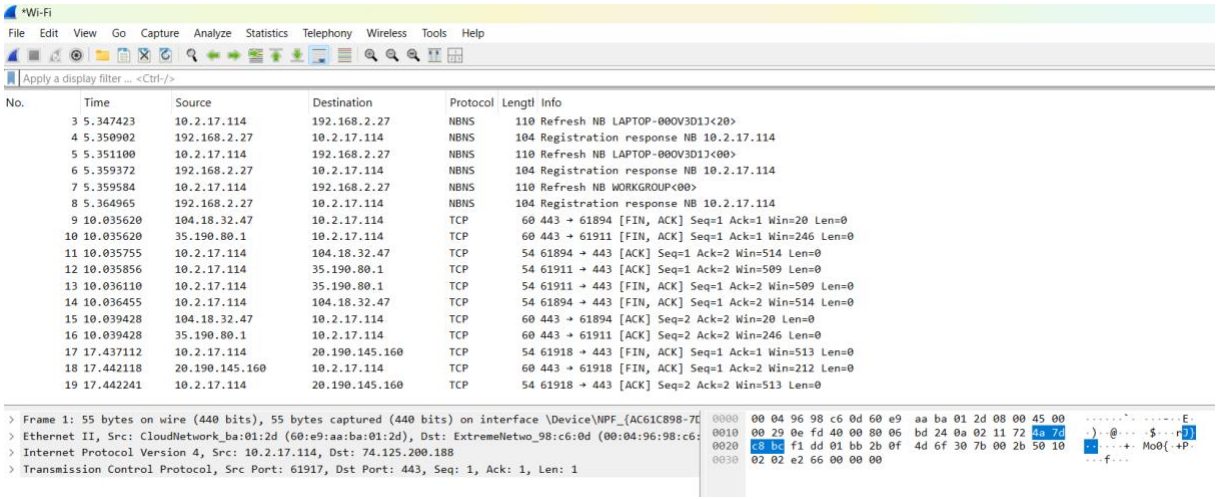
**Practical Examples:**

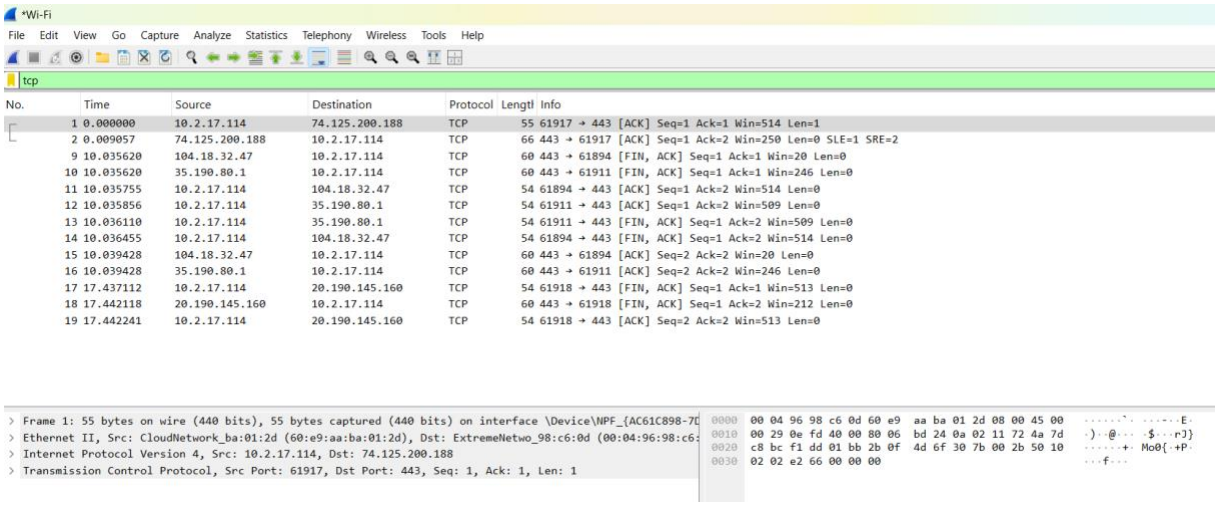
- **HTTP Capture**: Show how to filter HTTP traffic to observe request and response details, such as status codes or specific endpoints accessed.
- **DNS Analysis**: Demonstrate isolating DNS queries to identify patterns, such as high-frequency requests to a particular domain, which could indicate malware communication.
- **SSL/TLS Decryption**: Example of decrypting HTTPS traffic (if SSL keys are available), showing sensitive information contained within secured transmissions.

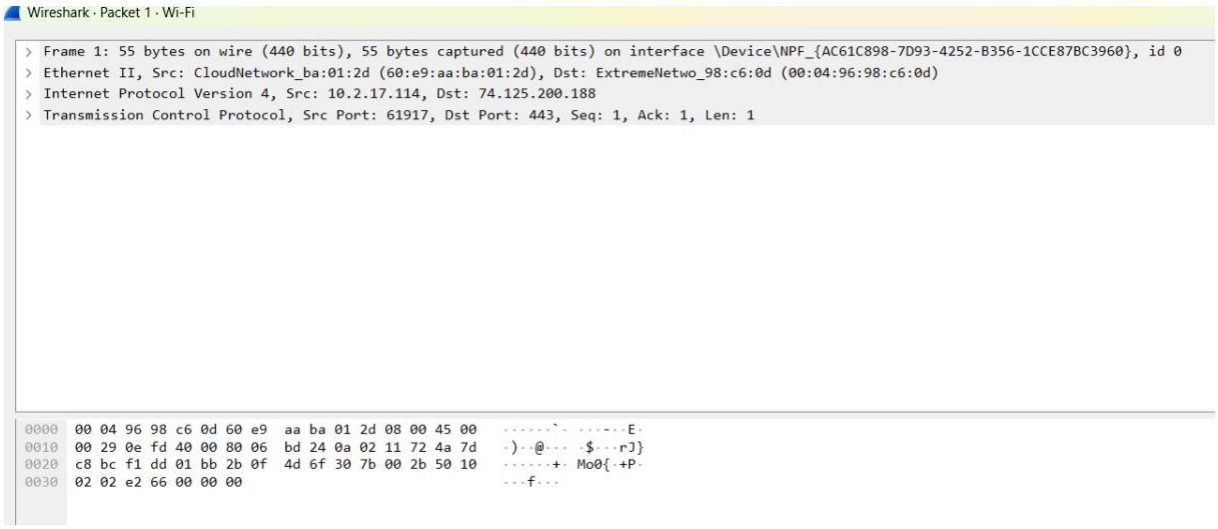
**Comparative Analysis with Similar Tools:**

- Compare Wireshark with **tcpdump** (a command-line packet analyzer) and **Nmap** (a network scanning tool), focusing on Wireshark's graphical interface, protocol support, and detailed packet views.
- Emphasize Wireshark's advantage for in-depth analysis compared to tcpdump's command-line output and its focus on packet capture over Nmap's broader network scanning features.

# Visual Aids:


*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Lengtl Info
3 5.347423 10.2.17.114 192.168.2.27 NBNS 110 Refresh NB LAPTOP-00OV3D1J<20>
4 5.350902 192.168.2.27 10.2.17.114 NBNS 104 Registration response NB 10.2.17.114
5 5.351100 10.2.17.114 192.168.2.27 NBNS 110 Refresh NB LAPTOP-00OV3D1J<00>
6 5.359372 192.168.2.27 10.2.17.114 NBNS 104 Registration response NB 10.2.17.114
7 5.359584 10.2.17.114 192.168.2.27 NBNS 110 Refresh NB WORKGROUP<00>
8 5.364965 192.168.2.27 10.2.17.114 NBNS 104 Registration response NB 10.2.17.114
9 10.035620 104.18.32.47 10.2.17.114 TCP 60 443 → 61894 [FIN, ACK] Seq=1 Ack=1 Win=20 Len=0
10 10.035620 35.190.80.1 10.2.17.114 TCP 60 443 → 61911 [FIN, ACK] Seq=1 Ack=1 Win=246 Len=0
11 10.035755 10.2.17.114 104.18.32.47 TCP 54 61894 → 443 [ACK] Seq=1 Ack=2 Win=514 Len=0
12 10.035856 10.2.17.114 35.190.80.1 TCP 54 61911 → 443 [ACK] Seq=1 Ack=2 Win=509 Len=0
13 10.036110 10.2.17.114 35.190.80.1 TCP 54 61911 → 443 [FIN, ACK] Seq=1 Ack=2 Win=509 Len=0
14 10.036455 10.2.17.114 104.18.32.47 TCP 54 61894 → 443 [FIN, ACK] Seq=1 Ack=2 Win=514 Len=0
15 10.039428 104.18.32.47 10.2.17.114 TCP 60 443 → 61894 [ACK] Seq=2 Ack=2 Win=20 Len=0
16 10.039428 35.190.80.1 10.2.17.114 TCP 60 443 → 61911 [ACK] Seq=2 Ack=2 Win=246 Len=0
17 17.437112 10.2.17.114 20.190.145.160 TCP 54 61918 → 443 [FIN, ACK] Seq=1 Ack=1 Win=513 Len=0
18 17.442118 20.190.145.160 10.2.17.114 TCP 60 443 → 61918 [FIN, ACK] Seq=1 Ack=2 Win=212 Len=0
19 17.442241 10.2.17.114 20.190.145.160 TCP 54 61918 → 443 [ACK] Seq=2 Ack=2 Win=513 Len=0
Frame 1: 55 bytes on wire (440 bits), 55 bytes captured (440 bits) on interface \Device\NPF_{AC61C898-7D
Ethernet II, Src: CloudNetwork_ba:01:2d (60:e9:aa:ba:01:2d), Dst: ExtremeNetwo_98:c6:0d (00:04:96:98:c6:
Internet Protocol Version 4, Src: 10.2.17.114, Dst: 74.125.200.188
Transmission Control Protocol, Src Port: 61917, Dst Port: 443, Seq: 1, Ack: 1, Len: 1
0000 00 04 96 98 c6 0d 60 e9 aa ba 01 2d 08 00 45 00
0010 00 29 0e fd 40 00 80 06 bd 24 0a 02 11 72 4a 7d
0020 c8 bc f1 dd 01 bb 2b 0f 4d 6f 30 7b 00 2b 50 10
0030 02 02 e2 66 00 00 00


*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp
No. Time Source Destination Protocol Lengtl Info
1 0.000000 10.2.17.114 74.125.200.188 TCP 55 61917 → 443 [ACK] Seq=1 Ack=1 Win=514 Len=1
2 0.009057 74.125.200.188 10.2.17.114 TCP 66 443 → 61917 [ACK] Seq=1 Ack=2 Win=250 Len=0 SLE=1 SRE=2
9 10.035620 104.18.32.47 10.2.17.114 TCP 60 443 → 61894 [FIN, ACK] Seq=1 Ack=1 Win=20 Len=0
10 10.035620 35.190.80.1 10.2.17.114 TCP 60 443 → 61911 [FIN, ACK] Seq=1 Ack=1 Win=246 Len=0
11 10.035755 10.2.17.114 104.18.32.47 TCP 54 61894 → 443 [ACK] Seq=1 Ack=2 Win=514 Len=0
12 10.035856 10.2.17.114 35.190.80.1 TCP 54 61911 → 443 [ACK] Seq=1 Ack=2 Win=509 Len=0
13 10.036110 10.2.17.114 35.190.80.1 TCP 54 61911 → 443 [FIN, ACK] Seq=1 Ack=2 Win=509 Len=0
14 10.036455 10.2.17.114 104.18.32.47 TCP 54 61894 → 443 [FIN, ACK] Seq=1 Ack=2 Win=514 Len=0
15 10.039428 104.18.32.47 10.2.17.114 TCP 60 443 → 61894 [ACK] Seq=2 Ack=2 Win=20 Len=0
16 10.039428 35.190.80.1 10.2.17.114 TCP 60 443 → 61911 [ACK] Seq=2 Ack=2 Win=246 Len=0
17 17.437112 10.2.17.114 20.190.145.160 TCP 54 61918 → 443 [FIN, ACK] Seq=1 Ack=1 Win=513 Len=0
18 17.442118 20.190.145.160 10.2.17.114 TCP 60 443 → 61918 [FIN, ACK] Seq=1 Ack=2 Win=212 Len=0
19 17.442241 10.2.17.114 20.190.145.160 TCP 54 61918 → 443 [ACK] Seq=2 Ack=2 Win=513 Len=0
Frame 1: 55 bytes on wire (440 bits), 55 bytes captured (440 bits) on interface \Device\NPF_{AC61C898-7D
Ethernet II, Src: CloudNetwork_ba:01:2d (60:e9:aa:ba:01:2d), Dst: ExtremeNetwo_98:c6:0d (00:04:96:98:c6:
Internet Protocol Version 4, Src: 10.2.17.114, Dst: 74.125.200.188
Transmission Control Protocol, Src Port: 61917, Dst Port: 443, Seq: 1, Ack: 1, Len: 1
0000 00 04 96 98 c6 0d 60 e9 aa ba 01 2d 08 00 45 00
0010 00 29 0e fd 40 00 80 06 bd 24 0a 02 11 72 4a 7d
0020 c8 bc f1 dd 01 bb 2b 0f 4d 6f 30 7b 00 2b 50 10
0030 02 02 e2 66 00 00 00




***Practical Implementation/Case Study***

**Case Study**: **Investigating Suspicious WiFi Activity on a Corporate Network**:

- **Scenario**: A network administrator suspects unauthorized devices accessing the company's WiFi network during off-hours.

**Steps for Implementation**:

- **Step 1**: Set up Wireshark with a capture filter for the IP range used by corporate devices.
- **Step 2**: Enable monitor mode to capture all packets within WiFi range.
- **Step 3**: Use display filters to isolate packets from unidentified devices or unusual MAC addresses.
- **Step 4**: Generate protocol hierarchy and endpoint statistics to identify high-traffic sources.

**Results and Analysis**:

- Detail any abnormal findings, such as high traffic from unfamiliar IPs or devices that appear to be scanning for network resources. Include screenshots and charts to support these findings.

**Effectiveness in Problem Solving**:

- Assess how Wireshark's capabilities allowed you to detect unauthorized access, such as isolating suspicious MAC addresses or examining peak traffic times. Discuss Wireshark's role in addressing the network security issue effectively.

***Security Implications and Analysis***

**Security Benefits:**

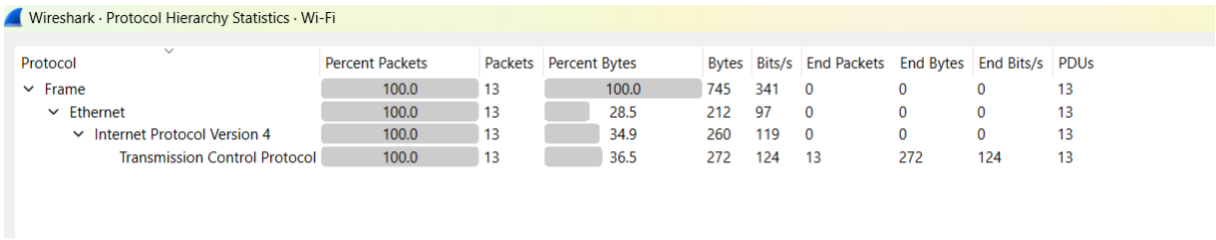
- Wireshark enables real-time monitoring, making it invaluable for detecting and mitigating network-based attacks quickly.

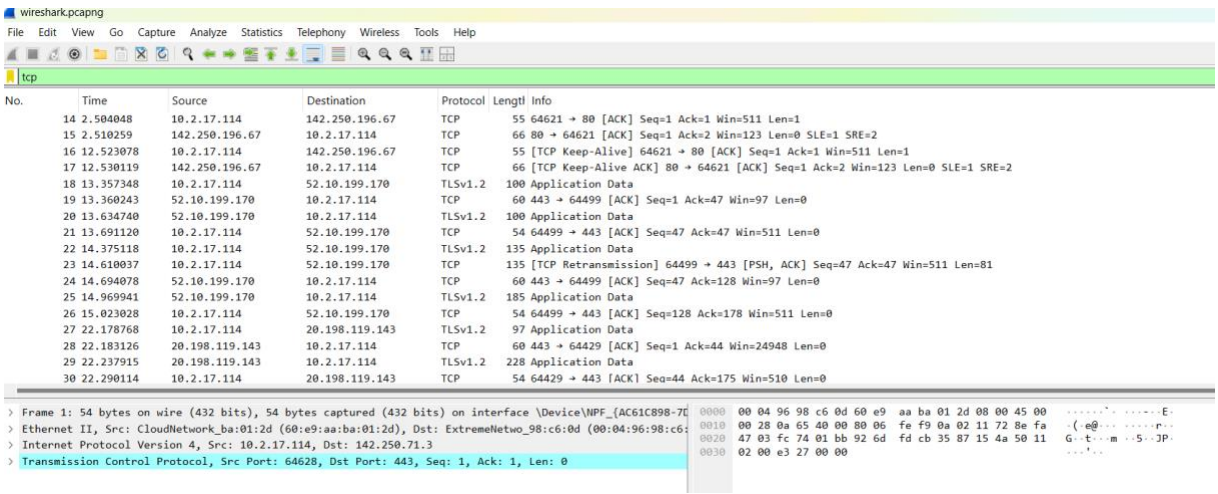
**Potential Vulnerabilities and Limitations:**

- **Privacy Concerns**: Capturing sensitive, unencrypted data could expose private information. Network access controls should be enforced.

- **Encrypted Traffic**: Wireshark cannot decrypt SSL/TLS traffic without the necessary keys, limiting its effectiveness for complete security monitoring.
- **System Resource Constraints**: High-traffic captures may strain system resources, leading to packet loss or incomplete captures.

### *Sample Output/Screenshots*




### *Findings of the Study*

**Summary of Observations**:

- The WiFi traffic analysis revealed specific patterns during business hours, with most traffic consisting of HTTP, HTTPS, and DNS protocols, which aligned with normal usage. However, during off-hours, unusual patterns emerged, such as high UDP traffic from unknown IP addresses and repeated ARP requests, which may indicate unauthorized access or network scanning.
- Certain devices exhibited abnormal traffic volumes, including continuous requests to external IPs, suggesting possible malware or data exfiltration attempts. Wireshark's filtering and protocol analysis allowed us to narrow down these devices and monitor them closely.

**Effectiveness of Wireshark**:

- **Filtering and Protocol Analysis**: Wireshark's display filters and protocol analysis were crucial in isolating specific traffic patterns, such as unusual DNS queries and suspicious IP addresses, which might have otherwise gone unnoticed.
- **Statistical Visualization**: The statistical tools, like I/O graphs, provided an overview of traffic spikes and anomalies, aiding in quickly identifying deviations from normal patterns.

***Conclusion***

- Wireshark is a powerful tool for WiFi traffic analysis, offering packet-level insights and robust filtering options that enable quick identification of anomalies and potential security threats. The tool's user-friendly interface and protocol support make it highly effective for both routine monitoring and advanced security diagnostics.
- Wireshark's comprehensive protocol analysis and filtering features make it invaluable for pinpointing issues and detecting unauthorized activity. The ability to visualize traffic statistics further enhances its effectiveness in spotting unusual patterns.