Lab 3

Exploring UDP with DNS and Sockets using Wireshark

Experiment 3.2: Exploring UDP with Sockets - Wireshark

1. Objective

To implement and observe a bi-directional chat application using UDP sockets. The goal is to understand how UDP communication works, analyze its characteristics using Wireshark, and explore the differences between UDP and TCP.

2. Prerequisites

- Two laptops running Windows, connected to the same network (Wi-Fi or mobile hotspot)
- Wireshark installed for network packet analysis
- Basic knowledge of networking concepts

Note = While opening the tool if you get a warning which says "Are you sure to run this Executable, click on Run Anyway"

You are free to see the source code here: https://github.com/NemaAdarsh/Udp-Chat-Tool.git

3. Theory

UDP (User Datagram Protocol) is a connectionless, lightweight transport layer protocol. Unlike TCP, UDP does not establish a connection before data transfer and does not provide reliability mechanisms like acknowledgments or retransmissions. This makes UDP faster but less reliable than TCP. UDP is used in applications where speed is preferred over reliability, such as video streaming, online gaming, and VoIP.

4. Steps to Execute

Step 1: Find the IP Addresses

1. Open Command Prompt on both laptops.
2. Run the following command:
3. ipconfig
4. Note the IPv4 Address (e.g., 192.168.x.x) and share it with the other user.

Step 2: Launch the Chat Application (Make sure to run as Administrator)

1. Open Computer Networks Lab 3 Tool on both laptops.
2. Enter your friend's IPv4 Address in the provided field.
3. Enter the desired Port Number (for example = 12345), make sure both of you both the same port number.
4. Click on Start chat to begin the Chat.
5. Start sending and receiving messages in real-time.

Step 3: Capture UDP Packets in Wireshark

1. Open Wireshark on both laptops.
2. Select the active network adapter (Wi-Fi or Ethernet).
3. Apply the following filter to see only UDP packets related to the chat:

```
udp.port == (port number of your choice eg:12345)
```

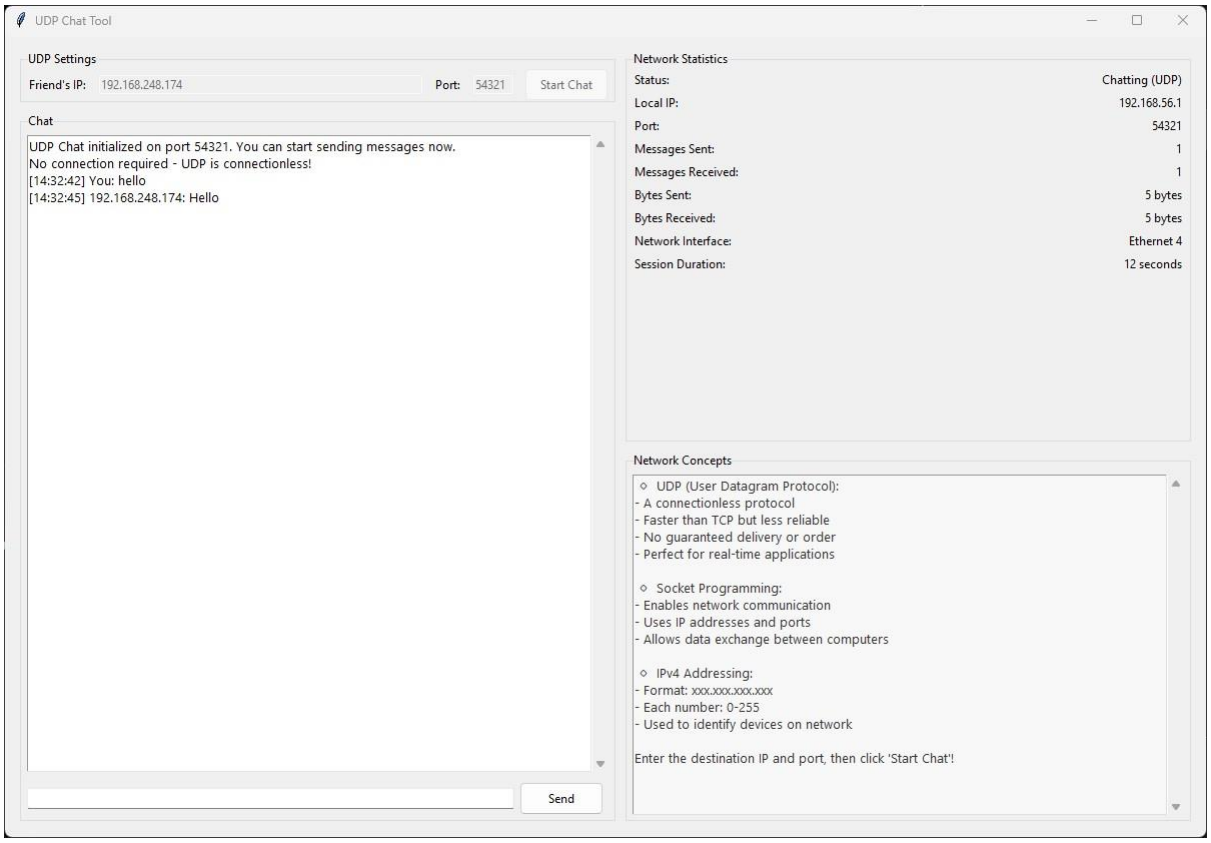
4. Start capturing packets and observe the data being transmitted.

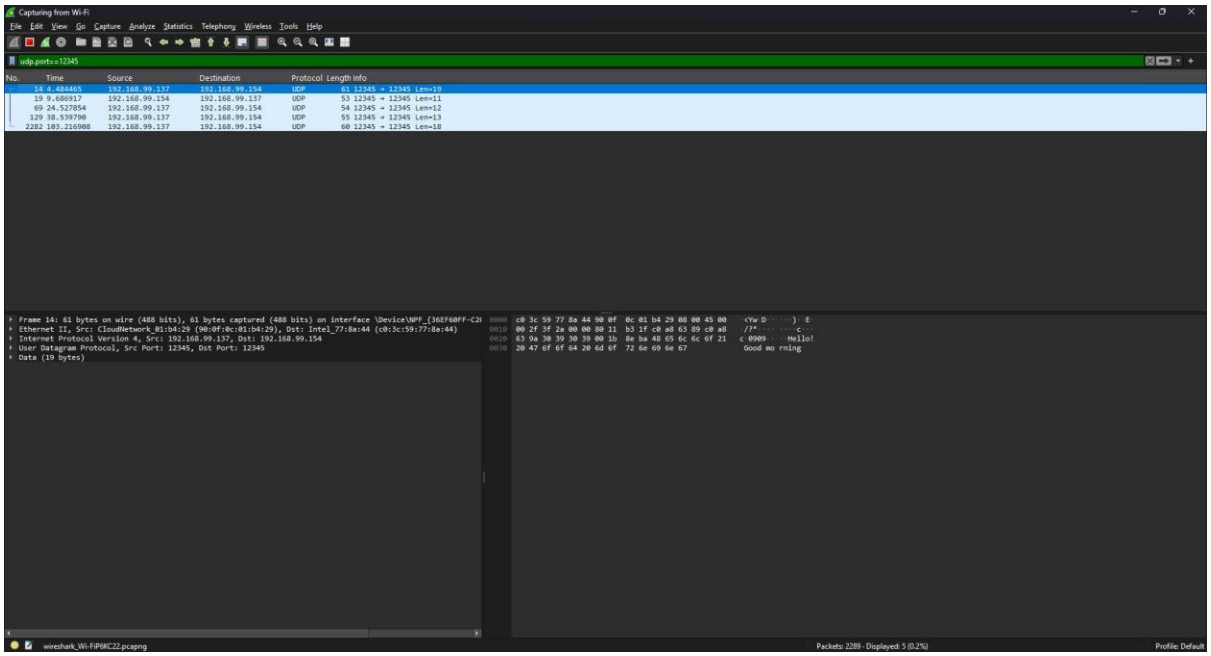
5. Observations

- Messages are transmitted as UDP packets between the two laptops.
- UDP communication is connectionless, meaning messages are sent without establishing a dedicated connection.
- Wireshark captures show the message content in plain text within the UDP packets.
- If either user disconnects, the chat session stops immediately without warning.

6. Conclusion

This experiment demonstrated a real-time chat application using UDP sockets, highlighting UDP's connectionless nature and real-time communication capabilities. The captured packets in Wireshark confirmed successful message transmission over the network, reinforcing the differences between UDP and TCP.





Homework Assignment

- **Packet Analysis**

Use Wireshark to capture UDP packets while using the chat tool. Identify the source and destination IP addresses and ports. Take a screenshot of your findings and write a short explanation of how UDP packets are transmitted.

Explanation of UDP Packet Transmission:

UDP (User Datagram Protocol) is a connectionless transport layer protocol that transmits data without establishing a session between sender and receiver. Here's how UDP packets are transmitted:

1. **Source and Destination Details**: Each UDP packet contains a source port (indicating the sender's application), a destination port (indicating the recipient application), source IP address, and destination IP address.

2. **No Handshake Process**: Unlike TCP, UDP does not establish a connection before sending data. It simply encapsulates the payload in a UDP datagram and sends it to the specified destination.

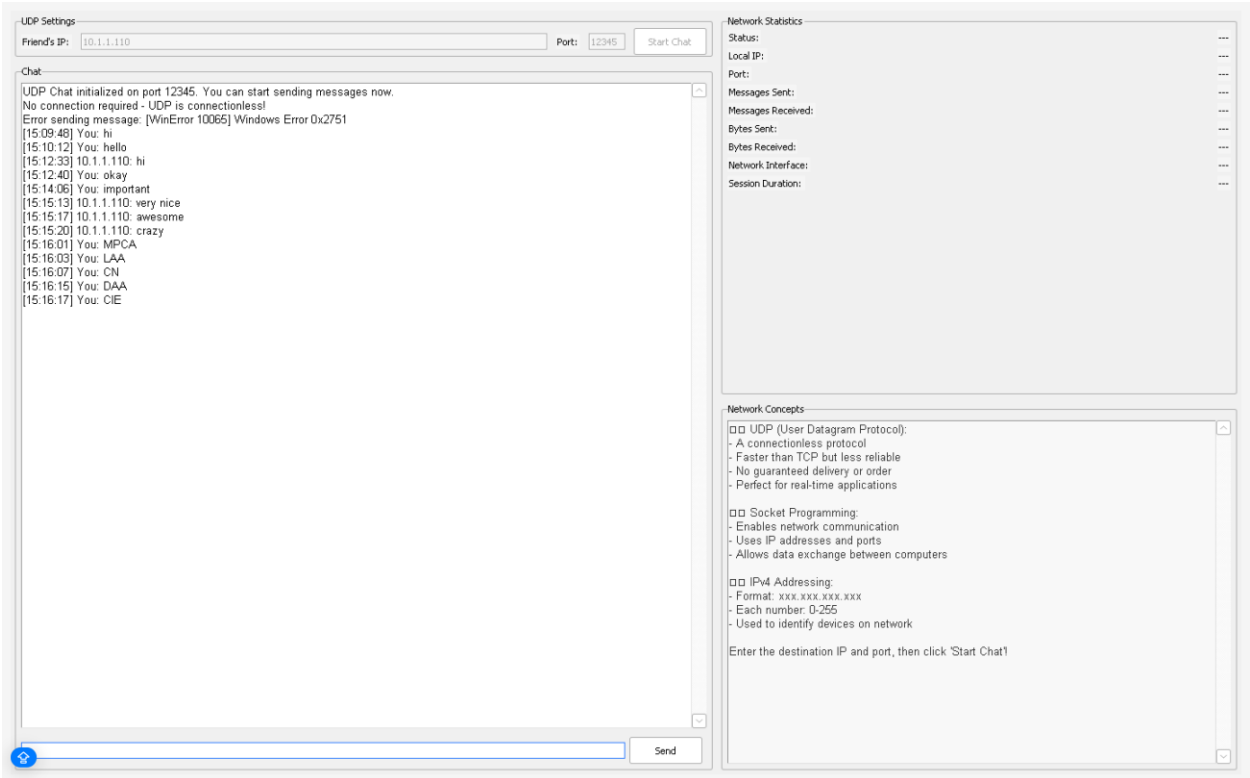
3. **Packet Structure**: A UDP packet consists of:

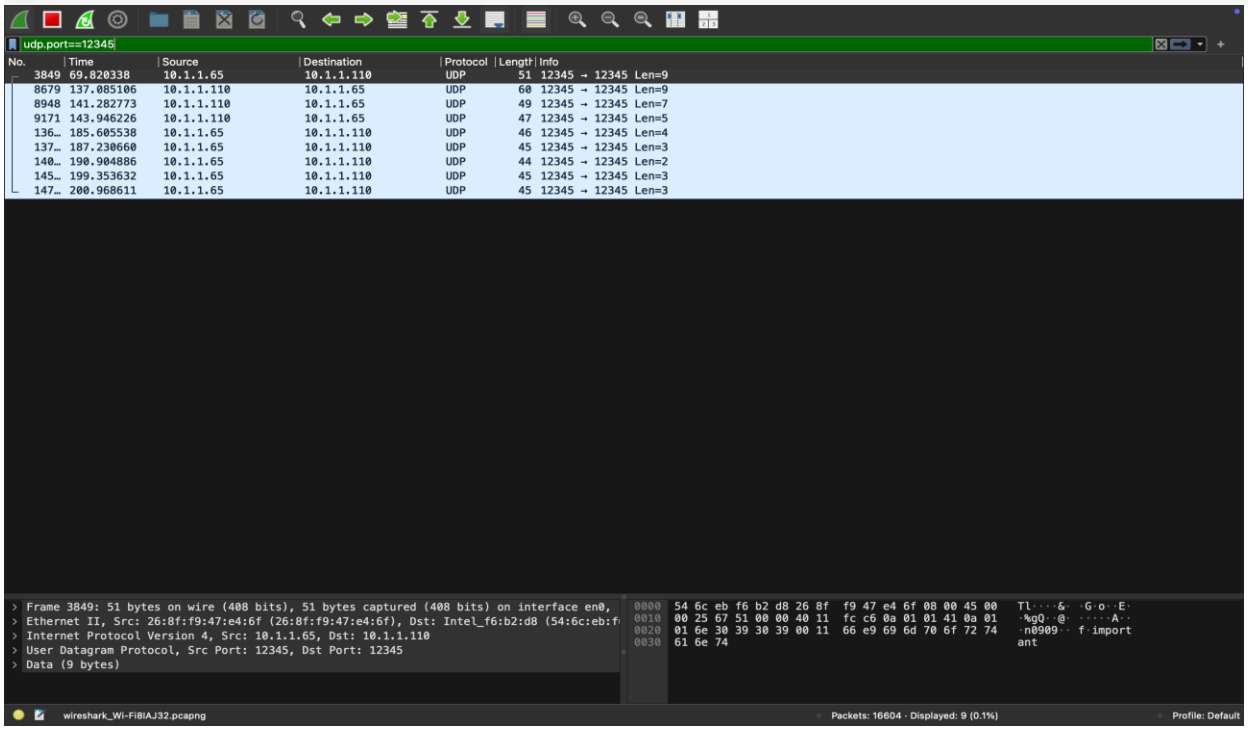
- **Header (8 bytes)**:
- Source Port (2 bytes)
- Destination Port (2 bytes)
- Length (2 bytes) – Specifies the total packet size.
- Checksum (2 bytes) – Provides error detection but is optional.
- **Data (Variable size)** – The actual payload.

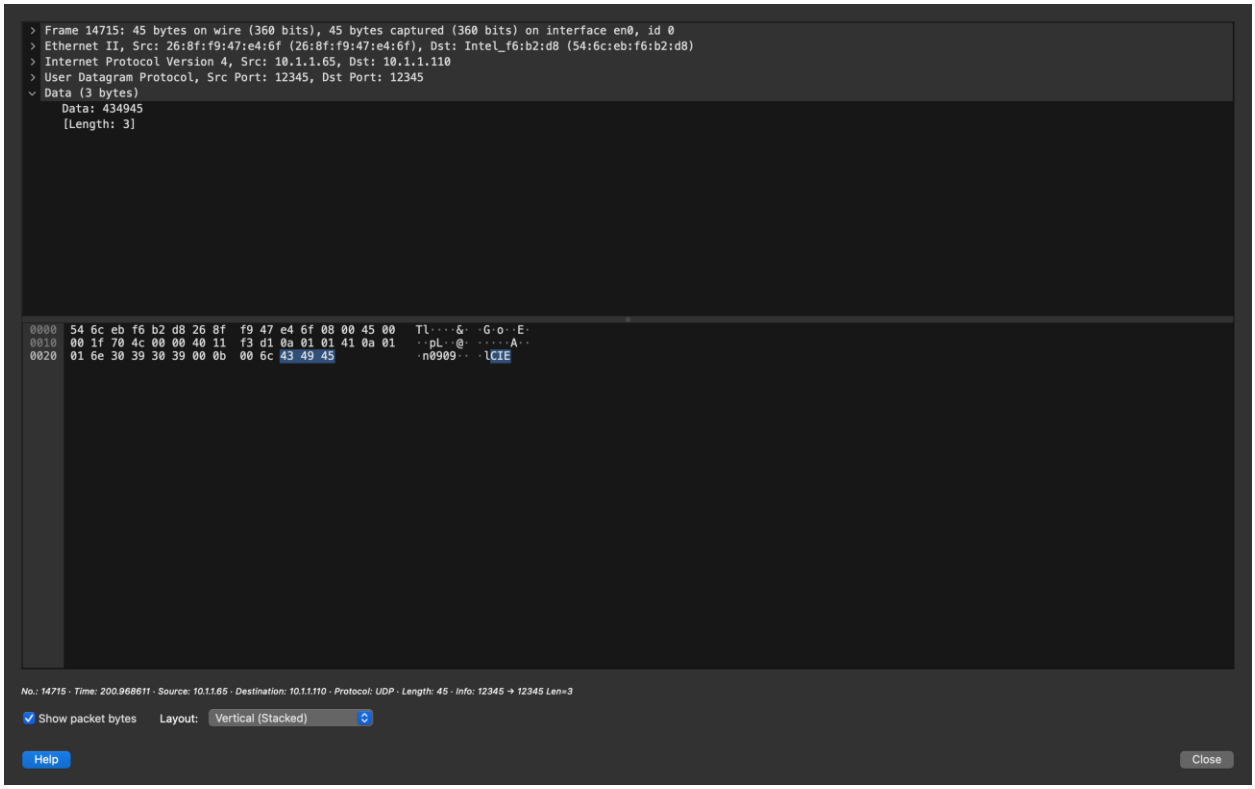
4. **Transmission and Reception**:

- The sender transmits a UDP packet to the recipient.
- The recipient receives the packet but does not acknowledge it.
- If a packet is lost, it is not retransmitted unless handled at the application layer.

5. **Use in Chat Tools**: In real-time communication applications like chat tools, UDP is often preferred because it provides low-latency data transfer without the overhead of TCP. However, since there is no built-in error correction, some messages may be lost during network congestion.

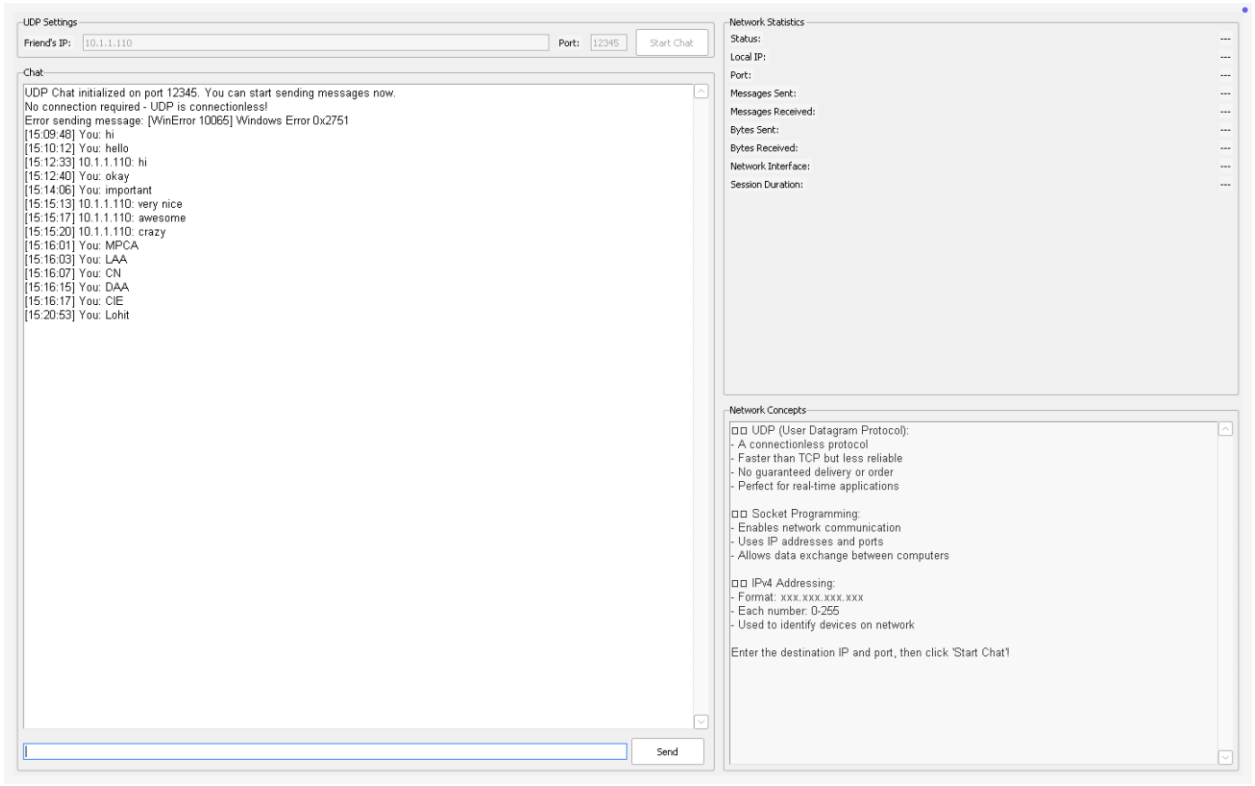

UDP Settings
Friend's IP: 10.1.1.110
Port: 12345
Start Chat
Chat
UDP Chat initialized on port 12345. You can start sending messages now.
No connection required - UDP is connectionless!
Error sending message: [WinError 10065] Windows Error 0x2751
[15:09:48] You: hi
[15:10:12] You: hello
[15:12:33] 10.1.1.110: hi
[15:12:40] You: okay
[15:14:06] You: important
[15:15:13] 10.1.1.110: very nice
[15:15:17] 10.1.1.110: awesome
[15:15:20] 10.1.1.110: crazy
[15:16:01] You: MPCA
[15:16:03] You: LAA
[15:16:07] You: CN
[15:16:15] You: DAA
[15:16:17] You: CIE
Send
Network Statistics
Status:
Local IP:
Port:
Messages Sent:
Messages Received:
Bytes Sent:
Bytes Received:
Network Interface:
Session Duration:
Network Concepts
UDP (User Datagram Protocol):
- A connectionless protocol
- Faster than TCP but less reliable
- No guaranteed delivery or order
- Perfect for real-time applications
Socket Programming:
- Enables network communication
- Uses IP addresses and ports
- Allows data exchange between computers
IPv4 Addressing:
- Format: xxx.xxx.xxx.xxx
- Each number: 0-255
- Used to identify devices on network
Enter the destination IP and port, then click 'Start Chat'!

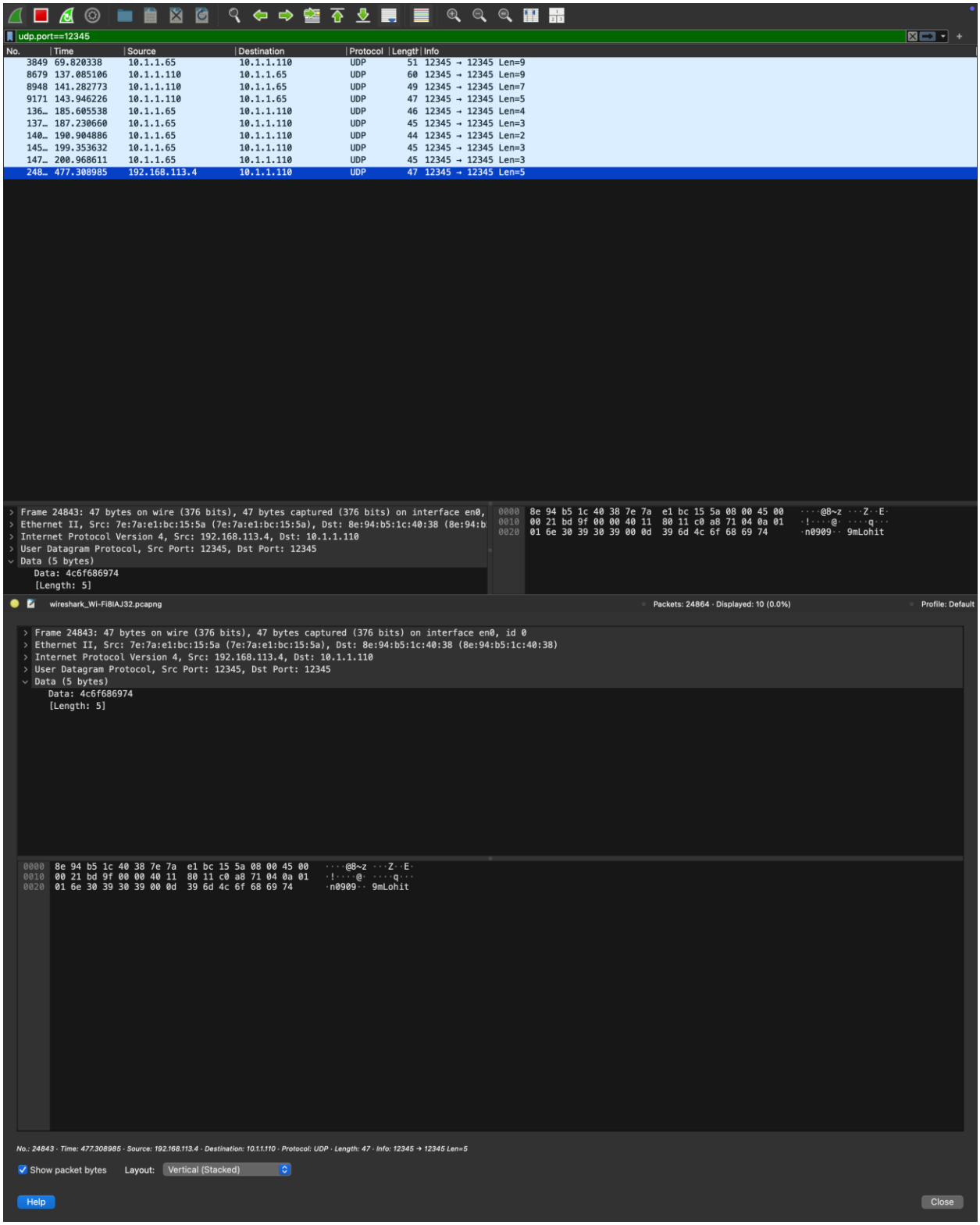

udp.port==12345
No. Time Source Destination Protocol Length Info
3849 69.820338 10.1.1.65 10.1.1.110 UDP 51 12345 → 12345 Len=9
8679 137.085106 10.1.1.110 10.1.1.65 UDP 60 12345 → 12345 Len=9
8948 141.282773 10.1.1.110 10.1.1.65 UDP 49 12345 → 12345 Len=7
9171 143.946226 10.1.1.110 10.1.1.65 UDP 47 12345 → 12345 Len=5
136… 185.605538 10.1.1.65 10.1.1.110 UDP 46 12345 → 12345 Len=4
137… 187.230660 10.1.1.65 10.1.1.110 UDP 45 12345 → 12345 Len=3
140… 190.904886 10.1.1.65 10.1.1.110 UDP 44 12345 → 12345 Len=2
145… 199.353632 10.1.1.65 10.1.1.110 UDP 45 12345 → 12345 Len=3
147… 200.968611 10.1.1.65 10.1.1.110 UDP 45 12345 → 12345 Len=3
Frame 3849: 51 bytes on wire (408 bits), 51 bytes captured (408 bits) on interface en0,
Ethernet II, Src: 26:8f:f9:47:e4:6f (26:8f:f9:47:e4:6f), Dst: Intel_f6:b2:d8 (54:6c:eb:f
Internet Protocol Version 4, Src: 10.1.1.65, Dst: 10.1.1.110
User Datagram Protocol, Src Port: 12345, Dst Port: 12345
Data (9 bytes)
0000 54 6c eb f6 b2 d8 26 8f f9 47 e4 6f 08 00 45 00
0010 00 25 67 51 00 00 40 11 fc c6 0a 01 01 41 0a 01
0020 01 6e 30 39 30 39 00 11 66 e9 69 6d 70 6f 72 74
0030 61 6e 74
wireshark_Wi-FiBIAJ32.pcapng
Packets: 16604 · Displayed: 9 (0.1%)
Profile: Default



- **Network Behavior Report**

Run the chat tool on two different networks (e.g., a home Wi-Fi network and a mobile hotspot). Observe and report any differences in message delays, lost packets, or overall performance. What might be causing these differences?





☐ Security Considerations

Since UDP is connectionless and does not guarantee delivery, attackers can exploit it for spoofing or flooding attacks. Research and write a short report (300-400 words) on common UDP security threats and possible mitigation strategies.

Understanding UDP Security Risks and Protective Measures

Introduction User Datagram Protocol (UDP) is widely used in networking due to its speed and efficiency. However, its connectionless nature makes it vulnerable to various cyber threats. Unlike TCP, UDP lacks built-in mechanisms for ensuring packet integrity or sender authentication, making it an attractive target for attackers. This document highlights the major security risks associated with UDP and suggests measures to mitigate them.

Key UDP Security Threats

1. **IP Spoofing and Reflection Attacks** Since UDP does not verify the source of a packet, attackers can forge sender addresses, making it difficult to trace malicious activity. This tactic is frequently used in reflection-based attacks, where a small request generates a disproportionately large response to overwhelm a target.

2. **UDP Flooding** Attackers can send an overwhelming number of UDP packets to a target machine, exhausting its resources and causing disruptions. This type of attack can bring down services or make networks unresponsive.

3. **Amplification Attacks** UDP services such as DNS, NTP, and SSDP can be exploited for amplification attacks, where attackers send small requests that trigger large responses directed at a victim's system, leading to denial-of-service conditions.

4. **Exploitation of UDP-Based Services** Protocols like TFTP and SNMP, which rely on UDP, are often targeted due to weak authentication controls. Attackers may use these vulnerabilities to gain unauthorized access or inject malicious data into a system.

5. **Disruption of Real-Time Communications** UDP is widely used for VoIP, gaming, and video streaming. Attackers can disrupt these services by injecting artificial latency, dropping packets, or overwhelming the network with junk data.

Defensive Strategies

1. **Traffic Rate Limiting and Filtering** Implementing firewall rules and rate-limiting policies can help prevent excessive UDP traffic and mitigate flood attacks.

2. **Authentication and Encryption** Using Datagram Transport Layer Security (DTLS) ensures encrypted and authenticated communication, making it harder for attackers to manipulate UDP traffic.

3. **Monitoring and Intrusion Detection** Network monitoring tools and anomaly detection systems can identify suspicious UDP activity and help prevent attacks before they escalate.

4. **Reducing Attack Surface** Disabling unnecessary UDP services and restricting access to essential ones using access control lists (ACLs) can limit exposure to attacks.

5. **Implementing Source Address Validation** Enforcing ingress filtering (BCP38) at the network level can block spoofed packets and reduce the effectiveness of reflection and amplification attacks.

Conclusion While UDP is indispensable for fast and real-time applications, its security weaknesses require proactive defense measures. By implementing proper filtering, monitoring, and authentication mechanisms, organizations can significantly reduce the risks associated with UDP-based threats and ensure a more secure network environment.