

Title of the Experiment: Comparative Traffic Analysis of Real-time VoIP (RTP/UDP) vs. Bulk Data Transfer (TCP) using Wireshark

Submitted by:

Name: Prachi Sharnagat Roll No: 20 Branch / Semester: A2 CSE (III SEM) Batch: B2	Name: Mansi Gupta Roll No: 32 Branch / Semester: A2 CSE (III SEM) Batch: B2	Name: Rashmi Sukhwani Roll No: 13 Branch / Semester: A2 CSE (III SEM) Batch: B1
---	--	--

Submission Deadline: 31/10/2025

1. Aim / Objective

To capture, analyze, and compare the network traffic characteristics of two different application types:

1. A real-time Voice over IP (VoIP) call, to observe the behavior of UDP and RTP.
2. A large file download, to observe the behavior of the TCP. The primary goal is to use the Wireshark tool to gather empirical evidence of the protocol differences between applications that prioritize speed (VoIP) versus those that prioritize reliability (file download).

2. Theory / Conceptual Background

Computer networks use a suite of protocols to manage communication. The Transport Layer is primarily governed by two key protocols: TCP and UDP.

- **UDP (User Datagram Protocol):** This is a "connectionless" protocol. It is very fast and has low overhead because it simply sends data packets (datagrams) without establishing a formal connection or checking if they arrive. It is ideal for time-sensitive applications like online gaming or voice calls, where speed is more important than perfect data integrity.
- **RTP (Real-time Transport Protocol):** This protocol runs *on top* of UDP and is designed specifically for streaming media. It adds timestamps and sequence numbers to the UDP packets, allowing the receiving application

to handle packets in the correct order and measure performance metrics like jitter and packet loss.

- **TCP (Transmission Control Protocol):** This is a "connection-oriented" protocol. It is built for reliability. Before sending any data, TCP establishes a stable connection via a **three-way handshake (SYN, SYN-ACK, ACK)**. It then meticulously tracks every packet sent, requiring acknowledgments from the receiver. If a packet is lost, TCP automatically re-transmits it, ensuring the data is 100% complete and uncorrupted. It is used for web browsing, file downloads, and email.

This experiment will capture and visualize this fundamental trade-off: UDP/RTP's speed vs. TCP's reliability.

3. Software / Simulation Tool Used

- **Tool:** Wireshark (v4.x)
- **Applications Used for Traffic Generation:** Discord (desktop client), Google Chrome (for file download)

4. Hardware / Software Requirements

- **Hardware:**
 - Processor: Intel Core i3 or above
 - RAM: 8 GB
 - Network: Active Internet Connection (Wi-Fi or Ethernet)
- **Software:**
 - Operating System: Windows 11
 - Simulation Tool: Wireshark (v4.2.0 or similar)
 - Other: Discord, Web Browser

5. Procedure / Steps Performed

The experiment was conducted in two distinct phases:

Phase A: VoIP (RTP/UDP) Traffic Capture & Analysis

1. Launched Wireshark and selected the active network interface.
2. Started a new packet capture.
3. Immediately started a voice call on the Discord desktop application and spoke for approximately 90 seconds.
4. Ended the voice call and then immediately stopped the Wireshark capture.
5. Filtered the captured packets by **udp** to locate the conversation.
6. Right-clicked the relevant UDP stream and used the **Decode As...** feature to correctly interpret the packets as **RTP**.
7. Filtered the display for **rtp** to isolate the voice stream.

- Opened the **Telephony -> RTP -> RTP Streams** analysis tool to gather statistics on packet loss and jitter.

Phase B: File Download (TCP) Traffic Capture & Analysis

- Started a new, separate capture in Wireshark.
- Immediately opened a web browser and began downloading a large file (the Ubuntu Desktop ISO).
- Allowed the download to run for approximately 30 seconds.
- Paused the download in the browser and then immediately stopped the Wireshark capture.
- Located a **TLSv1.3** packet from the download stream and right-clicked.
- Used the **Follow -> TCP Stream** feature to isolate the entire conversation.
- Analyzed the filtered stream, starting from the top, to observe the TCP handshake and data transfer.

6. Simulation Diagram / Screenshot

Screenshot 1: VoIP (RTP) Stream Analysis

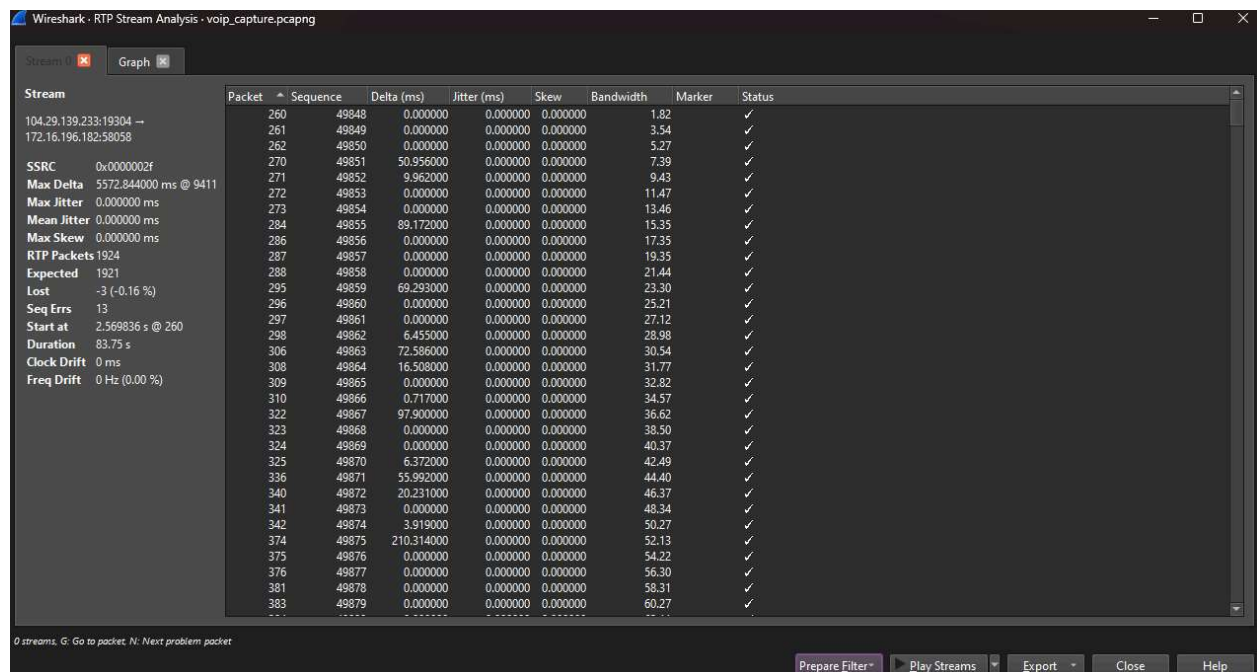


Figure 1: Analysis of the RTP stream from a Discord call, showing packet loss and jitter statistics.

Screenshot 2: TCP Stream Analysis

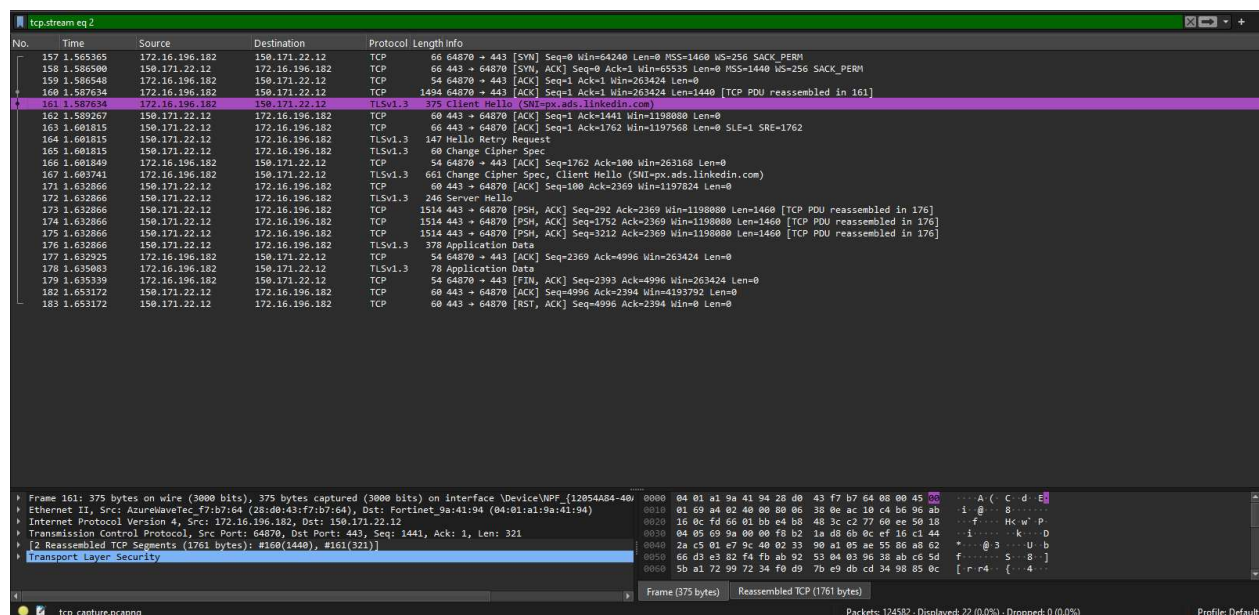


Figure 2: Analysis of the filtered TCP stream from a file download, showing the three-way handshake and data packets.

7. Observations

From the captured data and screenshots, we made the following observations:

- **RTP/UDP Traffic:**
 - The traffic consisted of a large number of small, continuous packets.
 - The **RTP Stream Analysis** tool (Figure 1) showed a packet loss of -3 (-0.16%), which is extremely low and acceptable for a real-time call.
 - The protocol did not show any connection setup (like a handshake); it began sending data immediately.
- **TCP Traffic:**
 - The conversation (Figure 2) clearly began with the **TCP three-way handshake** ([SYN], [SYN, ACK], [ACK]), which establishes the connection.
 - The data was sent in large, efficient "Application Data" packets.
 - The TCP stream also included acknowledgment ([ACK]) packets being sent back to the server to confirm receipt.

8. Result / Output

We successfully used Wireshark to capture and analyze two distinct types of network traffic. We were able to:

1. Isolate a UDP stream, decode it as RTP, and analyze its real-time performance metrics (packet loss, jitter).
2. Isolate a TCP stream and identify its key reliability features, most notably the three-way handshake.

9. Conclusion

This experiment successfully provided a practical demonstration of the core differences between UDP and TCP. The results confirm the theoretical concepts:

- **VoIP (RTP/UDP)** prioritizes speed and low latency by sacrificing the overhead of guaranteed delivery. This is essential for a real-time call.
- **File Download (TCP)** prioritizes 100% data integrity by using a robust, connection-oriented protocol that performs handshakes and acknowledgments. This is essential for ensuring a file is not corrupted.

We conclude that the choice of transport protocol is fundamentally dictated by the specific needs and goals of the application.

10. Viva Questions

1. What is the main difference between TCP and UDP?
2. Why can't you use TCP for a real-time voice call?
3. What is the purpose of the three-way handshake in TCP?
4. What is RTP, and why is it used on top of UDP?
5. In our TCP capture (Figure 2), what does [SYN] stand for, and what is its purpose?