COMPUTER NETWORKS

Instructor:

Prof. Dr. İlker Bekmezci

Report prepared by:

Aleyna Ünver - 042001017
Ali Boray Celpiş - 042001013
Enes Yıldız - 042001058
Gizem Şahin - 042001084
Göktuğ Kocauşu - 042001019
Kemal Saraç - 042001044

Date: 7 Jan 2024

1. Introduction

The SEQ & ACK Number Game project is an exploration of network programming, specifically creating a client-server program using the User Datagram Protocol (UDP). The main goal of the project is to offer practical experience in implementing a sequencing and acknowledgment mechanism to ensure the reliable transfer of data over an unreliable medium.

In this project, communication between the client and server occurs through UDP, with messages containing SEQ (Sequence) and ACK (Acknowledgment) numbers, along with information about packet length. The program also incorporates mechanisms for detecting errors to manage duplicate or lost packets, replicating real-world challenges encountered in network communication.

2. Architecture of Project

2.1 Communication Between Client and Server Using UDP

This project utilizes the UDP protocol for communication between the client and server. UDP is chosen for its lightweight and connectionless nature, enabling faster communication. However, it lacks guarantees regarding delivery, order, or protection against duplicates. This choice is made with a practical approach to understanding the difficulties of transmitting data reliably in an environment where these guarantees are not inherently provided.

2.2 The SEQ & ACK Number Game

2.2.1 How the Client Operates

The client sends messages to the server, each containing a SEQ (Sequence) number that indicates the message's sequence. Additionally, the message includes information about the length of the packet, ensuring the ordered and reliable delivery of data. To improve reliability, the client integrates a mechanism for message retransmission in case an acknowledgment is not received within a predetermined timeout period. This simulates situations where packets might be lost or delayed.

2.2.2 Server Operation

The server has two modes: auto and manual.

Auto Mode: The server checks if a received message is damaged and responds with the correct SEQ & ACK number and packet length automatically. This mode tests the client's ability to handle automatic error detection.

Manual Mode: In this mode, the user manually calculates SEQ & ACK numbers and packet lengths. This helps in better understanding the underlying SEQ & ACK logic and allows for user involvement in testing and troubleshooting. Additionally, the server may simulate not sending messages to test the client's ability to handle lost packets, creating a scenario similar to a timeout.

2.3 Detecting Errors

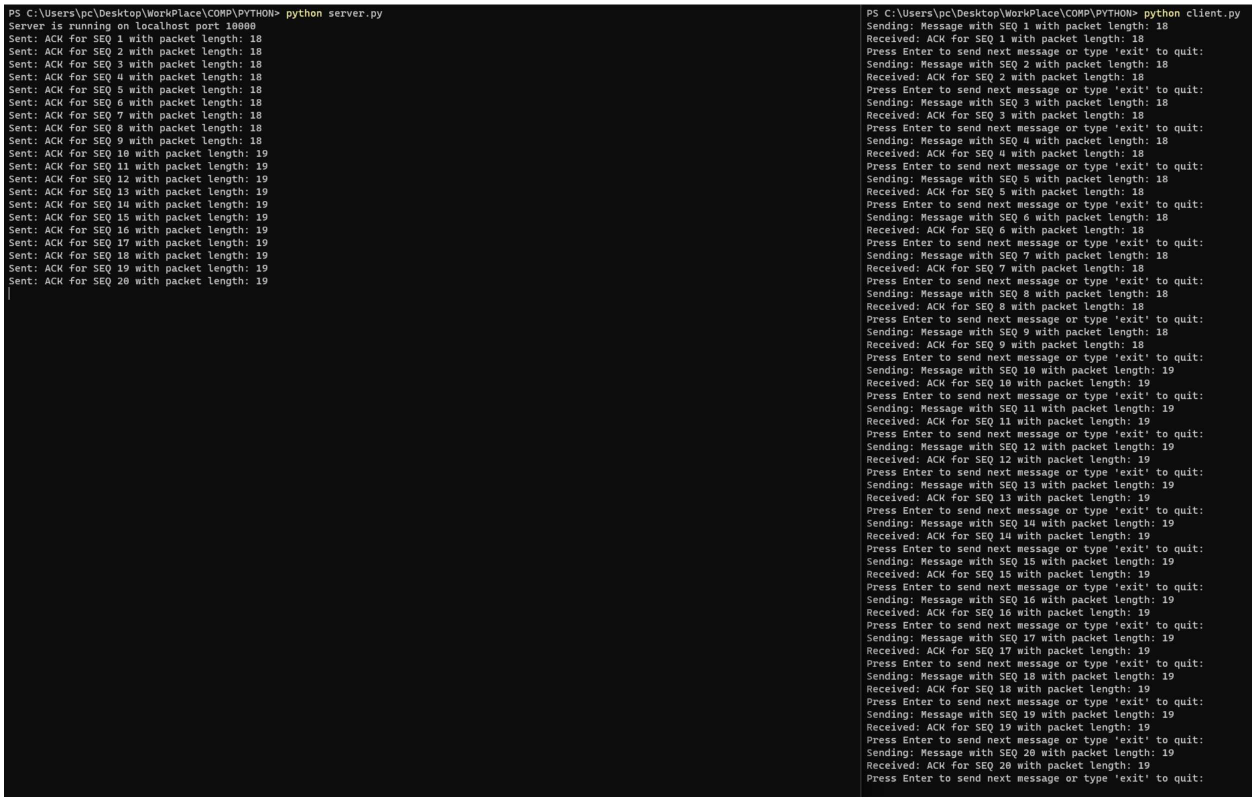
Both the client and server have mechanisms to detect duplicate or lost packets. The way they behave may vary between auto and manual modes, offering a thorough understanding of how error detection works in networking.

The project's architecture is designed to provide a holistic learning experience in network programming, emphasizing the challenges and solutions associated with reliable data transfer over UDP. The combination of sequencing, acknowledgment, and error handling mechanisms ensures a thorough exploration of key concepts in networking.

3. Results & Conclusion

The SEQ & ACK Number Game project, based on UDP, provides important insights into the intricacies of network programming, focusing on establishing dependable data transfer methods over UDP. By blending hands-on experiences with theoretical principles, this project prepares participants to tackle challenges in practical networking situations. Through building the client-server program and incorporating SEQ&ACK logic, error detection features, and user intervention functions, participants acquire a thorough grasp of UDP communication and the complexities of guaranteeing data reliability in an uncertain environment.

The output of the project sample is shown in Figure 1.



```
PS C:\Users\pc\Desktop\WorkPlace\COMP\PYTHON> python server.py
Server is running on localhost port 10000
Sent: ACK for SEQ 1 with packet length: 18
Sent: ACK for SEQ 2 with packet length: 18
Sent: ACK for SEQ 3 with packet length: 18
Sent: ACK for SEQ 4 with packet length: 18
Sent: ACK for SEQ 5 with packet length: 18
Sent: ACK for SEQ 6 with packet length: 18
Sent: ACK for SEQ 7 with packet length: 18
Sent: ACK for SEQ 8 with packet length: 18
Sent: ACK for SEQ 9 with packet length: 18
Sent: ACK for SEQ 10 with packet length: 19
Sent: ACK for SEQ 11 with packet length: 19
Sent: ACK for SEQ 12 with packet length: 19
Sent: ACK for SEQ 13 with packet length: 19
Sent: ACK for SEQ 14 with packet length: 19
Sent: ACK for SEQ 15 with packet length: 19
Sent: ACK for SEQ 16 with packet length: 19
Sent: ACK for SEQ 17 with packet length: 19
Sent: ACK for SEQ 18 with packet length: 19
Sent: ACK for SEQ 19 with packet length: 19
Sent: ACK for SEQ 20 with packet length: 19
```

```
PS C:\Users\pc\Desktop\WorkPlace\COMP\PYTHON> python client.py
Sending: Message with SEQ 1 with packet length: 18
Received: ACK for SEQ 1 with packet length: 18
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 2 with packet length: 18
Received: ACK for SEQ 2 with packet length: 18
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 3 with packet length: 18
Received: ACK for SEQ 3 with packet length: 18
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 4 with packet length: 18
Received: ACK for SEQ 4 with packet length: 18
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 5 with packet length: 18
Received: ACK for SEQ 5 with packet length: 18
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 6 with packet length: 18
Received: ACK for SEQ 6 with packet length: 18
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 7 with packet length: 18
Received: ACK for SEQ 7 with packet length: 18
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 8 with packet length: 18
Received: ACK for SEQ 8 with packet length: 18
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 9 with packet length: 18
Received: ACK for SEQ 9 with packet length: 18
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 10 with packet length: 19
Received: ACK for SEQ 10 with packet length: 19
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 11 with packet length: 19
Received: ACK for SEQ 11 with packet length: 19
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 12 with packet length: 19
Received: ACK for SEQ 12 with packet length: 19
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 13 with packet length: 19
Received: ACK for SEQ 13 with packet length: 19
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 14 with packet length: 19
Received: ACK for SEQ 14 with packet length: 19
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 15 with packet length: 19
Received: ACK for SEQ 15 with packet length: 19
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 16 with packet length: 19
Received: ACK for SEQ 16 with packet length: 19
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 17 with packet length: 19
Received: ACK for SEQ 17 with packet length: 19
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 18 with packet length: 19
Received: ACK for SEQ 18 with packet length: 19
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 19 with packet length: 19
Received: ACK for SEQ 19 with packet length: 19
Press Enter to send next message or type 'exit' to quit:
Sending: Message with SEQ 20 with packet length: 19
Received: ACK for SEQ 20 with packet length: 19
Press Enter to send next message or type 'exit' to quit:
```

Figure1. The output of the program

4. References

[1] “Differences Between TCP and UDP,” *Spiceworks*, Apr. 18, 2022. https://www.spiceworks.com/tech/networking/articles/tcp-vs-udp/#:~:text=Last%20Updated%3A%20April%2018%2C%202022,UDP%20prioritizes%20speed%20and%20efficiency. (accessed Jan. 07, 2024).

[2] “Differences between TCP and UDP,” *GeeksforGeeks*, Jun. 18, 2018. https://www.geeksforgeeks.org/differences-between-tcp-and-udp/ (accessed Jan. 07, 2024).

[3] L. Rosencrance, G. Lawton, and C. Moozakis, “User Datagram Protocol (UDP),” *Networking*, 2023. https://www.techtarget.com/searchnetworking/definition/UDP-User-Datagram-Protocol (accessed Jan. 07, 2024).

[4] “TCP Sequence and Acknowledgement Numbers Explained,” *MadPackets*, Apr. 25, 2018. https://madpackets.com/2018/04/25/tcp-sequence-and-acknowledgement-numbers-explained/#:~:text=TCP%20Sequence%20(seq)%20and%20Acknowledgement,the%20byte%2Dorder%20number). (accessed Jan. 07, 2024).