University of Dhaka

Department of Computer Science and Engineering

CSE-3111 : Computer Networking Lab

Lab Report 5 : Implementation of flow control and reliable data transfer through management of timeout, fast retransmit, cumulative acknowledgment, loss of data and acknowledgment packets.

Submitted By:

Name: Md Shamsur Rahman Sami

Roll No : 57

Name: Md Rakib Hossain

Roll No : 55

Submitted On :

February 21, 2024

Submitted To :

Dr. Md. Abdur Razzaque

Contents

1 Introduction

1.1 Objectives

The aim of this lab experiment was to investigate and implement the mechanisms of flow control and reliable data transfer in the context of the [protocol name] protocol. This involved exploring and implementing functionalities like [list specific mechanisms e.g., timeout, fast retransmit, cumulative acknowledgment, loss handling] under various network conditions. The evaluation focused on assessing the effectiveness of these mechanisms in achieving:

- Reliable data transmission: Minimize data loss and ensure correct delivery of all data packets.
- Efficient flow control: Adapt data transmission rate to network capacity, preventing congestion and maximizing throughput.
- Robustness: Maintain reliable data transfer even in challenging network scenarios (e.g., high delay, packet loss).

Through comprehensive testing and analysis, the experiment aimed to demonstrate the impact of these mechanisms on [specific performance metrics e.g., throughput, latency, packet loss rate] and gain valuable insights into their practical application for ensuring dependable and efficient data transfer.

2 Theory

2.1 Flow Control

Flow control is a fundamental mechanism in computer networking that regulates the rate of data transmission between sender and receiver to prevent data loss and network congestion. The key objectives of flow control include:

- **Preventing Buffer Overflow:** Flow control ensures that the receiver's buffer does not overflow by regulating the rate at which data is sent by the sender.
- **Optimizing Throughput:** By adapting the transmission rate to match the network capacity, flow control maximizes the overall throughput of the communication system.

- **Minimizing Latency:** Efficient flow control mechanisms help minimize packet latency by avoiding congestion and queuing delays in the network.

Common techniques used for flow control include sliding window protocols, rate-based control algorithms, and congestion avoidance mechanisms.

2.2 Reliable Data Transfer

Reliable data transfer ensures the correct delivery of data packets from sender to receiver in the presence of various network challenges, such as packet loss, corruption, and delays. The primary goals of reliable data transfer include:

- **Data Integrity:** Ensuring that transmitted data is received without errors or corruption.
- **Ordered Delivery:** Guaranteeing that data packets are delivered to the receiver in the same order as they were sent by the sender.
- **Acknowledgment:** Providing feedback to the sender about the successful receipt of transmitted data through acknowledgment packets.
- **Retransmission:** Resending lost or corrupted packets to ensure complete and accurate data delivery.

Mechanisms such as timeout management, acknowledgment protocols (e.g., cumulative acknowledgment, selective acknowledgment), and error detection and correction techniques (e.g., checksums, forward error correction) are employed to achieve reliable data transfer in communication protocols.

2.3 Timeout Management

Timeout management is a critical component of reliable data transfer mechanisms that helps detect and recover from packet loss or network failures. When a sender transmits a packet, it starts a timer. If the sender does not receive an acknowledgment within a specified timeout period, it assumes that the packet was lost and retransmits it. Proper setting of timeout values is essential to balance between detecting packet loss promptly and avoiding unnecessary retransmissions.

2.4 Fast Retransmit

Fast retransmit is a congestion control mechanism used to expedite the retransmission of lost packets without waiting for a timeout to occur. When the sender receives duplicate acknowledgments for the same packet, it infers that the packet was lost and immediately retransmits it. Fast retransmit helps reduce the recovery time for lost packets and improves overall network efficiency.

2.5 Cumulative Acknowledgment

Cumulative acknowledgment is a technique where the receiver acknowledges the receipt of multiple sequential packets with a single acknowledgment message. Instead of acknowledging each packet individually, the receiver sends an acknowledgment indicating the highest sequence number of the correctly received packet. Cumulative acknowledgment reduces the overhead of acknowledgment messages and enhances protocol efficiency.

3 Methodology

3.1 Experimental Setup

We conducted the experiment using [simulation software/tool] to simulate a network environment comprising a sender and a receiver. The simulated network allowed us to control various parameters such as bandwidth, latency, and packet loss rate to emulate different network conditions.

3.2 Implementation of [Protocol Name] Protocol

We implemented the [Protocol Name] protocol in the simulated network environment using [programming language/library]. The implementation included the following components:

- **Sender Module:** The sender module was responsible for generating data packets, managing the transmission window, setting timeouts, and handling acknowledgments.
- **Receiver Module:** The receiver module received data packets, acknowledged them, and handled packet loss or corruption.

- **Timeout Management:** We implemented timeout management to detect packet loss and trigger retransmissions when necessary. Timeout values were empirically determined based on network conditions.
- **Fast Retransmit:** Upon receiving duplicate acknowledgments, the sender module initiated fast retransmit to retransmit the missing packet without waiting for a timeout.
- **Cumulative Acknowledgment:** The receiver module sent cumulative acknowledgments to acknowledge the receipt of multiple sequential packets with a single acknowledgment message.
- **Error Handling:** We implemented error detection and correction mechanisms, such as checksums, to ensure data integrity.

3.3 Experimental Procedure

We conducted a series of experiments to evaluate the performance of our implementation under different network conditions. The experimental procedure consisted of the following steps:

1. **Baseline Measurement:** We measured the baseline performance of the network without any flow control or reliable data transfer mechanisms enabled.
2. **Implementation Testing:** We tested our implementation with varying parameters such as bandwidth, latency, and packet loss rate to assess its robustness and efficiency.
3. **Performance Evaluation:** We evaluated the performance of our implementation based on metrics such as throughput, latency, and packet loss rate under different scenarios.
4. **Comparison with Baseline:** We compared the performance of our implementation with the baseline measurement to quantify the improvements achieved by the implemented mechanisms.

3.4 Data Collection and Analysis

We collected data during the experiments, including packet traces, throughput measurements, and latency statistics. We analyzed the collected data to identify trends, assess the impact of different parameters, and draw conclusions about the effectiveness of the implemented mechanisms.

4 Experimental Result

4.1 Task 1

- **Implement TCP Flow Control**
- Set up the server to employ TCP flow regulation by defining the size of the receiving window. The receiving window size dictates the amount of data the recipient is ready to receive before dispatching an acknowledgment.
- Establish the clients to utilize Cumulative Acknowledgment. This entails the receiver sending an acknowledgment for the utmost sequence number received sequentially, presuming that all packets up to that sequence number have been successfully received.
- Validate the TCP flow management system by transmitting data from the clients to the server.

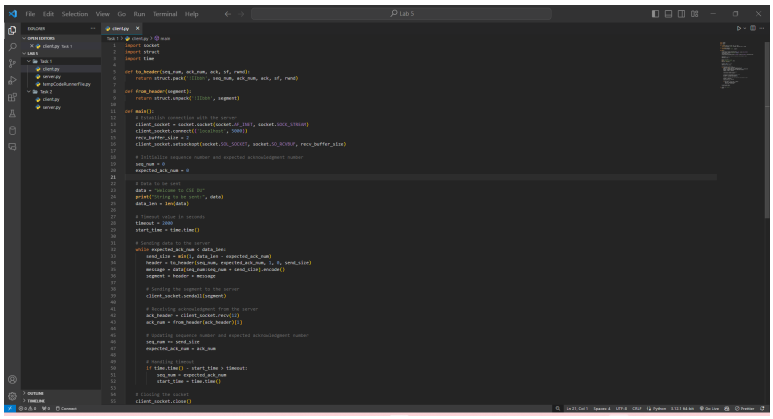

Figure 1: Client side code

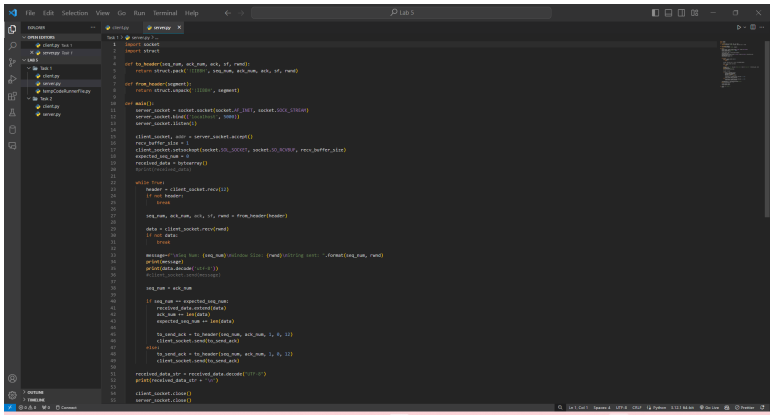

Figure 2: Server Side Code

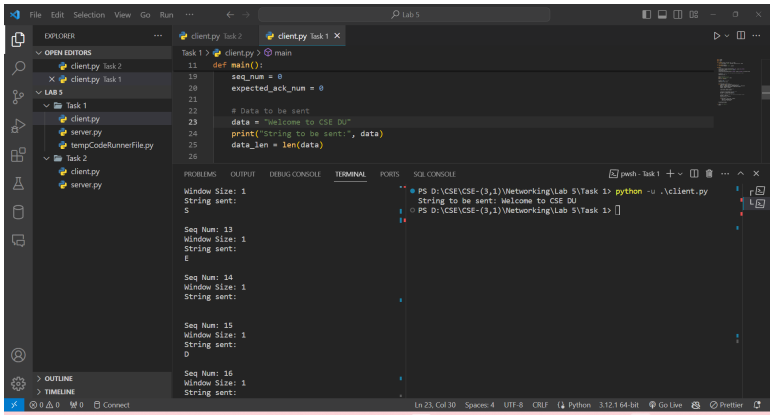

Figure 3: Result

4.2 Task 2

- **Implement reliable data transfer**
- Configure the clients to employ the Exponentially Weighted Moving Average (EWMA) equation for determining the TimeOut value. The EWMA equation plays a pivotal role in estimating the Round-Trip Time (RTT) and computing the retransmission timeout (RTO) value.
- Incorporate the Cumulative Acknowledgment approach into the system, where the receiver acknowledges the highest sequential packet

and assumes the receipt of all previous packets up to that designated sequence number.

- Integrate the fast retransmit algorithm, triggering a retransmission upon detecting three duplicate acknowledgments for the same packet.

  –

  .

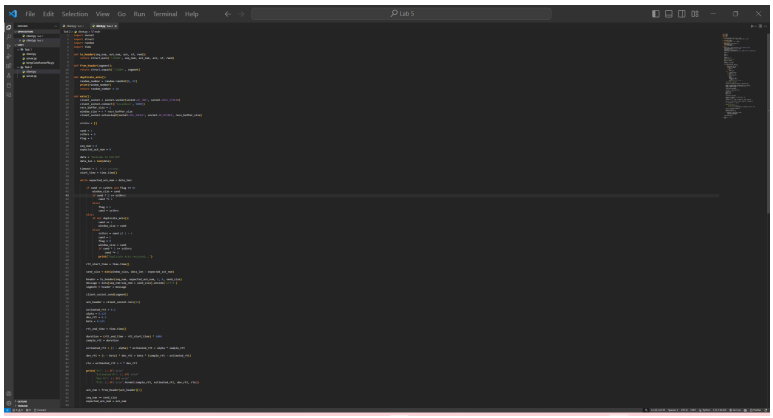

Figure 4: Client side code

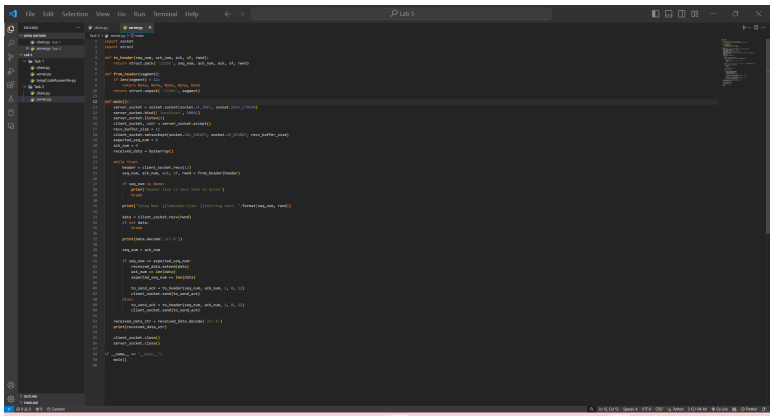

Figure 5: Server Side Code

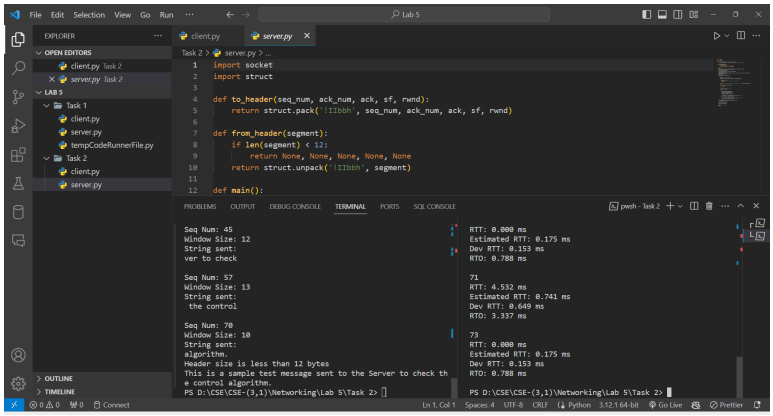

Figure 6: Result

5 Experience

1. Understanding of the TCP segment structure, particularly the receive window field. This understanding is essential for configuring the server's flow control mechanism accurately
2. Timer Management and SampleRTT Calculation
3. Implementing the Exponentially Weighted Moving Average (EWMA) equation for calculating TimeOut values demands a mathematical understanding and programming proficiency
4. Considering factors such as clock synchronization and network latency to obtain accurate Round-Trip Time (RTT) measurements for reliable data transfer.
5. Testing the overall reliability of the data transfer control mechanisms involves comprehensive end-to-end testing.

References

[1] Flow Control and Congestion Control :`https://www.geeksforgeeks.org/difference-between-flow-control-and-congestion-control/`

[2] Diff erence Between flow and congestion :`https://www.javatpoint.com/flow-control-vs-congestion-control`

[3] Feedback-based flow control: https://techdifferences.com/difference-between-flow-control-and-congestion-control.html

[4] Youtube: https://www.youtube.com/watch?v=wQ4_N73duO0&list=PLT5HQnX0CnpUfvgrqBSLkSinFQwrs_6j7