جامعة العلوم والتكنولوجيا
University of Science and Technology

CSAI 252
Computer Networks
Project

Project Report

Implementation of
Reliable Transport Protocol
Using
Go-Back-N (GBN) Protocol over UDP

By

Ahmed Ibrahim - 202202064

Haneen Alaa - 202201463

Rihana Mahmoud - 202201092

Tasneem Mohammed - 202201798

14/5/2024

1. Introduction

This project aimed to implement a reliable transport protocol utilizing the Go-Back-N (GBN) protocol over UDP.

The GBN protocol enhances the reliability of UDP by enabling the sender to transmit multiple packets without awaiting acknowledgments, while maintaining a limited number of unacknowledged packets in the pipeline.

2. Implementation Details:

- **GBN Algorithm:**

We used GBN for reliable data transfer.

GBN ensures cumulative acknowledgments, which enables the sender to slide its window forward upon receiving acknowledgments.

- **Sender Implementation:**

The sender script takes 3 arguments: filename, receiver IP address, and receiver port.

It divides the file into smaller chunks, assigns unique packet and file IDs, and appends a trailer indicating the end of the file.

The sender transmits packets in a sliding window manner, waiting for acknowledgments and handling timeout events.

- **Receiver Implementation:**

Upon receiving packets, the receiver parses the headers and trailers to identify the file and packet IDs.

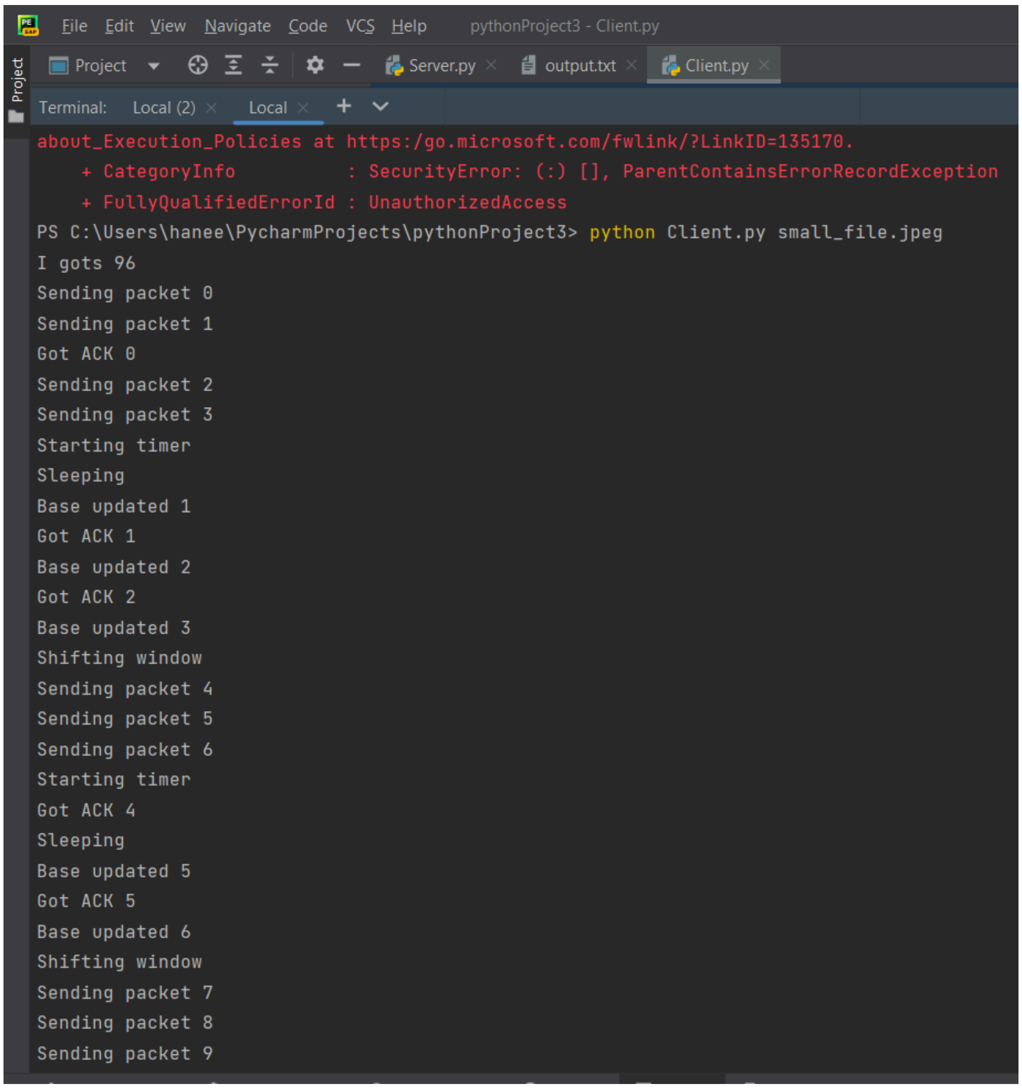
If the received packet is the expected one, it stores the application data; otherwise, it discards the data.

The receiver sends acknowledgments for correctly received packets, advancing the expected packet ID.

3. Results:

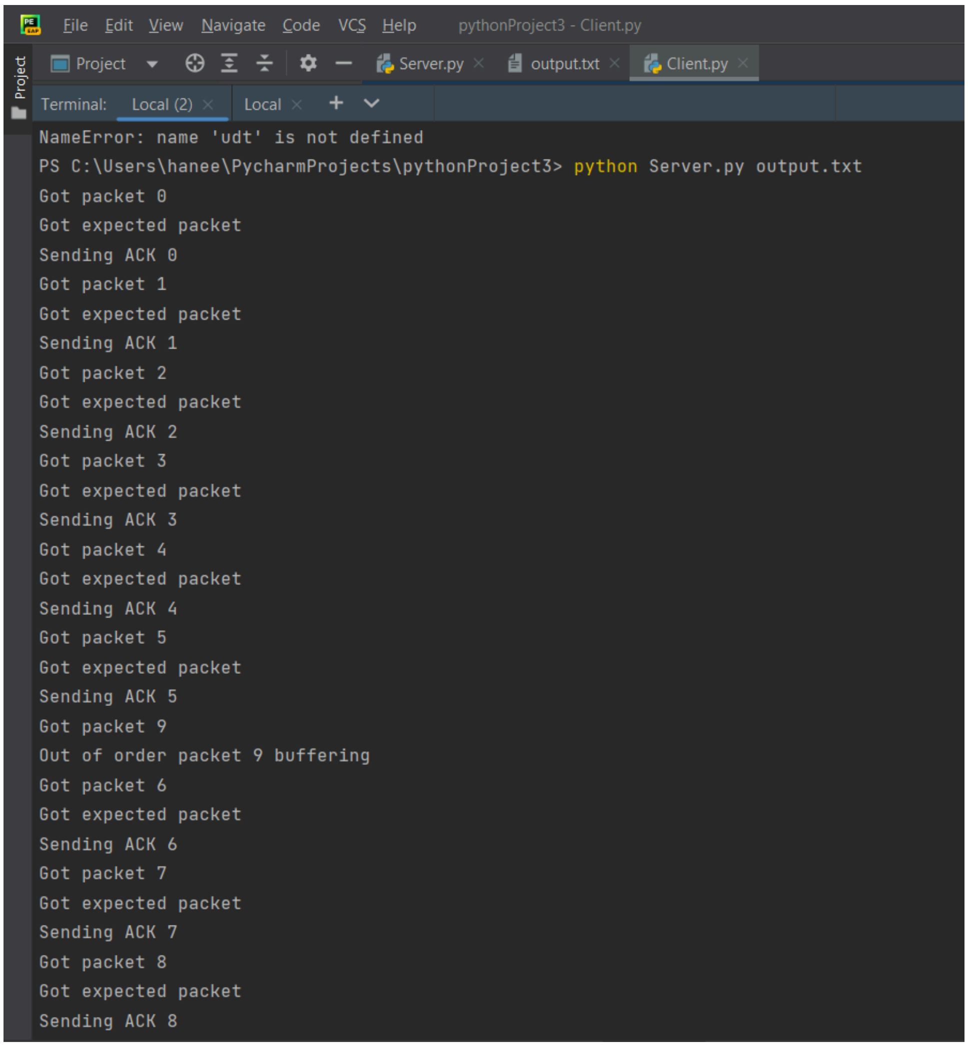
- **Packet Listing Screenshots:**

Client.py


File Edit View Navigate Code VCS Help pythonProject3 - Client.py
Project
Server.py
output.txt
Client.py
Terminal: Local (2) Local

```
about_Execution_Policies at https:/go.microsoft.com/fwlink/?LinkID=135170.
    + CategoryInfo          : SecurityError: (:) [], ParentContainsErrorRecordException
    + FullyQualifiedErrorId : UnauthorizedAccess
PS C:\Users\hanee\PycharmProjects\pythonProject3> python Client.py small_file.jpeg
I gots 96
Sending packet 0
Sending packet 1
Got ACK 0
Sending packet 2
Sending packet 3
Starting timer
Sleeping
Base updated 1
Got ACK 1
Base updated 2
Got ACK 2
Base updated 3
Shifting window
Sending packet 4
Sending packet 5
Sending packet 6
Starting timer
Got ACK 4
Sleeping
Base updated 5
Got ACK 5
Base updated 6
Shifting window
Sending packet 7
Sending packet 8
Sending packet 9
```

Server.py


File Edit View Navigate Code VCS Help pythonProject3 - Client.py
Project
Server.py
output.txt
Client.py
Terminal: Local (2) Local

```
NameError: name 'udt' is not defined
PS C:\Users\hanee\PycharmProjects\pythonProject3> python Server.py output.txt
Got packet 0
Got expected packet
Sending ACK 0
Got packet 1
Got expected packet
Sending ACK 1
Got packet 2
Got expected packet
Sending ACK 2
Got packet 3
Got expected packet
Sending ACK 3
Got packet 4
Got expected packet
Sending ACK 4
Got packet 5
Got expected packet
Sending ACK 5
Got packet 9
Out of order packet 9 buffering
Got packet 6
Got expected packet
Sending ACK 6
Got packet 7
Got expected packet
Sending ACK 7
Got packet 8
Got expected packet
Sending ACK 8
```

- **Tests iterations :**

We conducted tests with different window sizes (N) and timeout intervals, testing the protocol performance with different network conditions.

4. Bonus Point 5 :

The protocol can be enhanced for better performance by adding features from the TCP protocol standard.

For example : Flow control, congestion model and hashing.

And these examples are what we are going to discuss.

A. Flow Control to Prevent Overloading :

Flow control is crucial for preventing the sender from overwhelming the receiver with too much data at once.

In TCP, flow control is implemented using a mechanism called the receiver window.

- The TCP Feature inspired from is the receiver window
- TCP uses a sliding window for flow control, allowing the receiver to advertise how much data it can accept at any given time. This prevents buffer overflow at the receiver.
- Implementation in GBN: we can introduce a receiver window size parameter that the sender must respect, and adjust its transmission rate based on the receiver's capacity.
- Reference: RFC 793, Section 3.7.

B. Congestion Control for Resource Management

Congestion control mechanisms help in managing network traffic to avoid congestion, which can lead to packet loss and delays.

- TCP Feature that inspired from is Congestion Control Algorithms like : Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery.

- The TCP uses algorithms to detect and react to network congestion. Furthermore, Slow Start gradually increases the transmission rate, Congestion Avoidance adjusts the rate based on network feedback, and Fast Retransmit/Recovery quickly deals with lost packets.
- Implementation in GBN: it can be done by implementing a basic congestion control algorithm such as Slow Start and Congestion Avoidance to dynamically adjust the sender's transmission rate based on network conditions.

C. Hashing:

- The purpose is to ensure the integrity of each data packet, and verifying that the data received is the same as what was sent.
- TCP includes a checksum in each segment to detect errors in transmitted data and not a hash, it serves a similar purpose in ensuring data integrity.
- Hash Function: Use a cryptographic hash function like SHA-256 to generate a hash for the data portion of each data segment. This hash is sent along with the data, and the receiver verifies the integrity by comparing the received hash with a newly calculated hash from the received data.

5. Bonus Point (6) :

First : Modifying the sender in a way that exploits any weakness in the protocol specifications:

- We can use packet flooding which works on sending large numbers of packets through the network to see how will the network deal with the high traffic, so we can know the limits of the network also identifies if there is a problem that can be solved like a hardware problem like the memory or a software problem like software issues.
- Sending corrupted data, we can send corrupted or altered data through the network in order to know how the system will handle it and by applying this we can check if the system has different strategies to detect those errors and handle them.
- Also there is the fragmentation attack which is breaking down large packets and sending them to the receiver and it should gather those packets again in the corrected order.

Second : How the proposed attack method works and the suggested mitigation methods on the receiver side:

a) Fragmentation attack: which is simply sending sending large number of fragmented packets to the receiver so the receiver will be focused on gathering the packets to be in the correct order and format and it may lead to system crashing

b) Mitigation Method: implementing packet limit on the receiver side to put a threshold for the packets to be received so that the system is safe from crashing, another way is to put a timeout for the packets gathering process if it took more than the specified time to discard them.

Third : Possible protocol updates for a relatively more secure version of this Protocol:

- Limiting and controlling the number of received packets from one single source to avoid system crashing.
- Making sure the fragments order is correct.
- Dropping incomplete sent fragmented packets after specific time.
- Specifying the fragmented packet number in the header with the total number of packets.

Fourth : Regulations and Laws:

1) Local :

a. Computer Fraud and Abuse Act (CFAA)

In the US, the CFAA prohibits unauthorized access to computer systems, including intentional disruption of network communication.
Violators may face fines and imprisonment, with penalties varying based on the severity of the offense.

b. Data Protection Laws (e.g. GDPR in Europe):

Many regions have data protection laws that include provisions for safeguarding against unauthorized access and disruption of data transmission.
Violations can result also in fines and legal consequences for both individuals and organizations.

c. Telecommunications Act

Telecommunications laws in many countries regulate network communication and may include provisions against malicious interference with networks. Penalties for violations can include fines, imprisonment, and civil liability.

2) International :

a) Council of Europe Convention on Cybercrime (Budapest Convention):

This treaty addresses various forms of cybercrime, including unauthorized access to computer systems and interference with network communication. Signatory countries are required to implement laws to combat cybercrime and provide for legal cooperation and mutual assistance.

b) International Telecommunication Union (ITU) Regulations:

The ITU develops international regulations and standards for telecommunications, including measures to protect the integrity and security of networks. Member states are expected to comply with ITU regulations and may face diplomatic consequences for non-compliance.

3) Penalties:

- **Fines:** Monetary penalties imposed on individuals or organizations found guilty of violating relevant laws and regulations.

- **Imprisonment:** offenders may face jail time for serious violations, especially if they cause significant harm or financial loss.

- **Civil Liability:** Individuals or organizations affected by network disruptions may seek compensation through civil lawsuits for damages incurred.

- **Legal Restrictions:** offenders may be subject to legal restrictions, such as probation or restrictions on computer and internet use.

4) Economic and Societal Impact:

The disruptions in network communication can have financial and societal impacts. Since businesses depend on stable and secure networks for their operations, any disruptions can result in financial losses due to downtime, lost productivity, and reputational damage. Additionally, critical infrastructure networks in sectors such as healthcare, transportation, and utilities are essential for public safety and well-being.

The widespread availability of tools that disrupt network communication can erode trust in online services and the internet, hindering the growth of digital economies and stifling innovation. In response, governments and regulatory bodies might implement stricter regulations and enforcement measures, increasing the compliance burden on businesses and individuals.

In conclusion, the regulations and laws against disrupting network communication aim to safeguard the integrity, security, and reliability of digital networks. Violations of these laws can result in severe penalties, including fines and civil liability.

References

Calderoni, F. (2010). The European legal framework on cybercrime: striving for an effective implementation. *Crime, Law and Social Change*, *54*(5), 339–357. https://doi.org/10.1007/s10611-010-9261-6

Datta, S., & Datta, S. (2024, March 18). *Flow Control vs. Congestion Control in TCP | Baeldung on Computer Science*. Baeldung on Computer Science. https://www.baeldung.com/cs/tcp-flow-control-vs-congestion-control

J. Postel, "Transmission Control Protocol," Sep. 1981. doi: 10.17487/rfc0793.

Pawan, & Renu. (2019). Impact of Cyber Crime: issues and challenges. *International Journal of Trend in Scientific Research and Development*, *Volume-3*(Issue-3), 1569–1572. https://doi.org/10.31142/ijtsrd23456

Wright, D., & Kumar, R. (2023). Assessing the socio-economic impacts of cybercrime. *Societal Impacts*, *1*(1–2), 100013. https://doi.org/10.1016/j.socimp.2023.100013

W. M. Eddy, "RFC 9293: Transmission Control Protocol (TCP)," Aug. 01, 2022.
https://www.ietf.org/rfc/rfc9293.html#name-functional-specification