Homework 3: Developing a Reliable Data Transfer Protocol

Due Date: March 21, 2025

Goal

The IP protocol enables addressing, packet routing, and forwarding in the network as a best-effort service, which does not guarantee packet delivery. In this project, you will use datagrams in UDP to send and receive data and develop your own protocol for reliable data transfer.

Part 1: Reliable Data Transfer

Implement a reliable data transfer mechanism using acknowledgments, retransmissions, and timeouts. The protocol will include features such as data integrity checking, sequence numbers for reordering packets etc. For better link utilization your protocol should support multiple packets that are sent by the sender which are not yet acknowledged by the receiver. You don't have to implement any congestion control algorithm. Because of this, for the purposes of this assignment, just use a very low sending rate by waiting for sometime between packet transmissions (for example less than 500 bits per second)

For testing/debugging purposes, write a Python program that acts as an intermediary between two entities communicating using your reliable data transfer protocol. This program should receive packets from one entity and forward them to the other, simulating network conditions. It must have the ability to introduce loss, reordering and corruption of packets. The purpose of this simulator is to test and evaluate the reliability of a data transfer protocol by emulating various network impairments.

Part 2: File Transfer Application

Develop a simple file transfer program (i.e. reading and writing files to a remote server) using your reliable transfer protocol. You will have to write a client and server program for this part.

Note: If any functionality is not explicitly mentioned, you are free to make design decisions as long as the core requirements are met.

Instructions

1. **Programming Language:**
    a. You must use **Python** to implement your code (Useful link: https://www.python.org)
    b. Follow the PEP 8 coding guidelines for writing clean and professional Python code. (Here is a link: https://peps.python.org/pep-0008/)
    c. Use `git` for version control of your code
2. **Code Documentation and Comments:**
    a. Properly comment your code to make it clear and understandable
    b. Generate a PDF documentation for your code using a tool like Sphinx that automatically generates it for you. (see this link: https://www.sphinx-doc.org/en/master/). There are other tools that provide similar functionality (i.e. automatically generates documentation for you). You can use any of them if you dont like sphinx
3. **What to Submit:** All files must be submitted in a **single zip file**. Name the zip file <firstname>_<lastname>_hw3.zip (replace <firstname> and <lastname> with your name). The zip will have **atleast the following files**:
    a. **Python code**: Python files containing the code you wrote
    b. **Requirements File**: Submit a `requirements.txt` file listing all the Python dependencies required to run your code. (See this link: https://www.geeksforgeeks.org/how-to-create-requirements-txt-file-in-python/)
    c. **Readme**: A `README.md` file that clearly explains: 1- How to compile and run your code. 2- Examples of command-line usage
    d. **Report (PDF)**: The report must be in pdf format. It should include:
        i. Screenshots showing certain packets were corrupted and your protocol was able to recover from it
        ii. Screenshots showing certain packets were lost and your protocol was able to recover from it

   iii. Screenshots showing certain packets were reordered and your protocol was able to recover from it
   iv. Screenshots showing your protocol successfully wrote and read a file from a remote server
e. **Code Documentation**: It must be in PDF format. Use a tool like Sphinx to automatically generate documentation. You can use any other similar tool if you want.
f. **GIT Log**: Use Git for version control and make commits at appropriate points during development. Submit a git log file showing the commit history. This file should be name `revisions.txt`