Project 2: UDP with Go-Back-N

Kacey Ryan
Greg Futami

Design and Implementation

- Sender sets up socket and binds it
- Receiver sets up socket and requests a file
- Sender divides the file into packets if it exists and starts to send them to the receiver based on the window size.
- The receiver ACKs each packet if it is the one it is expecting, and then writes the data to a file. If it's not the packet it is expecting it ignores it. And if it is corrupt it states it is corrupt and then ignores it.
- The sender continues to receive ACKs and slide the window except once it times out, it restarts from the base of the window.
- The sender marks the final packet so the receiver knows to close the file and the sender can reset.

Experiences Gained

- Start Early on Code!!
  - Debugging with network issues is a lot more tedious than debugging syntax and logic errors. With UDP this added a lot more room for error!
- We learned how to build a protocol from scratch without a browser interface. This added some complication but I felt it showed a more in depth side of networking.
- UDP is a much more tedious protocol but one can see its advantages once coding it. It is much more versatile because of its simplicity. Also for streaming and speedy transfer with little care over reliability it is simple to see why UDP is used.

Lesson Learnt/ Suggestion

- Learned that networking code is difficult to debug.
- UDP interface can be done without browser interaction and be quite successful with a reliable data transfer protocol.
- Some suggestions:
  - Some aid to approach would be helpful. This project was a lot more open ended in the students' approach. This made the understanding of what was expected a little confusing. Maybe a skeleton code such as for project 1, or a guide on how to convert project 1's code to support the features of project 2 would be nice.
  - Also, a little more direction on what is expected of Go Back N and Selective Repeat because in OH, the TA stated we could ignore certain aspects of Go Back N.