CS 118 - Fall 2014 - Project 2

Students

Joseph Cox	Chih Chang
703956821	304541390

Implementation

Header Format

```
struct packet
{
    int seq_num;

    short flags;

    int d_length;
    char data[DATA_SIZE];
};
```

The header is as written to the left. It contains a 4-byte sequence number, which the sender and receiver use to communicate which packets are being sent and received. A 2-byte short is used to store up to 8 1-bit flags, including SYN, ACK, FIN, and NONE. A 4-byte int is used to denote how much data the packet is actually transferring (max DATA_SIZE), and then there is a char array for the data of size DATA_SIZE. The packet's max size is 1000 bytes, and because the seq_num, flags, and d_length fields take up 10 bytes, DATA_SIZE is 990 bytes.

Establishing Connection

The connection is established by the receiver first sending a SYN packet to the sender, which contains the requested filename in the data field. Once the sender receives the SYN packet, it opens the file if it exists and begins sending DATA packets (which have no flags set). If the file doesn't exist, the sender sends a NONE packet to the receiver and exits. The receiver receives the NONE packet and exits.

Go-Back-N

We use GBN to guarantee reliable data transfer. We use a sliding window of user-specified size cwnd, divided by the packet size to get the number of packets that will fit in the window. The sender sends the correct number of DATA packets based on the window size and slides its window only when it receives an ACK for the first packet in the window. If the sender receives a duplicate ACK, loses a packet, or receives a corrupted packet, it does nothing and waits for its timeout, at which time it resends all the packets in its window. The receiver ACKs DATA packets with sequence numbers that match the expected sequence number. Any other DATA packets or a corrupted packet result in the sending of a duplicate ACK.

Teardown

The connection is torn down by the sender, which sends a FIN packet once it reads an EOF from the file. Once the receiver gets the FIN packet, it sends back a FIN-ACK packet and exits. Once the sender receives the FIN-ACK, it tears down its connection.

Difficulties

Our primary difficulty was on the sender side with the timeout. We initially forked a new process so that the recvfrom call would not block the timer and used signals to manipulate the timer from the forked listener process. We had so many time-related issues that we switched to a single process and began using siginterrupt() instead of signal(), so that our blocking recvfrom call would be interrupted on timeout.