CS 118

Project 2: Go-Back-N over UDP

Partner #1: Hang Yeh

UID: 804169208

SEASnet: yehh

Partner #2: Emily Hwang

UID: 504158819

SEASnet: emilyh

# Implementation

**Header Format**

Inside our header, we include:

packetType:

0: request from the client

1: ACK packet

2: DATA packet

3: FIN-ACK packet

4: Error handling

size: the size of the actual data. If it’s a ACK packet then the size is zero.

file[PACKETSIZE]: it’s a character array that stores the data

seqNum: the sequence number of the packet

ackNum: the acknowledgment number of the packet, which is the expected value of the sequence number

fin: it’s a fin flag that indicates the end of the file

**Messages**

On the Server side:

./server <port\_number> <probability of lost> <probability of corruption>

Wait for request message…

Upon receiving request message, send ACK to client.

Open file descriptor to file and read in 1024 bytes at a time.

Immediately send the first WINDOW\_SIZE data messages.

For each ACK received:

if ack number == base, send one more packet and slide window.

if ack number > base, send ack number – base packets and slide window

if ack number < base, send last sent packet ?????

When file transfer complete, send empty message with just a FIN flag.

Set timer and wait for FINACK

When receive FINACK, send ACK message.

Set timer.

When timer expires, close connection.

On the Client side:

./client localhost <server\_port\_number> <filename> <probability\_of\_lost> <probability\_of corruption>

Client sent request message

Upon receiving the acknowledgement from the server, client wait for the data package

After receiving the DATA packet, check for the sequence number. If the sequence number doesn’t match, ignore the packet

Client sent ACK message to indicate received data packet

When the transfer is completed, an FIN message is sent the client and client respond with a FIN-ACK packet

Client received an acknowledgement from server and closes the connection

**Timeouts**

A timer is set for:

each ACK the client sends to the server

the oldest unACKed packet in the server

the last ACK after FINACK sent by server

The select function is used.

**select** (*intnfds, fd\_set \*read-fds, fd\_set \*write-fds, fd\_set \*except-fds, structtimeval \*timeout*)

select() returns zero when time out, -1 for error or > 0 if a signal is detected in the file descriptor’s change of state. We use this function before recvfrom() to receive the incoming packet in the socket.

Time out value is set to 2 seconds.

**Window-Based Protocol:**

We use the Go-Back-N protocol

# Difficulties

1. Keeping track of the ACK number and the SEQ number: we had difficulties in figuring out the correct ack numbers and sequence numbers for the client and server. It was difficult figuring out who had to keep track of what numbers. We solved it by basically reading the go-back-n protocol over and over again until we completely understood how the numbers were supposed to work.
2. select() function: We had difficulties using the select() function mainly because we didn’t really understand how it worked. It gave us a lot of bugs while we were learning how to use it. For example, we didn’t realize we had to reset the descriptors and timeout each time we called it. We solved this difficulty by basically doing a lot of Googling.
3. We had difficulties figuring out if there was a way to effectively pause and resume the timer in the select function once activity has occurred in the socket. We solved it by just allowing the timer to reset each time.