18-345 Project 2 - Protocol Design

Michael Nye (mnye) and Michael Ryan (mer1)

Our UDP Header will consist of four four-byte integers.

First: ClientID number from the originating backend server.

Second: Sequence number, for tracking in our ARQ. This is linked to the ClientID and the sender's IP address.

Third: A RequestID assigned by the backend server that uniquely identifies the transaction of one file. Before one has been assigned, it is set to 0.

Fourth: UDP Packet type (mapped to integer values), one of:

> REQUEST: A request from a frontend to a backend peer for a file
> DATA: File sent to a reqeuster
> ACK: Acknowledgement of recieval for DATA packet
> NACK: Acknowledgement of a packet we need but haven't received
> KILL: A message from a frontend informing a backend peer to stop transmitting its file in case of disconnect

SENDER:
Many threads, one packet. Waits for a packet to send

RECEIVE ACK
Mark packet as received. Refuse to resend later

WAIT:
Sends packet, sets timeout. Waits for response.

RECEIVE NACK
Trigger a timeout early to resend

TIMEOUT:
Attempt to re-send

RECEIVER:
One thread, many clients to pass packets to.

WAIT:
Listens on specified port for incoming packets

INCOMING:
Receives packet. Uniquely identifies its client using ClientID and RequestIP.

PASS-ON:
Using unique ID, hands off packet to a ClientHandler, who adds it to a priority queue or spins off a RequestHandler, depending on UDP Packet Type

CLIENTHANDLER:
Keeps PriorityQueue of packets sorted by the packet's Sequence ID

RECEIVE:
RECEIVER gives us a packet to sort. ACK the packet. Also NACK a missing a packet if we have not received something before this newly received sequence number.

FLUSH TO CLIENT:
Add to PriorityQueue. If the sequence number is the next one we haven't seen, flush all consecutive packets from queue to client