client index
/Users/aayansayed/Documents/CSCI - 351/Sayed_Aayan_hw03/HW_03/Reliable-UDP-Protocol/Part_2/client.py

Modules

os sys
socket threading time

Functions

createPacket(data, seqNum)
Function called from the send_packet which takes in
already split data and assigns it a sequence number
and checksum to create a packet to send.
Params:
Data: Split data for 1 packet

generateCheckSum(data)
Function called from the createPacket function which takes in
already split data and generates a checksum for it.
Params:
Data: Split data for 1 packet

handle_timeout()
Function called when packets have been sent but Acks havent been
received. Resends all packets in the current window.

listen_for_ack()
Function called from send_packet which operates on its thread
and listens for Acks send from previously sent packets

main()
Main function that reads in a filePath and
creates packets which are sent to the server

send_packets(all_packets)
Function that takes in a list of packets and sends it to
the server with a generated checksum and expected seq num

split_into_packets(filepath, packet_size)
Helper function with takes in the contents of a file
and splits it into an array of length 5

Data

base = 0
client_socket = <socket.socket fd=3, family=2, type=2, proto=0, laddr=('127.0.0.1', 6969)>
lock = <unlocked _thread.lock object>
next_seq = 0
num_packets = 0
sent_packets = {}
server_address = ('127.0.0.1', 7979)
timer = None

mediator index
/Users/aayansayed/Documents/CSCI - 351/Sayed_Aayan_hw03/HW_03/Reliable-UDP-Protocol/Part_1/mediator.py

Modules

random socket time

Functions

simulateNetwork()
Function that listens for packets from the sender and simulates the network conditions.

Data

CORRUPT_PROB = 0.2
LOSS_PROB = 0.5
PACKET_SIZE = 5
REORDER_PROB = 0.2
expected_seq_num = 0
mediatorPort = 9000
receiver_address = ('127.0.0.1', 10000)
reorder_buffer = []
socket_mediator = <socket.socket fd=3, family=2, type=2, proto=0, laddr=('127.0.0.1', 9000)>

receiver index
/Users/aayansayed/Documents/CSCI - 351/Sayed_Aayan_hw03/HW_03/Reliable-UDP-Protocol/Part_1/receiver.py

Modules

random socket time

Functions

compute_checksum(data)
Function recomputes the checksum for the data to
check for corrupted data.

receiver()
Function that listens for packets and accounts for Dropped, reordered, and corrupted packets.

Data

PACKET_SIZE = 5
buffer = {}
expected_seq_num = 0
received_packets = []
receiverPort = 10000
sender_address = ('127.0.0.1', 8000)
socket_receiver = <socket.socket fd=3, family=2, type=2, proto=0, laddr=('127.0.0.1', 10000)>

sender index
/Users/aayansayed/Documents/CSCI - 351/Sayed_Aayan_hw03/HW_03/Reliable-UDP-Protocol/Part_1/sender.py

Modules

socket struct threading time

Functions

createPacket(data, seqNum)
Function called from the send_packet which takes in
already split data and assigns it a sequence number
and checksum to create a packet to send.
Params:
Data: Split data for 1 packet

generateCheckSum(data)
Function called from the createPacket function which takes in
already split data and generates a checksum for it.
Params:
Data: Split data for 1 packet

handle_timeout()
Function called when packets have been sent but Acks havent been
received. Resends all packets in the current window.

listen_for_ack()
Function called from send_packet which operates on its thread
and listens for Acks send from previously sent packets

main()
Main function that reads in input from user and sends to the mediator

send_packet(data)
Send Packets is a function that takes in data from the user
splits it into packets and starts a thread which listens for Acks
before sending all the packets to the mediator

Params:
Data: A string read in from the user

Data

base = 0
lock = <unlocked _thread.lock object>
next_seq = 0

num_packets = 0
packetSize = 5
receiver_address = ('127.0.0.1', 9000)
sent_packets = {}
sock = <socket.socket fd=3, family=2, type=2, proto=0, laddr=('127.0.0.1', 8000)>
timeBetweenPackets = 5
timer = None
windowSize = 5

server index
/Users/aayansayed/Documents/CSCI - 351/Sayed_Aayan_hw03/HW_03/Reliable-UDP-Protocol/Part_2/server.py

Modules

random socket time

Functions

compute_checksum(data)
Function called from the createPacket function which takes in
already split data and generates a checksum for it.
Params:
Data: Split data for 1 packet

server()
The server listens for incoming packets from the client and has code
which account for corrupted, out-of-order, and reordered

write_file(received_packets)
Fuction with takes all the packets and writes them to a
file in the parent directory

Data

PACKET_SIZE = 5
buffer = {}
client_address = ('127.0.0.1', 6969)
expected_seq_num = 0
received_packets = []
server_port = 7979
server_socket = <socket.socket fd=3, family=2, type=2, proto=0, laddr=('127.0.0.1', 7979)>