**Demonstrate TCP retransmission and TCP fast retransmit scenarios in a client-server architecture.**

## Requirements:

1. Create a C-based client-server architecture using sockets
2. The server should be able to accept and service single client’s requests
3. The server should run on server machine and the client should run on client machine
4. Demonstrate TCP retransmission when a segment is lost for a given window size
5. Demonstrate TCP retransmission when an acknowledgement is lost for a given window size
6. Demonstrate TCP fast retransmit when a segment or an acknowledgment is lost for a given window size.

## Procedure:

1. Create a C-based server that can accept a single client’s request using sockets
2. Make sure the server runs on server machine and the format to start the server is as follows

server <port\_number> window size: timeout:

where server is the server executable and port\_number is the port number on which the server listens. The window size is a non-negative number greater than zero and less than 11. Timeout is a non-negative number greater than zero and less than 11 seconds.

1. Create a C-based client that can connect to the server using sockets
2. Make sure the client runs on client machine and connects to the server.

client <port\_number> test conditions:

where client is the client executable, port\_number is the port number on which the client is connected to the server and test conditions are the test conditions given to the client (receiver)

1. The server is trying to send 2048 bytes of data to the client. The fixed segment size in the demonstration is 128 bytes
2. Create a 2048-byte text file and transmit the file to the client as part of the demonstration
3. Before starting the transmission, set the window size on the server and set only one test condition on the client
4. The test conditions are (1) SL num (2) AL num where SL stands for segment loss and num stands for the sequence number (ex: 0, 128, 256, or etc) and AL stands acknowledgment loss and num stands for the acknowledgement number (ex: 128, 256, 384, or etc)
5. When the first segment is transmitted by the server, the transmit time (time stamp) is recorded and when the acknowledgment is received by the server, the window slides by one segment. Only one segment is transmitted every second
6. Measure the elapsed time after transmitting the segment. If timed out, resend the segment
7. Demonstrate TCP retransmission by setting a segment loss on the client side. When a segment loss is set on the client side with a sequence number, the client drops the corresponding segment once. Dropping a segment is simply not storing the segment in the receive buffer
8. When a segment is lost, the server waits for timeout and then retransmits the segment. The client accepts the received segment and stores it in the buffer
9. Demonstrate TCP retransmission by setting an acknowledgement loss on the client side. When an acknowledgment loss is set on the client side with an acknowledgement number, corresponding acknowledgement is not sent to the server.
10. When an acknowledgment is lost, the server waits for timeout and then retransmits the segment. The client accepts the received segment and discards the duplicate segment.
11. Demonstrate TCP fast retransmit by setting a segment loss or an acknowledgment loss
12. When all the segments are received, the segments are written to a file on the client side
13. Assume the connection management is taken care by the kernel’s transport layer
14. Check to make sure that the sent and received files are the same. Also, test the server and client to make sure the demonstrations are working.

## Deliverables:

1. Commented server and client C code
2. A readme file that describes how to compile, execute, and test the C codes.