Computer Networks Lab (CS 353): Lab 4

This assignment is graded.

Question 1: [7 marks]

The goal of this lab exercise is to **simulate** the Stop-and-Wait-ARQ protocol over sockets, as follows:

A client connects over a TCP socket to a server. After the client (receiver) connects, the server (sender) sends data contained in a file as frames. Each frame from the sender contains the following fields: a sequence number and the information to be sent. The client receives the frames and keeps copying the information received into a file. Each ACK frame from the receiver contains the ackNo.

Implement the protocol such that you can:

1) Demonstrate the normal scenario where the server sends data in the file as frames and acknowledgements are received. **[1 mark for demonstration + 1 mark for viva]**
2) Demonstrate a scenario where the client deliberately skips acknowledging a frame, thus simulating a lost ACK. The sender will timeout and resend the frame it has sent. After the frame is resent, the client will acknowledge it. **[3 marks for demonstration + 2 marks for viva]**

   For both the scenarios, you need to do the following: At the server, you need to print the frames that are sent and the ack frames that are received. At the client, you need to print the frames that are received and the ack frames that are sent. After receiving all frames at the client, compare the file sent from the server with the original file and state whether the files are identical.

Hints:

1. To implement a timer, create a thread called timer_thread in the server. See https://man7.org/linux/man-pages/man2/timerfd_create.2.html for system calls related to timers. The main thread will call timerfd_create() to create a file descriptor fd for a timer. Then it can start a timer using timerfd_settime (), giving fd as the input. In an infinite loop in timer_thread, use poll() to poll fd. If the return result indicates that there is data to read (that is, fd.revents & POLLIN), use read(). If read() returns an unsigned 8 bytes integer, the timer has expired and appropriate action may be taken, such as resending the data.
2. Implement the code for timers first and check if it works. After this, send one frame of data and check if an ack is received. Then send 2 frames, but do not ack the second one.
3. You may use the client-server code written for earlier labs.
4. Use a small text file for demonstration.

Question 2: [3 Marks]
Write and test a program that simulates byte stuffing and byte unstuffing in the data link layer. You can assume your own values for the flag and the Esc bytes.