CS5313: Computer Networks **Instructor: Dr. Deepak Tosh**
Fall 2020 **Due Date:** 10/02/2020

Assignment # 2

GOAL:

The goal of this assignment is to implement both non-pipelined and pipelined versions of the reliable data transfer protocol: (1) Stop-and-wait protocol and (2) Go-Back-N (GBN) Protocol. Additionally, interested students can attempt to implement the Selective Repeat (SR) protocol for **bonus**. We know that GBN and SR use sliding window to send more than one packets to a receiver over an unreliable medium where packets can get corrupted and lost too. However, the Stop-and-Wait protocol only sends one packet at a time and waits for the ACK to be received until it attempts to send the next packet.

Disclaimer: This is a group assignment. Each group has two members. Your groups are pre-decided and should be activated in the Blackboard where you can interact with your teammate.

Environment Setup:

- The starter code (in Python) is given. There are 5 important python programs. You can run them on same machine or on two separate VMs.
- The "receiver.py" needs to be running first, and then you can run the sender program ("sender.py") to send any number of packets.
- When you run both programs for the first time, you will observe that the sender sends 20 packets of random payload, and receiver prints them upon receival. This is given to make you understand how those two programs use the other auxiliary programs (*udt, packet, and timer* modules).

(**Note**: you are given with 3-auxiliary helper modules: ***timer***, ***packet***, ***udt***).

- Timer module: Provides *start(), stop(), timeout(), running()* functions
- Packet module: Provides *make(seqNo, data), extract(packet), make_empty()*
- udt module: Provides *send(packet, sock, addr), rcv(sock)* to send/recv from unreliable channel

Task-1 [File Transfer using Stop-and-Wait (SnW) Protocol] (40pts)

In this part, your job is to implement the stop-and-wait protocol that we discussed in the class to transfer a file ("*Bio.txt*"). The state diagram of this protocol is same to **RDT3.0**, which you can refer from Section 3.4 of our textbook.

(i) Implementing SnW Sender

In this task, you will program following python functions in *"Sender.py"*.

a. Modify the function "*send_snw(sock)*" that will read the file "*Bio.txt*" to transfer it by following the stop-and-wait protocol. Note that each packet should have a payload of 512 Bytes, which is defined in PACKET_SIZE. You need to handle packet losses by using the timer module.

b. Implement the function "*receive_snw(sock, pkt)*" to handle the acknowledgements in stop-and-wait protocol. If the ACK gets lost, this function will retransmit the *pkt* again.

(ii) Implementing SnW Receiver

a. Modify the function "*receive_snw(sock)*" to receive all the payloads in correct sequence and write them into a file "*receiver_bio.txt*" at the receiver, reconstructing the sender's file *"bio.txt"*.

Task-2 [File Transfer using Go-Back-N (GBN) Protocol] (60pts)

(i) Implementing GBN-Sender

a. Implement *send_gbn(sock)* that will read the file "*Bio.txt*" to transfer it to the GBN receiver. Its job is to calculate how many packets to send, and then send N packets based on defined window size. Ensure to implement the actions when packets get lost happens.

b. Implement *receive_gbn(sock)* that checks if the expected ACKs are being received or not. Based on that the sender should take necessary actions.

(ii) Implementing GBN-Receiver:

a. Implement *receive_gbn(sock)* at the receiver which receives the packets and writes them into the file *"receiver_bio.txt"*. Ensure that if the expected sequence number is not received, it sends the last in-ordered packet's sequence # as ACK.

(Bonus) File Transfer using Selective Repeat (SR) Protocol (15%)

Repeat the same task of transferring the file between sender and receiver as above, but this time you need to implement the Selective Repeat (SR) protocol instead of the GBN.

a. Read the SR-sender's events and actions from Fig 3.24 in the Book and modify the "Sender.py" program accordingly.

b. Read the SR-receiver's events and actions from Fig 3.25 in the Book and modify the "Receiver.py" program accordingly.

What to be submitted:

1. A **PDF formatted report** with all evidences, codes, execution samples, instructions for running the submitted programs, and references used. The title should be formatted as "*Group_x_Report_Assign2.pdf*".
2. Well-documented source codes and *README.txt* in the form of **one single Zip** file.

- Each student's contribution should also be reported with a comparative chart as shown below and should be reported through the report.

Task lists	Student-1's contribution	Student-2's contribution
…	…	…

Submission:

Submit your **Python codes** for all the implemented algorithms, *README.txt* and the report on **Blackboard only.**