COMP3825 Course Project (Part A is 2 points.  Part B is 10 points and bonus is 2 extra points.)

**Part A: WireShark Lab - Exploring TCP (page 302 of textbook)**

You’ll use your Web browser to access a file from a Web server. As in earlier Wireshark labs, you’ll use Wireshark to capture the packets arriving at your computer. Unlike earlier labs, you’ll *also*be able to download a Wireshark-readable packet trace from the Web server from which you downloaded the file. In this server trace, you’ll find the packets that were generated by your own access of the Web server. You’ll analyze the client- and server-side traces to explore aspects of TCP. In particular, you’ll evaluate the performance of the TCP connection between your computer and the Web server. You’ll trace TCP’s window behavior, and infer packet loss, retransmission, flow control and congestion control behavior, and estimated round-trip time. Deliverable: submit the lab report by 11:59pm, Saturday Oct. 21.

**Part B: design and implement reliable data transfer using UDP to download files.**

There should be a client program and a server program. The server program hosts files and responds to requests for files. It breaks the requested file into segments and sends them to the client over UDP. The client program takes a file name as input and requests the file from the server. The programs should handle UDP packet losses to make sure that the entire file is correctly received by the client program. Your code needs to include implementation of the **Alternating Bit** and **Selective Repeat** protocols. Please read Chapter 3.4 (Principles of Reliable Data Transfer) to understand these two protocols.

Client program should print detailed information about any requests sent and data received at byte level (e.g., time X received byte Y to Z) during the file download. Server program should print detailed information about any requests received and data sent at byte level.

You can choose the programming language. During your development, you can run the client and server on the same machine. However, for comprehensive testing and evaluation, your client program and server program should be run on different machines in different networks. My research lab has Unix machines that can run your server program, but you need to learn simple Unix commands to login to the machine and run your program. Let our TA know if you need access to my server.

Deliverables:

1. Submit design document for the client and server by 11:59pm, Friday Oct. 27. In addition to text description, use class diagram, flow chart and state transition diagram to explain your design.
2. Submit code framework for both client and server by 11:59pm, Friday Nov. 3. It should have the major data structures and functions (APIs) defined based on the design document.
3. Submit (a) code for Alternating Bit implementation, (b) instructions on how to install and run the program, (c) screenshots of the input and output by 11:59pm, Friday Nov. 10.
4. Submit (a) code for the Selective Repeat implementation, (b) instructions on how to install and run the program, (c) screenshots of the input and output by 11:59pm, Friday Nov. 17.
5. Submit (a) final report (>= 3 pages) including design & implementation details, (b) Bonus: run your programs to compare the performance of the two protocols by downloading different files with different sizes and include your evaluation results in the report by 11:59pm, Friday Dec. 1.

The Computer science department has a cluster for class usage. Your server-side program needs to be tested and run on the cluster. The client side can be run on your computer/laptop (connected to the UofM VPN). Here is the getting started page with all the details about the cluster.

<https://docs.google.com/document/d/136bAS2Ln-9Banb1imkTFLCmBNwGhuPCaKpHcYnaGO2o>