Sabir Kirpal
11-23-2021
CSE 160 Proj 3 Design

For project 3, we were required to design a TCP like reliability layer in our network. To do this we had to use sockets which bound to file descriptors or FD's. To establish a socket the server node must have a listening socket available to take requests from a client node. The client node must send a SYN to the server in order to begin this process. Next the server must reply with a SYN+ACK message back to the client. The client uses some information about the destination and returns a SYN_EST or established flag in order to notify the server that the three way hand shake has completed. From here both nodes theoretically have an established bit stream. To implement reliability, we must send only a certain "effective window" amount of bits from the socket. As we keep getting replies from client that data is being received the sliding window increased the left pointer over and sends one more message from the right side of the window. Using this we are sure that this transportation mechanism will reliably be able to send as much data as it needs without losing packets or stressing out the network.

Discussion Questions

1. The sequence number should always begin with 1 especially in a test environment because it is easier to debug. There is no performance benefits to starting with different sequence numbers.
2. This buffer should be picked as the difference between Last Written and Last Read which is considered to be the window in our project.
3. The server receiving the SYN attack packs will eventually run out of space for sockets and will overload the network. One way I can redesign this is to keep track of a tuple (lastSYNsrc, count). Every time the server receives a new SYN it can check if the src is the same as lastSYNsrc and if it is increment count. If it isn't update lastSYNsrc and count = 1. If count ever hits around 4 you know that node is up to malicious activity.

4. If the sender fails to close a socket, that socket will remain open and other sources may be able to transfer data to it. One way to combat this is to shut off the socket after a set amount of time that new data isn't read.