Raivat Alwar, Ritesh Patro
Project 3 Discussion Questions

Discussion Questions:

1. Your transport protocol implementation picks an initial sequence number when establishing a new connection. This might be 1, or it could be a random value. Which is better, and why?

2. Your transport protocol implementation picks the size of a buffer for received data that is used as part of flow control. How large should this buffer be, and why?

3. Our connection setup protocol is vulnerable to the following attack. The attacker sends a large number of connection requests (SYN) packets to a particular node but never sends any data. (This is called a SYN flood.) What would happen to your implementation if it were attacked in this way? How might you have designed the initial handshake protocol (or the protocol implementation) differently to be more robust to this attack?

4. What happens in your implementation when a sender transfers data but never closes a connection? (This is called a FIN attack.) How might we design the protocol differently to better manage this case?

---

1. In regards to the sequence number, I would say having the sequence start from a random value would be the more beneficial option. The reason this is better is because it reduces the likelihood of the host confusing the new sequence number with a previous sequence that came from an earlier connection to the same client.

2. The buffer size of received data depends on many factors such as how much data is being sent and the type of data being sent. As of project 3, we are only sending numbers to a maximum value of 1000. In this case, it would make things simpler if the buffer size could hold at least 1000 values so at least 2^10.

3. My implementation continues to send SYN_ACK packets back to the initial source. In the case it gets attacked by a SYN flood, I would continue to open new connections and send SYN_ACK packets until the host is unable to open any more connections. To make it more robust to this attack, I could have made it so that, at the arrival of the SYN packet, I check to see if a connection between the server and client is already existing. In the case that a connection already exists, I can discard the SYN packet without returning anything. This will prevent another connection from getting created.

4. To better manage this case, we could utilize a timer to close the connection. In the case that data has not come into the node for a while, we can close the connection between the client and server. An issue that can come with this method however is in the case that a data packet takes a long time to arrive at its destination, the connection might already be closed. To mitigate this issue, we can have the timer wait a time of 3 times the rtt and close the connection is no data packet is received.