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

It is better to pick a random sequence number because if you always start with the same sequence number, packets from previous connections have a higher chance of accidentally coinciding with the sequence numbers of the ongoing connection, which can result in unexpected behavior

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

The buffer should be as large as the remaining amount of space in the memory buffer on the receiving end, because if more data is sent than the receiver can store, then that data will literally be unable to be stored.

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?

If 10 SYN packets are sent to a listening port then never used, those connections will never close automatically, so the node will be completely incapable of forming any more connections until it is rebooted. One way of defending against this attack is to automatically garbage collect sockets which have not been synchronized after a long time, by taking a timestamp at the point when the first SYN packet is received.

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?

If many senders send data and then never close their connections, the node will run out of space for sockets. We can stop this from happening by garbage collecting sockets which are ESTABLISHED but have not received any data in a long time, by timestamping the last point in time a packet was sent on that socket connection.