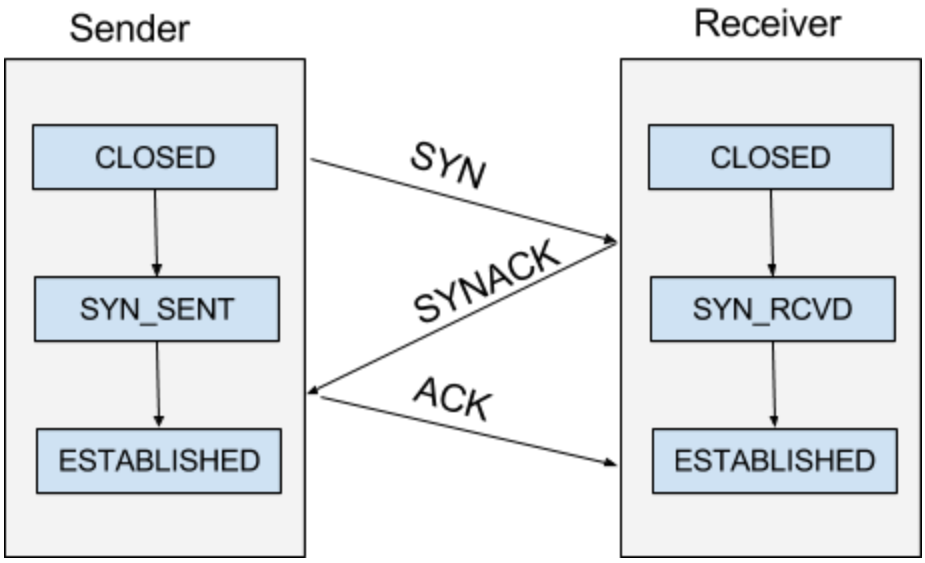
CS5450 Networks & Distributed Computing
Project 2 : Go-Back-n
Chumeng Xu, cx87@cornell.edu
Saran Kumar Krishnasamy, sk2998@cornell.edu

3-way handshake -

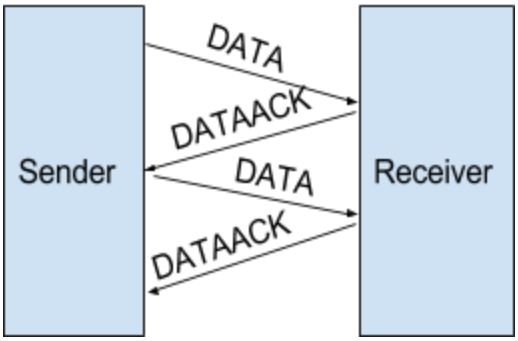
In order to establish a connection, both the sender and receiver must send a *SYN* and receive an *ACK* for it from the other party. This means a total of 4 messages. However, in a 3-way handshake mechanism, one of the *SYNs* and one of the *ACKs* is sent together(*SYNACK*).



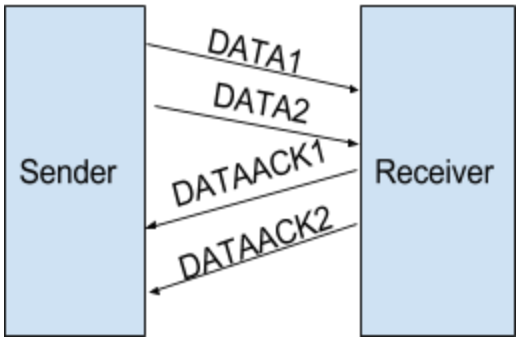
Both the Sender and Receiver start with an initial state *CLOSED*. The sender initiates the connection by sending a *SYN* packet and moves into the *SYN_SENT* state. The receiver on getting the sender's *SYN* packet , changes it's state to *SYN_RCVD* and sends a *SYNACK*(which comprises of the *ACK* for sender's *SYN* and a *SYN* for the receiver) . When the sender receives the *SYNACK* , the state changes to *ESTABLISHED* and sends over a *ACK* to the receiver. Once the receiver obtains the *ACK* , state changes to *ESTABLISHED* and data transfer can begin.

Data Transfer -

As soon as the connection is *ESTABLISHED,* data transfer begins. The data is split into packets of size *DATA_LEN* and is sent by the gbn_send() to the receiver. The gbn_recv() of the receiver on receiving a *DATA* packet will send a *DATAACK* to acknowledge the packet. This process continues till the receiver gets a *FIN* packet which signals the beginning of connection teardown process.



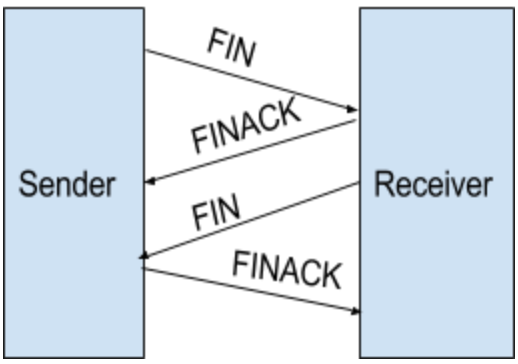
1. Slow mode (go-back-1)



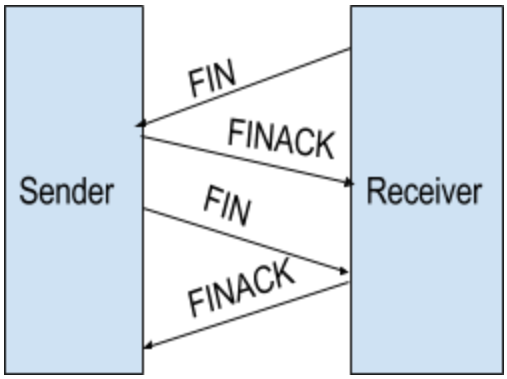
2. Fast mode (go-back-2)

The go-back-n system implemented in this project has 2 modes -
Slow mode : the next packet will not be sent until the sender receives acknowledgement for the previous packet.
Fast mode : a second packet may be sent before receiving an acknowledgment for the previous packet.

Connection Teardown -
Once the sender sends over all the data, it sends a *FIN* packet to the receiver that serves as a connection termination request to the receiver. The receiver on getting the *FIN* responds with a *FINACK* to indicate that it was received. The connection as a whole is not considered terminated until both sides have finished the shutdown procedure by sending a *FIN* and receiving an *ACK*.
Ideally the party that wants to finish the connection sends a *FIN* to the other end.



2. When sender finishes sending all the data, it initiates teardown.



2. When receiver finishes before sender(eg.: receiver is unable to open outputfile)

Tricky parts of the implementation -
The actual data transfer in the system was one of the tricky parts of the implementation - particularly the logic for the **gbn_send() :** figuring out the seqnum when the data is huge(Giant

test cases). Clearly we had to consider buffer overflow in the case of seqnum size. One approach we took was to reset the state.seqnum to 1 once we had reached 1024 packets.

Test cases -

./receiver 10000 output
./sender localhost 10000 Tiny1.test
diff Tiny1.test output

Input	sendto( )	maybe_sendto( )
Tiny1.test	Pass	Pass
Tiny2.test	Pass	Pass
Tiny3.test	Pass	Pass
Tiny4.test	Pass	Pass
Small1.test	Pass	Pass
Small2.test	Pass	Pass
Small3.test	Pass	Pass
Small4.test	Pass	Pass
Large1.test	Pass	Pass
Large2.test	Pass	Pass
Large3.test	Pass	Pass
Large4.test	Pass	Pass
Giant1.test	Pass	Pass
Giant2.test	Pass	Pass
Giant3.test	Pass	Pass
Giant4.test	Pass	Pass

Known issues-

If maybe_sendto( ) messes up packets during 3-way handshake in gbn_connect( ), system fails. Didn't handle that scenario. All other cases have been handled.