**ASSIGNMENT 1**

STP over UDP

COMP3331

MICHAEL JIN

3216720

**1. Features Implemented**

* Initiate connection establishment with a 3way handshake.
* End connection using a 4-segment connection termination.
* Sender has a single timeout operation. If timeout occurs, the sender will send the last unacknowledged packet again.
* Cumulative acknowledgement. The sender will send multiple packets within the window to send packets. The sender will then wait for all of the acknowledgements before sending the next set of packets.
* The sender will fast transmit any packets which are not acknowledged. Duplicate ACKs received will indicate packets been dropped.
* Receive will immediately acknowledge packets received and will have a window of how many packets is received. Any packets sent outside the window will be ignored.
* The maximum segment size can be set on the sender
* Maximum window can also be set to the sender
* PLD has been implemented to mimic packets dropped.

**2. STP**



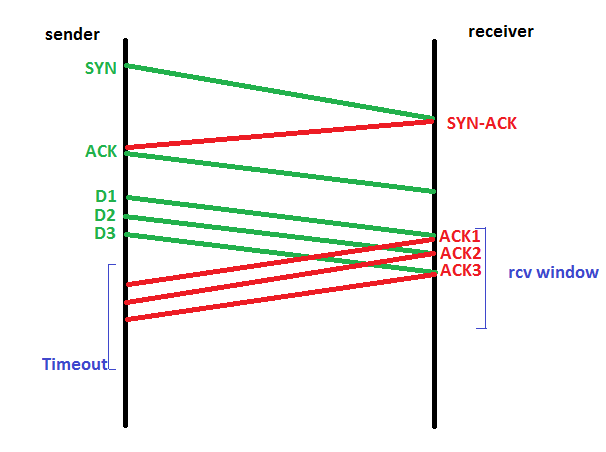
Packet Type is the type of packet sent. Data, SYN, ACK, FIN

The Pkt Number is use to track the order of the packets

Seq is the sequence number of the packet using the bytes progression

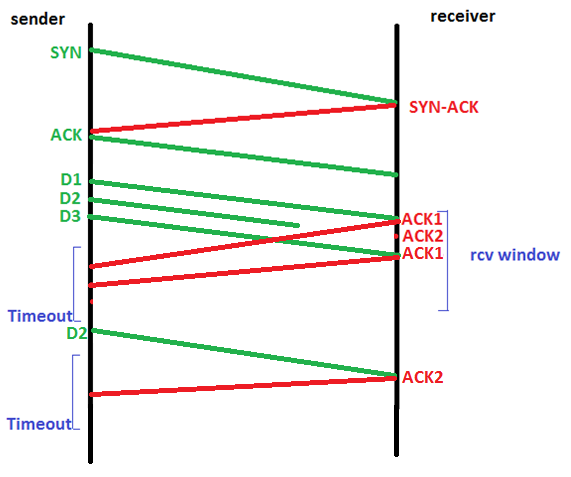
Data is the content sent. SYN, FIN will not have data

Following diagram shows packet transfers.



If D2 is not received but D3 is received. The receiver will send ACK1 again to indicate that D2 is not received but D3 is received. If both D2 and D3 is not received then sender will wait till timeout and send D2 and D3 again.

Following diagram shows packet loss.



In the case of a packet is lost, using cumulative ACKs, the receiver sends back an ACK of the previously received packet. This will let he sender know that a packet was receive but a packet was also lost. The packet lost would be the one numbered after the previously received ACK.

**3a. Timeout**

Timeout length is important as it deals with sender sending duplicate packets receiver may have already received. If 3 packets are sent successfully, depending on the size of the packets, the receiver may not be able to return an ACK for the 3rd packet before the sender times out. This will cause the sender to send the 3rd packet again.

However, if the timeout is too long the sender may be wasting time waiting for an ACK that is never going to be sent. Using the example of sending 3 packets again. Assume that D3 is dropped which mean the receiver will not send an ACK for D3. The sender will have to wait until timeout before sending D3 again.

To determine the length of timeout, it is important to know the size of the packets sent and the time it usually take to receive an ACK. If we are sending 3 packets of 20bytes and it takes 5milliseconds to receive 20bytes and return an ACK. We can set timeout to a little over 15milliseconds. This will give us enough time to receive the ACKs before needing to send the packets that did not receive ACKS.

**3b.**

*Timeout = 100:*

TIME: 0.0273

Total number of bytes receives: 2510

Total number of segments receives: 36

Total number of duplicates receives: 0

*Timeout = 400:*

TIME: 0.0318

Total number of bytes receives: 2510

Total number of segments receives: 36

Total number of duplicates receives: 0

*Timeout = 25:*

TIME: 0.0179

Total number of bytes receives: 2510

Total number of segments receives: 36

Total number of duplicates receives: 0

The only difference between the different tests sending test2.txt is the time it took to complete the transfer. As you can see there are no duplicates in the different transfers. However, I believe if the files were bigger or if there was a bigger delay, there would be duplicates in the tests caused by timeout on the sender end. The sender would not receive an ACK and would therefore send the past packet again.