# Section #6 Handout

## 1. TCP Sequence Numbers

A TCP connection has been established between hosts A and B. B receives a packet from A with the following field values shown below,

Sequence: 101
Acknowledgement: 10001
Window size: 5000
Payload size: 1400

B has processed this packet and updated its position of its sliding window. Everything is in terms of bytes.

(a) B wants to send out the next packet to A. What is the lowest sequence number that B can send out in the next packet?

(b) What is the highest sequence number that B can send?

(c) What is the acknowledgement number sent along with the next packet?

(d) Suppose B sends out the following packet to A:

Sequence: 13001
Acknowledgement: [Answer to part (c) above]
Window size: 50
Payload size: 1400

What is the range of sequence numbers that A can send in response?

# Congestion Control

In lecture we considered the case where CWND is expressed in terms of packets (MSS). We now consider the more realistic case where CWND is expressed in terms of bytes. In particular, assume ***1 MSS = 100 bytes***.

## 2a. Slow Start and Advertised Window

Doc needs to send data to Dopey. Doc wants to send the data as fast as possible, but doesn't want to overwhelm Dopey, so Doc decides to use ***slow start***. Dopey has a ***receive window size of 450***. However he's a little confused and only removes data from the buffer when it is full.

Write down the specified fields of Doc and Dopey's sent packets and state. If no packets are sent, mark it explicitly.

*Hint:*

- During slow start, **CWND += MSS** on new ACK
- Sending window = **min(CWND, RWND)**

### Doc (sender)

### Dopey (receiver)

| In response to which ACK? | CWND, RWND | Seq # | Payload size | | ACK | Advertised Window |
|---|---|---|---|---|---|---|
| --- | 100, 450 | 1 | 100 | | | |
| | | | | | | |
| | | | | | | |
| | | | | | leave blank | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

## 2b. Additive Increase and Simple Fast Retransmit (Reno)

Sneezy needs to send data to Grouchy. Sneezy doesn't want to risk getting Grouchy too upset with missing packets. So he uses ***simple fast retransmit***.

Assuming in additive increase state, fill in CWND and SSTHRESH values for the following series of ACKs. When dividing integers, use round it down (e.g. 55/10 = 5).

*Hint:*

- During congestion avoidance, **CWND += MSS /** $\lfloor \mathbf{CWND}/\mathbf{MSS} \rfloor$ on new ACK
- On triple duplicate ACK, **SSTHRESH = CWND/2**, then **CWND = SSTHRESH**

| ACK | CWND | SSTHRESH |
|---|---|---|
| 801 | 1000 | 800 |
| 901 | | |
| 1001 | | |
| 1101 | | |
| 1101 (1) | | |
| 1101 (2) | | |
| 1101 (3) | | |

## 2c. Advanced Fast Retransmit (NewReno)

But Grouchy is still upset whenever there is a loss, which is manifest as a period of silence on the receiver side. So Sneezy uses ***advanced fast retransmit***.

Assuming in additive increase state. Do the same as in the previous part.

*Hint:*

- In congestion avoidance, handle acks as before
- On triple duplicate ACK, **SSTHRESH = CWND/2,** then **CWND = SSTHRESH + 3 MSS**
- In fast recovery, **CWND += MSS** on every dupack
- Exit fast recovery on new ack, setting **CWND = SSTHRESH**

| ACK | CWND | SSTHRESH |
|---|---|---|
| 801 | 1000 | 800 |
| 901 | | |
| 1001 | | |
| 1101 | | |
| 1101 (1) | | |
| 1101 (2) | | |
| 1101 (3) | | |
| 1101 | | |
| 1101 | | |
| 1101 | | |
| 1801 | | |
| 1901 | | |

## 2d. Loss by Timeout

Finally, Bashful is sending a file to Happy, but Sleepy is carrying the packets. Once in a while Sleepy unfortunately takes a nap and drops the packets he's carrying on the floor and they are lost. Bashful is embarrassed whenever he catches Sleepy asleep on the job and is very slow to ask the still waking up Sleepy to carry packets. To make the most of it Bashful uses **TCP loss by timeout**.

*Hint:*

- On timeout, **SSTHRESH = CWND/2**, then **CWND = 100**
- During slow start, **CWND += MSS** on new ACK
- During congestion avoidance, **CWND += MSS /** $\lfloor \mathbf{CWND}/\mathbf{MSS} \rfloor$ on new ACK
- The algorithm leaves slow-start when **CWND >= SSTHRESH**

Assuming in additive increase state, fill in CWND and SSTHRESH's value for the following series of ACKs.

| ACK | CWND | SSTHRESH |
|---|---|---|
| 801 | 800 | MAX_INT |
| Timeout! | | |
| 901 | | |
| 1001 | | |
| 1101 | | |
| 1201 | | |
| 1301 | | |