Assignment #5

1. Suppose two hosts have a long-lived TCP session over a path with a 100ms round-trip time (RTT). Then, a link fails, causing the traffic to flow over a longer path with a 500ms RTT.



1. Suppose the router on the left recognizes the failure immediately and starts forwarding data packets over the new path, without losing any packets. (Assume also that the router on the right recognizes the failure immediately and starts directing ACKs over the new path, without losing any ACK packets.) Why might the TCP sender retransmit some of the data packets anyway?

解：在使用前一条链路时，EstimatedRTT应该在100ms左右，与此对应的TimeoutInterval应该也不会比EstimatedRTT大太多。当链路出现故障导致超时出现后，TimeoutInterval会被加倍，在200ms左右，但是，新链路的RTT仍旧很可能远大于现在的TimeoutInterval，因此，TCP发送方可能仍旧会重传数据包。

1. Suppose instead that the routers do not switch to the new paths all that quickly, and the data packets (and ACK packets) in flight are all lost. What new congestion window size does the TCP sender use? Why?

解：新的拥塞窗口值应当为1，因为在题设情况下所有ACK丢失将会出现超时现象，而根据拥塞控制，当出现超时现象时，拥塞窗口将会被置为1，阈值将会被置为原来值的一半，故新的拥塞窗口被置为1。

2. Consider the following behavior of a TCP connection (using the congestion control algorithm we learned in class).

At time 0, a TCP sender initiates a connection. As soon as the connection is established, the TCP sender will begin sending data. The MSS is 1KB and RTT is 100 ms.

1) Assuming the connection does not lose any data or experience any timeouts, at what time will the sender’s congestion window be 16KB? (Assuming *threshold* is 32MSS)

解：建立连接需要一个RTT，之后，第一个RTT后，拥塞窗口变为1KB，第二个RTT后，拥塞窗口变为2KB，第三个RTT后，拥塞窗口变为4KB，第四个RTT后，拥塞窗口变为8KB，第五个RTT后，拥塞窗口变为16KB。

综上所述，经过5个RTT即500ms后，拥塞窗口变为16KB。

Right after the sender’s congestion window has reached a size of 16KB, a timeout occurs. After the timeout is detected, the sender continues sending more data over the established connection.

2) Assuming no additional packets loss or timeouts, how long (since the observed timeout) will it take for the congestion window to build to size 14KB?

解：遇到超时现象后，阈值会被置为16KB的一半即8KB。此时连接再次进入慢启动状态，在第一个RTT后，拥塞窗口变为2KB，第二个RTT后，拥塞窗口变为4KB，第三个RTT后，拥塞窗口变为8KB，拥塞窗口与阈值相等，此时进入拥塞避免状态，之后第四个RTT过后，拥塞窗口变为9KB，此后每个RTT过后且在拥塞窗口到达14RTT之前，拥塞窗口增一。

综上所述，当超时现象被检测到之后，将会经历9个RTT即900ms后，拥塞窗口变为14KB。

3) While its congestion window is at 14KB, the sender receives triple duplicate acknowledgements for the same sequence number. How long after receiving the third duplicate acknowledgement will it take for the sender’s congestion window to be at least 9KB again?

解：若采用Reno算法，则发生超时之后，拥塞窗口将变为原来的一半即7KB，然后每过一个RTT，拥塞窗口增一，经过两个RTT，拥塞窗口变为9KB。

若采用Tahoe算法，则发生超时之后，拥塞窗口将变为1KB，之后进入快速恢复，经过3个RTT，拥塞窗口变为7KB，然后进入拥塞避免阶段，经过2个RTT，拥塞窗口变为9KB。

综上所述，若采用Reno算法，则需要2个RTT即200ms，若采用Tahoe算法，则需要5个RTT即500ms。

3. Consider a scenario with two hosts, Alice and Bob. A web server running on Alice is trying to send data to a browser on Bob. For each TCP connection, Alice’s TCP stack maintains a send buffer of 512 bytes and Bob’s TCP stack maintains a receive buffer of 1024 bytes. For simplicity, assume TCP sequence numbers began at 0 in this problem.

1) Bob’s stack received up to byte 560 in order from Alice, although its browser has only read up to the first 60 bytes. What will be the ***ACK#***and ***rcvr window size***in the TCP headers that Bob next sends to Alice?

解：因为Bob端已经接收直到560byte的数据，因此，Bob发送给Alice的ACK#将是561。并且，由于Bob的浏览器已经取走60bytes的数据，因此接收窗口将是1024bytes – (1+560)bytes + 60bytes = 523bytes。

2) Later in the same connection, Alice’s congestion window is set to 1 MSS = 536 bytes and the advertised flow-control window from Bob is 560 bytes. The last ***ACK#*** that Alice received from Bob is byte 700, and the last byte that Alice sends to Bob is byte 900.

A) What is the smallest byte number that Bob will not accept?

B) Assuming that Alice doesn’t receive any more ACKs and her window does not change, what is the greatest byte number that Alice can send?

C) Again assuming that Alice doesn’t receive any additional ACKs, how many more bytes can the web server running on Alice write to its network socket before blocking?

A)解：Bob目前的接收窗口大小为560bytes,而Alice最后接收到的ACK#为700，即表示699及之前已被收到，因此Bob所能接受的bytenumber最大为699+560 =1259，故所不能接受的bytenumber最小为1260。

B)解：Alice缓存所剩余容量为512-（900-699）= 311bytes，故Alice所能发送的最大byte number为311+900 = 1211

C)解：由于min(536,311,560) = 311bytes,故Alice还能像缓存中写入311bytes