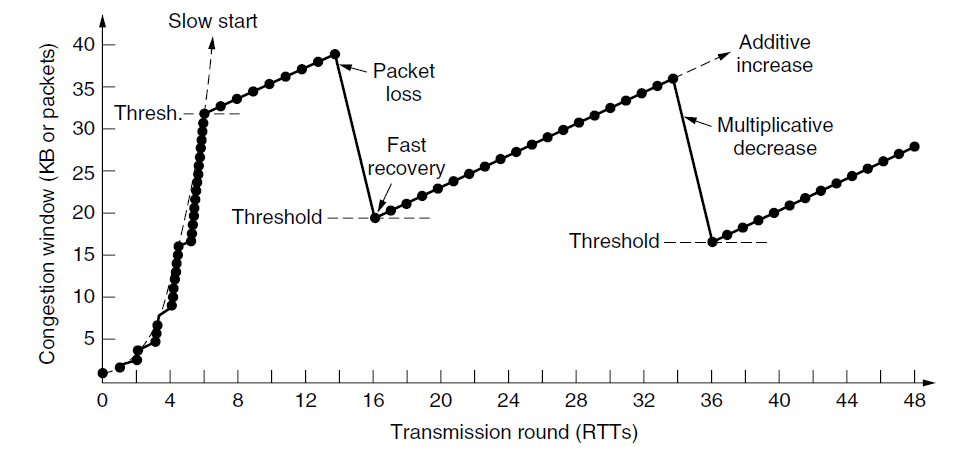
1. Which socket primitive is used to block the caller until a connection attempt arrives?
2. Bind b) listen c) connect d) accept
3. What is used at the transport layer to stop a receiving host’s buffer from overflowing?
4. Segmentation b) Packets c) Acknowledgements d) Flow control
5. If the window size field of the acknowledgement TCP segment is 50 KB, and the congestion window size is 50 KB, how many bytes could the sender transmit next time?
6. Consider the effect of using slow start on a line with a 10 ms round-trip time and no congestion. The receive window is 77 KB and the maximum segment size is 2 KB. How long does it take before the first full window can be sent?

With slow start, the first RTT sends out 1 segment (or 2KB), and 2nd RTT sends out 2 segments (or 4KB), the 3rd RTT 4 segments (or 8 KB), the 4th 8 segments (or 16 KB), the 5th 16 segments (or 32 KB), the 6th 32 segments (or 64 KB), The 7th RTT would have sent out 64 segments (or 128 KB), however, it will exceed the receiver’s window. Therefore, the amount of time it takes before the 7th RTT (or full window, that is, 64 KB) is 6 ×10 = 60 ms.

1. In TCP Reno suppose that the TCP congestion window is set to 64 KB and a timeout occurs. How big will the window be if the next 5 transmission bursts are all successful? Assume that the maximum segment size is 1 KB.

In TCP Reno, when a timeout occurs, the slow start threshold (ssthresh) will be a half of the previous TCP congestion window, therefore, the ssthresh = 64/2 = 32 KB (fast recovery). Then TCP is performing congestion avoidance, that is additively increase 1 to expand the congestion window after each successful transmission. So after the next 5 successful transmission, the congestion window will be 32 + 5 = 37. (这里没有slow start！！！)



1. In TCP Tahoe suppose that the TCP congestion window is set to 64 KB and a timeout occurs. How big will the window be if the next 5 transmission bursts are all successful? Assume that the maximum segment size is 1 KB.

16

When a timeout occurs, three things happened. First, slow start will be initiated. Second, the congestion window would start at 1. Third, the threshold will be reset to 64 KB/2 = 32 KB. If the next 5 transmission burst are all successful, then

1st transmission: 1 segment (1 KB)

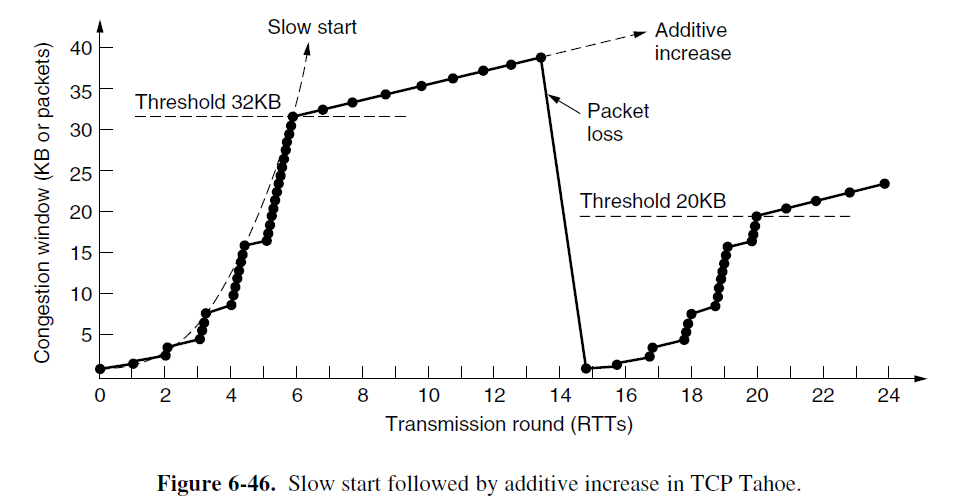
2nd transmission: 2 segments (2 KB)

3rd transmission: 4 segments (4 KB)

4th transmission: 8 segments (8 KB)

5th transmission: 16 segments (16 KB)

6th transmission: 32 segments (32 KB)



1. In TCP Tahoe suppose that the TCP congestion window is set to 18 KB and a timeout occurs. How big will the window be if the next 4 transmission bursts are all successful? Assume that the maximum segment size is 1 KB.

Here's the relevant information from RFC 2001:

"If cwnd is less than or equal to ssthresh, TCP is in slow start; otherwise TCP is performing congestion avoidance. Slow start continues until TCP is halfway to where it was when congestion occurred (since it recorded half of the window size that caused the problem in step 2), and then congestion avoidance takes over. Slow start has cwnd begin at one segment, and be incremented by one segment every time an ACK is received. As mentioned earlier, this opens the window exponentially: send one segment, then two, then four, and so on. Congestion avoidance dictates that cwnd be incremented by segsize\*segsize/cwnd each time an ACK is received, where segsize is the segment size and cwnd is maintained in bytes. This is a linear growth of cwnd, compared to slow start's exponential growth."

So we start this scenario with the Slow-start threshold (ssthresh) set to 18KB/2 = 9KB and the congestion windows (cwnd) set to 1KB due to the timeout. Then we do the following for each ACK we receive:

if (cwnd <= ssthresh)

    { cwnd += 1KB; }

else

   { cwnd += (1KB\*1KB)/cwnd; }

Using this we get the following:

After 1st successful burst of 1 segment

cwnd = 1KB + 1KB = 2KB

After 2nd successful burst of 2 segments

cwnd = 2KB + 2KB = 4KB

After 3rd successful burst of 4 segments

cwnd = 4KB + 4KB = 8KB

After 4th successful burst of 8 segments, things get interesting

ACK 1: (cwnd <= ssthresh) so cwnd = 8KB + 1KB = 9KB

ACK 2: (cwnd <= ssthresh) so cwnd = 9KB + 1KB = 10KB = 10KB

ACK 3: (cwnd > ssthresh) so cwnd = 10240B + (1024B\*1024B)/10240B = 10342B

ACK 4: (cwnd > ssthresh) so cwnd = 10342B + (1024B\*1024B)/10342B = 10443B

ACK 5: (cwnd > ssthresh) so cwnd = 10443B + (1024B\*1024B)/10443B = 10543B

ACK 6: (cwnd > ssthresh) so cwnd = 10543B + (1024B\*1024B)/10543B = 10642B

ACK 7: (cwnd > ssthresh) so cwnd = 10642B + (1024B\*1024B)/10642B = 10740B

ACK 8: (cwnd > ssthresh) so cwnd = 10740B + (1024B\*1024B)/10740B = 10837B