

1. 0101 0011 + 0110 0110 = 1011 1001

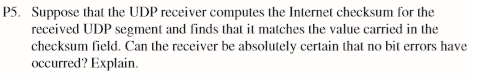
1011 1001 + 0111 0100 = 1 0010 1101 = 0010 1101 + 1 = 00101110

So, the 1’s complement is 11010001.

1. It’s easier to detect errors with the 1’s complement of the sum instead of just the sum.

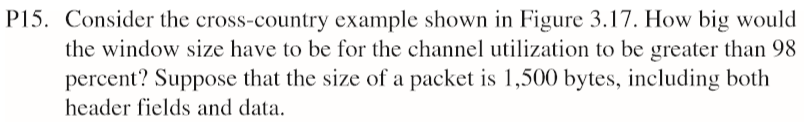
And we don’t need to care for whether the mode of the receiver is big-endian or little-endian.

1. To detect errors, the receiver just need to add the sum it calculated to the 1’s complement it received. If the new sum contains 0, an error occurs.
2. 1-bit error will not go undetected, but 2-bit error may. For example, the first two bytes turn to 01110011 and 01000110, the sum will be the same.



No. Double-bit error may go undetected.

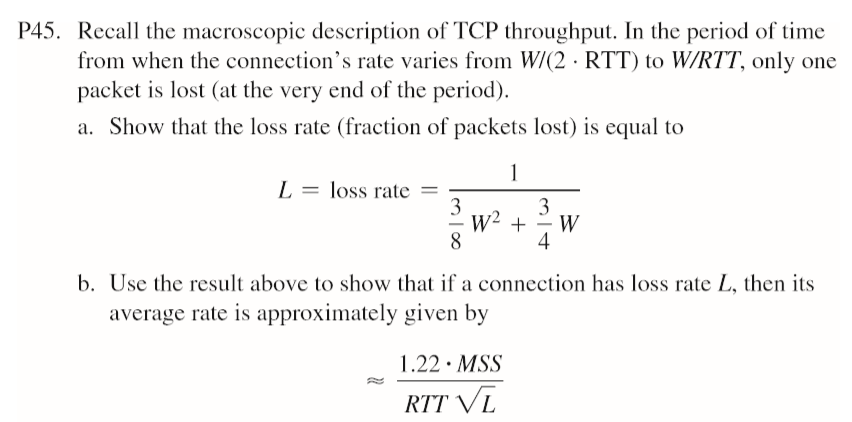
For example, two 8-bit bytes, 0000 1010 and 0000 0101. If they turn to 0000 1011 and 0000 0100, their sum doesn’t change. Although errors occurred, they can’t be detected. So we can’t absolutely certain that no bit errors have occurred.



U = (nL/R) / (RTT + L/R) > 98%

n>2450.92

So, n should be 2451 at least.



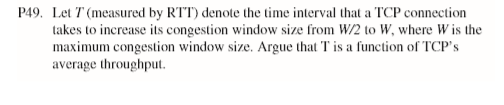
1. packet number form W/(2\*RTT) to W/RTT is:

W/2+(W/2+1)+……+W =

Only one packet is lost, so the loss rate is:

1. When , , ,

The average throughput is:



TCP’s average throughput:

=>

=>