CHAPTER 12

TCP

Exercises

1.

Common fields in UDP and TCP header:

source port number
destination port number
checksum

Fields present in TCP header that are absent from UDP header:

sequence number	(for flow and error control)
acknowledgment number	(for flow and error control)
header length	(TCP has variable header length)
control bits	(flow and error control and connection)
urgent pointer	(needed for flow and error control)
options and padding	(for better performance)

Fields present in UDP header that are absent from TCP header:

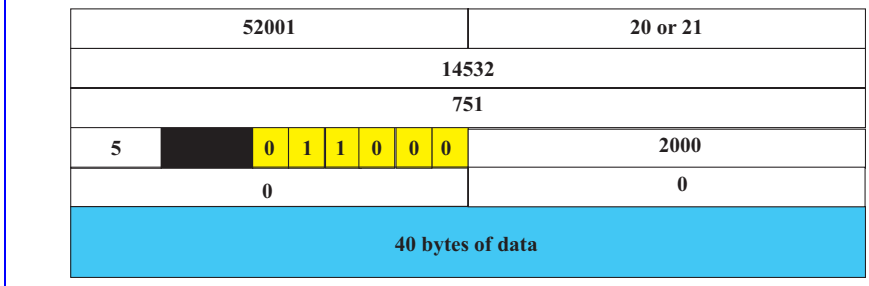
total length	(actually not needed)

3. The port is not listed in the transmission control block.

5. The maximum size of the TCP header is 60 bytes. The minimum size of the TCP header is 20 bytes.

7. See Figure 12.1.

Figure 12.1 *Exercise 7*



9.

a. The source port number is 0532 in hex and 1330 in decimal.
b. The destination port number is 0017 in hex and 23 in decimal.
c. The sequence number is 00000001 in hex and 1 in decimal.
d. The acknowledgment number is 00000000 in hex and 0 in decimal.
e. The header length is 5 in hex and 5 in decimal. 5 × 4 is 20 bytes of header.
f. The control field is 002 in hex. This indicates a SYN segment used for connection establishment.
g. The window size field is 07FF in hex and 2047 in decimal. The window is 2047 bytes.

11. Every second the counter is incremented by 64,000 × 2 = 128,000. The sequence number field is 32 bits long and can hold only 2^{32}–1. So it takes (2^{32}–1)/(128,000) seconds or 33554 seconds.

13.

a. 2171
b. 2172
c. 3172

15. See Figure 12.2.

17. See Figure 12.3.

Figure 12.2 *Exercise 15*

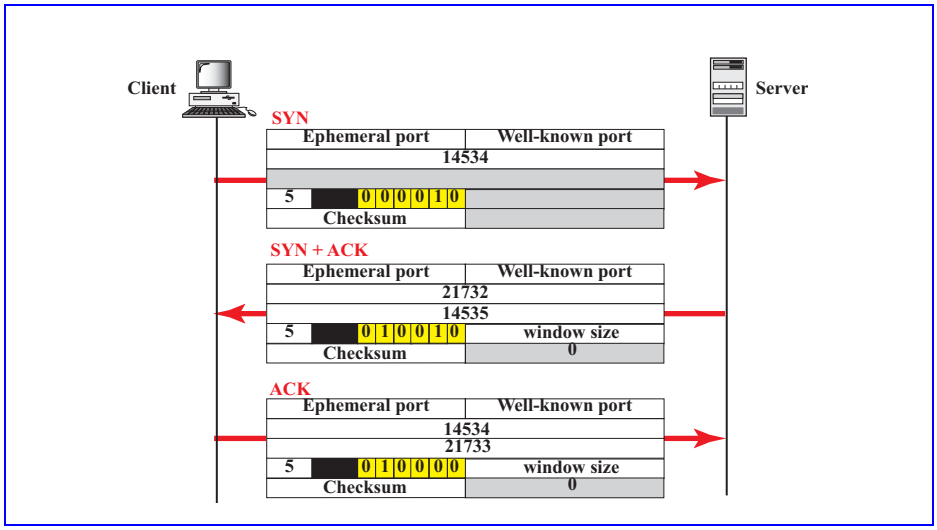
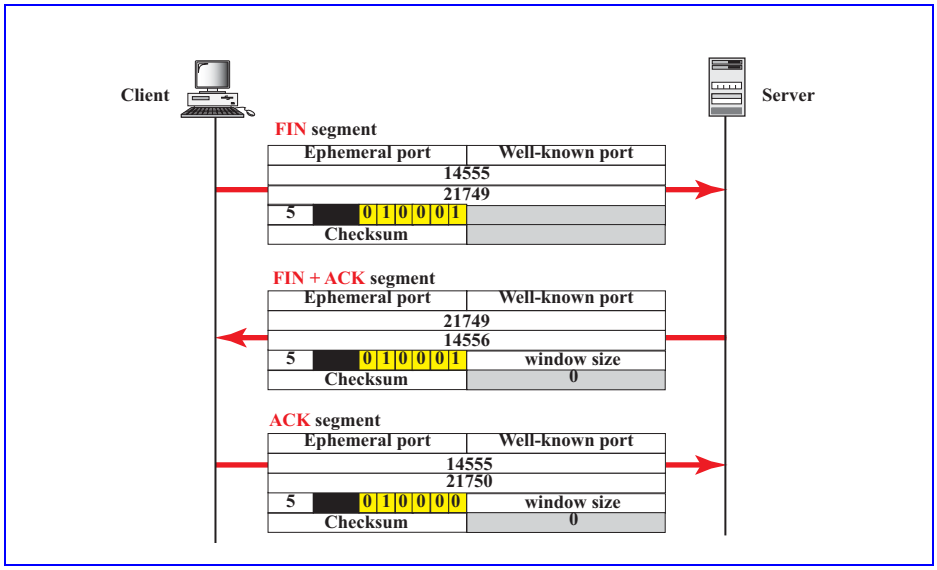


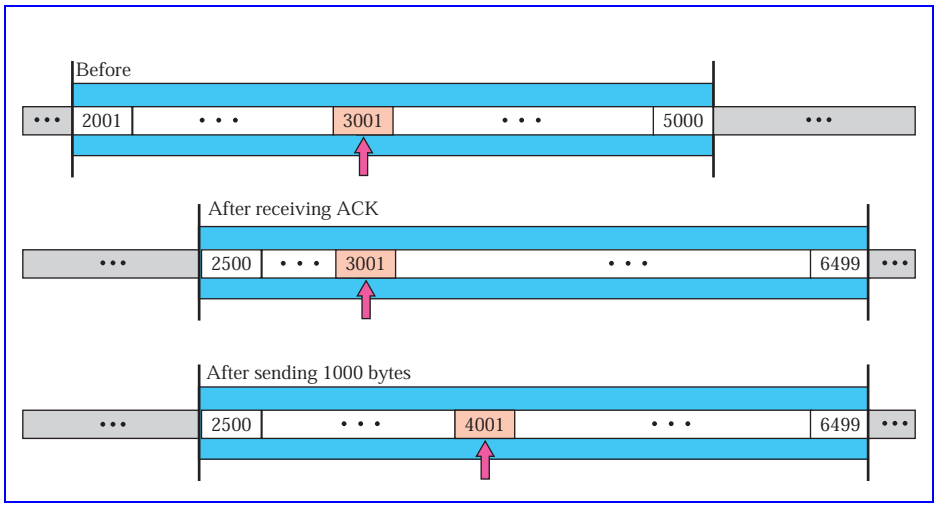
Figure 12.3 *Exercise 17*



19. The largest number in the sequence number field is $2^{32} - 1$. If we start at 7000, it takes $[(2^{32} - 1) - 7000] / 1{,}000{,}000 =$ **4295** seconds.

21. See Figure 12.4.

Figure 12.4 *Exercise 21*



23.

a. When the "close" message is received from the application, the client TCP sends a FIN segment; the client goes to the FIN-WAIT-1 state and waits for an ACK.

b. When the client receives an ACK segment from the server, it moves to the FIN-WAIT-2 state, and waits for the server to send a FIN segment.

25. See Figure 12.5.

27. Although some implementations reacts to the congestion detection (3ACKs or timeout) only during the congestion avoidance, we assume that the system reacts to the 3-ACK events even during the slows tart as shown in See Figure 12.6.

Figure 12.5 *Exercise 25*

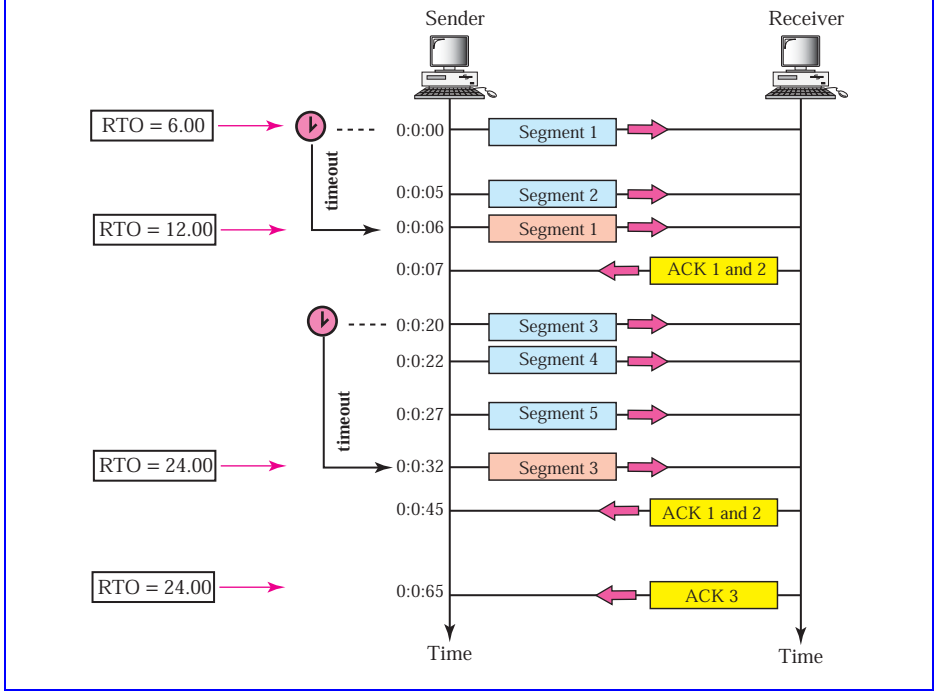


Figure 12.6 *Exercise 27*