Tutorial 3 & 4 Solutions

Instructions

1. Form ad-hoc groups of 2 to 3 students to solve this week's exercise.
2. Each group must answer the following review Q's
3. Each group will use shared google docs to work with all group members and tutor. The document must include the group member's names and the tutorial sheet number.

Review Questions

1. Q3-3. Assume you need to write and test a client-server application program on two hosts you have at home.
   a. What is the range of port numbers you would choose for the client program?
   b. What is the range of port numbers you would choose for the server program?
   c. Can the two port numbers be the same?

   Answer: Although any port number can be used for the client and server and they can be the same in this private communication, it is recommended to follow the division specified by ICANN:

   a. The client port number should be chosen from the dynamic range, 49,152 to 65,535.

   b. The server port number also should be chosen from the dynamic range, 49,152 to 65,535.

   c. It is advisable to choose different port numbers for the server and the client to be able to better debug the programs.

2. Q3-5. In a network, the size of the receive window is 1 packet. Which of the following protocols is being used by the network?
   a) Stop-and-Wait
   b) Go-Back-N &
   c) Selective-Repeat

   *Answer: We check each protocol **one by one**:*

   *d. The protocol can be Stop-and-Wait with the **receive** window size of 1 and the send window size of 1.*

   *e. The protocol can also be Go-Back-N with the **receive** window size of 1 and the send window size of n packets.*

   *f. The protocol **cannot** be Selective-Repeat because the size of both windows should be the same (1), which means the protocol is Stop-and-Wait, not Selective-Repeat.*

3. Q3-7. In a network with fixed value for windows size ***m*** **>** ***1***, we can use either the Go-Back-N or the Selective-Repeat protocol. Describe the advantage and the disadvantage of using each. What other network criteria should be considered to select either of these protocols?

   Answer: We describe the advantage and disadvantage of each first:

   g. The advantage of using the Go-Back-N protocol is that we can have a larger send window size. We can send more packets before waiting for their acknowledgment. The disadvantage of using this protocol is that the receive window size is only 1. The receiver

cannot accept and store the out- of-order received packets; they will be discarded. Discarding of the out-of- order packets means resending these packets by the sender, resulting in congestion of the network and reducing the capacity of the pipe. So the advantage seen by a larger send window may disappear by filling the net- work with resent packets.

h. *The advantage of using the Selective-Repeat protocol is that the receive window can be much larger than 1. This allows the receive window to store the out-of-order packets and avoids resending them to congest the network. On the other, the send window size of this protocol is half of the Go-Back- N, which means that we can send fewer packets before waiting for the acknowledgment.*

i. *If the bandwidth-delay product of the network is large, the reliability is good, and the delay is low, we should choose the Go-Back-N protocol to use more of the network capacity. On the other hand, if the bandwidth- delay product is small, and the network is not very reliable, or the network creates long delays, we need to use Selective-Repeat.*

4. Q3-11. Can you explain why some transport-layer packets may be lost in the Internet?

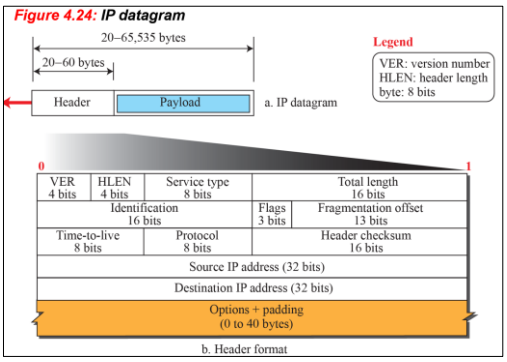
   Answer: The transport-layer packets are encapsulated in the datagram at the network layer. The router through which the datagrams need to pass to reach their destination may be congested and drop the packets.

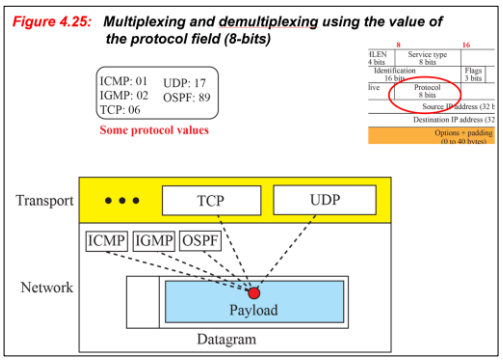
5. Q3-13. In the **Go-Back-N protocol**, the size of the send window can be 2^{m-1}, while the size of the receive window is only 1. How can flow control be accomplished when there is a big difference between the size of the send and receive windows?

   Answer: The rest of the packets (2^m ^{2}) are supposed to be in transit, filling the pipe.

   The size of the receive window is chosen to be 1 to accept only one packet, the one expected, and not out-of-order packets. The receiver cannot be over- whelmed because it holds only one packet in its window. When the only packet in the window is consumed by the upper-layer protocol, the receive window slides to make it possible to receive the next packet in transit. If any packet in transit arrives before the window slides, it is discarded.

6. Q3-15. Some of the application programs can use the services of two transport-layer protocols (UDP or TCP). When a packet arrives at the destination, how can the computer find which transport layer is involved?





Answer: The protocol field of the datagram (see Figures 4.24 and 4.25 in Chapter 4) defines the transport-layer protocol that should receive the transport-layer packet. If the value is 06, the protocol is TCP; if the value is 17, the protocol is UDP.

7. Q3-17. UDP is a message-oriented protocol. TCP is a byte-oriented protocol. If an application needs to protect the boundaries of its message, which protocol should be used, UDP or TCP?

   Answer: UDP is preferred because each user datagram can be used for each chunk of data. However, a better solution is the new transport protocol, SCTP, discussed in other chapter (chapter 8).

8. Q3-19. Assume a private internet, which uses point-to-point communication between the hosts and needs no routing, has eliminated the use of the network layer. Can this internet still benefit from the service of UDP or TCP? In other words, can user datagrams or segments be encapsulated in the Ethernet frames?

   *Answer: The answer is **positive**.*

   There is nothing in the UDP or TCP protocol that requires the use of the IP protocol. A UDP user datagram or a TCP segment can be encapsulated in an Ethernet frame. However, the protocol field of the Ethernet needs to define which protocol is directly used in this case.

9. Q3-21. Can you explain why we need four (or sometimes three) message-segments for connection termination in TCP?

   Answer: There are two parties involved in a two-way communication. TCP allows each party to stop sending while the other party is still sending data. This means we need at least two FIN segments (or data segments in which the FIN bit is set). Since each FIN segment should be acknowledged, we normally need four segments for connection termination. Sometimes the second FIN segment can be combined with the ACK segment to reduce the number of segments to three.

10. Q3-25. In TCP, how many sequence numbers are consumed by each of the following segments:

    a. SYN b. ACK c. SYN+ACK d. Data

    Answer:
    - *a. A SYN segment consumes one sequence number.*
    - *b. An ACK segment does not consume any sequence numbers.*
    - *c. A SYN + ACK segment consumes one sequence number because it is a SYN segment.*
    - *d. A data segment consumes **n** sequence numbers, where **n** is the number of bytes carried by the segment.*

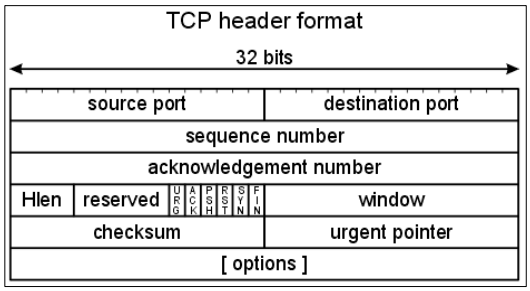
11. Q3-27. Looking at the TCP header (Figure 3.44), we find that the sequence number is 32 bits long, while the window size is only 16 bits long. Does this mean that TCP is closer to the Go-Back-Nor Select-Repeat protocol in this respect?

    Answer: We cannot say. The size of the window field makes the size of the window 2^{16} bytes, but the sequence number is 2^{32}. The size of the window is definitely much smaller (actually 2^{16} times smaller) than half of the range of the sequence numbers. The requirement of each protocol, Go-Back-N and Selective-Repeat, is satisfied. However, as we mentioned in the text, TCP is closer to Selective-Repeat in other aspects.

12. Q3-29. What is the maximum size of the TCP header? What is the minimum size of the TCP header?

    Answer:

    - *The maximum size of the TCP header is **60** bytes (20 bytes of header and a maximum of 40 bytes of options).*
    - *The minimum size of the TCP header is **20** bytes.*



13. Q3-31. In TCP, does a FIN segment close a connection in only one direction or in both directions?

    Answer:

    *A FIN segment closes the connection in only one direction. The party that issues a FIN segment cannot send data to the other party, but needs to acknowledge the data received from the other party. The other party can still send data and acknowledgments. To completely close the connection, **two FIN segments are required**, one in each direction. Of course, the FIN and ACK segments can always be combined.*

14. Q3-35. Can you explain how TCP, which uses the services provided by the unreliable IP, can provide reliable communication?

    Answer: This is done through acknowledgment and retransmission. If a packet is lost or corrupted, it will be resent. As an analogy, assume the postal service is unreliable. When we send a letter to a friend, we can ask for confirmation with a postcard. If we do not receive the postcard, we can resend the copy of the letter until we finally get the confirmation.

15. Q3-37. Assume Alice uses her browser to open two connections to the HTTP server running on Bob's server. How can these two connections be distinguished by the TCP?

    Answer: A connection is distinguished by a pair of socket addresses, one for each end. Although the socket addresses at Bob's site are the same in this case, the socket addresses at Alice's site are different. Each socket address at Alice's site has a different ephemeral port number.

16. Q3-39. In TCP, can the sender window be smaller, larger, or the same size as the receiver window?

    Answer: The sender window in TCP is originally the same size as the receiver window, but as the congestion builds up in the network, it can become smaller than the receiver window. It can never become larger than the receiver window. The size of the sender window is the smaller of the window size either dictated by the congestion or determined by the receiver.

17. Q3-41. In a TCP segment, what does a **sequence number** identify?

    Answer: The answer depends on whether the segment is carrying data or not.

    a. *If the segment carries data, the sequence number defines the number of the first bytes in the segment.*

b. *If the segment carries no data (pure control segment), the sequence number identifies the segment itself.*

18. Q3-43. Is the use of checksum for error control optional or mandatory in:

    a. UDP?
    b. TCP?

Answer:
*The use of checksum in UDP is **optional**. If the sender does not use the check- sum, it fills the checksum field with sixteen 0s.*

*The use of the checksum in TCP, on the other hand, is **mandatory**. The sender should calculate the check- sum; otherwise, the checksum calculation at the receiver fails and the segment is dropped.*

19. Q3-45. Assume a TCP client expects to receive byte 2001, but it receives a segment with sequence number 1201. What is the reaction of the TCP client to this event? Can you justify the reaction?

Answer: The received segment is a duplicate. The TCP client needs to discard the segment and immediately send an ACK with acknowledgment number 2001. This reaction helps the server to update itself if the previous ACK with acknowledgment number 2001 is somehow lost (Rule 6 in ACK generation).

20. P3-1. Compare the range of 16-bit addresses, 0 to 65,535, with the range of 32-bit IP addresses, 0 to 4,294,967,295 (discussed in Chapter 4). Why do we need such a large range of IP addresses, but only a relatively small range of port numbers?

Answer: The domain of IP addresses is universal. A device directly connected to the Internet needs a unique IP address (see exceptions in Chapter 4). The domain of port numbers is local; they can be repeated. Two computers running the HTTP server process use the same well-known port number (80); two computers running the HTTP client process can use the same ephemeral port number.

21. P3-5. Using 5-bit sequence numbers, what is the maximum size of the send-receive windows for each of the following protocols.

    a) Stop-and-Wait
    b) Go-Back-N
    c) Selective-Repeat

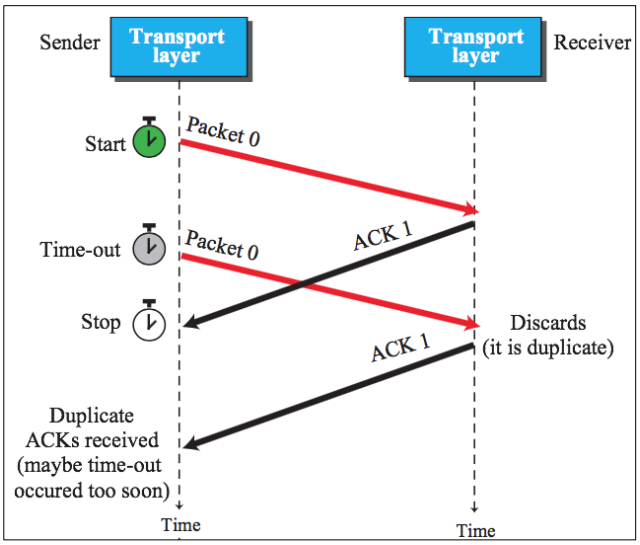
Answer:

*a) **Stop-and-Wait:** Max **Send** Wsize = 1; Max **Receive** Wsize = 1*
*b) **Go-Back-N:** Max **Send** Wsize = $2^5 - 1 = 31$; Max **Receive** Wsize = 1*
*c) **Selective-Repeat:** Max **Send** Wsize = $2^5 / 2 = 16$; Max **Recieve** Wsize = $2^5 / 2 = 16$*

22. P3-9. In the Stop-and-Wait protocol, show the case in which the receiver receives a duplicate packet (which is also out of order). Hint: Think about a delayed ACK. What is the reaction of the receiver to this event?

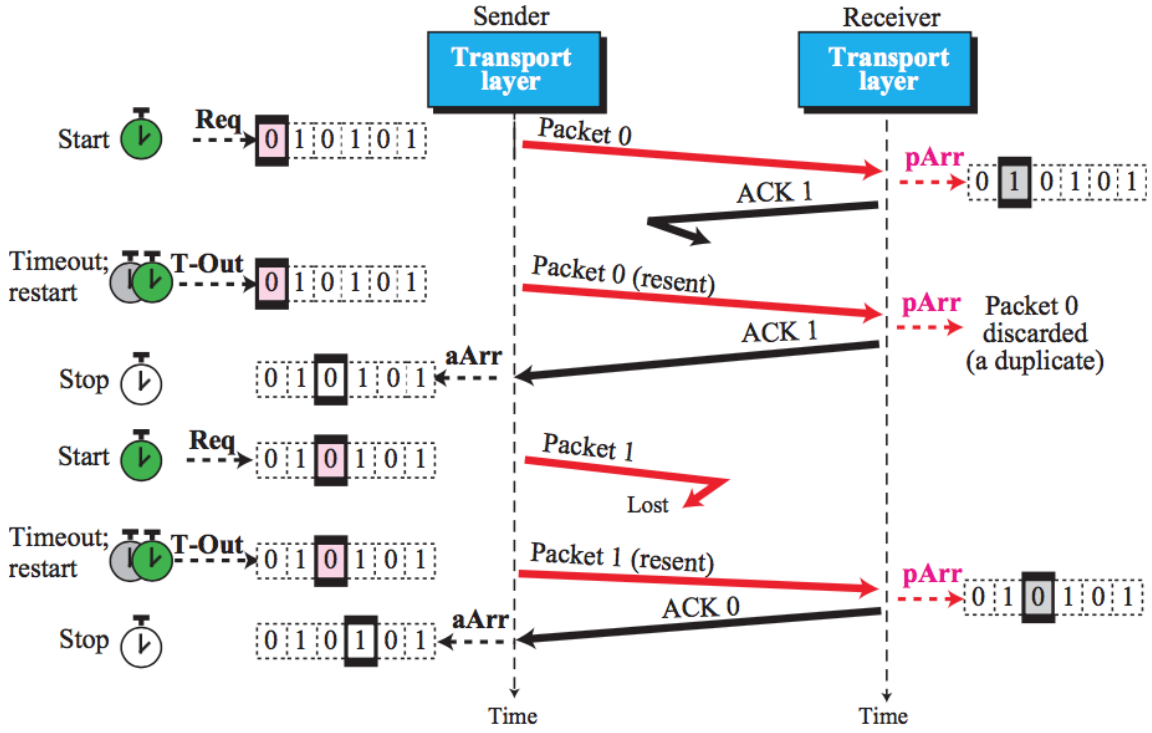
Answer: The following shows the case. It happens when an ACK is delayed and the time-out occurs. The sender resends the packet that is already acknowledged by the receiver. The receiver discards the duplicate packet, but resends the previous ACK to inform the sender that there is a delay in the network.

Receiving a duplicate ACK may alert the sender to increase the time-out to prevent resending the packets prematurely. Note that if the receiver ignores the duplicate packet and does not send the second ACK, the sender believes that the first packet is lost and the first ACK is actually acknowledging the packet that is resent. Duplicate ACKs give the clue about what is happening.



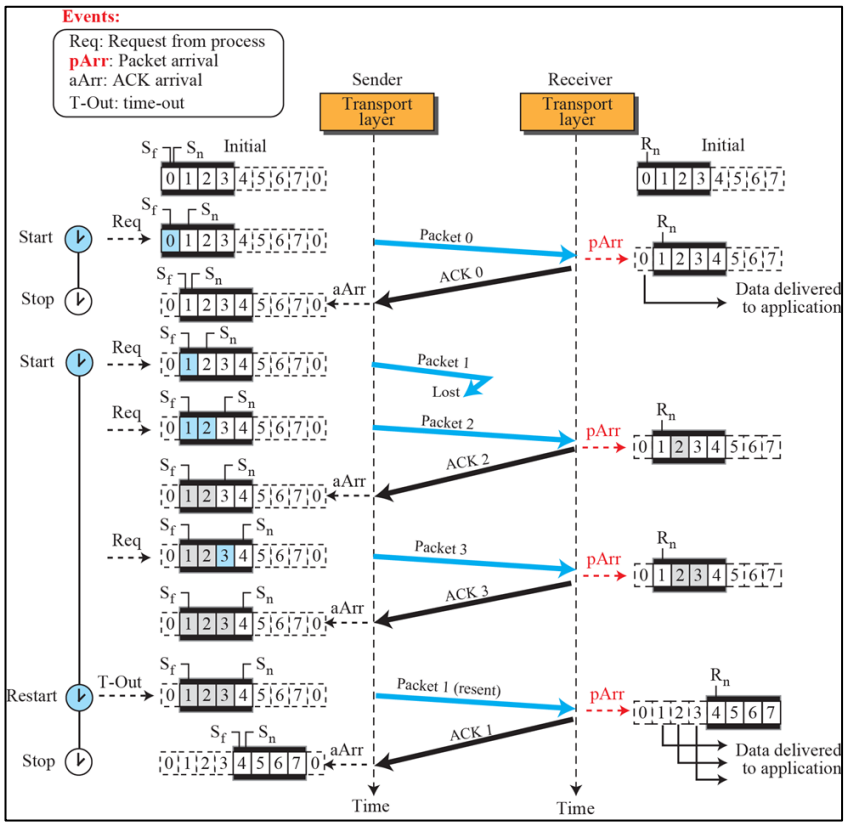
23. P3-13. Create a scenario similar to Figure 3.22 in which the sender sends two packets. The first packet is received and acknowledged, but the acknowledgment is lost. The sender resends the packet after time-out. The second packet is lost and resent.

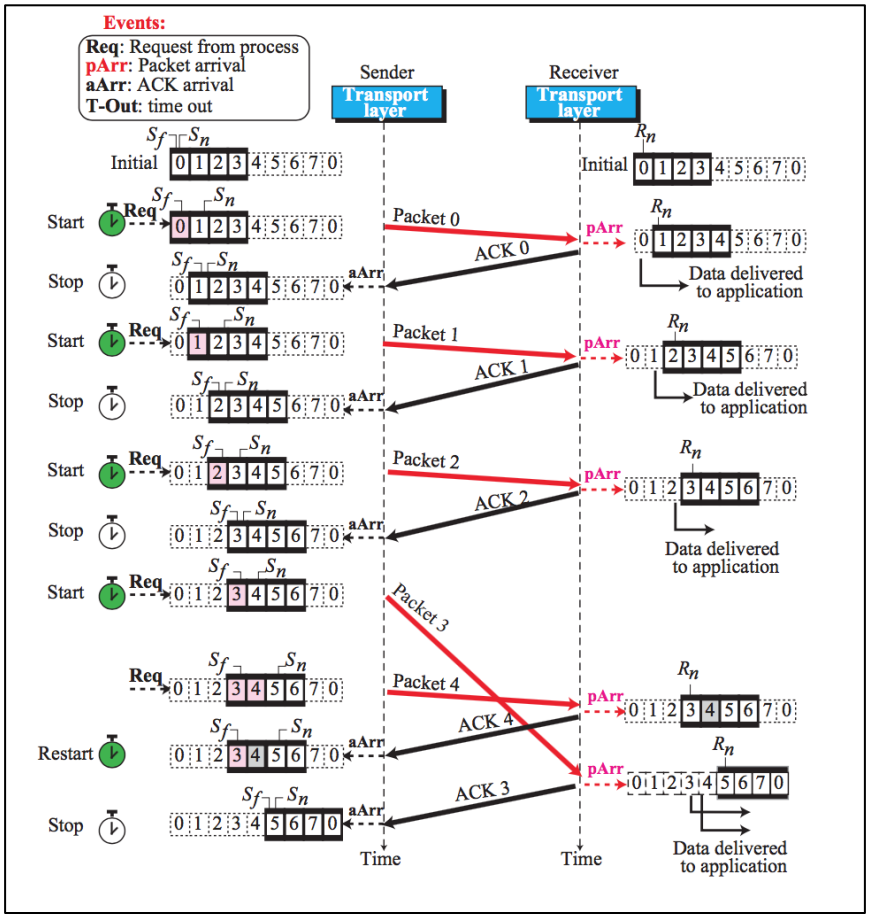
Answer: See the following:



24. P3-15. Redraw **Figure 3.35** if the sender sends 5 packets (0, 1, 2, 3, and 4). Packet 1 and 2 are received in order and acknowledged, one by one. Packet 3 delayed and received after packet 4.

Answer: The following shows the solutions:





25. P3-19. . We can define the bandwidth-delay product in a network as the number of packets that can be in the pipe during the round-trip time (RTT). What is the bandwidth-delay product in each of the following situation s?

    a. Bandwidth: 1 Mbps, RTT: 20 ms, packet size: 1000 bits
    b. Bandwidth: 10 Mbps, RTT: 20 ms, packet size: 2000 bits
    c. Bandwidth: 1 Gbps, RTT: 4 ms, packet size: 10,000 bits

    Answer: In each case we first define the bandwidth-delay product (BDP) in bits and then find it in the number of packets:

    a. BDP = 1 Mbps × 20 ms = 20,000 bits = 20 packets

    b. BDP = 10 Mbps × 20 ms = 200,000 bits = 100 packets

    c. BDP = 1 Gbps × 4 ms = 4,000,000 bits = 400 packets

26. P3-21. Assume we need to design a **Selective-Repeat sliding-window** protocol for a network in which the bandwidth is 1 Gbps and the average distance between the sender and receiver is 5,000 Km. Assume the average packet size is 50,000 bits and the propagation speed in the media is 2×10^8 m.

    a. Find the maximum size of the send and receive windows,

    b. Find the number of bits in the sequence number field (m), and

    c. Appropriate time-out value for the timer.

    Answer: We first calculate the average round-trip time (RTT) and the number of packets in the pipe before finding the sizes of the windows, the value of m, and the time-out value.

    - *Average RTT = 2 × (5,000 Km) / (2 × 10^8) = 50 ms.*
    - *The bandwidth-delay product = 1 Gbps × 50 ms = 50,000,000 bits.*
    - *The number of Packets ==bandwidth-delay product/packet_size = 50,000,000 bits / 50,000 bits = 1000 packets.*

    ➢ *The maximum **send** window size should be 1000 to allow not more than 1000 packets in the pipe.*
    ➢ *The maximum **receive** window size should also be 1000 packets.*
    ➢ *We know that the (window size) ≤ ($2^m - 1$) or 1000 ≤ ($2^m - 1$). This means that we need to choose (m − 1) to be at least 10 or m = 11. The sequence numbers are then 0 to 2047.*
    ➢ *The timeout value should be at least the average RTT = 50 ms to avoid early retransmission of the packets and to prevent congestion.*

27. P3-25. Answer the following questions:

    a. What is the minimum size of a UDP user datagram ?
    b. What is the maximum size of a UDP user datagram?
    c. What is the minimum size of the application-layer payload data that can be encapsulated in a UDP user datagram?
    d. What is the maximum size of the application-layer payload that can be encapsulated in a UDP user datagram?

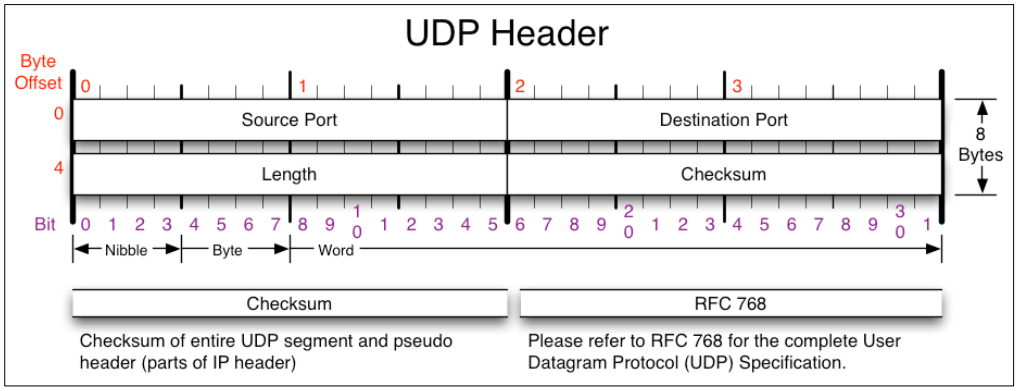
Answer:

a. The minimum size is 8 bytes (header without payload).

b. Although the theoretical maximum size is 65,535 bytes, since a user data- gram needs to be encapsulated in a single IP datagram (UDP is a connectionless protocol) and the maximum payload of an IP datagram is 65,515 bytes (see Figure 4.24 in Chapter 4), we should say the maximum size of a UDP datagram is only 65,515 bytes.

c. The minimum size of the application-layer payload is zero bytes.

d. The maximum size of the application-layer payload is 65,507 bytes (65,515 − 8).



28. P3-27. The following is a dump (contents) of a UDP header in hexadecimal format
0045DF0000580000

a. What is the source port number?
b. What is the destination port number?
c. What is the total length of the user datagram?
d. What is the length of the data?
e. Is the packet directed from a client to a server or vice versa?
f. What is the application -layer protocol?
g. Has the sender calculated a checksum for this packet?

Answer:

a. The source port number is the first 16 bits or $(0045)_{16}$ = 69.

b. The destination port number is the second 16 bits $(DF00)_{16}$ = 57,088.

c. The total length of the datagram is the third 16 bits $(0058)_{16}$ = 88 bytes.

d. The length of the data is 88 − 8 = 80 bytes.

e. The message is from a server with a small (well-known) port number(69) to a client with a large (ephemeral) port number(57088).

f. The well-known source-port number = 69 belongs to TFTP.

*g. The sender has **not** calculated the checksum for this packet because the value of the checksum is all zeros.*

29. P3-37. A client uses TCP to send data to a server. The data consist of 16 bytes. Calculate the efficiency of this transmission at the TCP level (ratio of useful bytes to total bytes).

Answer: The data section is only 16 bytes. The TCP header is 20 bytes. The efficiency is (16) / (16 + 20) = 0.444 → 44.4%

30. P3-55. In a TCP connection, assume that maximum segment size (MSS) is 1000 bytes. The client process has 5400 bytes to send to the server process, which has no bytes to

respond (unidirectional communication). The TCP server generates ACKs according to the rules we discussed in the text. Show the time line for the transactions during the slow start phase, indicating the value of *cwnd* at the beginning, at the end, and after each change. Assume that each segment header is only 20 bytes.

Answer: The data from the client process, 5400 bytes, can be divided into six chunks (five chunks of 980 bytes and one chunk of 500 bytes). After adding a header of 20 bytes, we have six segments (five segments of 1000 bytes and one segment of 520 bytes).

The segments and the ACKs are created according to the rule we mentioned in the text. The size of the congestion window is increased by one MSS for each ACK received.

If we follow the growth of the cwnd, we can see the pattern is exponential, but the base is decreased from 2 to 1.5 ($2^0 = 1$, $2^1 = 2$, $1.75^2 \approx 3$, $1.60^3 \approx 4$, and $1.5^4 \approx 5$).