|  |  |
| --- | --- |
| Course – Section | Computer Networks (CS3001 - Spring 2024) – *BCS 6B* |
| Assignment Num. | 03 |
| Total Marks | **60** |
| Start Date | 13-March-2024 |
| Due Date/Time | 24-March-2024 |
| Submission | ***Submit the handwritten assignment in class as well as google classroom*** |
| Author | Hammad Ali ([l215273@lhr.nu.edu.pk](mailto:l215273@lhr.nu.edu.pk)) |
| Approved By | Ms. Umme Ammarah ([umme.ammarah@lhr.nu.edu.pk](mailto:umme.ammarah@lhr.nu.edu.pk)) |

1. Suppose the network layer provides the following service. The network layer in the source host accepts a segment of maximum size 1,200 bytes and a destination host address from the transport layer. The network layer then guarantees to deliver the segment to the transport layer at the destination host. Suppose many network application processes can be running at the destination host.
   1. Design the simplest possible transport-layer protocol that will get application data to the desired process at the destination host. Assume the operating system in the destination host has assigned a 4-byte port number to each running application process.

Call this protocol Simple Transport Protocol (STP). At the sender side, STP accepts from the sending process a chunk of data not exceeding 1196 bytes, a destination host address, and a destination port number. STP adds a four-byte header to each chunk and puts the port number of the destination process in this header. STP then gives the destination host address and the resulting segment to the network layer. The network layer delivers the segment to STP at the destination host. STP then examines the port number in the segment, extracts the data from the segment, and passes the data to the process identified by the port number.

* 1. Modify this protocol so that it provides a “return address” to the destination process.

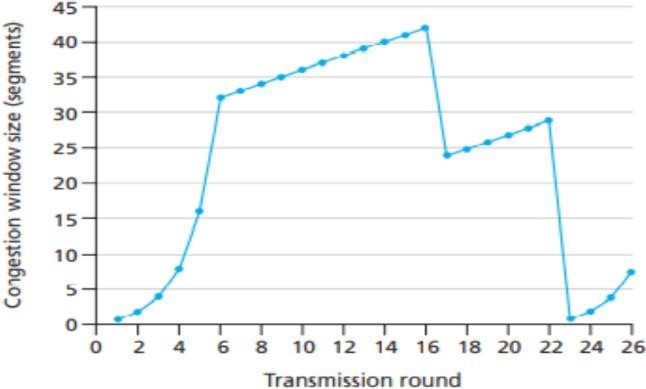
The segment now has two header fields: a source port field and destination port field. At the sender side, STP accepts a chunk of data not exceeding 1192 bytes, a destination host address, a source port number, and a destination port number. STP creates a segment which contains the application data, source port number, and destination port number. It then gives the segment and the destination host address to the network layer. After receiving the segment, STP at the receiving host gives the application process the application data and the source port number.

* 1. In your protocols, does the transport layer “have to do anything” in the core of the computer

network.

No, the transport layer does not have to do anything in the core; the transport layer “lives” in the end systems

# (8 marks)

1. For the figure given below, assume TCP Reno is the protocol experiencing the behavior shown.
   1. Identify the intervals of time when TCP slow start is operating.

[0,6],[23,26]

* 1. Identify the intervals of time when TCP congestion avoidance is operating.

[6, 16] [17, 22], linear increase

* 1. After the 16th transmission round, is segment loss detected by a triple duplicate ACK or by a timeout?

Triple ACK, otherwise cwnd would have been 1

* 1. After the 22nd transmission round, is segment loss detected by a triple duplicate ACK or by a timeout?

Timeout because cwnd is 1.

* 1. What is the initial value of *ssthresh* at the first transmission round?

32

* 1. What is the value of *ssthresh* at the 18th transmission round?

21

* 1. What is the value of *ssthresh* at the 24th transmission round?

14

* 1. During what transmission round is the 70th segment sent?

7

* 1. Assuming a packet loss is detected after the 26th round by the receipt of a triple duplicate ACK, what will be the values of the congestion window size and of *ssthresh*?

Cwnd=4+3, ssthresh=4

# (8 marks)

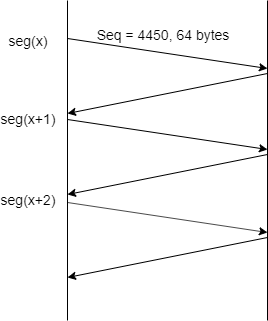
1. Consider a planet where everyone belongs to a family of six, every family lives in its own house, each house has a unique address, and each person in each house has a unique name. Suppose this planet has a mail service that delivers letters from source house to destination house. The mail service requires that
2. the letter be in an envelope, and that (2) the address of the destination house (and nothing more) be clearly written on the envelope. Suppose each family has a delegate family member who collects and distributes letters for the other family members. The letters do not necessarily provide any indication of the recipients of the letters.
   1. Describe a protocol that the delegates can use to deliver letters from a sending family member to a receiving family member.

For sending a letter, the family member is required to give the delegate the letter itself, the address of the destination house, and the name of the recipient. The delegate clearly writes the recipient’s name on the top of the letter. The delegate then puts the letter in an envelope and writes the address of the destination house on the envelope. The delegate then gives the letter to the planet’s mail service. At the receiving side, the delegate receives the letter from the mail service, takes the letter out of the envelope, and takes note of the recipient name written at the top of the letter. The delegate then gives the letter to the family member with this name.

* 1. In your protocol, does the mail service ever have to open the envelope and examine the letter to provide its service?

No, the mail service does not have to open the envelope; it only examines the address on the envelope.

# (8 marks)

1. Suppose Node A (sender) and B (receiver) have a TCP connection between them. Assume that a single segment, i.e., seg(x-1) is timed-out. Consider the size of the TCP receiver buffer is 300 bytes. Assuming all packets of equal size i.e., 64 bytes, if *ssthreshold* = 6, then **answer the following questions** by looking at the figure provided below, by providing:
   1. Sequence number of seg(x+2)

4578

* 1. Acknowledgement of seg(x+2)

4386

* 1. Sequence number of bytes of seg(x+4)

Starting byte number: 4386 Ending byte number: 4449

* 1. *Receiver window* field value in acknowledgment of seg(x+3)

44

* 1. *Receiver window* field value in acknowledgement of seg(x+4)

300

* 1. Value of *window size* and *ssthreshold* after acknowledgement of seg(x+6) is received?

Window size: 6 ssthreshold: 6

* 1. TCP receiver sometimes waits for 500ms before sending an acknowledgement of a newly arrived segment. For how long will receiver wait before sending ack for seg(x+4)?

It will not wait and send ack 4706

* 1. Assuming seg(x) to be sent in the first transmission round, how many segments will be sent in the third transmission round?

4

* 1. Assume no loss occurs, what will be the last segment that will be sent in the *slow start* phase starting from seg(x)?

Seg(x+12)

* 1. What is the link utilization during the *slow start* phase if the link capacity is 10 Mbps and the RTT between node A and B is 15ms?

Formula used = (L/R)/(RTT+L/R)

Input values: (13∗64∗8/10Mbps)/(15ms + 64∗8/10Mbps)

Final answer after calculation = 0.044.

* 1. Suppose after receiving 50 segments from the source, node B lost synchronization with A. Write the name and value of the field used by destination B to notify source node A.

RST bit = 1

***NOTE: Answer all numbers in decimal number system ONLY (where applies). For segment numbers, use the notation of seg(x) where x is the number of a segment.***

# (10 marks)

1. Suppose that TCP's current estimated values for the round-trip time (estimated RTT) and deviation in the RTT (Dev RTT) are 210 msec and 49 msec, respectively. Suppose that the next two measured values of the RTT are 280, 210. Compute TCP's new value of estimated RTT, Dev RTT, and the TCP timeout value after each of these two measured RTT values is obtained. Use the values of α = 0.125 and β = 0.25.

# (6 marks)

After the first RTT estimate is made:

estimatedRTT = 0.875\*210 + 0.125\*280 = 218.75 msecs

DevRTT = 0.75\*49 + 0.25\*(abs(280 - 218.75)) = 52.0625 msecs

TimeoutInterval = 218.75 + 4\*52.0625 = 427 msecs

After the second RTT estimate is made:

EestimatedRTT = 0.875\*218.75 + 0.125\*210 = 217.65625 msecs

DevRTT = 0.75\*52.0625 + 0.25\*(abs(210 - 217.65625)) = 40.9609375 msecs

TimeoutInterval = 217.65625 + 4\*40.9609375 = 381.5 msecs

1. UDP and TCP use 1s complement for their checksums. Suppose you have the following three 8-bit bytes: 01010011, 01100110, 01110100. What is the 1s complement of the sum of these 8-bit bytes? (Note that although UDP and TCP use 16-bit words in computing the checksum, for this problem you are being asked to consider 8-bit sums.) Show all work. Why is it that UDP takes the 1s complement of the sum; that is, why not just use the sum? With the 1s complement scheme, how does the receiver detect errors? Is it possible that a 1-bit error will go undetected? How about a 2-bit error?

# (6 marks)

# Note, wrap around if overflow.

# 0 1 0 1 0 0 1 1

# + 0 1 1 0 0 1 1 0

# -------------------------

# 1 0 1 1 1 0 0 1

# + 0 1 1 1 0 1 0 0

# -------------------------

# 0 0 1 0 1 1 0 1

# + 1

# -------------------------

# 0 0 1 0 1 1 1 0

# One's complement = 1 1 0 1 0 0 0 1.

# To detect errors, the receiver adds the four words (the three original words and the checksum). If the sum contains a zero, the receiver knows there has been an error. All one-bit errors will be detected, but two-bit errors can be undetected (e.g., if the last digit of the first word is converted to a 0 and the last digit of the second word is converted to a 1)

1. Host A and B are communicating over a TCP connection, and Host B has already received from Host A all bytes up through byte 126. Suppose Host A then sends two segments to Host B back-to-back. The first and second segments contain 80 and 40 bytes of data, respectively. In the first segment, the sequence number is 127, the source port number is 302, and the destination port number is 80. Host B sends an acknowledgment whenever it receives a segment from Host A.
   1. In the second segment sent from Host A to B, what are the sequence number, source port number, and destination port number?

In the second segment from Host A to B, the sequence number is 207, source port number is 302 and destination port number is 80.

* 1. If the first segment arrives before the second segment, in the acknowledgment of the first arriving segment, what is the acknowledgment number, the source port number, and the destination port number?

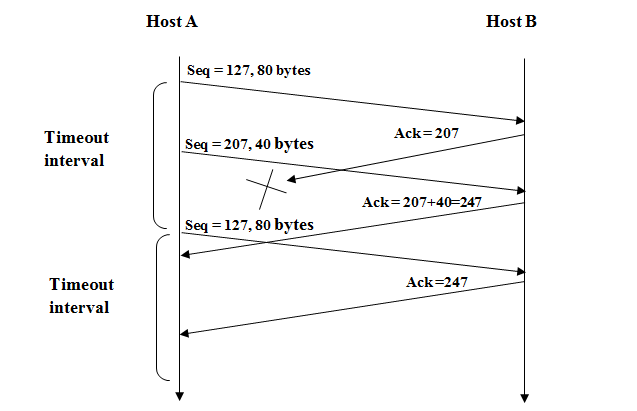
If the first segment arrives before the second, in the acknowledgement of the first arriving segment, the acknowledgement number is 207, the source port number is 80 and the destination port number is 302.

* 1. If the second segment arrives before the first segment, in the acknowledgment of the first arriving segment, what is the acknowledgment number?

If the second segment arrives before the first segment, in the acknowledgement of the first arriving segment, the acknowledgement number is 127, indicating that it is still waiting for bytes 127 and onwards.

* 1. Suppose the two segments sent by A arrive in order at B. The first acknowledgment is lost, and the second acknowledgment arrives after the first timeout interval. Draw a timing diagram, showing these segments and all other segments and acknowledgments sent (***assume there is no additional packet loss***). For each segment in your figure, provide the sequence number and the number of bytes of data; for each acknowledgment that you add, provide the acknowledgment number.

# (8 marks)



1. Consider a channel that can lose packets but has a maximum delay that is known. Modify protocol RDT 2.1 to include sender timeout and retransmit. Informally argue why your protocol can communicate correctly over this channel.

# (6 marks)

# Here, we add a timer, whose value is greater than the known round-trip propagation delay. We add a timeout event to the “Wait for ACK or NAK0” and “Wait for ACK or NAK1” states. If the timeout event occurs, the most recently transmitted packet is retransmitted. Let us see why this protocol will still work with the rdt2.1 receiver.

# Suppose the timeout is caused by a lost data packet, i.e., a packet on the senderto-receiver channel. In this case, the receiver never received the previous transmission and, from the receiver's viewpoint, if the timeout retransmission is received, it looks exactly the same as if the original transmission is being received

# Suppose now that an ACK is lost. The receiver will eventually retransmit the packet on a timeout. But a retransmission is exactly the same action that if an ACK is garbled. Thus the sender's reaction is the same with a loss, as with a garbled ACK. The rdt 2.1 receiver can already handle the case of a garbled ACK

(Good Luck)