ANNA UNIVERSITY, GUINDY, CHENNAI:: 600 025
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Course Code: CS6111

Course Name: Computer Networks

Semester : V

Batch : N&Q

DATE:13.09.2024

Lab 8 – Flow Control Simulation

Consider a scenario where a sender (S) has '**n**' bytes of data to be sent to a receiver (R) which maintains a buffer of size '**m**' from which the data is read at a rate of '**i**' bytes for every '**j**' bytes received successfully. S sends the data as multiple packets, as the amount of data that can be sent at a time is limited by the maximum segment size (**MSS**). S uses sliding window protocol to determine the number of packets that can be sent at a time before receiving an acknowledgement from R. The size of the window is purely determined based on the advertised window sent by R as follows:

effective_window = maximum_window – {last_byte_sent – last_byte_acknowledged}

When a connection is initiated between S and R, the entire buffer in R is assumed to be free and hence the initial advertised window would be equal to 'm'. Eventually, as R receives more data from S, the size of the buffer decreases and hence the advertised window.

The k^{th} packet received by R is erroneous and hence acknowledgment is not sent by R for the packet.

Simulate this scenario using socket programming with S sending messages carrying the sequence number (starting with an initial sequence number (**ISN**) for the first packet transmitted) and the byte range of data to be sent in the packet. In return, R sends acknowledgements carrying the next expected byte and the advertised window for each received packet. On receiving an erroneous packet, R acknowledges the subsequent packets with the next expected byte as that sent for the previous packet. Complete the simulation when the effective window becomes 0 or when the entire data is transmitted, whichever occurs earlier. Print the messages received on either side. In addition, print the effective window every time it is updated in S.

As a follow-up, simulate fast retransmission on receiving 3 duplicate acknowledgements.

Sample:

Input on S: n = 2500, MSS = 100, ISN = 0

Input on R: m = 600, i = 50, j = 200, k = 5

Simulation:

R sends ACK = -1, AW = 600

1. S calculates effective_window = 600 and sends SEQ = 0, DATA = 0-99
2. S sends SEQ = 100, DATA = 100-199
3. S sends SEQ = 200, DATA = 200-299
4. S sends SEQ = 300, DATA = 300-399
5. S sends SEQ = 400, DATA = 400-499
6. S sends SEQ = 500, DATA = 500-599

1. R sends ACK = 100, AW = 500
2. R sends ACK = 200, AW = 450
3. R sends ACK = 300, AW = 350
4. R sends ACK = 400, AW = 300
6. R sends ACK = 400, AW = 300
7. S calculates effective_window = 100 and sends SEQ = 600, DATA = 600-699
7. R sends ACK = 400, AW = 350
8. S calculates effective_window = 50 and sends SEQ = 700, DATA = 700-749
9. R sends ACK = 400, AW = 350

   S calculates effective_window = 0 and stops sending data.

Spot Question :TCP Flow Control (NAGLE's)

Assume that a client has opened a TCP connection with a server to download a file. During setting up the connection, the client set its receive window to 4000 bytes (rwnd =4000) and the server set its Initial Sequence Number (ISN) to 100. Assume that no segments were lost for the whole duration of the connection.

The following events happened in order between the client and the server:

(a) (4 points) Once the connection is opened, the server sent 1000 bytes. How much more the server can send to the client without waiting for an ACK?

(b) (4 points) The server then received an ACK with a value of 600 . How many more bytes the server can send before receiving the next ACK?

(c) (4 points) The server then sent 2000 bytes segment. What is the sequence number of this TCP segment?

(d) (4 points) The server then received an ACK with a value of 2500 and a rwnd with a value of 7000 . How many more bytes the server can send before receiving the next ACK? Hint : don't forget to take into account the 3-way TCP handshake when calculating the acknowledgements and sequence number