Practical No :5

Problem statement:-Write a program to simulate Go back N and Selective Repeat Modes of Sliding Window Protocol in peer to peer mode and demonstrate the packets captured traces using Wireshark Packet Analyzer Tool for peer to peer mode.

Objectives:

1. To Learn Sliding Window Protocol
2. To understand the working of Go Back N and Selective Repeat Sliding Window Protocol

Theory:-

Sliding window protocol:-

Sliding Window Protocols assumes two-way communication (full duplex). It uses two types of frames:

Data

Ack (sequence number of last correctly received frame)

The basic idea of sliding window protocol is that both sender and receiver keep a ``window'' of acknowledgment. The sender keeps the value of expected acknowledgment; while the receiver keeps the value of expected receiving frame. When it receives an acknowledgment from the receiver, the sender advances the window. When it receives the expected frame, the receiver advances the window.

In transmit flow control, sliding window is a variable-duration window that allows a sender to transmit a specified number of data units before an acknowledgement is received or before a specified event occurs.

An example of a sliding window in packet transmission is one in which, after the sender fails to receive an acknowledgement for the first transmitted packet, the sender "slides" the window, i.e. resets the window, and sends a second packet. This process is repeated for the specified number of times before the sender interrupts transmission. Sliding window is sometimes (loosely) called *acknowledgement delay period*.

For example, supposing a fixed window size of *m* packets, a sender may send out packets

$[n \ldots (n+m-1)]$ before receiving any acknowledgement. If acknowledgement arrives from the receiver for packet *n*, then the range (window) of unacknowledged

packets slides to $[(n+1) \ldots (n+m)]$, and the sender is able to send out packet (*n*

+ *m*). In some way, "sliding" signifies a FIFO operation, trimming the range at one end, extending it at the other end.

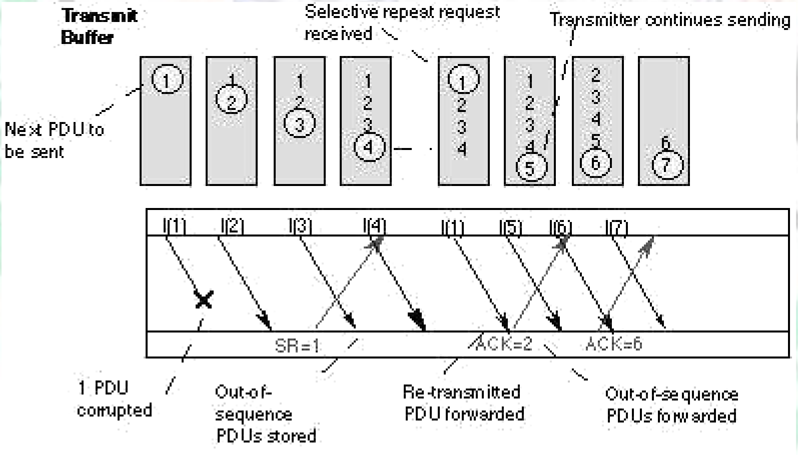
The purpose of the sliding window is to increase throughput. Let's denote the round trip time with RTT. The time necessary to transfer and acknowledge *K* (a big number of)

packets is roughly $\mathrm{RTT} \cdot K/(2m)$ (in one round trip, 2*m* packets and 2*m* ACKs are delivered). However, the size of the window (in bytes) should not grow above "capacity of the path" (the sum of affected network buffer sizes of all hops along the path): windows that are too big do not increase throughput; they only increase latency, the number of packets transmitted out-of-order, and memory usage.

In practice, protocols often adapt the window size to the link's speed and actual saturation or congestion.

Selective Repeat ARQ

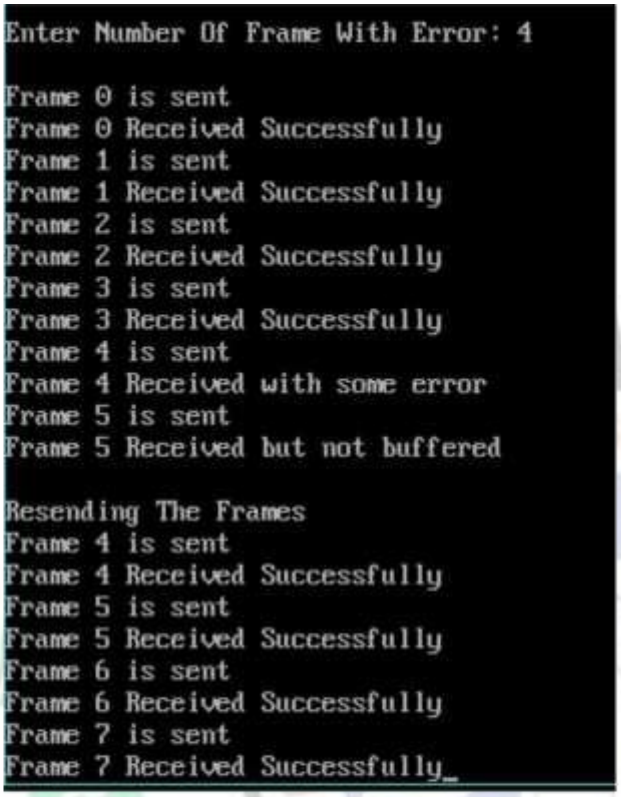
Selective Repeat ARQ is a specific instance of the Automatic Repeat-reQuest (ARQ) Protocol, in which the sending process continues to send a number of frames specified by a *window size* even after a frame loss. Unlike Go-Back-N ARQ, the receiving process will continue to accept and acknowledge frames sent after an initial error.


Transmit Buffer
Selective repeat request received
Transmitter continues sending
Next PDU to be sent
1
1 2
1 2 3
1 2 3 4
1 2 3 4
1 2 3 4 5
2 3 4 5 6
6 7
I(1)
I(2)
I(3)
I(4)
I(1)
I(5)
I(6)
I(7)
SR=1
ACK=2
ACK=6
1 PDU corrupted
Out-of-sequence PDUs stored
Re-transmitted PDU forwarded
Out-of-sequence PDUs forwarded

The receiver process keeps track of the sequence number of the earliest frame it has not received, and sends that number with every ACK it sends. If a frame from the sender does not reach the receiver, the sender continues to send subsequent frames until it has emptied its *window*. The receiver continues to fill its receiving window with the subsequent frames, replying each time with an ACK containing the sequence number of the earliest missing frame. Once the sender has sent all the frames in its *window*, it re- sends the frame number given by the ACKs, and then continues where it left off.

The size of the sending and recieving windows must be equal, and half the maximum sequence number (assuming that sequence numbers are numbered from 0 to n-1) to avoid miscommunication in all cases of packets being dropped. The sender moves its window for every packet that is acknowledged.

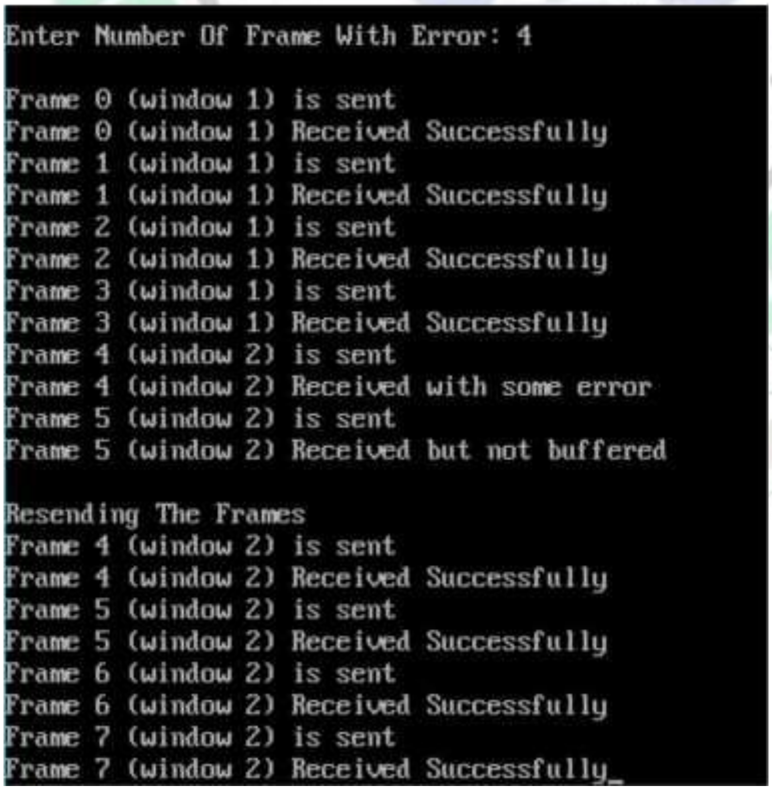
Sample Output:



```
Enter Number Of Frame With Error: 4

Frame 0 is sent
Frame 0 Received Successfully
Frame 1 is sent
Frame 1 Received Successfully
Frame 2 is sent
Frame 2 Received Successfully
Frame 3 is sent
Frame 3 Received Successfully
Frame 4 is sent
Frame 4 Received with some error
Frame 5 is sent
Frame 5 Received but not buffered

Resending The Frames
Frame 4 is sent
Frame 4 Received Successfully
Frame 5 is sent
Frame 5 Received Successfully
Frame 6 is sent
Frame 6 Received Successfully
Frame 7 is sent
Frame 7 Received Successfully_
```



```
Enter Number Of Frame With Error: 4

Frame 0 (window 1) is sent
Frame 0 (window 1) Received Successfully
Frame 1 (window 1) is sent
Frame 1 (window 1) Received Successfully
Frame 2 (window 1) is sent
Frame 2 (window 1) Received Successfully
Frame 3 (window 1) is sent
Frame 3 (window 1) Received Successfully
Frame 4 (window 2) is sent
Frame 4 (window 2) Received with some error
Frame 5 (window 2) is sent
Frame 5 (window 2) Received but not buffered

Resending The Frames
Frame 4 (window 2) is sent
Frame 4 (window 2) Received Successfully
Frame 5 (window 2) is sent
Frame 5 (window 2) Received Successfully
Frame 6 (window 2) is sent
Frame 6 (window 2) Received Successfully
Frame 7 (window 2) is sent
Frame 7 (window 2) Received Successfully_
```

APPLICATION:-

1. The sliding window implements reliability at both the data-link layer and the transport layer of the network protocol stack, like TCP/IP.

Conclusion: students understand and learn Sliding Window Protocol as well as the working of Go Back N and Selective Repeat Sliding window Protocol.

Signature with Date