

Implementation and Simulation of Go-Back-N and Selective Repeat Protocols

By: Hrishikesh Kasar (801053462) and Avadhut Naik (801045233)

UDP (User Datagram Protocol), defined in RFC 768 provides a connectionless and unreliable data transfer service between end users. Doing little beyond multiplexing and demultiplexing with addition of light error checking, it adds little to improve the unreliable data transfer service of the network layer protocol, IP. Unlike its counterpart, the Transmission Control Protocol, TCP, UDP doesn't implement the technique of handshaking between sender and receiver before initiating communication, (hence, connectionless). Neither does it provide a reliable data transfer service through ACK's and sequence numbers. These limitations however, are advantages for some applications like Internet Telephony and other loss-tolerant applications. With UDP providing greater control over data transmission as it almost immediately transmits data sent by the application utilizing it thus, TCP's retransmission mechanism which is quite favourable for loss – tolerant applications, it also skips TCP's handshaking mechanism for connection establishment along with congestion and flow control, all of which could lead to unnecessary latencies for applications that demand immediate data transmission.

Switching on to protocols that provide reliable data transfer service, Go-Back-N (GBN), also known as sliding window protocol, has been implemented as an enhancement rdt (reliable data transfer 3.0) protocol. Allowing sender to transmit multiple packets without waiting for an acknowledgement, the former is constrained by the window size N which defines the maximum number of unacknowledged segments with each segment in the window being assigned a unique sequence number, in accordance to which an ACK shall be sent by the receiver. With a timer implemented at the sender, retransmission of the entire window shall be initiated upon the event of timeout.

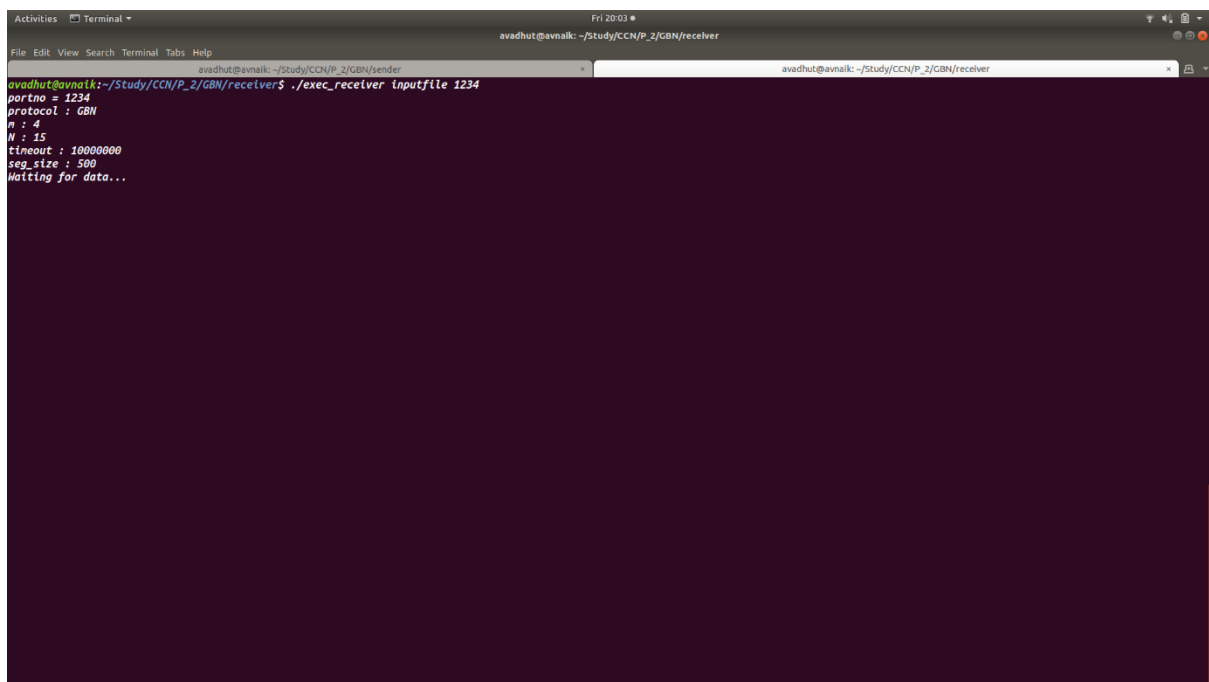
Selective Repeat (SR) is another example of a protocol providing reliable data transfer service while attempting to mitigate the disadvantages of Go-Back-N which can potentially lead to large retransmissions in case of timeout. Incorporating a transmitting window for packets in a similar manner to Go-Back-N, the Selective repeat however, does not provide cumulative but acknowledgements for individual segments. With timer implemented at sender, only the segment for which acknowledgement was not received shall resent in the event of a timeout, thereby reducing the Go-Back-N's need of large retransmissions.

In this project, we have implemented, the two aforesaid reliable data transfer protocols i.e. GBN and SR over the unreliable transport layer protocol UDP. Mechanisms to demonstrate their robustness in events of errors such as checksum error, lost packets and ACKS's have also

Implementation and Simulation of Go-Back-N and Selective Repeat Protocols

By: Hrishikesh Kasar (801053462) and Avadhut Naik (801045233)

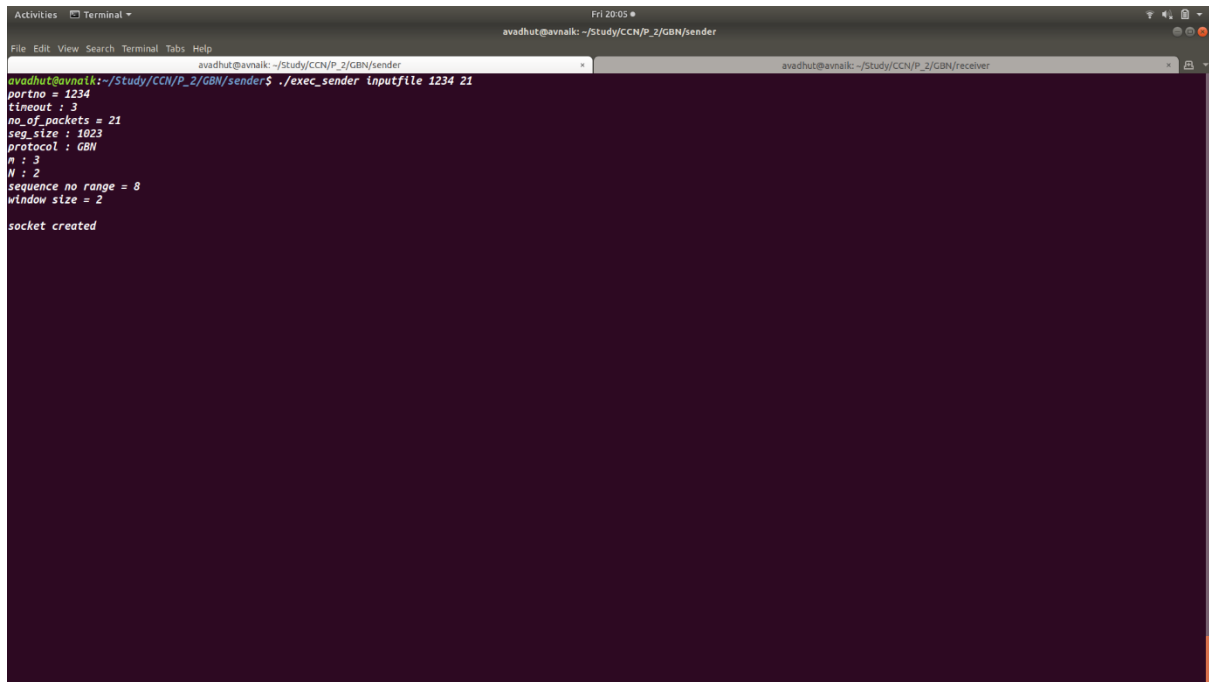
been deployed in the program, the implementation of which has been undertaken in C language. The codes for both Client and Server have been written, implemented and tested on the Desktop version of the Linux Operating System's 64-bit Ubuntu 17.10.1. Having completed writing the codes for both client and server in accordance with the requirements of the project and the useful tips provided for the same, the output, when the Server program was compiled and executed over the Linux terminal followed by the client, for the project requirements is as follows:

A screenshot of a Linux terminal window. The terminal has a dark purple background. At the top, there's a title bar with 'Activities' and 'Terminal' tabs. Below that, the terminal shows the command prompt 'avadhut@avnalk: ~/Study/CCN/P_2/GBN/receiver'. The user has entered the command './exec_receiver inputfile 1234'. The output of the command is displayed as follows: 'portno = 1234', 'protocol : GBN', 'n : 4', 'W : 15', 'timeout : 10000000', 'seg_size : 500', and 'Waiting for data...'. The terminal window also shows a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', 'Tabs', and 'Help'.

Giving the required inputs through the command line which included the input file, as per project requirements, we attempted to start the receiver.

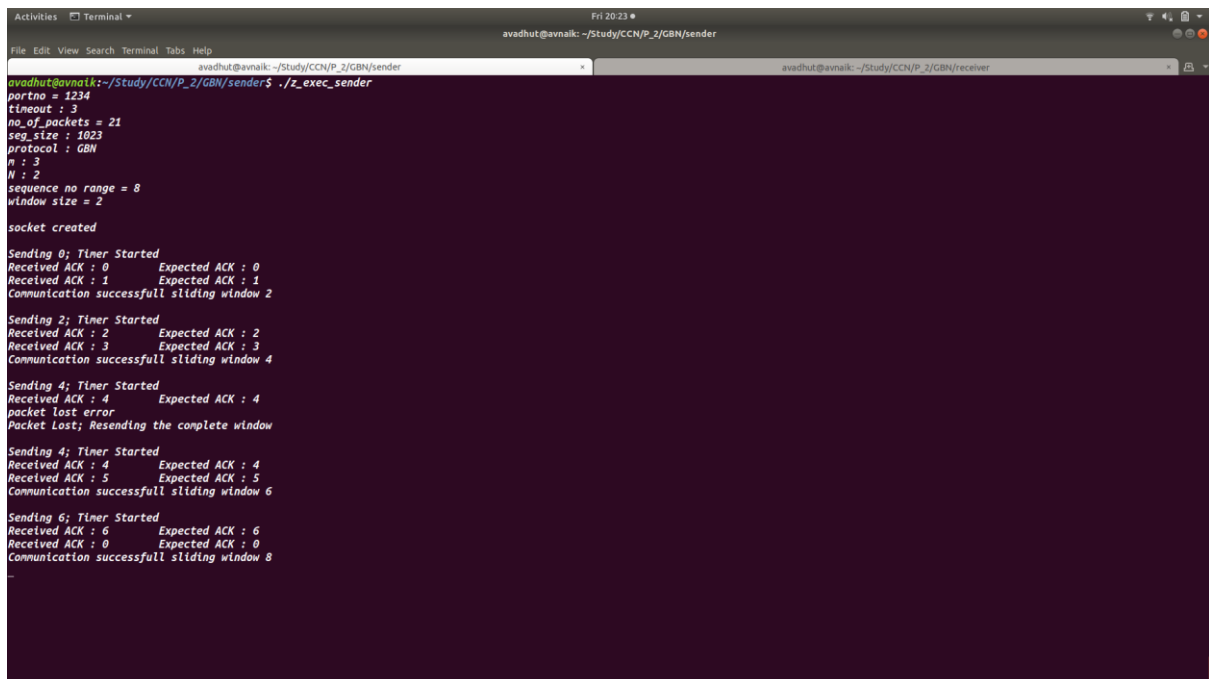
Implementation and Simulation of Go-Back-N and Selective Repeat Protocols

By: Hrishikesh Kasar (801053462) and Avadhut Naik (801045233)



```
avadhut@avnaiik: ~/Study/CCN/P_2/GBN/sender
avadhut@avnaiik:~/Study/CCN/P_2/GBN/sender$ ./exec_sender inputfile 1234 21
portno = 1234
timeout : 3
no_of_packets = 21
seg_size : 1023
protocol : GBN
n : 3
N : 2
sequence no range = 8
window size = 2
socket created
```

Giving the required inputs through the command line which included the input file, as per project requirements, we attempted to start the sender.



```
avadhut@avnaiik:~/Study/CCN/P_2/GBN/sender$ ./z_exec_sender
portno = 1234
timeout : 3
no_of_packets = 21
seg_size : 1023
protocol : GBN
n : 3
N : 2
sequence no range = 8
window size = 2
socket created

Sending 0; Timer Started
Received ACK : 0 Expected ACK : 0
Received ACK : 1 Expected ACK : 1
Communication successfull sliding window 2

Sending 2; Timer Started
Received ACK : 2 Expected ACK : 2
Received ACK : 3 Expected ACK : 3
Communication successfull sliding window 4

Sending 4; Timer Started
Received ACK : 4 Expected ACK : 4
packet lost error
Packet Lost; Resending the complete window

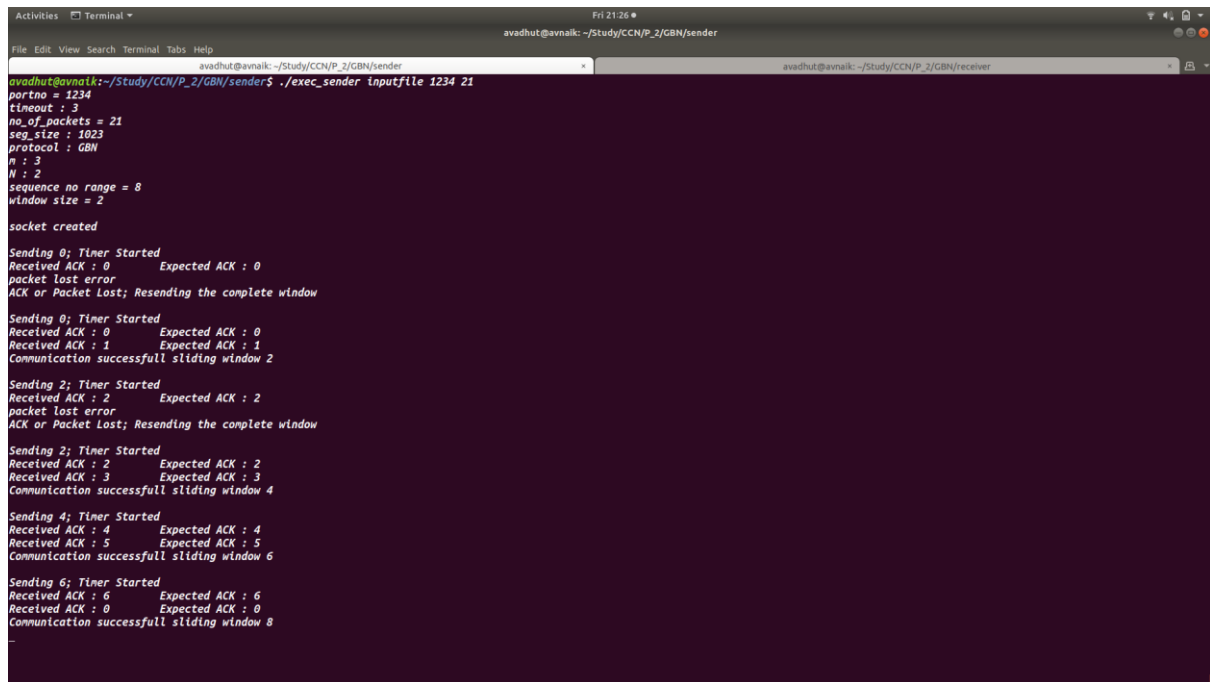
Sending 4; Timer Started
Received ACK : 4 Expected ACK : 4
Received ACK : 5 Expected ACK : 5
Communication successfull sliding window 6

Sending 6; Timer Started
Received ACK : 6 Expected ACK : 6
Received ACK : 0 Expected ACK : 0
Communication successfull sliding window 8
```

Implementation and Simulation of Go-Back-N and Selective Repeat Protocols

By: Hrishikesh Kasar (801053462) and Avadhut Naik (801045233)

With both the client and server running the transmission has been initiated. As per project requirements, the segments being sent are displayed over the command line along with expected and received ACK's.



```
avadhut@avnaiik: ~/Study/CCN/P_2/GBN/sender
avadhut@avnaiik:~/Study/CCN/P_2/GBN/sender$ ./exec_sender inputfile 1234 21
partno = 1234
timeout : 3
no_of_packets = 21
seq_size = 1023
protocol : GBN
n : 3
N : 2
sequence no range = 8
window size = 2
socket created

Sending 0; Timer Started
Received ACK : 0      Expected ACK : 0
packet lost error
ACK or Packet Lost; Resending the complete window

Sending 0; Timer Started
Received ACK : 0      Expected ACK : 0
Received ACK : 1      Expected ACK : 1
Communication successfull sliding window 2

Sending 2; Timer Started
Received ACK : 2      Expected ACK : 2
packet lost error
ACK or Packet Lost; Resending the complete window

Sending 2; Timer Started
Received ACK : 2      Expected ACK : 2
Received ACK : 3      Expected ACK : 3
Communication successfull sliding window 4

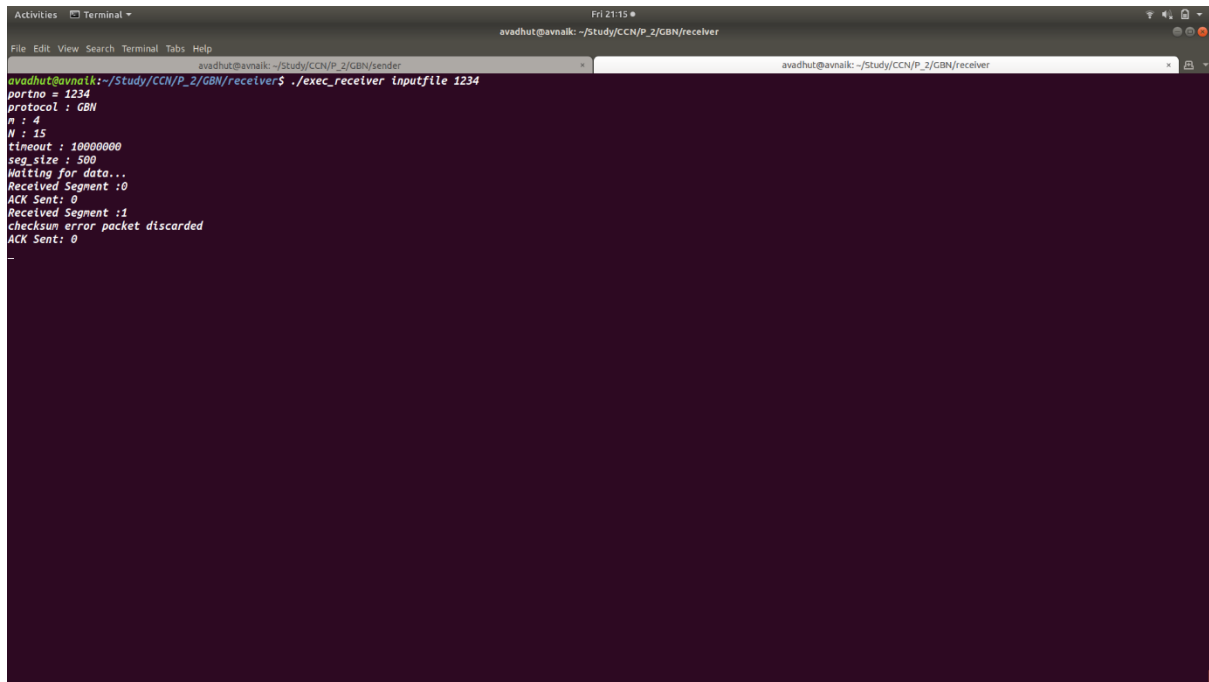
Sending 4; Timer Started
Received ACK : 4      Expected ACK : 4
Received ACK : 5      Expected ACK : 5
Communication successfull sliding window 6

Sending 6; Timer Started
Received ACK : 6      Expected ACK : 6
Received ACK : 0      Expected ACK : 0
Communication successfull sliding window 8
-
```

Introducing error mechanism, entire windows one and three were retransmitted due to lost ACK or lost Packet, as per GBN's retransmission mechanism.

Implementation and Simulation of Go-Back-N and Selective Repeat Protocols

By: Hrishikesh Kasar (801053462) and Avadhut Naik (801045233)

A terminal window with a dark purple background. The title bar shows 'Fri 21:15' and the user is 'avadhut@avnalk'. The terminal shows the execution of a receiver program. The output includes configuration parameters like portno, protocol, n, N, timeout, and seg_size. It shows the receiver waiting for data, receiving segment 0, and then segment 1. A checksum error is reported for segment 1, and the packet is discarded. The ACK sent remains 0.

```
avadhut@avnalk:~/Study/CCN/P_2/GBN/receiver$ ./exec_receiver inputfile 1234
portno = 1234
protocol : GBN
n : 4
N : 15
timeout : 10000000
seg_size : 500
Waiting for data...
Received Segment :0
ACK Sent: 0
Received Segment :1
checksum error packet discarded
ACK Sent: 0
-
```

Introducing Checksum error mechanism for Packet 1, ACK of the packet last received correctly is being sent.

Implementation and Simulation of Go-Back-N and Selective Repeat Protocols

By: Hrishikesh Kasar (801053462) and Avadhut Naik (801045233)

```
Activities Terminal
Fri 20/24
avadhut@avnalk: ~/Study/CCN/P_2/GBN/receiver
File Edit View Search Terminal Tabs Help
avadhut@avnalk:~/Study/CCN/P_2/GBN/sender avadhut@avnalk:~/Study/CCN/P_2/GBN/receiver
N : 15
timeout : 10000000
seg_size : 500
Waiting for data...
Received Segment :0
ACK Sent: 0
Received Segment :1
ACK Sent: 1
Received Segment :2
ACK Sent: 2
Received Segment :3
ACK Sent: 3
Received Segment :4
ACK Sent: 4
Received Segment :5
ACK Sent: 4
Received Segment :4
ACK Sent: 4
Received Segment :5
ACK Sent: 5
Received Segment :6
ACK Sent: 6
Received Segment :0
ACK Sent: 0
Received Segment :1
ACK Sent: 1
Received Segment :2
ACK Sent: 2
Received Segment :3
ACK Sent: 3
Received Segment :4
ACK Sent: 4
Received Segment :5
ACK Sent: 5
Received Segment :6
ACK Sent: 6
Received Segment :0
ACK Sent: 0
Received Segment :1
ACK Sent: 1
Received Segment :2
ACK Sent: 2
Received Segment :3
ACK Sent: 3
Received Segment :4
ACK Sent: 4
```

ON the receiver side, as packet number 5 is incorrectly received ACK of packet 4 is sent, in the aftermath of which packet 5 is retransmitted.

```
Activities Terminal
Fri 22/24
avadhut@avnalk: ~/Study/CCN/P_2/sender
File Edit View Search Terminal Tabs Help
avadhut@avnalk:~/Study/CCN/P_2/sender avadhut@avnalk:~/Study/CCN/P_2/GBN/receiver
avadhut@avnalk:~/Study/CCN/P_2/sender$ ./exec_sender inputfile 1234 21
portno = 1234
timeout : 3
no_of_packets = 21
seg_size : 1023
protocol : SR
n : 3
N : 4
sequence no range = 8
window size = 4
socket created

Sending 0; Timer Started
Received ACK : 0 Expected ACK : 0
Received ACK : 1 Expected ACK : 1
Received ACK : 2 Expected ACK : 2
Received ACK : 3 Expected ACK : 3
Communication successfull sliding window 4

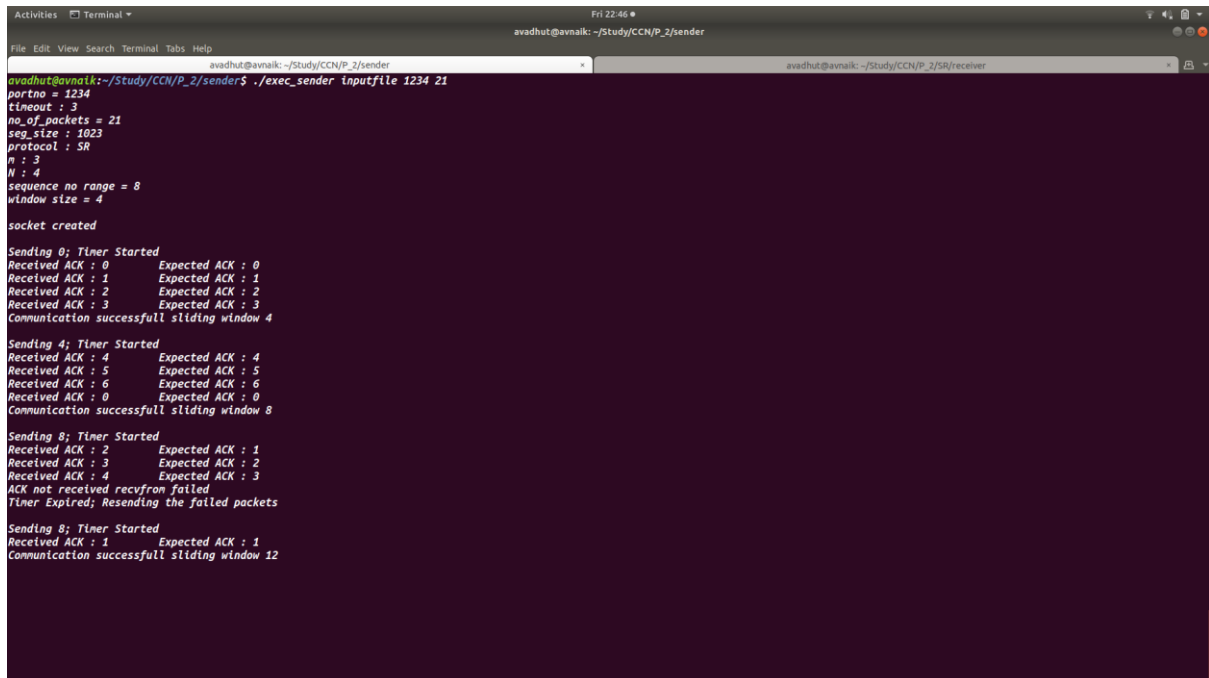
Sending 4; Timer Started
Received ACK : 4 Expected ACK : 4
Received ACK : 5 Expected ACK : 5
Received ACK : 6 Expected ACK : 6
Received ACK : 0 Expected ACK : 0
Communication successfull sliding window 8

Sending 8; Timer Started
Received ACK : 1 Expected ACK : 1
Received ACK : 2 Expected ACK : 2
Received ACK : 3 Expected ACK : 3
Received ACK : 4 Expected ACK : 4
Communication successfull sliding window 12
```

Implementation and Simulation of Go-Back-N and Selective Repeat Protocols

By: Hrishikesh Kasar (801053462) and Avadhut Naik (801045233)

Giving the parameters through the command line to initiate transmission through the SR Protocol, the sending and receiving is successful.



```
avadhut@avnai: ~/Study/CCN/P_2/sender
avadhut@avnai:~/Study/CCN/P_2/sender$ ./exec_sender inputfile 1234 21
portno = 1234
timeout : 3
no_of_packets = 21
seg_size : 1023
protocol : SR
n : 3
N : 4
sequence no range = 8
window size = 4
socket created

Sending 0; Timer Started
Received ACK : 0      Expected ACK : 0
Received ACK : 1      Expected ACK : 1
Received ACK : 2      Expected ACK : 2
Received ACK : 3      Expected ACK : 3
Communication successfull sliding window 4

Sending 4; Timer Started
Received ACK : 4      Expected ACK : 4
Received ACK : 5      Expected ACK : 5
Received ACK : 6      Expected ACK : 6
Received ACK : 0      Expected ACK : 0
Communication successfull sliding window 8

Sending 8; Timer Started
Received ACK : 2      Expected ACK : 1
Received ACK : 3      Expected ACK : 2
Received ACK : 4      Expected ACK : 3
ACK not received rcvfrom failed
Timer Expired; Resending the failed packets

Sending 8; Timer Started
Received ACK : 1      Expected ACK : 1
Communication successfull sliding window 12
```

Introducing error mechanism in the form of lost ACK, the segment 1 was retransmitted as ACK wasn't received before its timer expired

Implementation and Simulation of Go-Back-N and Selective Repeat Protocols

By: Hrishikesh Kasar (801053462) and Avadhut Naik (801045233)

Instructions for Execution of Program

1) Heading to the directory where the receiver program is stored from the terminal of Linux, execute the make file for the server, to compile the program.

```
$ make
```

2) Execute the z_exec_receiver file to start the receiver.

```
$ ./exec_receiver inputfile 1234
```

3) On a new terminal window, heading to the directory of sender program, execute the make file to compile the server program.

```
$ make
```

4) Execute the z_exec_sender file to start the sender.

```
$ ./exec_sender inputfile 1234 21
```


Implementation and Simulation of Go-Back-N and Selective Repeat Protocols

By: Hrishikesh Kasar (801053462) and Avadhut Naik (801045233)

References: -

- 1) James F. Kurose and Keith W. Ross COMPUTER NETWORKING: A Top-Down Approach