Lab 3 : Socket Programming

Team Details:

Paarth Jain : 190050076

Sahasra Ranjan : 190050102

Sibasis Nayak : 190050115

Overview

We implemented two C socket programs, **sender.c** and **receiver.c** hat communicate together using “Data- gram Sockets (UDP)”

We also implemented the ARQ using the time.h library and the setsockopt command in socket programming

Usage Instructions

Sender:

```
gcc sender.c -o sender
```

```
sender.c <SenderPort> <ReceiverPort> <RetransmissionTimer>
<NoOfPacketsToBeSent>
```

Receiver:

```
gcc receiver.c -o receiver
```

```
receiver.c <ReceiverPort> <SenderPort> <PacketDropProbability>
```

P.S:

- Might need to use `-lm` flag for both the codes in case it fails to compile.
- Need to run receiver.c first as our code doesn't handle exceptions well

Two files will be generated with the receiver and sender side output:

- sender.txt
- receiver.txt

Overview of the code

Structure of the frame:

```
struct frame{
    int ack;
    int seqNo;
    char data[1024];
};
```

Packet Format : “Packet:”

Acknowledgment Format:“Acknowledgment:”

sender.c

Socket binding:

```
if((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0){
    perror("socket creation failed");
    exit(EXIT_FAILURE);
}

memset(&receiver, '\0', sizeof(receiver));
memset(&sender, '\0', sizeof(sender));
receiver.sin_family = AF_INET;
receiver.sin_port = htons(receiverPort);
receiver.sin_addr.s_addr = inet_addr("127.0.0.1");
unsigned int size_rec = sizeof(receiver);
```

Send and receive data:

```
sendto(sockfd, &send, sizeof(send), 0, (struct sockaddr*)&receiver,
sizeof(receiver));

int rec_size = recvfrom(sockfd, &rec, sizeof(rec), 0, (struct
sockaddr*)&receiver, &size_rec);
```

Main program:

```
while(seqNo <= P){
        // Prepare frame
        // Send frame instructions
        send_frame:
            // Instructions

        if(rec_size > 0 && rec.ack == 1){
            if(rec.seqNo == seqNo+1){
                // Accept the ACK and send next packet
            }else{
                    // Ignore the ACK
                    continue;
                }else{
                  // Invalid ACK, so program will send the frame
again with the same timer
                    goto send_frame;
                }
            }
```

```
        }else{
            // Timer Expired
        }
    }
```

receiver.c

Socket binding:

```
if((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0){
        perror("socket creation failed");
        exit(EXIT_FAILURE);
    }

  memset(&receiver, '\0', sizeof(receiver));
  memset(&sender, '\0', sizeof(sender));
  receiver.sin_family = AF_INET;
  receiver.sin_port = htons(ReceiverPort);
  receiver.sin_addr.s_addr = inet_addr("127.0.0.1");

  if(bind(sockfd, (struct sockaddr*)&receiver, sizeof(receiver)) < 0)
{
    perror("bind failure");
    exit(EXIT_FAILURE);
  }
  socket_size = sizeof(sender);
```

Receive and send data:

```
int rec_size = recvfrom(sockfd, &rec, sizeof(rec), 0, (struct
sockaddr*)&sender, &socket_size);

sendto(sockfd, &send, sizeof(send), 0, (struct sockaddr*)&sender,
socket_size);
```

Main program:

```
while(1){
  // Receive data
  // Check the received data
    if (rec_size > 0 && seqNo == rec.seqNo && rec.ack == 0){
      if(seqNo == rec.seqNo){
        float random = (float)rand()/RAND_MAX;
        if(random < dropProb){
          // No ACK generated
          continue;
        }else{
          // Generate ACK
          // Send ACK
          seqNo++;
        }
      }else{
        // Incorrect seq no
```

```
        // Send ACK with old seq no (as mentioned in the prob
statement)
      }
    }else{
      // Frame not received
    }
  }
```

References

UDP Server-Client implementation in C - GeeksforGeeks

https://github.com/nikhilroxtomar/UDP-Client-Server-Program-in-C

Beej's Guide to Network Programming