Foundation of Computer Networks - Project 3

Bhavin Oza (bo2115)

Structure of the project:
There are following files in the project:
Sender.java
Receiver.java
CustomPacket.java
Utils.java
Main.java
The Sender and Receiver are child of Thread class
Main class is the entry point for the program.
On the basis of the number of parameters passed to main() in Main.java, it decides whether it is a Sender or Receiver instance.
Sender expects 2 parameters: filename and receive address
Receiver expectes 0 parameter

Implementation of the project:
Two main points covered in this project are:
1. Maintaining the order of the transmission of the packets,
2. Avoiding sending of previous packets if they are lost.

The implementation of the project is done using datagram sockets and packets, following the tcp method of hand shake and packet acknowledgement.
The data is sent sequentially here, with each packet sent and its acknowledgement just after it. The next packet is sent after the acknowledgement for the previous packet is received. Thus, this way no reordering of packets is required at the receiving end and no data is lost in between the transmission. The data at the receiving end is stored as it is received.
Instead of the TCP method of receiver sending an ACK with the sequence number to be sent next, here we have a simple way of maintaining the sequence of packet transfer. The receiver sends back the same sequence number to the sender, the sequence number is the first byte of the packet that is received. The sender gets it easy to verify the received ACK with the recently sent packet's SEQ, thereby validating if the packet was successfully sent.
The Sender file reads the data of the file, and then sends the data in packets to the receiver.
The SEQ is the starting byte of the data being sent to the receiver.
The Receiver sends an ACK back to the server, the ACK being the same SEQ that the Sender had sent.
The Sender on getting the ACK checks if ACK == SEQ, then it confirms that the packet was delivered.
Until the sender receives any ACK, it won't transmit any other data to the Receiver.
The sender at the end of complete file data transmission sends a packet with FIN request, to which the receiver accepts and closes the socket connection, and the same is done by the sender.

Following commands are used for starting the sender and receiver containers:

Foundation of Computer Networks - Project 3
Bhavin Oza (bo2115)

Sender command:

```
docker run -it -p 8081:8080 --cap-add=NET_ADMIN --net nodenet --ip 172.18.0.21 javaapptest rit.jpg 172.18.0.22
```

It takes filename and receiverIP as input parameters for the program.

Receiver command:

```
docker run -it -p 8082:8080 --cap-add=NET_ADMIN --net nodenet --ip 172.18.0.22 javaapptest
```