CSC361: Programming Assignment 3

Shirley Shu

V00886800

Section A:

1. Documentation available? (1 point)

Yes, the implementation of my RDP will be described in Section B.

2. Without setting any packet loss in the link, code works perfectly for transmitting small.html? (4.5 points)

Yes, see video demo for details.

3. Without setting any packet loss in the link, code works perfectly for transmitting medium.html or large.html? (3.5 points)

Yes, see video demo for details.

4. With setting 5% packet loss rate in each link, code works perfectly for transmitting small.html? (3.5 points)

Yes, see video demo for details.

5. With setting 5% packet loss rate in each link, code works perfectly for transmitting medium.html or large.html? (3.5 points)

Yes, see video demo for details.

6. Bonus part.

No, I didn't do any bonus part.

Section B:

There are three code files in my submission, which are client, server, and helper. To run the protocol, you need to put these three files together with the html file you want to get in the same folder in Mininet machine. Use code `$sudo mn –x` to run the RDP without packet loss and code `$sudo mn --link tc,loss=5 –x` to run the RDP with 5 percent packet loss.

In host h1, go to the directory of the code. Use code `$python server.py` to run the server.

In host h2, go to the directory of the code. Use code `$python client.py 10.0.0.1 6789 large.html` to run the client.

Then, the scree will show the information of transmission and packet timeout. After the transmission finish, it will show the MD5 code for both original file and received file. If they are same, it means the transmission works well.

RDP Header:

1. GET: only 0 or 1, indicate whether it is an HTTP request packet
2. SYN: only 0 or 1, indicate whether it is a synchronization packet
3. ACK: only 0 or 1, indicate whether it is an acknowledgement packet
4. FIN: only 0 or 1, indicate whether it is a finish packet
5. Data length: length of transmission data
6. Sequence number: sequence number
7. Acknowledge number: acknowledge number

RDP payload: transmission data or just empty for the flag packet

Implementation:

Use python socket to bind two host as client and server.

Use dictionary to include RDP header and payload as the transmission packet.

Use 800 bytes as the maximum segment size to contain the data.

Use md5 to check the correction of data at the end of transmission.

Use socket timeout to avoid packet loss and resend the packet.

Transmission Steps:

Step1: 3-way hand shake

Server is waiting. Client send the SYN packet to server. If server receive the SYN, it will send the ACK packet. If client receive the ACK packet, it will send another ACK packet. Overall, connection between the Client and Server is established.

Step2: Data Transmission

Once the connection is established, client send a GET to request for data. If server receive the GET, it will return the data and ack number. If the data is too large to put in one packet, server will split them and send them one by one. For here, I use the stop-and-wait concept. Thus, server will send the next part of data until it receives the ack from the client.

Step3: finish connection

Once the server sent the last part of data, it will send a FIN to finish the transmission. If client receive the FIN, it will return a ack and close the socket.