**Project 4 Zhengguan Li 113383597**

1. **Problem solving:**

***The flow of this project is as follows:***

***Basic info:***

***udpclient:***

*ask user to enter the username*

*ask user to enter the password*

***udpserver****:*

*verify the username and password, if not, send feedback to udpclient*

***details:***

***Udpclient****: send information in the form: group name + “$” + “option”--->Udpsender*

***Udpserver****: split the info received from udpclient: group name, request(according to the special character ‘$’)*

*Check if the group name received from udpclient matches the group name created before:*

*If matches, get the request join and the array has space to join the array:*

*Then add the udpclient to the array of client which is of maximum size: send “Positive match“ message to udpclient*

*Else(exception handling):*

*if group name does not match: send “Negative match” message to udpclient*

***udpclient:***

*ask user to enter the filename*

***udpserver****:*

*check if the file is valid, if not, send feedback to udpclient*

***details:***

***udpsender****:*

*fopen() to open the file, it will return a file descriptor fp, if fp == null, then file not valid,*

*send feeback to udpclient*

1. **Data Structure:**
   1. ***The data for this project mainly include username, password and value****,* ***two data structure can be used here(I derived it from project 1):***
2. *Username, password and value can be grouped by array, i.e.*

*char\* gusername[NUMBER];*

*char\* gpassword[NUMBER];*

1. *Use struct:*

*Struct company {*

*Char \* username;*

*Char \* password;*

*}*

***For this project, I choose the 1st way to store my data:***

I read “./user.txt” file using file I/O:

*FILE\* fp = fopen("user.txt","r");*

And Read every item:

*for(int i = 0; i < 6;i++){*

*gusername[i]=(char\*)malloc(256\*sizeof(char));*

*gpassword[i]=(char\*)malloc(256\*sizeof(char));*

*}*

*for(int i = 0; i < 6; i++){*

*for(int j = 0; j < 2; j++){*

*fscanf(fp, "%s", buffer);*

*if(j%2 == 0){*

*strcpy(gusername[i],buffer);*

*}*

*else if(j%2 == 1){*

*strcpy(gpassword[i], buffer);*

*}*

*}*

*}*

* 1. *I also define a struct to store the info of frames, it includes 4 fields：*

*struct frame{*

*int kind;//transmission(0) or retransmission(1)*

*int seq;*

*int ack;*

*char info[FRAMESIZE];*

*};*

*The kind field tells us whether this frame is transmission frame ot retransmission frame,*

*The seq and ack fields are used for sequence numbers and acknowledgements, respectively;*

*The info field of a data frame contains a single packet*

*It is defined both in udpsender, as well as udpclient*

* 1. *I create a list to store each frame called frameList in udpclient, it is defined below:*

*struct frame frameList[windowSize];*

*//it is an array and it stores a list of struct frame.*

*So everytime I receive a frame from sender, I will put the frame into frameList.(details in 2.4)*

*And I also create a list to record if the frame was received before called ackList, it is defined below:*

*int ackList[windowSize];*

*So everytime I receive a frame from sender, I will set the corresponding elment in ackList to 1.(details in 2.4)*

* 1. *When I receive an frame, I will chech its’ sequence number if it falls within the window. If it is, then it’s accepted and stored. Here I just use an if condition to check this:*

*if(recvFrame.seq >= 0 && recvFrame.seq < windowSize && ackList[recvFrame.seq]！= 0){*

*//store the frame in frameList*

*frameList[recvFrame.seq] = recvFrame;*

*ackList[recvFrame.seq] = 1;*

*}*

*else{*

*printf("Sequence number does not fall within the window\n");*

*}*

1. **User define function:**

*I defined multiple functions in udpserver.c and udpclient.c file:*

1. *The declaration of these functions in udpserve.c are as follows:*

*int Authenticate(char username[], char password[]){*

*int index = -1;//-1 means no matching username;*

*for(int i = 0; i < 6; i++){*

*if(strcmp(gusername[i],username) == 0 && strcmp(gpassword[i], password) == 0){*

*return 0;*

*}*

*}*

*return -1;*

*};*

*This function is used to check the pair, username and password, if they are right, then return 0,*

*Or else return -1*

1. *Time out*

*Every time I send a frame, I will create a new thread for timing, the thread function is defined as below:*

*void\* timer(void\* arg){*

*struct argsThread args;*

*args = \*(struct argsThread\*)arg;*

*int seq = args.seq;*

*sleep(0.00001);*

*if(args.ackList[seq] == 1){*

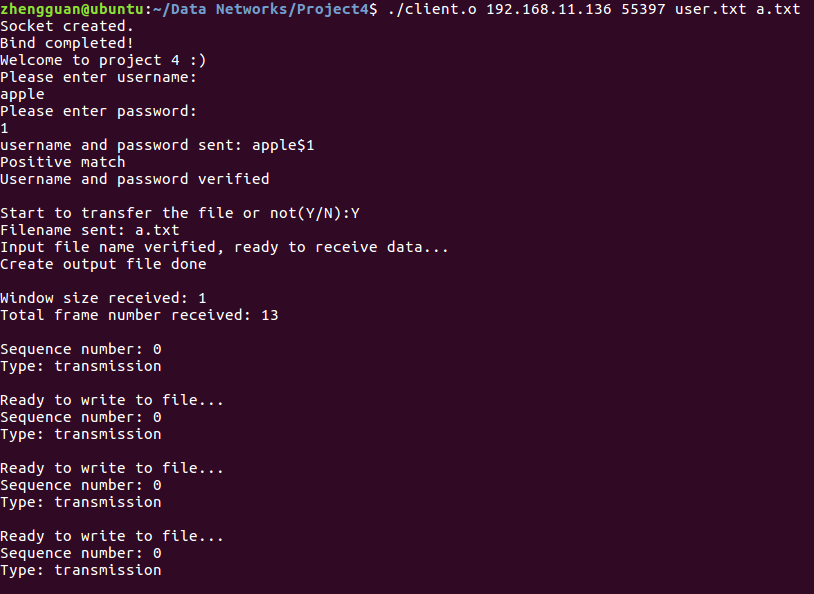
*printf("Time out\n");*

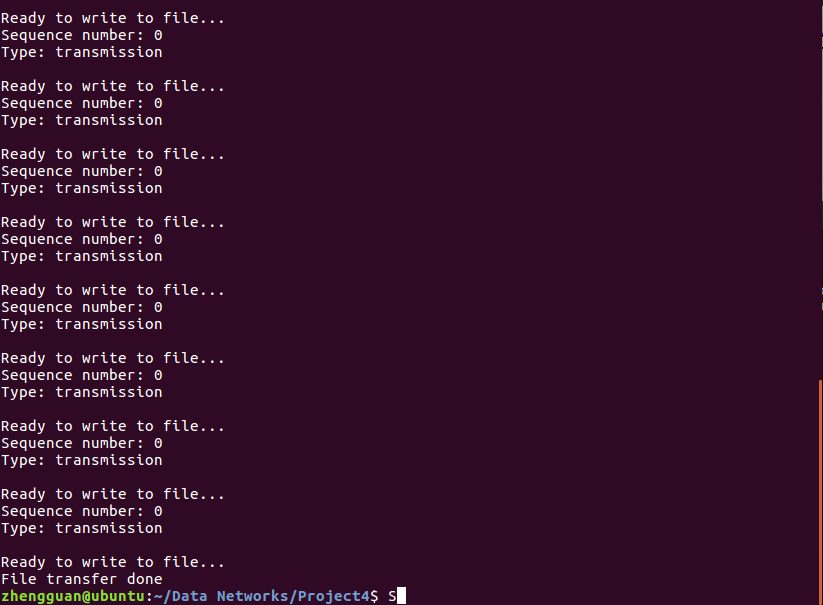
*}*

*};*

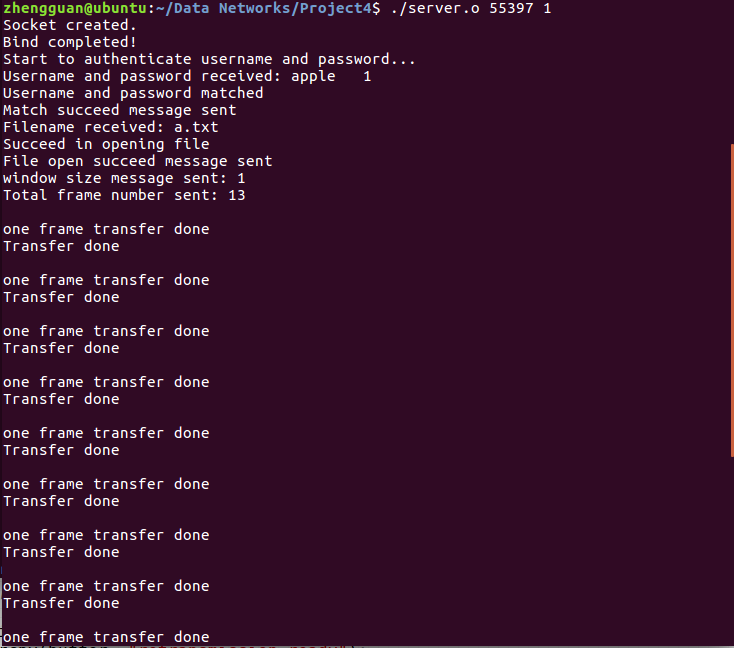
1. **Screen shots of outputs:**

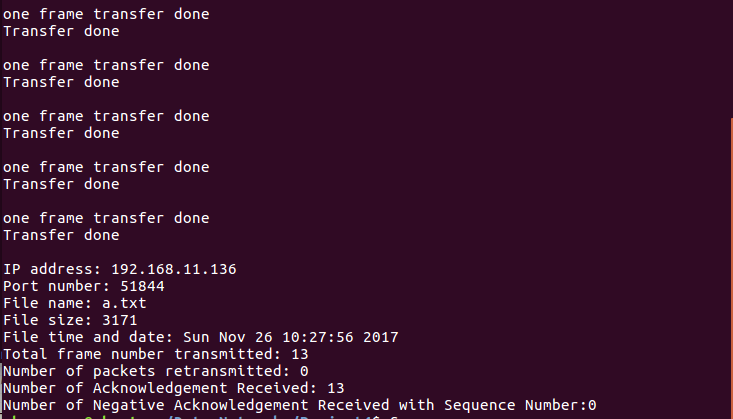
*Client:*





*Server:*

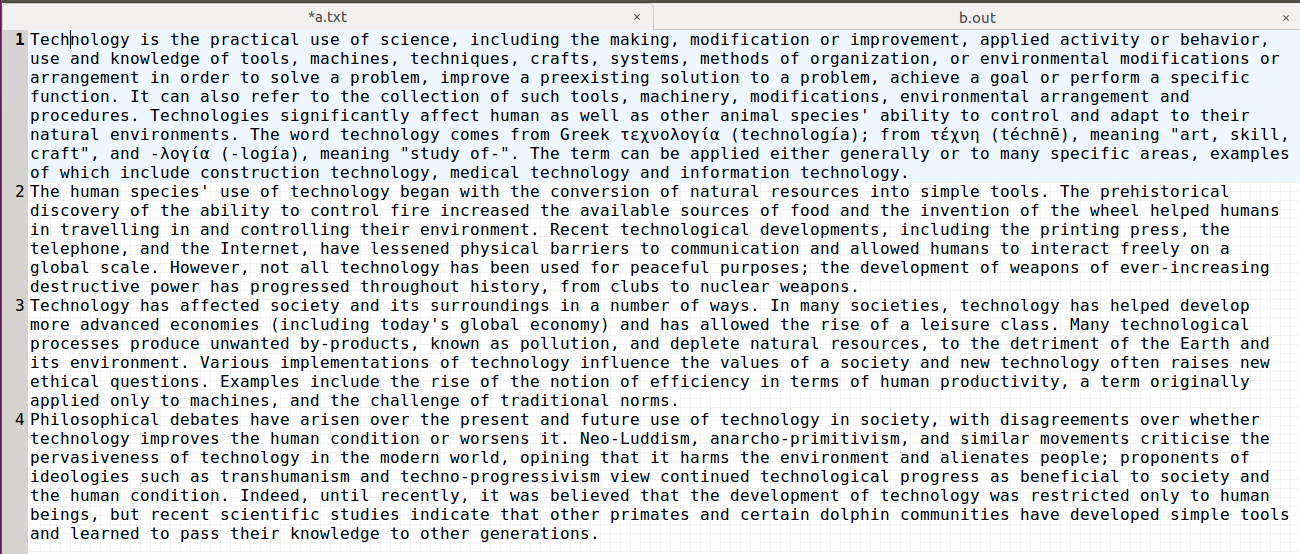




*Admission ☹:*

*I have to admit, my program still has some flaws, I copy the buffer into the frame info, but the content of them are not exactly the same, I don’t know why. Need to fix later. Thanks.*

*File compare:*

*Original:* 

*My hurtbreaking result:*

