COMP3015 Data Communication and Networking

Course Project

Name: Zou Xueyan

Student Id: 13253956

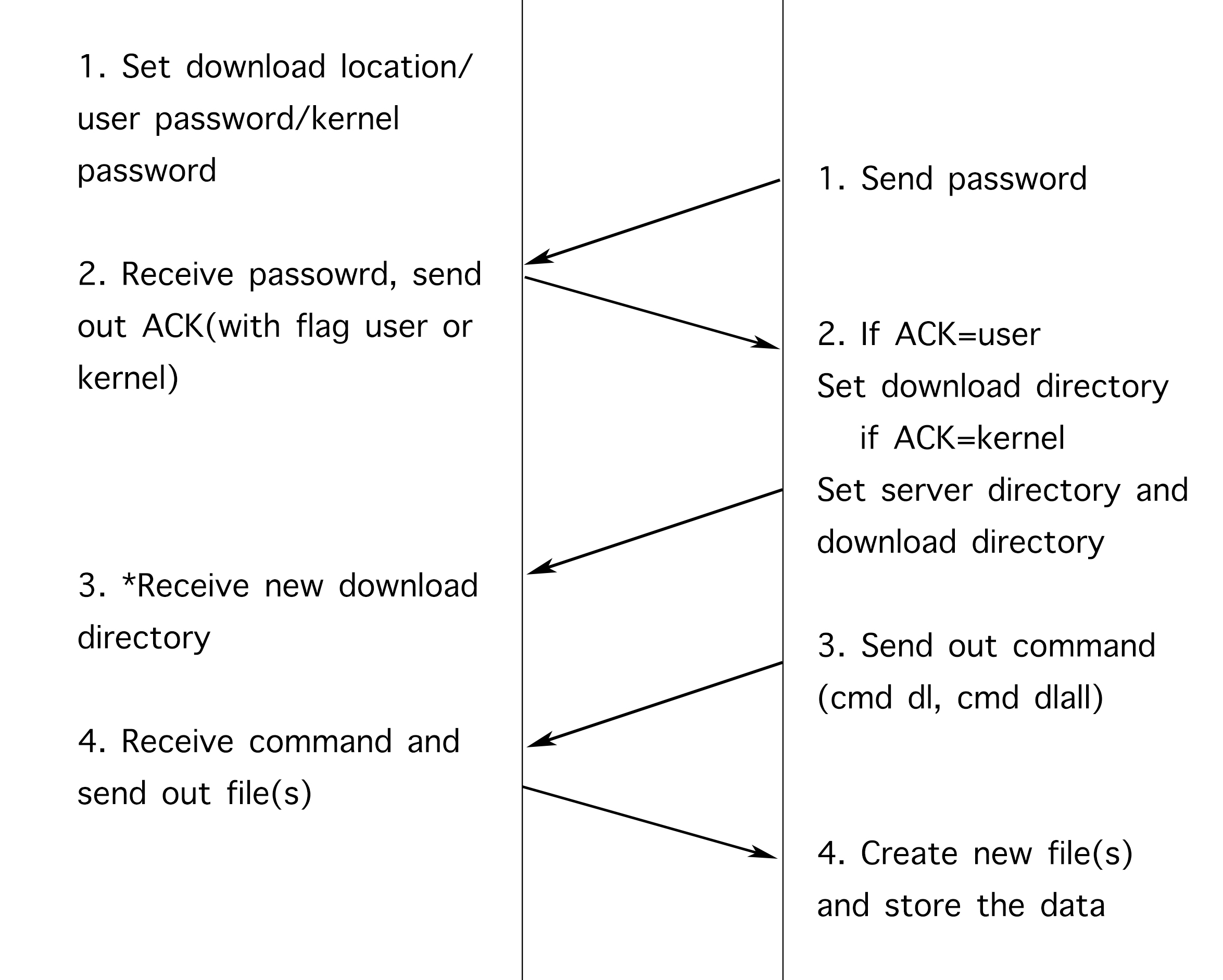
1. Description

This is a graphical user interface supported file transfer system, which could transfer all types of files between two end systems that are connecting to the internet. It supports the following function:

1. Simutaneously Multi-Thread client connection
2. Simutaneously Multi-Thread file downloading

*\*In the test procedure, for downloading a 102.6M files inside loacal computer, without multi-thread downloading, it needs 0.51sec, however, with multi-thread mode, it could reduce the time to 0.29sec.*

1. User and Kernel mode supported
2. Support directory operation
3. Support encryption data transfer(The data was encoded before sending out)
4. Support graphical user interface
5. Protocol Demonstration



1

The above is the main transferring protocol between server and client. As it is a GUI program, which doesn’t need the internal command, below is the demonstration of using the “Application” for both client and server.

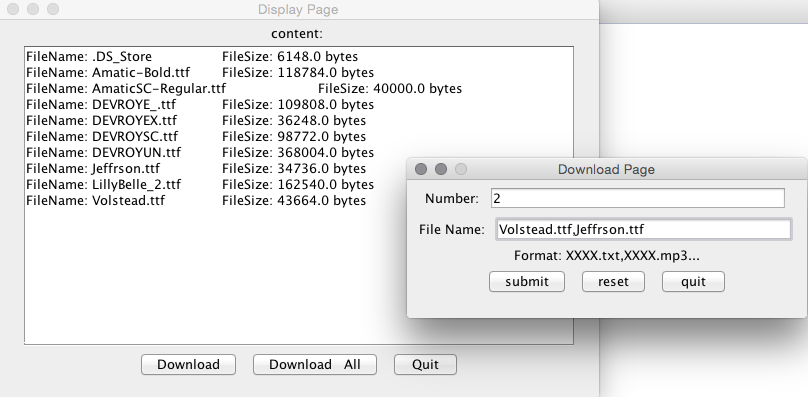
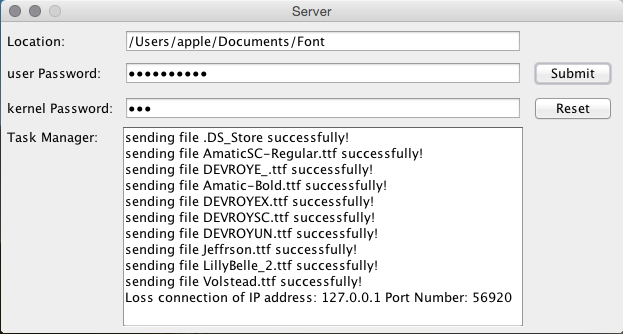


Figure 1.1 Server Figure 1.2 Client

Server:

Figure 1.1 is the display page of server. And the Location should be entered in the format of directory like “/Users/apple/Documents/Font”. Server should enter both password for user mode(The download directory is set by the server) and kernel mode(The download directory could reset by the client).

The task manager can display the information of the files that have

already been sent out, and the disconnected information. After fill in all

the blanks, click on submit to establish a server socket.

Client:

After correctly entering all the information, including password and

download location. The client would establish connection with server.

And figure 1.2 is displayed. If we click on the download all button,

all the files would be downloaded to the current directory. Or, if download

button is clicked, a new window would be pop out, in which you should

enter the number of files that you can download and the exact file name

split by ‘,’.

1. Flow chart

In the server application, there exist two server socket in order to realize the multi-tread downloading files for every client. One socket is used for connect to the client, the other one is used to connect to the client socket which is used for downloading file.

Macintosh HD:Users:apple:Desktop:COMP3015.png

Pseudo Code:

Server:

public class server{

public server(){

new connection\_ServerSocket();

preprocessing();

accept();

disconnect();

}

public void preprocessing(){

set directory, userPassword, kernelPassword;

display GUI;

}

public void accept(){

while(true){

accpt clientSocket;

new connectClient().start();

}

}

public void disconnect(){

connection\_ServerSocket().close();

fileDownload\_ServerSocket().close();

}

public void static void main(String[] args){

new server();

}

}

class connectClient extends Thread{

public connectClient(){

set object variables;

}

static{

new download\_ServerSocket();

}

public void run(){

passwordConnection();

instruction();

}

public void passwordConnection(){

while(true){

if(msg=userPassword)

break;

else if(mes=kernelPassword)

change directory; break;

else

send(wrong\_password\_info);

}

}

public void instruction(){

while(true){

receive(cmd);

if(msg="cmd fs")

send(file\_info);

else if(msg="cmd dl")

for(i<fileNum)

accept clientSocket; new DownloadFile().start();

else if(msg="cmd dlall")

for(i<totalFileNume)

accept clientSocket; new DownloadFile().start();

//break when IOException was thrown.

}

}

}

class DownloadFile extends Thread{

Public DownloadFile(){

set object variables;

}

public void run(){

sendFile();

}

public void sendFile(){

input Data from file;

send(Data);

}

}

Client:

public class Client{

public Client(){

connect();

process();

disconnect(); //clientSocket().close();

}

public void connect(){

new connect\_clientSocket();

while(true){

framePassword(); //display GUI of password connection

if(passwordNum=4)

throw new IOException();

if(msg=kernelPassword)

set\_serverDirectory; break;

if(msg=userPassword)

break;

}

}

public void process(){

frameLocation();//display GUI for input location, only at the first time

while(locationLock==true){}//stay inside until location setup finish

frameDisplay();

while(totalLock==true){}//stay inside until download finish

}

}

class frameDisplay extends JFrame{

downloadButton.addActionListener(new ActionListener{

send("cmd dl"); new ReceiveFile().start();});

downloadAllButton.addActionListener(new ActionListener{

send("cmd dlall"); new ReceiveFile().start();});

}

class ReceiveFile extends Thread{

public ReceiveFile(){

new download\_clientSocket();

}

public void run(){

crFile();

}

public crFile(){

create file;

receive(data);

output(data);

}

}