Indiana University Purdue University Indianapolis

Department of Computer and Information Science

CSCI 53700

Fall 2018

Assignment – 3

Devanapally Sai Krishna

2000276942

Contents

1.Introduction:

The aim for this assignment is to implement Remote Procedure calls to develop a simple distributed environment where we will be using multiple clients and single server. The server will be multi-threaded will a set of predefined functions like return host name, merge sort, encrypted echo, list files, add complex numbers. And the clients will be able to send concurrent requests.

2.Concepts and Design:

RPC stands for remote procedure calls. It's used to issue instructions from machine to another machine remotely. When program statements that use RPC framework are compiled into an executable program, a stub is included in the compiled code that acts as the representative of the remote procedure code. When the program is run and the procedure call is issued, the stub receives the request and forwards it to a client runtime program in the local computer.

The client runtime program has the knowledge of how to address the remote computer and server application and sends the message across the network that requests the remote procedure. Similarly, the server includes a runtime program and stub that interface with the remote procedure itself. Response-request protocols are returned the same way

For us to implement RPC we first develop a protocol which contains the list of functions data types, return types, parameters. This file is a .x file.

Next, we use rpcgen with –A and –C flags command to generate server and client here –A is used for automatic mode which enables auto threading and –C to generate stubs in C. After the generation of stubs, we can add out logic in server and client program.

Our Design:

add.x: Here we predefined our functions which we will be using further.

add_server.c: Here we implemented the following functions

1. merge sort: Here merge sort is implemented on two different lists of user input.
2. gethostname: Prints out IP address of the server.
3. printencryptedecho: Here the client gives an input of characters which will be send to server and the encrypted version of the text will be generated and sent to client. Here the encryption is done by subtracting –15 from the ASCII value of the character given as input.
4. listfiles: Server sends out the list of files on it to the client.
5. Addcomplex: Here the client gives the real part and complex parts of 2 complex numbers and the sum will be printed.

Add_client.c: All the client-side part which can implement here, All the functions to invoked from the server through remote procedure calls.

3.Pros and Cons:

1. Pros:

a. Server is always independent which gives us features of concurrency.

b. Always easy to implement threads by automatic flag which is easy to give control over concurrency.

c. The code re-writing and re-development is reduced.

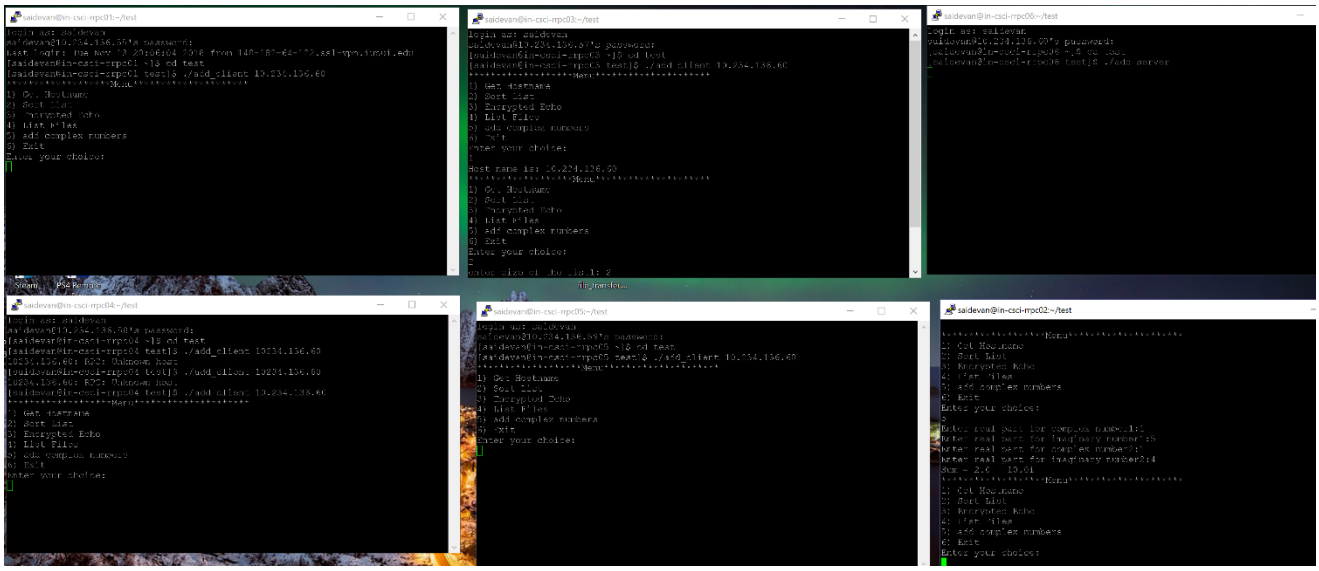
2. Cons:

a. Context switching causes an overhead in cost.

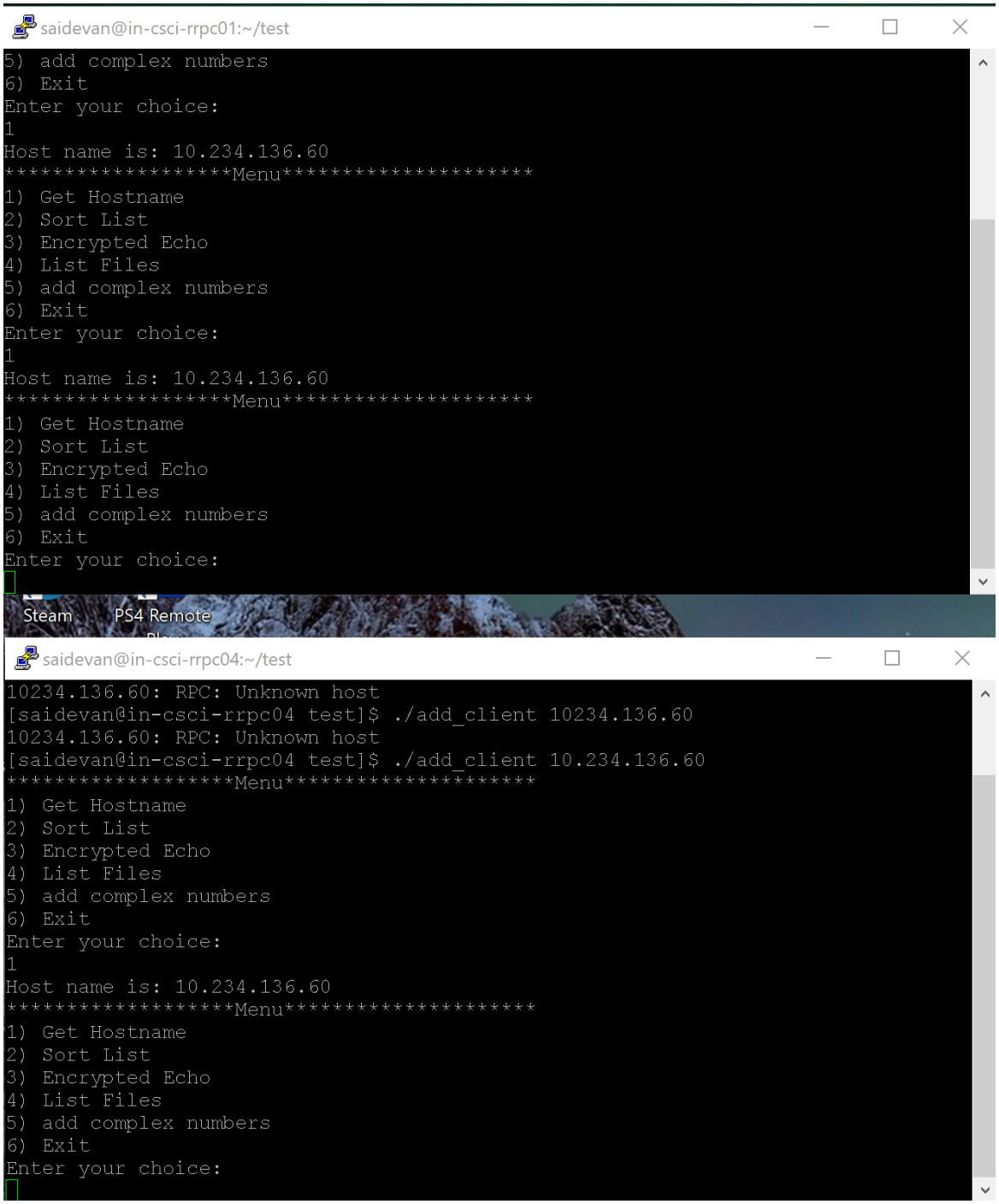
b. It is only feasible with TCP/IP protocol.

4.Results:

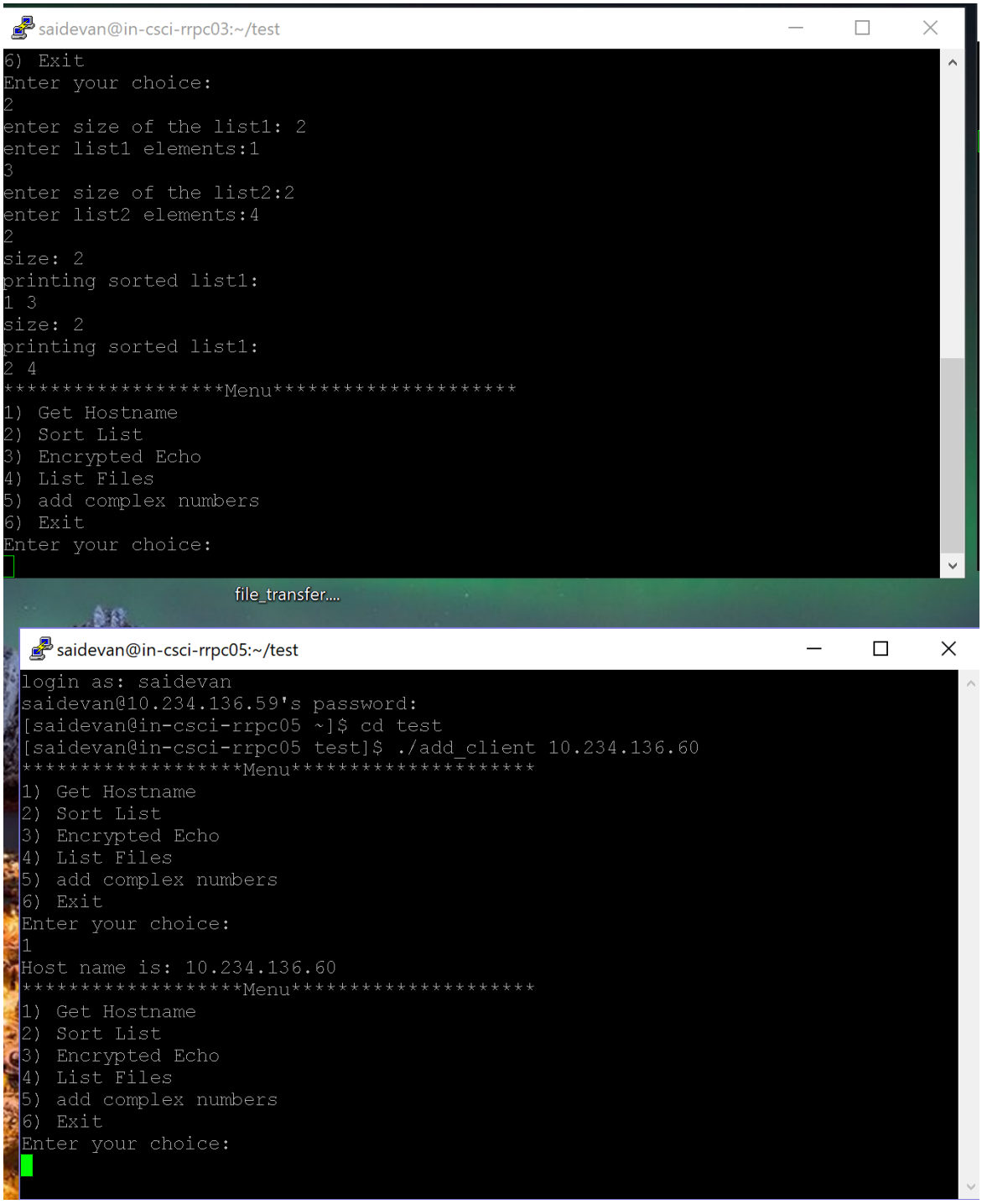
1. All 5 clients and 1 server.



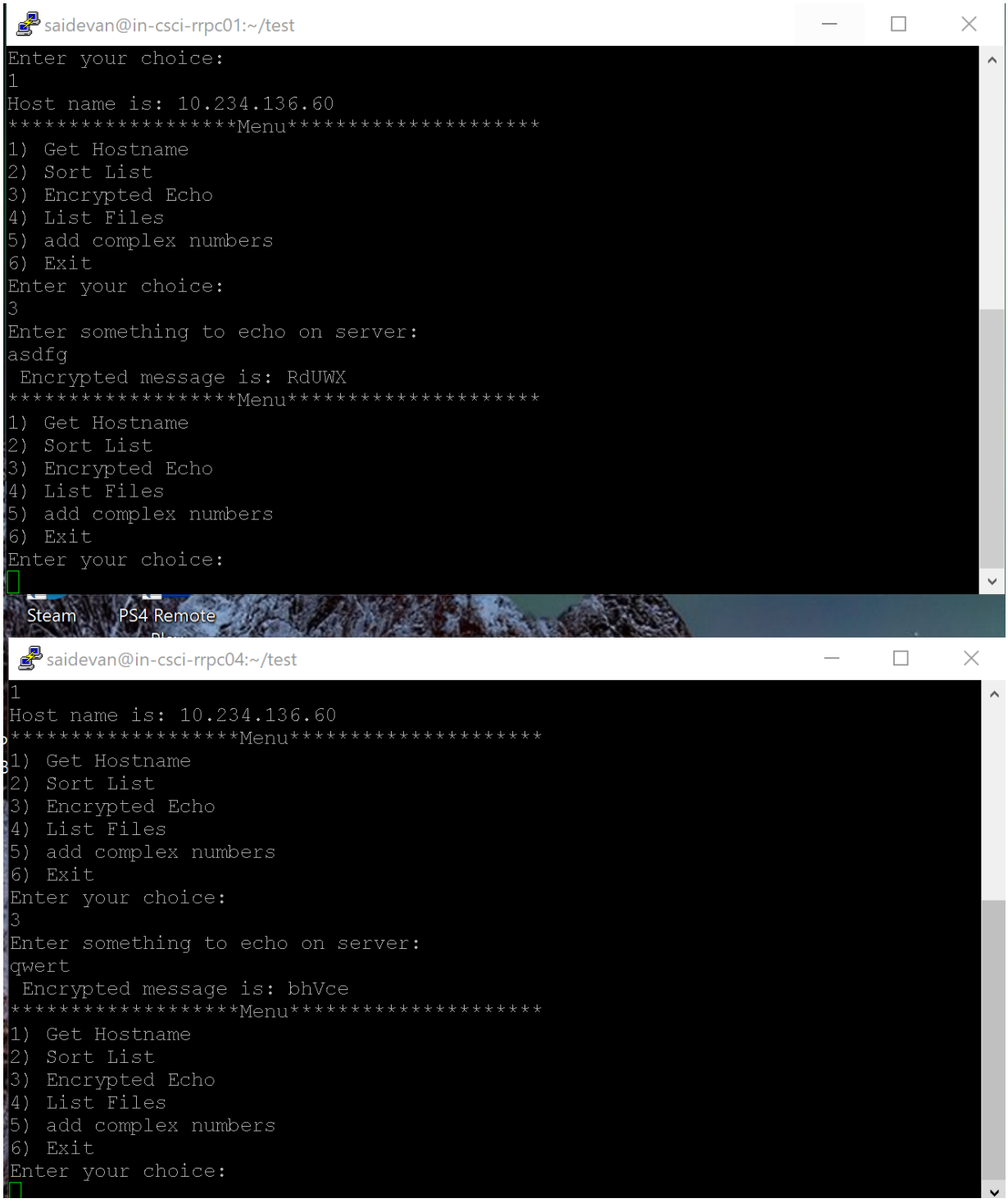
2. 2 clients implementing gethostname:



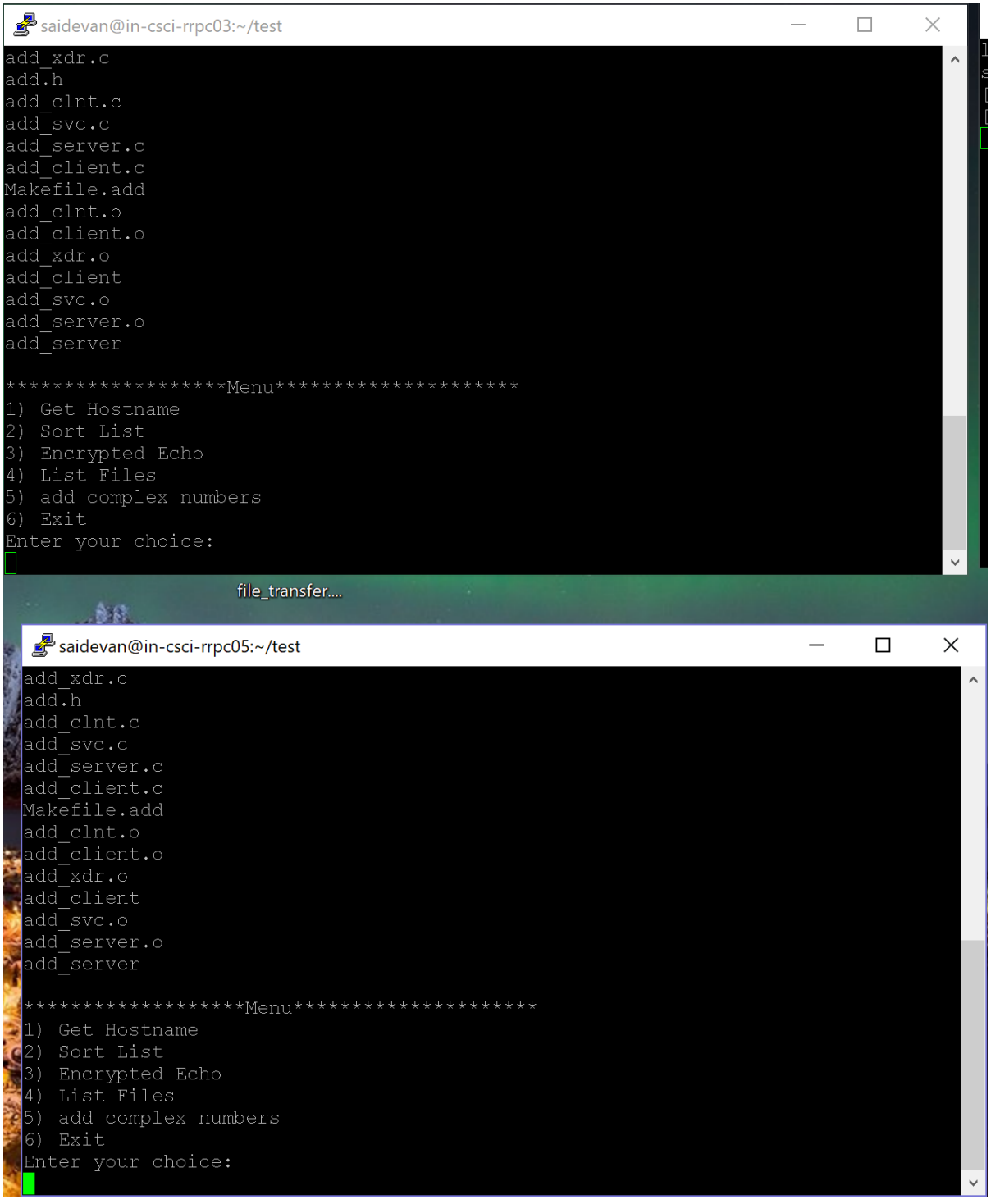
3. 2 clients one running merge sort and another runner displaying host name



4. 2 clients taking input and printing encrypted version of that text.



5. 2 clients printing out the list of files on the server.



6. 2 clients accepting 2 complex numbers and displaying their sum