2. Implement command line interface FTP using raw socket

What you have to do in this term project is to implement FTP protocol. All client interface is followed from the existing program “sftp”. For simplicity, your program will be tested in the Ubuntu system. Actually, your implementation is working exactly same as the “sftp”.

Experimental setup

1. In server side, we execute “python ftp_server.py” to construct server.
2. In client slide, we execute “python ftp_client.py” to execute client. Then, the program will immediately require authentication information.

Requirements

Correctness

- A client has to authenticate their identity.
- A client can traverse the server directory structure.
- A client can traverse the client directory structure.
- A file needs to be uploaded from client to server
- A file needs to be downloaded from server to client.
- Multiple file could be uploaded from client to server.
- Multiple file could be downloaded from server to client.
- A server can handle more than 100 clients at the same time.

Performance

- The upload speed will be similar to that of existing implementation.
- The download speed will be similar to that of existing implementation.

Term project will be rigorously tested so that you must check many edge cases. If you think some edge cases could be handled in a variety of ways, provide how to handle that case. Just ignoring it without proper reasoning will be considered as not implementing that requirement.

Submission

You have to submit a zip file with “**ftp_server.py**”, “**ftp_client.py**”, **pdf report**, **ppt** and **demo video**.

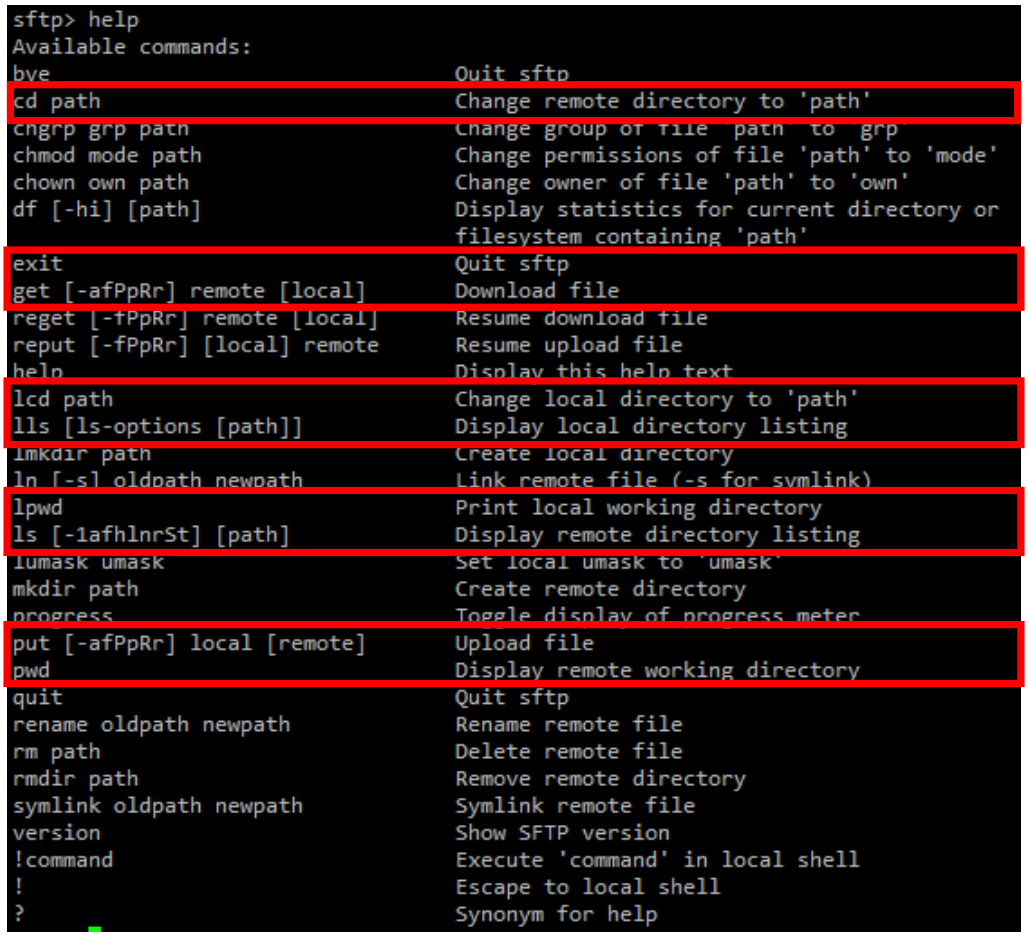

Fig. 2.a. Client Interface (the commands inside the red box are the requirement).

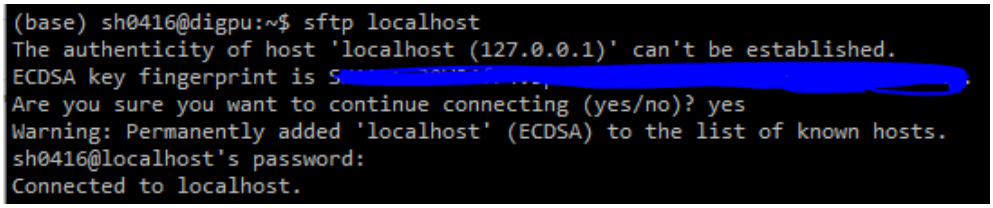

Fig. 2.b. Authentication.