Application-layer protocol

(for serving files over an Internet connection)

Candidate no: 52665

Server is running firs. Client is running second and connecting to the server.

- Upon connection server must verify if client is not forbidden.
- If not, server asks client for password.
- Client sends password to server
- If password is correct, client gains access to server, or is denied otherwise.
- If access granted client can do following:
    - list files in its current local directory
    - list files in servers remote current directory
    - put a file from the current directory of the local machine and store it in the current directory of the remote machine
    - get a file from the current directory of the remote machine and store it in the current directory of the local machine
    - exit - connection to a client is closed
    - if command is not recognised, server will return appropriate message

This behaviour is implemented by the following request commands:

connect_req
password
lls
rls
put <filename>
get <filename>
exit

Server response messages:

unauthorised
passwd_req
access_ok
rls_ok
put_ok
get_ok
unknown_command
exit_ok

Detailed description of the protocol behaviour

In response to the command (connect_req), server will reply (unauthorised) if client is in the forbidden.txt record; or (passwd_req) if not. Client will reply to (passwd_req) by sending (password) in raw string. Server will respond to (password) by (access_ok) if password is correct or (unauthorised) if password sent wrong three times. After gaining access, (lls) command is processed locally. To (rls), server will respond (rls_ok) and sends the serialised object with captured data to client. Client then prints the details into the console. The(put) command will trigger the file name checker and if file found read the local file and sends the serialised object to the server. Object is reconstructed and file saved to which server replies (put_ok). To (get) command, server responds by checking the file name, if found, reads the file, serialise the object and sends it to client with message (get_ok). Client then saves the file in its current local directory. To (exit) command, server responds by (exit_ok) and closes the connection. If the command is not recognised the server will respond with (unknown_command).

For protocol illustration please see the ‘protocol.jpg’ in project’s root directory.