Computer Networking Assignment #1 Report

Trenton MacNinch Michael Lin

October 13, 2024

1 Description of Code

1.1 Server

The `server.py` program is used with the following syntax:

`./server.py [port] [numclients]`

Where `[port]` is the port the user wishes the server to listen on and `[numclients]` is the number of potential clients the user wishes to allow to connect to the server. Once activated, the server does not require user input. It can be exited by pressing CTRL+C.

1.2 Client

The `client.py` program is used with the following syntax:

`./client.py [ip] [port]`

Where `[ip]` is the ip address the user wishes to connect to, on the port supplied as `[port]`.

1.2.1 Client Commands

- `name`: Gets current client name from server.
- `status`: Gets all cached client data.
- `list`: Lists files and subdirectories in given directory (defaults to server repository folder root).
- `download`: Downloads file from server.
- `exit`: Closes connection with server and exits client program.

2 Difficulties Faced

Informing the client of when to stop receiving the file was an issue, as our first attempts involved transmitting a specific piece of text that would tell the client program to finish the transmission. This was suboptimal, as the client program would write the transmission end signal to the file and could conceivably see the text it is looking for within the regular text of the file. The solution was to transmit the size of the file as the first 4 bytes that were sent and received after confirmation the file would be sent.

The client program would also erroneously believe it could connect to the server after the client limit was reached, and then wait forever once the first message was sent. The solution to this was to set the client to timeout an unresponsive connection, and then start each connection with a HANDSHAKE message to wait for a response. The current behaviour also allows the client to wait a time for the number of connected clients to decrease, in cases where the client limit has been reached.

3 Test Results

3.1 Server Results

```
dirt@thewagon  /Files/School/NetworkAssignment1/src $ ./server.py
Creating server on localhost:8080 and listening for 3 clients...
Allowing access to directory /home/dirt/Files/School/NetworkAssignment1/src/repo.
Exit program with CTRL+C.
Connection found from address 127.0.0.1:36754
Receieved message from 127.0.0.1: HANDSHAKE
Sending message to 127.0.0.1: HANDSHAKE ACK
Receieved message from 127.0.0.1: name
Sending message to 127.0.0.1: Client00
Receieved message from 127.0.0.1: status
Sending message to 127.0.0.1:
Open connections:
Client00: ('127.0.0.1', 36754) connected at 2024-10-13 20:35:16.902788

Connection found from address 127.0.0.1:47370
Receieved message from 127.0.0.1: HANDSHAKE
Sending message to 127.0.0.1: HANDSHAKE ACK
Receieved message from 127.0.0.1: status
Sending message to 127.0.0.1:
Open connections:
Client00: ('127.0.0.1', 36754) connected at 2024-10-13 20:35:16.902788
Client01: ('127.0.0.1', 47370) connected at 2024-10-13 20:35:28.874887

Receieved message from 127.0.0.1: exit
Sending message to 127.0.0.1: exit ACK
Receieved message from 127.0.0.1: status
Sending message to 127.0.0.1:
Open connections:
Client00: ('127.0.0.1', 36754) connected at 2024-10-13 20:35:16.902788
Client01: ('127.0.0.1', 47370) connected at 2024 Oct 13 20:35:28:874887, closed at 2024 Oct 13 20:35:38:686

Receieved message from 127.0.0.1: list
Sending message to 127.0.0.1: folder file1 file2 beemovie.txt
Receieved message from 127.0.0.1: list folder
Sending message to 127.0.0.1: file4 file3
Receieved message from 127.0.0.1: download beemovie.txt
Sending message to 127.0.0.1: Transmitting file...
Sending Client00 file beemovie.txt (86092 bytes)
File transmission complete.
Receieved message from 127.0.0.1: exit
Sending message to 127.0.0.1: exit ACK
```

3.2 Client00 Results

```
dirt@thewagon  /Files/School/NetworkAssignment1/src $ ./client.py
Connecting to server on localhost:8080...
Sending HANDSHAKE...
Received "HANDSHAKE ACK"
Input message: name
Recieved message: Client00
Input message: status
Recieved message:
Open connections:
Client00: ('127.0.0.1', 36754) connected at 2024-10-13 20:35:16.902788
```

```
Input message: status
Recieved message:
Open connections:
Client00: ('127.0.0.1', 36754) connected at 2024-10-13 20:35:16.902788
Client01: ('127.0.0.1', 47370) connected at 2024-10-13 20:35:28.874887

Input message: status
Recieved message:
Open connections:
Client00: ('127.0.0.1', 36754) connected at 2024-10-13 20:35:16.902788
Client01: ('127.0.0.1', 47370) connected at 2024 Oct 13 20:35:28:874887, closed at 2024 Oct 13 20:35:38:686

Input message: list
Recieved message: folder file1 file2 beemovie.txt
Input message: list folder
Recieved message: file4 file3
Input message: download beemovie.txt
Recieved message: Transmitting file...
Downloading file "beemovie.txt" from server...
File download complete
Input message: exit
Recieved message: exit ACK
```

3.3 Client01 Results

```
dirt@thewagon  /Files/School/NetworkAssignment1/src $ ./client.py
Connecting to server on localhost:8080...
Sending HANDSHAKE...
Received "HANDSHAKE ACK"
Input message: exit
Recieved message: exit ACK
```

4 Possible Improvements

- Make it more convenient to daemonize the server
- Better error handling as there are probably edge cases that have been missed
- Clearer command formatting, as some commands like `status` add a newline to the end of their output. This provides much easier to read output from the program, and would likely be effective in the same capacity when applied to other commands.
- Currently, there is no actual sanitization of filepaths, and so commands like `list` are capable of being addressed to directories and files like `/home` or `../../report/report.pdf`. This is likely a security issue on some level, though it is of questionable importance in a local server made for educational purposes.