Name: Stephen
Date: July 1 2016
Current Module: Networking In C
Project Name: FDR

Project Goals:

The goal of this project was to create a server type program. This program was supposed to handle incoming requests via netcat on 3 distinct ports. It would then convert the input into hex and send it back to the requester

Considerations:

- One consideration would be what language to use
- Another consideration would be to consider how to have each port open and listening for whatever request comes
- Also the program should handle closing down gracefully

Initial Design:

At the beginning of the project I started in python by creating the 3 functions needed to convert the input into hexadecimal. Another thing I considered was how the server would handle multiple connections which made me use threads. However with using threads I had to find a way to handle closing down gracefully which lead me to setting my threads to nonblocking.

Data Flow:

The program starts by getting the uid of the user to set up the 3 ports needed for each open port. Then I created each of my listening server threads which just handled input and sent the correct response to the requester.

Communication Protocol:

Netcat

Potential Pitfalls:

Some pitfalls at the beginning of the program would be incorrect of out of bounds input. Another option would be not handling closing down of the server gracefully. The final pitfall I came across was making sure each port was open and could answer back without hesitation.

Test Plan:

User Test:

1. Tested the program with no request information
2. Tested the program with requests for all three possibilities(valid input

3. Tested the program with requests for all three possibilities(invalid input
4. Tested typing quit into the input line
5. Tested keyboard interrupts

Automated Test:

- None

Test Cases:

1. Just ran the program with no request to make sure it could stay up without any errors
2. Sent requests of valid requests of all 3 types to my running server to see if I got the correct hexadecimal value back
3. Sent requesting of invalid input of all 3 types to the server to see if it sent back the invalid input text as it should
4. Typed ‘quit’ into the running input line of my file to see if it would gracefully close down as I wanted it to
5. Tested running the server and hitting ctrl + c to see if it would catch it correctly and gracefully close down as wanted

Expected Result:

For each of these tests I expected that for correct input the correct conversion to hexadecimal would happen and for invalid input the requester would get an error message. For the input line on the server I expected both quit and ctrl + c to join my threads and close gracefully.

Conclusion:

For the future work on this project I could clean up the code to make it less bulky. Also it would be nice if the server could accept more input to affect things such as error messages and such. The program could also have a gui made and could support more options for conversions.