Project 1: Pirate Sockets

Due Date: 9/16/2019 at 11:59:59pm.

Late Penalty: Program can be turned in up to 2 day late for a deduction of 10% per day late. After 2 days late, no late submissions will be accepted.

Now hold onto your swashbuckles, this won't be the "typical" type of pirate server found on the dark parts of the Internet. Instead, you'll be exploring the basics of network programming using the client-server model to build a pirate-speak server translator! For this assignment, you'll be using the programming server at student.cs.uni.edu and the group of port numbers assigned to you.

This project is to be done individually. Typical plagiarism-checking software will be deployed. If you would like a reminder of what is and isn't permissible when helping each other, please re-read the academic integrity part of the class syllabus.

Step 1

Pick your language: C, Python3, or Java. Download the TCP starter code, and modify the code to use port numbers assigned to you. Play with it on the student server to make sure you understand how it works. *Be sure to change the server port number to a number assigned to you! (If you accidently used another student's port number, why would that be bad..?)*

Step 2

Modify the server to send the message back to the client. The client should print out the message it received. After that, the client should once again prompt for new input to send to the server. This pattern should continue until the user closes the client. **(30 pts)**

Hint: *Don't overthink this step. Remember that the send method/function sends data on the socket, and the recv method/function receives data from a socket. These methods/functions can work on both the client and server on any socket…*

Step 3

If the server is started with a –r flag on the command line (added by you), modify the server to send the message back to the client in a "translated" pirate-speak! In particular, the server should replace the following words with the pirate equivalent: hello -> ahoy; hi -> yo-ho-ho; my -> me; friend -> bucko; sir -> mate; miss -> proud beauty; stranger -> scurvy dog; officer -> foul blaggart; where -> whar; is -> be; the ->thar; you -> ye; water -> rum; nearby -> broadside; restroom -> head; restaurant -> galley; hotel -> fleabag inn. To make things simpler, don't worry about capital letters at the beginning of your sentences. (In other words, everything can remain lowercase.) **(10 pts)**

Step 4

If the server is started with a –n flag ("n" for "normal"), then any pirate words received by the server should be modified to be sent back in normal speak. For example, ahoy -> hello. **(10 pts)**

If the user closes the client (by hitting CTRL+C), the server should print "Where did ye go?" **(5 pts)**

Step 5

Also, create a separate text document called *whatsgoingon.txt* that tells me what the following calls do in Python and why they have to go in the client, server, or both to implement step 1. (If you used a programming language other than Python, also tell me how your chosen language implements/combines the same system calls into similar functionality.) **(10 pts)**

- socket
- bind
- listen
- accept
- connect
- send
- recv

Step 6

Create a README text file which contains the following **(10 pts)**:

- Your name
- Instructions for compiling/running your program
- Any helpful resources you found outside of class
- Any challenges you encountered along the way
- Anything that doesn't work correctly.

Other Notes

The programming submission will receive 0 points if the code does not compile (if applicable) and run on the Linux server and/or the code contains less than three useful comments. In addition, it must also contain a header comment containing at least your name and the name of the program for any points.

This is important: if you use other resources than the class resources for your assignment, you must cite them in a comment block above the affected code! Cite them by providing the URL of the resource and a short description of how the resource helped you. *I will be checking for this!* I also reserve the right to ask you to explain how a particular cited piece of code works, and I will probably do this. In other words, the only code in your submission should be code that you understand 100%!

If you save your whatsgoingon and README files as a format other than .txt, I may dock points for the annoyance of having to context switch out of the Linux server to read them.

Turning it in

Turn in your assignment by zipping an archive of the files and submitting to the appropriate project submission space on eLearning. Be sure that the files are at the root of the archive and not in any folders. Include your name in the text files and in the comments at the top of any programming files. Do not zip and send any executable files (for example, the C or Java executables). I want your source files, and I will compile/run them myself.

FAQ

[Q1]Will the client work from my home machine if I run the server on the class Linux machine (or the other way around)?

[A1]It might! However, your firewall or a firewall in between you and the class server may be getting in the way. As long as both the client and server work together on the class Linux machine, the problem probably isn't with you.

[Q2]Sometimes after I kill the server with (Ctrl+c), I can't restart the server program right away and I get a strange error message. Why?

[A2]This is because the socket was not closed cleanly, and the system has to wait a timeout period before the socket gets cleaned up so you can use it again. This could be fixed by catching the signal Ctrl+c with a signal handler, but that's beyond the scope of this assignment. If you wait 30 seconds to 1 minute, you should be able to restart your server. The official TCP timeout value is around 4 minutes, so in the worst case, that port number may not be available again for 4 minutes.

[Q3]What is the longest string length I have to worry about?

[A3]I will never send an *untranslated* message over 140 characters, because then I can't use it on Twitter. However, the translated message the server sends back could be over 140 characters. You might want to set your maximum string lengths to something higher to be safe.

[Q4]I still don't understand how to do this project. Help!

[A4]If you weren't in class when the project was introduced, did you watch the class screencasts for those days? Please do that first before asking for help.

* This part of the project has been adapted from a simpler pirate assignment by Dave Musicant at Carleton College: (http://www.cs.carleton.edu/faculty/dmusican/cs111s10/pirate.html).