Homework Assignment 4

This is an enhancement to the Client and Server from the previous homework assignments. Now that you can send data between a client and a server, you will modify your client and server programs to do a simulated TCP handshake.

Write a simple TCP client and a separate TCP server application in C. (You should already have that)

As soon as the client connects, the client will send 20 bytes of the TCP header simulating the 3-way handshake, the server should respond to it, and finally the client completes the handshake. You will be crafting the TCP header with binary data.

- Requirements for both client and server program
  - You must use the POSIX socket functions.
  - Do not implement the server handling multiple requests at the same time (multi-threading).
  - Implement reasonable output messages (to the console) that show BOTH the raw bytes (as a hex dump) of the header plus each of the required header fields in human readable format (decimals, flag names, etc.).
  - Your client and server should accept a single command line argument for the port number. You can assume that the client and server are running on the same machine.
- Create a `Makefile` to compile your program
  - The Makefile should compile your program into an executable using the "make" command.
  - A separate Makefile should be made to compile your client and server programs.
  - There should also be a clean function that removes any object or executable files using the command "make clean".
  - To get full credit, your code must compile without any warnings or errors.
- Required header fields:
  - Source TCP port number – Use a C function call to get this
  - Destination TCP port number – The real port you are connecting to
  - Sequence number – Create a random Initial Sequence Number
  - Acknowledgement Number – You should use the appropriate value
  - TCP data offset – Make it all zeroes
  - Reserved data – Make it all zeroes
  - Control flags – Flags should be set correctly for the 3-way handshake
  - Window size – Use a reasonable default value e.g., 17520 bytes
  - TCP checksum – Make it all ffff (hex)
  - Urgent pointer – Make it all zeroes
- Test your client and server programs together and save the output into a file named `output.txt`
- Write a summary of the status of your program in a `status.txt` file.

    - Does your program compile/run?
    - What features work/don't work?
    - Are there any known bugs in your program?
    - Any other relevant comments.
- Resources
    - You may use appropriate online resources or the textbook.
    - If you are using code from a resource, create a comment with a link or instructions on how to view the resource in a comment.
    - Do not use homework solution websites.
    - You are not allowed to collaborate with other classmates on this assignment.
    - If you have any questions on whether a source is acceptable to use, contact the TA.
- Expected files
    - All files should be in a TAR archive using this format "ics451_hw4_<uhusername>.tar". My file that I turn in would be named ics451_hw4_chadmmm.tar.
    - To create a tar file, use the command "tar cvf ics451_hw4_<uhusername>.tar ics451_hw4_<uhusername>".
    - Inside of the TAR archive, there should be a folder named "ics451_hw4_<uhusername>".
    - Follow this directory structure (inside of the tar archive):

```
ics451_hw4_chadmmm/
├── client
│   ├── Makefile
│   ├── client.c
│   └── output_client.txt
├── server
│   ├── Makefile
│   ├── output_server.txt
│   └── server.c
└── status.txt
2 directories, 7 files
```

- Do not make assumptions about the program specification. If any part of the specification is unclear, contact the TA.
- Deadline
    - The deadline is set on the Laulima assignment page.
    - Late work is accepted up to 2 days late.
        - 10% deduction for 1 day late.
        - 25% deduction for 2 days late.
        - 2+ days late will receive a 0.