CSCI 440: Computer Networks
Homework 2
Fall 2017

Summary:

We implement a file transfer client and server with TCP sockets in Python or Java or C/C++.

Objectives:

- Practice TCP socket programming
- Learn about the TCP protocol

Background

- Section 3.4-3.5 in Kurose and Ross (Computer Networks) details the basics of the TCP protocol.
- Section 2.7 in Kurose and Ross (Computer Networks) demonstrates how to use Python for socket programming, with 2.7.2 highlighting the TCP protocol you will use.

Collaboration:

You may complete this assignment in pairs.

Simple File Transfer Application

You shall write a simple client/server file transfer application using any programming language you are comfortable with (i.e., Java, Python, C++). Furthermore, the client/server application must exchange data using TCP sockets.

The functional requirements for client application are:

- Open an ASCII text file that resides on the file-system on the client machine. The
- Determine the number of <BYTES> in text file.
- Make a TCP socket connection to server at <IP> and <PORT>
- Read the <TEXT FILE> from file system and then write to the socket. When the last text byte is encountered, the program will automatically append the character sequence "\r\n\r\n" that indicates end-of-file (EOF) and write to the socket.

- Waits for the server to response, and then reads from the socket. The server will respond with the amount of <BYTES> it received.
- Close down the TCP socket.

To show the progress of the file transfer, the client program must display the following messages in a terminal.

- Successfully opened socket to server at <IP:PORT>
- Reading <TEXT FILE> from client file-system.
- Total number of byte in <TEXT FILE> = <BYTES>
- Writing <TEXT FILE> to socket
- Reading socket, successfully transferred <BYTES> bytes of data to server
- Closing down socket … good bye :)

The functional requirements for the server application are:

- Starts a process/thread that listens for client socket connection requests on <PORT>
- Shall read data from the socket and save to an ASCII text file named "transfer.dat" located on the server file-system.
- Shall record the number of <BYTES> received until the EOF character sequence is encountered.
- When the EOF sequence is encountered, the server shall write to the socket the total number of <BYTES> read (not including EOF character sequence).

To show the progress of the file transfer, the server program must display the following messages in a terminal.

- Starting service listening on <PORT>
- Socket successfully opened from client, beginning file transfer.
- File transfer complete, total number of bytes read = <BYTES>
- Writing transfer.dat to server file-system
- Sending file transfer complete message to client.
- Closing down socket … good bye :)

Lastly, please do not copy paste existing code found on the web, your assignment will not be graded. I'm fully aware that there are plenty of solutions out on the web, and this is not a "search the web" exercise.

Testing

For testing purposes, please create 5 different ASCII text files that each has a different size (i.e. the file size is measured in bytes, where 1 character is 1 byte, 50 KB means the file has roughly 50,000 characters that may include new-line etc.). For instance, generate 5 ASCII text files with the following sizes: 1 MB, 500 KB, 100 KB, 10 KB, and < 1KB.

Your test files must be included when you submit your solution.

Adopted from: Dr. Brent Munsell's Fall 2015 CSCI 440 class.