CS513/ECE506 Computer Networks Spring 2013

Program 0 {January 4, 2013} 4 points (with 4 extra credit points)

TCP Echo Server and Client
Due: Thursday, January 24, 2013 at 11:59 p.m.

This assignment provides all students an opportunity to work through the standard procedures for developing, executing and testing C socket programs on a Linux system. The task is to utilize any version of the TCP Echo Client and TCP Echo Server that you prefer. It can be essentially a **copy** of the programs in the D&C textbook or any other textbook (your comments must indicate the source). The procedures are:

1. Write the client program and the server program on a CCC Linux machine using your favorite editor.
2. Create and use a ***make*** file to build, test and debug both programs.
3. Create a **README** plain text file to assist in the grading of program 0.
4. Tar the two source programs, the make file and the **README** file into a single tar file for submission.
5. Use the Unix version of ***turnin*** to submit the tarred file for grading.

The **basic** assignment is to write both a TCP Echo Client and a TCP Echo Server in C or C++ using Unix socket commands. The client and server implement the ***echo*** protocol while running on **different** CCC Linux machines and communicating with each other using TCP.

The Echo Client

The **basic** Echo client connects to the Echo server and sends its data to the server. The data that the client sends is a string provided as the second client command-line argument. The **basic** Echo client prints the single string of data sent back by the Echo server.

The form of the command line and the print line for the **basic** Echo client are:

my_EchoClient 169.1.2.3 "echo this string!!"
Client received: echo this string!!

169.1.2.3 :: the first argument is the dotted-quad notation IP address of the Echo server.
my **::** should be replaced by the initials of the program author **.**

The Echo client accepts strings of length 1 to 32 bytes inclusive and prints out an error message for any out-of-range input string.

The Echo Server

After connecting to the **basic** Echo client, the **basic** Echo server (which is started first) simply echoes the string it receives back to the client, disconnects and terminates.

The Enhanced Echo Client

This assignment includes an optional **enhanced** Echo client and Echo server. To receive the extra credit, the student must turn in both the **basic** and **enhanced** Echo clients and servers **on time**. A student can receive up to four extra credit points.

my_EnhancedEchoClient ServerMachine "string 1" "string 2" "string 3" "string 4" "string 5"
EnhancedClient received: string 1
EnhancedClient received: string 4
EnhancedClient received: string 3
EnhancedClient received: string 4
EnhancedClient received: string 5
EnhancedClient: done

[1 pt] The **enhanced** Echo client receives the name of the computer where the server is running (e.g., CCCWORK2) as its first command-line argument and uses a Linux system call to convert the server name to the associated server IP address.

[2.5 pts] The other enhancement for both the Echo client and the Echo server permits the number of strings to be echoed to vary from one to five. The **enhanced** Echo client sends a series of data strings (one per TCP packet) to the enhanced Echo server. The enhanced Echo client prints out each data string sent by the enhanced Echo server. Once the **enhanced** Echo client has sent and printed all the echoed strings, it sends one additional TCP packet containing the two ASCII bytes **DLE ETX**. Once the enhanced Echo client receives the echoed **DLE ETX** bytes, it prints out a done message, disconnects and terminates.

[0.5 pts] The enhanced Echo client accepts strings of length 1 to 12 bytes inclusive and prints out an error message for any out-of-range input string. Any out-of-range string is not transmitted, echoed or printed. The Echo client then processes the next string (if any).

The Enhanced Echo Server

To earn the extra credit points, the **enhanced** Echo server runs in a loop accepting up to five strings from the enhanced Echo client as five TCP packets. When the **enhanced** Echo server receives a packet containing the two ASCII bytes **DLE ETX,** it echoes this packet back to the client and prints out a message of the form:

EnhancedServer echoed **n** strings.

where **n** is the number of strings echoed by the server prior to receiving the **DLE ETX**.

The enhanced Echo server then disconnects and terminates.

What to turn in for Program 0

Turn in Program 0 using the ***turnin*** program on the CCC machines. You must turn in a tarred file that includes: your source code files, a **make** file, a **README** file and a sample output file from a test run. The **make** files should include the ability to cleanup leftover output and intermediate files between compile and execution cycles. The **README** file provides any information to help the testing and grading of your Echo Client and Echo Server on CCC machines. The README file must indicate if you implemented the **enhanced** Echo Client and Server. You tarred file should include separate source files and make files for both the **basic** and the **enhanced** Echo Client and Server if you are trying to earn the extra credit points.