Sample Program

Requirements:

1) Program must be written in C and must compile using gcc
2) The application must be compiled to run on (x86/x86_64) Linux
3) All source files, binaries, documentation, and makefiles are expected to be delivered (tar archive, zip file, etc.)
    a. If using Netcat client, that program does *not* need to be bundled with deliverables (more information below).

Task Description:

Create a simple client/server program. The server should be written in C and accept client connections, receive XML data on that connection, and display the contents of the XML message. The client can simply be an external program such as Netcat or your own program written in C that initiates a connection with the server, sends XML data, and displays the server's response.

1. Socket Server Monitor:
    a. Components:
        i. Command line options:
            1. TCP/IP Address
                a. Default to 127.0.0.1
            2. Port Number
                a. Default to 5000
        ii. Socket Listener
        iii. XML Validator/Parser
    b. Functionality/Requirements:
        i. Open socket and listen for incoming data.
        ii. Receive data through this socket and determine if it is valid XML.
            1. If it is valid XML, pass to a work queue that "processes" the command.
                a. If not valid XML, display "Unknown Command" in the console.
            2. Parse the "Command" field out of the XML and display it to the console along with a receive date.
            3. Send response back to the originating socket.
2. Client using Netcat (http://netcat.sourceforge.net/):
    a. Functionality:
        i. Write a shell script to call the netcat program to send XML messages (from a text file) to the socket server monitor.
        ii. *OR* Write a client in C that can send XML messages (from a text file) to the socket server monitor.

A valid XML message will be in the following format (e.g. xml-message.txt):

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<Message>
  <Command>Print</Command>
  <Data>
     <Row>
       <Description>"Name"</Description>
       <Value>"Mr. Joe Chase"</Value>
```

```
        </Row>
        <Row>
          <Description>"Address"</Description>
          <Value>"123 Anywhere Lane"</Value>
        </Row>
    </Data>
</Message>
```

Extra Credit:

a. Support multiple simultaneous clients and requests (multi-threading support).
b. Implement mutexes within shared data elements.
c. Use C-based design patterns where appropriate.

Resources/Effort:
This is a test of your programming skill. The requirements are intentionally loose to allow you to flex your programming muscle. Using a “team” approach or help from others isn't acceptable. On the other hand, using a book or web resources is acceptable (and *highly* recommended). Open-source external libraries (e.g. libxml2 or expat for validating/parsing XML) are also permitted for use if they are included in the deliverables.

Your source code, complete or not, working or not, will be reviewed by our programmers. If your program is not complete, a partial, but working program will be scored better than an almost complete but not functioning program.

Additionally, spend some time giving a description of your implementation, what design and architectural trade-offs you may have made, and/or any other documentation you feel would clarify your solution.