Computer Networks

Practical Examination

Sample Questions

Assume that all numbers are unsigned 32-bit integers Assume no computation results into an overflow and/or underflow Properly organize your code and put comments as applicable

Part A: Socket Programming

Consider a client-server system that can send numbers back and forth. The client iteratively takes a number from the user and sends it to the server. Upon receiving a number the server does some computation on it, and then sends back another number as a reply. The client then prints the reply, and this process starts over. The process goes on until user enters a special value, say 0, as the input.

P1. **Building a Socket Program for Counting Even Numbers**

i. Write a client program that takes the integer from the user and sends them one by one to a server using Socket APIs. Furthermore, after each number sent to the server the client awaits for a reply, and then prints that reply value. If the input value is 0, the client closes the connection. [4]

ii. Write a corresponding server program that can receive the number and sends another as a reply. The server maintains a running counter for the number of evens seen so far. The server checks whether the received number is even or not, and updates the counter value accordingly. The server sends the counter value as a reply. If the received value is 0, the server closes the connection. [Try to be as efficient as possible for even/odd testing.] [6]

iii. Make server program able to simultaneously communicate with multiple clients with `poll()` API. Note that the server maintains only a single counter for the total number of evens seen so far considering all the clients. Use at least three clients for this experiment. [10]

Part B: Remote Procedure Call

Consider a client-server system that can perform some computation on a remote machine. The client takes some input from the user and invokes a remote procedure at the server with. The server executes the invoked procedure on the given parameters, and then returns a reply. The client then prints the return value.

P1. **Building an RPC Program for Vector Scalar Multiplication**

i. Suppose we need to perform a vector scalar multiplication operation using the Remote Procedure Call APIs. Therefore, we need to send a vector along with a scalar value to the remote procedure and get back another vector after the multiplication has been done. Write the specification file (.x file) for this purpose. Assume that the vectors are stored as fixed sized (max size = 20) integer arrays, and the scalar multiplier to be a positive integer. [Note that a vector can be of smaller size also, say 5.] [5]

ii. Use `rpcgen` to generate the required files. Modify the generated `makefile` if required. [2]

iii. Modify the client and server program so that the client first takes size of the vector n, followed by elements of the two vectors $A =< a_1, a_2, \ldots, a_n >$ and a positive integer k from the user. Then the client invokes the remote procedure with this A and k. The remote procedure does the scalar multiplication and returns another vector $B =< b_1, b_2, \ldots, b_n >$ where $b_i = a_i \times k$. The client also displays the returned vector. [6+7]