ECE 456 – Computer Networks

Programming Assignment # 4 : UDP Socket Programming

Aim

A socket is the most fundamental structure used for computer networking. Sockets allow applications running on same or different machines to communicate with each other. Sockets are created and used with a set of programming requests or "function calls" sometimes called the sockets application programming interface (APIs). The goal of this assignment is to understand the concepts of socket programming using UDP and to implement UDP based client-server systems.

Description

Implement and demonstrate a message exchange system that works as follows.
- A client contacts the server by sending a message of up 250 characters;
- At a given time the server displays the five most recent messages it received from clients, together with the arrival time and the sender's IP address for each message.
- Upon receiving a message from a client, the server responds by sending to the client the last five messages that it had received from clients (i.e., those being currently displayed).
- Upon receiving the response from the server, the client displays the information received from the server.

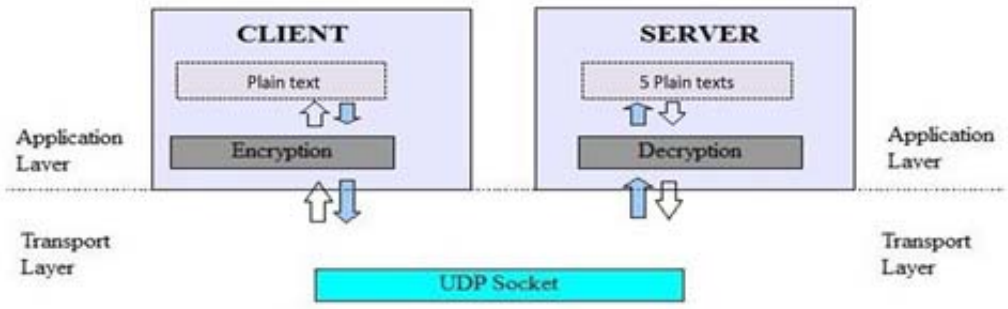

Figure 1

As shown in Figure 1, you need to create simple sockets to interact between the client and the server. The server is capable of handling requests from multiple clients. Each client is capable of accepting keyboard input, and also repeating the task for different messages.

Procedure

You need to implement 2 programs (or modules), the client and the server

Client:

1. Accept the server IP address, port and an input file from the user.
2. Accept the keyboard input and send the message to the server.
3. Wait for the response, and display the response. (How would you handle lost messages?)

Server:

1. Initialize (open socket, etc.)
2. Handle incoming client requests by sending the last five received messages together with time stamp and IP address information. Update the display.
3. **Server should be able to handle requests from different/multiple clients.**

Special notes - using the ENGR domain Linux/Solaris workstations

- **If you use a Linux workstation, note that network ports are blocked in those. For these assignments you may use Ports *13570 - 13579* in Linux workstations *linux1 - linux8.***

Suggestions

- Please provide detailed comments for your code even for functions / modules (http://en.wikipedia.org/wiki/Comment_(computer_programming))
- Take care of error checks at regular intervals.
- [C/C++] Always check for errors when making a system call. [Java] Handle all critical exceptions explicitly.
- Modularize your code such that you can reuse the modules.
- Include a readme file for details regarding compilation, code, etc

Useful Links

Network Programming [C] : http://beej.us/guide/bgnet/output/html/singlepage/bgnet.html

Network Programming [Java] : http://www.ashishmyles.com/tutorials/tcpchat/

Network Programming [Perl] : http://www.tutorialspoint.com/perl/perl_socket.htm

NOTES

- You should work individually. Each individual is responsible for the entire assignment, and should be able to demonstrate and explain the different aspects of the program.
- Submit your code and the makefile (if any) on Canvas in one Zip file.
- Include a complete readme.txt file with instructions for compilation.