Socket Programming

Goal: Create your own client server application using socket programming.

Extending Echo Client/Server for network analysis [Part 1]

1. Create your own UDP echo client and server application to measure round trip time between client and server (similar to "ping" command)

   The client should create a UDP socket and send echo packets to server at a given interval, number of echo messages, and given packet size (use command line arguments). On reception of the packet, server should send the packet back to the client. The client on reception of the packet should calculate and display the round-trip time. To calculate the round-trip time, you can have the timestamp in the packet or/and use some unique identifier in the packet. You should also calculate and print the loss percentage at the end.

2. Create an iperf like application using the above developed echo client and server program. Reduce the interval between two consecutive UDP echo packets generated by client to increase the number of echo packets sent from client for a given packet size. Calculate the throughput and average delay observed every one second. Plot the observed throughput and average delay vs time (1 second interval).

Extend Echo Client/Server and create your own client server application [Part 2]

Add any two features to Echo Client/Server and demonstrate them. In the report, you must describe the new features with their benefit. **Significance of the features will impact the marks given.**

Part 3: Let two computers talk with each other. Select one mode.

– **EASY Mode:** 1 server and 1 client.
Extend server program to accept typing on server itself. No echo.

– **NORMAL Mode:** 1 server and 2 clients.
Extend server program to forward a message from client 1 to client 2.
You may skip management of client ID.

– **HARD Mode:** 1 server and N clients.
Say N clients may connect to the server. Client 1 wants to talk with client m.
How do you manage multiple clients and select the one you want to talk to?

Making Echo Client/Server "protocol Independent" [Part 4]

Revise echo client and server to be protocol independent
(Support both IPv4 and IPv6).
Hint 1: sockaddr is too small for sockaddr_in6. sockaddr_storage has enough size to support both sockaddr_in and sockaddr_in6. (You will see this in server-side program.)
Hint 2: integrate getaddrinfo to avoid typing IPv6 address on your CLI
Hint 3: you may use hostname (IPv4: "localhost", IPv6: "ip6-localhost" address to develop / demonstrate the software on ubuntu. They're written in "/etc/hosts".