Assignment 3: Client-Server Applications using Socket Programming

Deadline: 20th November 2023, 11:59 PM

Goal of the Assignment: Get familiar with socket programming and learn to create your own client-server applications using socket programming.

Part 1: Implement SSH using socket programming. [20 Points]

SSH, which stands for Secure Shell, is a network communication protocol facilitating encrypted communication and data sharing between two computers. The supported commands are as follows

a. ls

b. cd

c. echo

d. mkdir

(You can provide support for other commands also)

The client-side should look like a terminal where you should be able to ssh to the server. The server should maintain two things one is the user-id: password, and the other is the last login information.

client$ ssh user@remoteserver

enter pass: user

authenticate the password at the server side, the server should reply with a yes or no message to the client as a response to authenticate the user, if authentication is successful, a welcome message should be sent by the server and will be displayed at the client side.

welcome message:

Welcome to devicename

system information
memory usage: 14%
processes: 141
ipv4 address: 192.168.13.17
last login: wed 08-11-23

user@remoteserver$
user@remoteserver$ ls
Report.pdf setup.txt
user@remoteserver$ echo hello
hello
user@remoteserver$ cd dir_name

any command that is typed on the client console should be sent to the server where it should be executed, and the output should be returned to the client side where it will be displayed.

Part 2: Build a DNS resolver application using socket programming. [15 Points]

A DNS resolver translates domain names to IP addresses, enabling devices to connect on the internet by converting human-readable names into numerical addresses.
A DNS request should be made at the client side, where the user will enter the domain name, this request should be forwarded to the server side, where the server maintains a text file of the domain name to IP address mapping the server should respond with the IP-address of the corresponding domain name. The text file should have a minimum of 5 entries of the domain name to IP address mapping.

The DNS request should have the following parameters
Domain Name (QNAME)
Transaction ID (ID)

The DNS response from the server should have the following parameters

Transaction ID (ID)

Answer Section (ANSWER)

Finally, the IP address should be displayed along with the domain name on the client side.

Part 3: Making Echo Client/Server "Protocol Independent" [15 Points]

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 the 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".

Part 4: Create your own client server application [15 Points]

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

Instructions for Implementation:

- You may choose any programming language (C, JAVA, Python, etc.)
- The software must be based on Socket Programming.
- Wrappers of API must not be used (messaging etc). Use send/recv or read/write using TCP/UDP socket.
- Keep a record of Reference.

Deliverables in a tarball on GC:

1. A report detailing your implementation and the results. The core idea of your answer to each question. Better visibility, like screenshots of the application, will be appreciated. One single report file (<your roll no>_<Name>_<Assignment3>.pdf for all four parts.
2. Screenshots showing the working of your code.
3. Create a separate folder for each part and include the source code and README as a separate file for each part so that TAs and instructor can compile source code and execute the binary anytime.
4. Submit all files in a single zip file named as <your roll no>_<Name>_<Assignment3>.zip

   Note: A plagiarism check will be done on your submitted code/report. We may call you for a presentation. In the presentation, you are expected to explain and demonstrate the applications you built.