Networked Shell

Typically if you want to execute commands on a remote machine, you will SSH in. Lets create a service which lets us easily execute commands on a computer over the network. We have already accomplished the networking portion, but there is still a lot to be desired.

Objective

Database integration to abstract logging logic

Your original shell used raw file IO for logging. Since then, we have implemented our own databases. We can use the database to write logs for us.

We have 2 options for how to add the database

1. As a library - We can import the database functions into our code. No IPC is used.
2. As a service - We make IPC calls (most likely socket) to connect to the database and tell it to perform operations. We can abstract the client side logic out as a library

Let's do the second option.

1. Make the database networked
2. Create a database library to be used by the shell, which abstracts out the connection logic. Use the sock_utils.c/h files for this.
    a. Recommended library functions are:
    Database database_connect(int port)
    int database_execute(Database database, char *command)
    database_close(Database database)
3. Replace logging logic in the original shell with database library calls.

See remote math examples for inspiration.

There are going to be 2 sections to the database, the commands you can execute on the database as functions, and the IO section which listens to the network and executes commands.

Note: Since your database does not protect against issues arising from writing to the same file from multiple threads (race conditions) - you should only allow for 1 active connection at a time.

TODO Later Features

- Software Visibility Features (Alerts, Metrics, Logs (Done))
- Client Side Application UI (Website, Desktop, Mobile App, etc)

- Multiple users and authentication
- Deployment Script
- Scalability and reliability features for scaling to millions of users