README: Server and Client Protocol

Overview

This document describes the operation and protocol of a server and client system implemented in Python. The system facilitates user authentication and processing of commands over a TCP connection. The server authenticates users and supports various operations such as arithmetic calculations, finding prime factors, and determining the maximum value among a list of numbers. The client connects to the server, authenticates, and sends commands for processing.

Server

Functionality

The server provides:

1. **User Authentication**: Reads a list of valid usernames and passwords from a file and validates incoming clients.
2. **Command Execution**:
   - **Arithmetic Operations** (calculate: X Y Z): Performs addition, subtraction, multiplication, division, or exponentiation.
   - **Finding Maximum** (max:(x1 x2 x3...)): Returns the maximum value from a list of integers.
   - **Prime Factorization** (factors: x): Calculates and returns the prime factors of a number.
   - **Quit Command** (quit): Terminates the client's connection.

Usage

To start the server:

```
python server.py <users_file> [port]
```

- <users_file>: Path to the file containing usernames and passwords (one username password pair per line).
- [port]: Optional. Default is 1337.

Protocol

1. The server listens for incoming connections.
2. Upon connection:
   - Authenticates the client using username and password.
   - Sends a welcome message upon successful login.
3. Processes commands received from the client, returning appropriate responses.

Error Handling

- Invalid commands result in error messages.
- Malformed user files (e.g., lines without exactly two fields) are ignored with a warning.

Client

Functionality

The client connects to the server, authenticates, and sends commands for processing. Commands and their responses are displayed to the user.

Usage

To start the client:

```
python client.py [hostname [port]]
```

- [hostname]: Optional. Default is localhost.
- [port]: Optional. Default is 1337.

Authentication

The client prompts the user for:

1. **Username**: Sent to the server.
2. **Password**: Sent to the server. If authentication fails, the client retries.

Command Protocol

After authentication, the client can send the following commands:

1. **Arithmetic Calculation**:

calculate: x <operator> y

Example: calculate:10 + 5

2. **Find Maximum**:

max:(x1 x2 x3...)

Example: max:(3 9 7 2)

3. **Prime Factorization**:

factors: x

Example: factors: 28

4. **Quit**:

quit

Example: quit

Response

The client prints the server's response for each command. On sending quit, the client disconnects.

Protocol Details

Communication

1. All communication between client and server uses TCP.
2. Messages are encoded as UTF-8 strings.
3. Each message ends with a newline character.

Command Format

Commands have the format: command: arguments

- calculate: Requires variable X, Operator Y, variable Z
- max: Requires a space-separated list within parentheses
- factors: Requires a single integer
- quit: No arguments

Error Handling

- The server responds with error: <description> for invalid commands or operations.

Example Usage

Starting the Server

```
python server.py users.txt
```

Running the Client

```
python client.py
```

Example Interaction

1. **Login**:

User: alice

Password: secret

Response: Hi alice, good to see you.

2. **Send Commands**:

calculate:10 + 5

Response: response: 15

max: (1 4 9 2)

Response: the maximum is 9

factors:28

Response: the prime factors of 28 are: 2, 2, 7

3. **Quit**:

quit

Response: Goodbye.

Notes

- Ensure the users file exists and is correctly formatted before starting the server.
- The server is designed to handle multiple client connections simultaneously.
- The server closes connections with clients who send invalid or malformed data repeatedly.