|  |
| --- |
| Lightbulb Icon Flat Vector Illustration Design Stock Vector ... |
| Design Document  Protocol Design & Proposed Implementation |
| |  |  |  | | --- | --- | --- | | Deirdre Lee | 3/12/24 | Distributed Computing | |

# Objectives and overview

An overview of the protocol's objectives and role in governing client-server communication…

# Design Philosophy

This protocol facilitates communication between clients and a server prioritising simplicity, security and concurrency. It offers the following functionality: allow users to log in, upload messages to the server, download single or all messages from the server, and log out. Security measures are implemented to ensure secure communication and authentication, while concurrency is achieved through asynchronous or non-blocking I/O by the server.

# Protocol for Client-Server Communication

A secure connection will be established using Secure Sockets Layer/Transport Layer Security (SSL/TLS).

The client will communicate with the server using TCP/IP as outlined in the requirements.

The server will be equipped to manage concurrent communication through the use of asynchronous I/O.

Messages will be exchanged between the client and server in plaintext format. Messages will follow a predefined structure, as outlined both below and in the protocol document, with appropriate error messages provided for incorrect input.

SSL/TLS will be used to encrypt and decrypt communication between the client and the server for the duration of the session.

# Format of Each Message Type

# --- Does every message need to be documented as they are being documented in protocol doc ---

## Request

Requests sent by the client to the server will be in string format.

Example:

LOGIN username password

## Response

## Responses received by the client from the server will be in string format.

## Example:

## **101 Login successful**

## Error

## Example:

## **102 Login unsuccessful, please try again**

# Pseudo-code for Functional Requirements

## Client Side

* Connect to the server
* Establish an initial handshake to establish a secure connection
* Send requests to the server
* Receive responses from the server
* Close the connection ( LOG\_OUT ) when done

## Server Side

* Listen for incoming client connections
* Accept client connections and establish communication
* Receive requests from the client(s)
* Process requests and generate responses
* Send responses to the client(s)
* Close the connection with the client(s) when done

# Inter-process Communication

In the system, communication between the client and the server processes is facilitated through a secure connection established using SSL/TLS.

## Sequence of Inter-process Communication

* 1. The client initiates a connection request to the server using SSL/TLS.
  2. The server listens for incoming client connections and accepts them.
  3. Upon successful connection establishment, the client and server perform an SSL/TLS handshake to establish a secure communication channel.
  4. The client acknowledges the server's acknowledgement over the secure connection.
  5. The client sends requests to the server.
  6. The server processes requests.
  7. The server sends back responses to the client.
  8. Communication continues until the session is terminated by either the client or the server.

# Conclusion

Summarise the key points of the protocol specification and emphasise its importance for ensuring proper communication between clients and servers…