CMSC 626 PRINCIPLES OF COMPUTER SECURITY
Design Document

Encrypted and Distributed File System

Team 7
Surya Theja Dokka - SL67413
Sathvik Reddy Musku - SX77250
Vishwanth Reddy Jakka - ZJ06092
Lakshmi Vivek Jagabathina - TN20930

Introduction

The Distributed File System project aims to create a robust and secure platform for file sharing and management. The system is designed to operate in a peer-to-peer (P2P) environment, ensuring data encryption, user authentication, and logging of file operations. This document outlines the architecture, components, and functionalities of the system.

System Overview

The system comprises two main components: the client application and the server application. Both components interact over a network to perform file-related operations securely.

Client Application

The client application provides a graphical user interface (GUI) for users to interact with the file system. It is developed using Python with the Tkinter library for GUI development.

Key Functionalities:

- User authentication (login, registration, password reset).
- File operations (create, read, update, delete, and restore files and directories).
- File encryption and decryption for secure data transmission.
- Logging of file operations for audit and debugging purposes.

GUI Components:

- Login and registration screens.
- File management interface with options to create, delete, restore, read, and write files.
- A logging window to display system logs.

Server Application

The server application handles requests from multiple clients simultaneously, managing file operations and ensuring data integrity and security.

Key Functionalities:

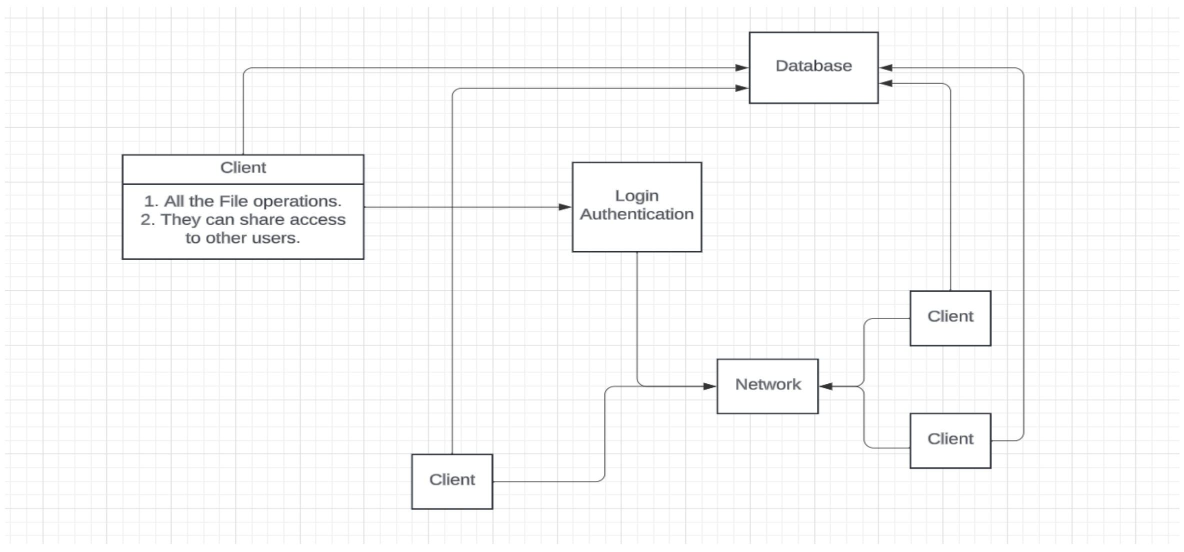
- Handling client requests for file operations.
- Managing user directories and files on the server.
- Encrypting and decrypting data received from and sent to the client.
- Concurrent handling of multiple client connections.

Security

The system uses AES encryption for secure data transmission. A PBKDF2-derived key is used for encryption and decryption, ensuring that file contents and names are secure during transit.

Architecture

The system architecture integrates a client-server model with aspects of peer-to-peer (P2P) networking. While a central server is responsible for user authentication, permissions management, and log maintenance, the system also enables direct file sharing between clients, embodying P2P characteristics. This approach allows clients to connect to a server for certain operations while also facilitating direct, decentralized file transfers among peers


Database
Client
1. All the File operations.
2. They can share access to other users.
Login Authentication
Client
Network
Client
Client

Client Architecture

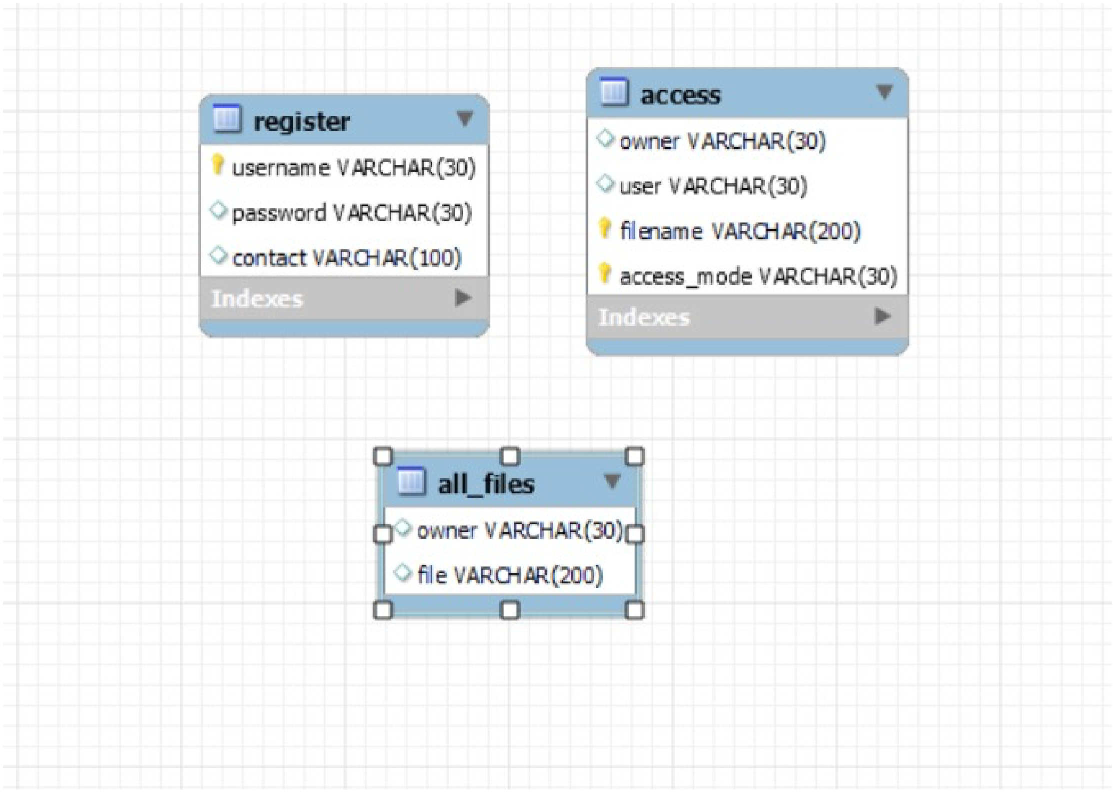
- User Interface Layer: Handles user interaction with the system.
- Network Communication Layer: Manages sending and receiving data to and from the server.
- Encryption Layer: Encrypts and decrypts data for secure communication.

Server Architecture

- Connection Handler: Manages client connections and requests.
- File System Manager: Handles operations on the file system, such as file creation, deletion, and retrieval.
- Security Layer: Encrypts and decrypts data received from and sent to clients.

Database Design

The system uses a MySQL database to store user credentials and file access permissions.



Tables:

- Users: Stores user login information.
- File Permissions: Stores file access permissions for each user.
- Logging: stores the logs(activities) and activities performed by user.

Network Communication

The system uses TCP sockets for network communication between the client and server.

Error Handling and Logging

Comprehensive error handling is implemented to manage exceptions during file operations, database access, and network communication. All significant events and errors are logged to a file for audit and debugging.

Testing and Benchmarking

To ensure the reliability and efficiency of the Distributed File System, a series of benchmarks will be conducted. These tests are aimed at evaluating the system's performance in handling file operations under various conditions.

Benchmark Criteria

The benchmark will primarily focus on the following file operations:

- Random Read: Assessing the time taken to read files randomly selected from the file system.
- Random Write: Measuring the performance in writing data to randomly chosen locations in the system.
- Combined Random Read and Write: Testing the system's ability to handle simultaneous read and write operations.

Conclusion

This Distributed File System project is designed to deliver a secure and efficient platform for file management in a peer-to-peer (P2P) environment. By integrating the principles of P2P networking with certain aspects of a client-server model approach to file sharing. Its architecture promotes direct file transfers between peers, ensuring both ease of use and enhanced data security. The user-friendly interface, coupled with robust security measures, addresses the essential requirements for secure and effective file sharing in a distributed network.