1. Introduction

The purpose of this project is to develop a decentralized file sharing system where users can share files securely and efficiently. The system consists of two major components: a tracker and a client. The tracker manages connections and file information, while the client facilitates user interactions.

2. System Architecture

The architecture comprises two primary components:

- Tracker Server: Manages client connections and maintains a list of active clients and the files they share.
- Clients: Connect to the tracker to register themselves and request files from other clients.

3. Tracker Implementation

3.1 File Structure

The tracker manages file metadata using a defined structure, including:

- File path
- File name
- SHA1 hash values
- Group ID

3.2 Tracking Logic

The tracker listens for incoming connections and handles requests from clients. It uses multithreading to manage multiple connections simultaneously. Key functions include:

- Accepting client connections
- Storing client information
- Managing file requests and responses

3.3 Multithreading

The tracker employs multithreading, allowing parallel execution of multiple clients that can connect through socket programming. This enhances the efficiency and responsiveness of the system.

4. Client Implementation

4.1 Communication with Tracker

Clients initiate a connection to the tracker to register themselves and retrieve the list of available seeders for file download. The communication protocol is based on sending and receiving messages formatted as strings.

4.2 Functionalities

Clients can perform various functionalities such as: create-user, login, create_group etc. These functionalities are communicated to the tracker, where they are stored in respective data structures.

4.3 File Download Logic

Upon receiving a file download request, the client connects to available seeders and retrieves the requested file in chunks. The file chunks are reassembled in the destination directory.

5. Hashing for Data Integrity

To ensure data integrity during file transfers, SHA1 hashing is implemented. Each chunk of the file is hashed, and the final hash of the complete file is also calculated. This guarantees that the downloaded file matches the original.