CS3240 Group 14 - Christopher Smith (cts2bu), Chun Wang (cw8df), Marlena Kauer (mak4wd), Justin Cayajon (jhc7ej)

Design Document

Components

Component #1 – The File Server

The File Server functions as the backend process that stores all of the users' uploaded files and folders. It has the following responsibilities:

- Store and synchronize files inside of server folder across all user machines.
- Share files with other users currently on the same server.
- Authenticate user login data through the User Information Database.
- Fields requests from the User Client and the Admin Client in relation to files and folders.

The File Server thus stores and manages user files and folders. It stays active as long as the OneDir system is running.

Component #2 – The User Client

The User Client handles file user generated tasks in relation to the File Server and the User Information Database. It has the following responsibilities:

- Initiate new user account creation.
- Turn automatic synchronization on or off.
- Change user password.
- Kick off file upload, moving, deletion, or sharing off of the server.

The User Client thus does not actually store any data, as it merely handles the requests to the

File Server and the User Information Database which actually handle things such as user information and file data.

Component #3 – The User Information Database

The User Information Database is a SQLite database that we are using to store the user information for all users of OneDir.

- Contains one table that holds username and passwords as text fields.
- Username is the primary key to ensure that all user accounts are unique.
- Will update itself every time a user creates an account or an admin changes an account.
- Database is persistent and does not lose content unless it is explicitly deleted.
- Can handle queries about user information if from a valid/admin user.

Component #4 - The Admin Client

The Admin Client handles admin user tasks in relation to the File Server and the User Information Database. It has the following responsibilities:

- Request to view user listing.
- View file information (size, number, etc).
- Change a user's password.
- View a traffic report log.

Like the User Client, the Admin Client does not actually store data about user files or user login information; rather, it makes calls to the File Server and the User Information Database, which actually handle storing and reading the information.

Technology for Communication

We use PyCharm as the environment to develop our code. It supports Flask, SQLite, and is very useful for unit testing. We elected to use Flask for the web framework because it has a relatively accessible interface for uploading files via HTTP server protocol. Flask is also thoroughly documented and compatible with PyCharm. It also provides specific abilities to integrate with the Linux curl scripting, which we found useful. We use subprocess which is part of the Python standard library to call curl and other shell commands from our python code. For our database to store user ids and passwords, we use a SQLite database by importing it in the relevant python files. We use SQLite because it is a free, open source database that integrates easily with Python and work well with Flask.

Interaction Examples

a) Client user decides to upload file to the flask server

1. Client requests to upload file.

2. System requests pathway to where file is located.

3. User provides pathway.

4. System stores the pathway strings as strings and then will use the pathway given by user as the pathway for the curl command to find the location of file to upload on local machine.

5. User provides filename.

6. System uses filename in curl command, and then Flask handles uploading file to server using HTTP protocol. File appears in User's OneDir folder.

Comment: for the actual product we should be able to recognize the file's filename and pathway automatically and upload the file to server.

b) Server has updated version of user's file

1. User makes modification to his file in local file system.

2. OS recognizes the file changes and send HTTP request with the file content to flask server.

3. Server compares the new content with existing file (if any) content and stores the changes.

c) File user signs in to the system

1. System prompts User for user id and password.

2. User enters user id and password.

3. System performs a SQL query to select the password of the entered username, if the password returned from the database is the same as the one entered then the sign in will be verified. System will display User's personalized OneDir folder with their files and files shared with them.

d) Admin user deletes a file user from the server

1. File user account information is deleted on the database

2. Username of deleted user becomes available for future use

3. If requested by server, folders of file user are deleted on the server machine

e) File user shares a file with another user.

1. File user requests to share a file with another user on the same server.

2. System requests filename and name of shared user.

3. User provides filename and name of shared user.

4. Using Flask's local server sharing feature through LocalTunnel, the server's specific shared file is sent over to the shared user's oneDir directory.