Deign Document: Manager Layer – User Info Manager

Hang Yuan ([hyuan211@gmail.com](mailto:hyuan211@gmail.com))

Version: 0.1 (09/02/19)

**1 Goals**

This module will manage the information of each user in both local client system and remote server system. Because client side and server side have different functionalities needs of user info management, some specific functions and file formats will be different. This design document will introduce all functions needed but the actual codes will have minute differences.

**2 Design**

The design for User Info Manager module includes two parts: (1) system design; (2) file type and format.

**2.1 System Design**

The module will manipulate the *user.sys* (on local client) or *user.data* (on remote server) to manage all the users’ basic information by reading, updating, creating or deleting user information or user account.

Notice that the *user.sys* and *user.data* are only standard names. The read *user.sys* will be adapted to *username@domain.sys*. The *user.data* will be adapted to *user@domain.data*. All those files will be directly contained under the *data* folder.

**2.1.1 Functions differ depending on sides**

Basic functions will be:

CheckExist (const string &userAccount)

CreateUser (const ServerUserInfo &userInfo)

CloseUser (const string &userAccount)

ReadUser (const string &userAccount, ClientUserInfo &userInfo) on client

ReadUser (ServerUserInfo &userInfo) on server

UpdateUser (const ServerUserInfo &userInfo)

SetupUser (const string &userAccount) only on client

Login (const ServerUserInfo &userInfo)

Logout (const string &userAccount)

|  |  |  |
| --- | --- | --- |
| **Function Name** | **Aim** | |
| **Local client** | **Remote server** |
| CheckExist | Check if the given user account has been ever registered in server database | Check if the given user account exists in current user database |
| CreateUser | Create a new *user.sys* file with the given account information and send user’s info to server | Add the new user into the *user.data* file |
| CloseUser | Remove this user’s *user.sys* file | Delete relevant user information from the *user.data* file |
| ReadUser | Read and return all information of this user from its system file | |
| UpdateUser | Update relevant proportion of user information | |
| SetupUser | Obtain the user’s information from the remote server. If the user info file doesn’t exist, create it first | N/A |
| Login | Verify account and password with the remote server. Then, update login time | Verify account and password with the local client. Then, update login time |
| Logout | Update logout time |  |

**2.1.2 UserInfo structure in different sides**

Based on different requirements of user info in client and server model, different structure will be adopted.

Under local client:

Struct ClientUserInfo {

string username;

string domainName;

time\_t lastLoginTime;

time\_t lastLogoutTime;

};

Under remote server:

Struct ServerUserInfo {

string username;

string domainName;

string password

time\_t lastLoginTime;

time\_t lastLogoutTime;

time\_t changeTimestamp;

};

Struct UserInfoHeader {

Uint32\_t totalUser;

};

**2.2 File Type and Format**

**2.2.1 File types**

Based on the different usage of storing user information, the local client adopts *.sys* as the file type and remote server adopts *.data* as the file type.

**2.2.2 File format**

*user.sys* (*username@domain.sys* in local client) will only contain the *ClientUserInfo* struct without a file header.

*user.data* (*user@domain.data* in remote server) will contain all users’ *SeverUserInfo* structs but with a file header *UserInfoHeader*.