CHERRY CREDITS PTE LTD

Cherry Exchange

Account Authentication API for Game

Created: 21st March 2011

Last modified: 18th May 2011

Version: 2.0.1

Contents

Terms Definition

Term	Definition
Cherry Account	User's account in Cherry Credits. User creates a Cherry Credits account through Cherry Credits portal site. With this account, user is able to use the utilities in Cherry Exchange and to login games offered by Cherry De Games.
Game Account	User's account in game. This account refers to the set of login ID and password used by user to login the game.
Member / User / Player	These basically refer to the same entity, which is the user. User can be known as Member when he has a Cherry Credits account. User can be known as Player when he has a game account.
Cherry Credits Member DB / Database	Database that consists of Cherry Credits member account information. This database is use for the authentication process via the *Authenticate* method of the API.
Cherry Credits Game DB / Database	Database that consists of Cherry Credits member's game accounts information. The game account information is a duplication of the game account information in the game. The existence of this database enables the change of game related login credentials (login ID and password) to be performed without affecting the user's main Cherry Credits account information.
CherryID	Cherry Credits account ID that uniquely identifies a user's Cherry Credits account. (This is different from user's game login ID.) This should be a field stored in the game's local account database's account table for internal reference. Datatype is varchar(15).
Cherry Exchange	This is a platform whereby Cherry Credits member will perform functions such as top-

	up. URL: https://exchange.cherrycredits.com
Cherry De Games	This is a section in Cherry Exchange that offer games which Cherry Credits members can login directly with their Cherry Credits account. Member will be able to check if his game account is activated or not under this section. Member will be able to change his game's password under this section too.

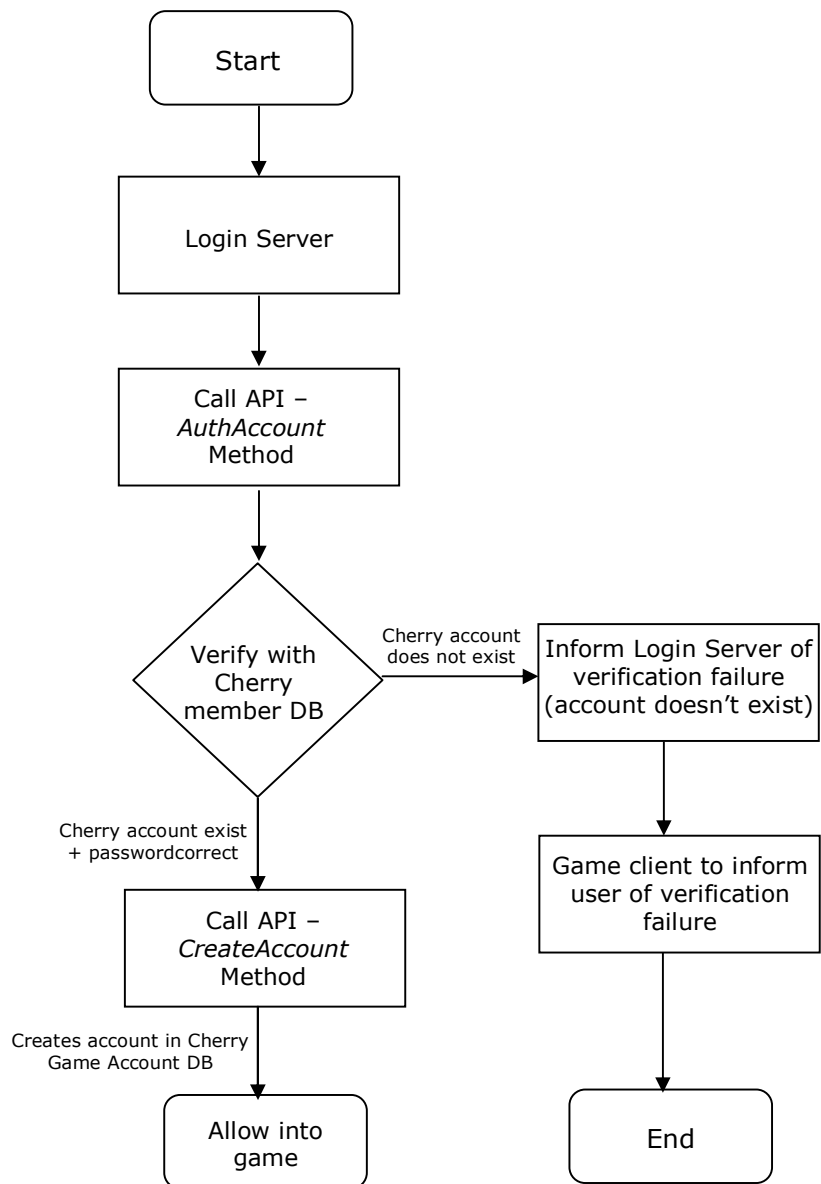# 1. Service Structure for Account Authentication
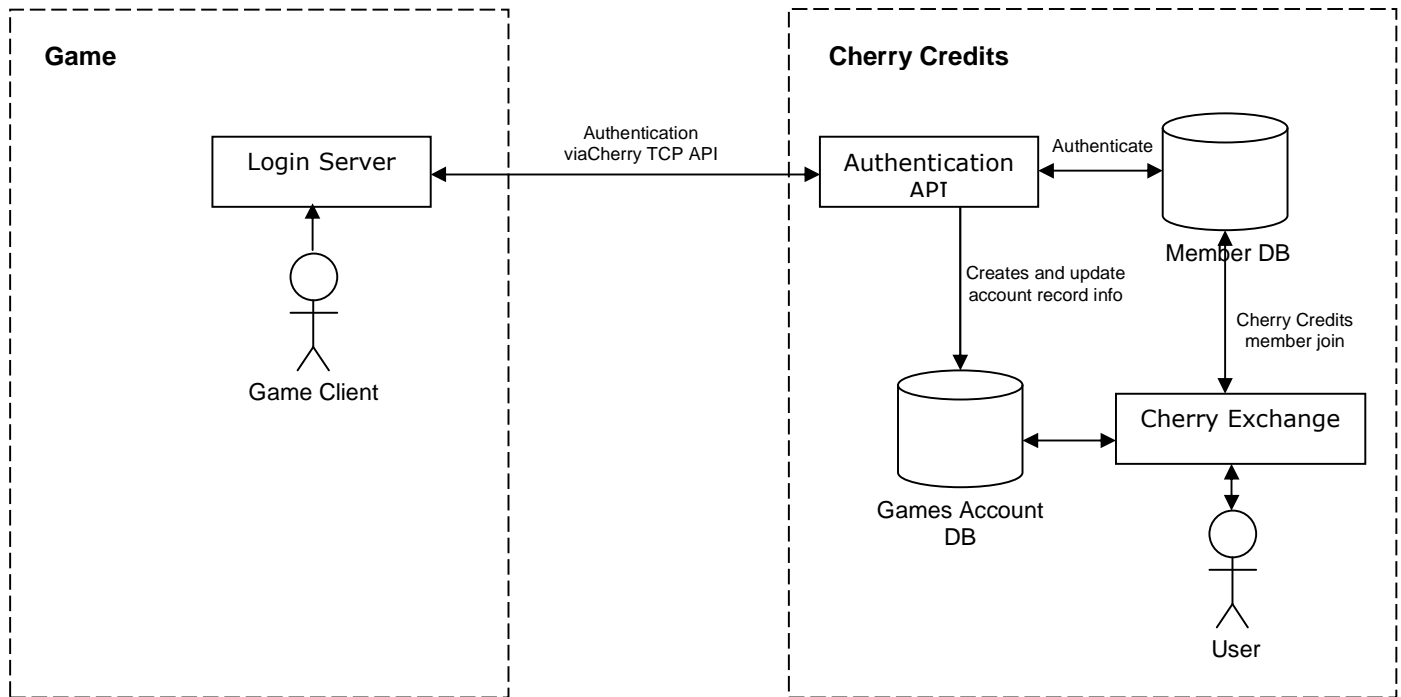
Note that the above diagram shows only the relevant service structure involved in the authentication process. It does not represent the whole service structure of the game nor Cherry Credits system.

The below describes the functionality of the servers and DB illustrated in the above diagram.

Server / DB	Functionality
Login Server (Game)	Game client connects to this server for authentication when player tries to log into the game. This server runs the login program which uses the header and configuration file provided by Cherry Credits for authentication. The header file contains functions that shall make use of Cherry Credits API to authenticate users with Cherry Member DB upon login credentials being sent from the client.
Authentication API (Cherry Credits)	This is the service whereby the game's authentication process shall connect to for authentication of Cherry Credits member. This service will connect to Cherry Member Database to verify account info sent over by the game's authentication / login server.
Cherry Exchange (Cherry Credits)	Cherry Exchange is a service whereby Cherry Credits members top-up their Cherry account for cash purchases in games. Users will need to have a Cherry account in order to play games offered under Cherry De Games. This service will create a Cherry account record in the Cherry Member DB when user sign up a Cherry Credits account via

	Cherry Exchange. This service will connect to the Cherry Credits Games Account DB to retrieve and display member's game account status.
Member DB (Cherry Credits)	This DB consist account info of Cherry Credits members. The game's authentication process will need to authenticate players with this DB via Cherry Credits' API. The Cherry Exchange Server also uses this Member DB for various member related activities such as login and account registration.
Games Account DB (Cherry Credits)	This DB stores the game login info of the Cherry Credits members. The game's authentication process will need to inform Cherry Credits API of successful account creation, the API server will then update the record in this DB of the account creation status.

2. Authentication Work Flow Diagram

```
┌─────────────┐
│   Start     │
└──────┬──────┘
       │
       ▼
┌─────────────┐
│ Login Server│
└──────┬──────┘
       │
       ▼
┌─────────────┐
│  Call API – │
│ AuthAccount │
│   Method    │
└──────┬──────┘
       │
       ▼
      ╱╲
     ╱  ╲          Cherry account      ┌──────────────────────┐
    ╱    ╲         does not exist      │ Inform Login Server of│
   ╱Verify╲───────────────────────────▶│ verification failure  │
   ╲ with ╱                            │(account doesn't exist)│
    ╲Cherry╱                           └───────────┬──────────┘
    ╱member╲                                       │
   ╱  DB   ╲                                       │
   ╲      ╱                                        ▼
    ╲    ╱                              ┌──────────────────────┐
     ╲  ╱                               │ Game client to inform│
      ╲╱                                │  user of verification│
       │                                │       failure        │
  Cherry account exist                  └───────────┬──────────┘
   + passwordcorrect                                │
       │                                            │
       ▼                                            ▼
┌─────────────┐                            ┌─────────────┐
│  Call API – │                            │     End     │
│CreateAccount│                            └─────────────┘
│   Method    │
└──────┬──────┘
       │
Creates account in Cherry
   Game Account DB
       │
       ▼
┌─────────────┐
│ Allow into  │
│    game     │
└─────────────┘
```

3. Authentication Functions

3.1 Connecting to the Authentication Service

To use the service, include files **GA.dll** to LIBRARY_PATH, and **IGA.h** to INCLUDE_PATH. GA.dll is compiled using VS2010. Recompilation is recommended to avoid the conflict due to difference in the internals.

Sample Code (C++):

```cpp
#include"stdafx.h"
#include"IGA.h"
#include<iostream>
#include<string>

int _tmain(intargc, _TCHAR* argv[])
{
        IGA iga = IGA();
        iga.Authenticate("http://192.168.1.240:22011",3,"test@cherrycredits.com","","127.0.0.1");
        std::cout<< iga.ai->GetStatusDescription();
        return 0;
}
```

e.g. code 3.1

The above service will only be accessible by the game's login server's IP provided.

3.2 IGAClass

This interface supports the main method, i.e. **Authenticate** (refer section 3.2.1)

3.2.1 Authenticate Method

This method allows the authorized party to authenticate the user login ID and password with Cherry Credits member DB.

3.2.1.1 Call

This method takes in the game's reference ID, user login ID and user login password to authenticate a user account in Cherry Credits member DB. It populates a public member, anAccountInfo object,that stores the required information for game account creation (refer to section 4.2.1.2 for method's response).

```
void IGA::Authenticate(char const *endpoint, intgameID, charconst *loginID, charconst
*processedPassword, charconst *ip)
```

Parameter	Type	Description
endpoint	string	The protocol, address and port of the service. (Example: net.tcp://116.12.134.130:21011)
gameID	integer	A reference ID that identify the game in Cherry Credits DB. This ID will be assigned by Cherry Credits.
loginID	string	Login ID entered by user to login to the game. Minimum 4 characters and maximum 80 characters. Alphanumeric. Allow '@', '_', '-', '.' symbols. (Login ID used can be a 4~16 character ID or a 4~80 character email address.)
password	string	Login password entered by user to login to the game. The password should be hashed* either using MD5 or Base64. *NOTE: Require login to add in a process for checking '@' symbol in the loginID user inputted. If loginID contains '@' symbol, encrypt password with Base64. Else, encrypt password with MD5.
userIP	string	User's IP address.

__NOTE:__ Use the original password string entered by user for hashing.Do not convert the string to all upper case or all lower case as this may cause a difference in hash result.

3.2.1.2 Public Member

`AccountInfo *ai`

Upon calling the Authenticate method,the public member ai of AccountInfo type will be populated.

Type	Description
AccountInfo	An object that holds the account information. It consist of 6 members namely CherryID, LoginID, Email, Token, StatusCode and StatusDescription. Refer to Section 3.2.1.3 for more information on the 6 members.

3.2.1.3 AccountInfo Class

The purpose of this class is for storing the required information returned by the *Authenticate* method. There are no methods supported for this class.

Members	Type	Description
GetCherryID	string	Cherry Credits account ID that uniquely identifies the user's Cherry Credits account.
GetLoginID	string	User's Cherry game login ID.
GetEmail	string	User's email address (used as alternative login credential).
GetToken	string	(To be ignored for Dragon Nest case)
GetStatusCode	integer	Indicates the result of authentication. Possible values: ▪ 0 *(Authentication failed)* ▪ 1 *(Authentication success)* ▪ -1 *(Account Locked)* ▪ -2*(Account suspended)* ▪ -999 *(Others)*
GetStatusDescription	string	Text description of StatusCode.

4. Information for Integration and Testing

Following information will be issued to the developer upon finalization of details for the integration process:

- Game ID: 3
- Test-bed Endpoint Address: http://116.12.134.130:22011
- Live Endpoint Address:To be confirmed when live server is up.
- Server Access Info: To be provided in separate file.
- Test accounts:
    - Login ID: eyetest01 ~ eyetest05
    - Email: eyetest01@cherrycredits.com ~ eyetest05@cherrycredits.com
    - Password: eyetest

- End -