1 Overview

Throughout this document, the term "web application" is used. The web application can be any forum, CMS, blogging software, etc. that uses MySQL for its database.

These are the major features of Community Bridge:

1. Minecraft player to web application user "linking".
2. Group synchronization (currently a misnomer, since the "synchronization" is only one way)
3. Recording statistical information about the players in web application custom fields.
4. Kicking of players who are banned from the web application.

The first of the features is essential, without it, none of the other features of the plugin would work. Given that the linking feature's importance, it is the first thing to evaluate when ensuring that CommunityBridge is operating correctly.

2 Database Handling

The config.yml section contains the hostname, port, database name, username, and password. It should correctly notify the server owner if the database information is incorrect or correct.

3 Linking Feature

For each of the following configurations, CB should:

- Recognize a player as not being registered, e.g., not present in the web application's database.
- Correctly identify a player as being registered–identifying the correct user.
- If kick-unregistered is turned on, kick unregistered players.

Possible Configurations (order of complexity):

- Minecraft playername and web application username are required to be the same. Required information: users table, user id column, username/playername column.
- Minecraft playername and web application username can be different; the Minecraft playername is in a separate column on the users table. Required information: users table, user id column, username/playername column.

- Minecraft playername and web application username can be different; the Minecraft playername is on a different table in its own column (frequently, this is because it is implemented through a custom profile field in the web application). Required information: tablename, user id column, username/playername column.
- Minecraft playername and web application username can be different; the Minecraft playername is on a different table that data is stored in the key-value format. Required information: tablename, user id column, key column, value column, data key name.

Fortunately, the overall result is the first three possible configurations require the same information. Only when the minecraft playername is stored in a key-value pair do we need different information.

4 Authentication

- **(NYI)** If authentication is off **DO NOT** force authentication.
- **(NYI)** If authentication is on, handle a valid authentication by...
- **(NYI)** If authentication is on, respond to an failed authentication attempt by...

5 Group Synchronization

Many web applications have a notion of a primary group. In fact, some only allow a member to have membership in a single group.

On the other hand, permissions systems appear to rarely have a notion of a primary group. In addition, they frequently have per-world settings, so the world name must be included any operations with a permissions system.

Possible configuration types for a “primary” group includes:

- A “primary” group_id stored on a table alongside a user_id. Required information includes: table_name, user_id column, group_id column.
- A “primary” group_id stored in a key-value pair. Required information includes: table_name, user_id column, key column, value column, data key name

Possible configuration types for “secondary” group includes:

- A list of “secondary” group IDs stored in a single field alongside a user_id. Required information includes: table_name, user_id column, group_ids column, group_id delimiter.

- A junction table to create the many-to-many relationship between user IDs and group IDs. Required information includes: table_name, user_id column, group_id column.
- A series of key-value pairs on a key-value table. Required information includes: table_name, user_id column, key volumn, value column, data key name