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.

1.1 config.yml Preamble and General Settings Section

```
#                  Community Bridge Configuration File
# ------------------------------------------------------------------------------

#                              General Settings
# ------------------------------------------------------------------------------
# Log level controls the degree of detail that is sent to the console/log
# The possible settings are (in order of quietest to noisiest:
#   info, config, fine, finer, finest, all
# - During configuration, I recommend using 'config'.
# - During normal operation, I recommend using 'info'.
# - If you want to see the notifications that the synchronize and reminder
#   notices have been sent, use 'fine'
# - During troubleshooting of problems, set this to either finest or all.
log-level: config

# Allow plugin metrics to start up for this plugin.
# Disable for all plugins by editing the plugins/PluginMetrics/config.yml file.
# They can be viewed at: http://mcstats.org
plugin-metrics: true

# Set the unit of measurement for sync and reminder scheduling. Options are:
# ticks, seconds, minutes, or hours. Note that this applies to both
# auto-sync-every and auto-remind-every.
auto-every-unit: minutes

# This is a timer that will check and sync all data with your database at a set
# interval. Otherwise group and player statistics will only be synchronized when
# they join and leave the server.
auto-sync: true

# The interval for the auto-sync timer, this should not be set any lower than
# 20 minutes.
auto-sync-every: 20
```

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.

2.1 config.yml Database Section

```
#                              Database Settings
# ------------------------------------------------------------------------------
# Please ensure if you run your Minecraft server on a remote host
# that your MySQL server allows REMOTE connections. This WILL NOT WORK unless
# your MySQL server is configured to allow connections from the machine that
# your Minecraft server is running on. If you don't know what that means,
# consult your hosting provider on how to configure your MySQL server correctly.
# All of this information should be the same server and database used by your
# web-application...
database:
  # Hostname for your MySQL Server
  hostname: localhost

  # MySQL Port
  port: 3306

  # Database Name
  name: databasename

  # Database Username
  username: username

  # Database Password
  password: password
```

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.

3.1 config.yml Section for Linking Feature

```
#                              Player Linking Settings
# ------------------------------------------------------------------------------
# Settings associated with linking a Minecraft player with a web application's
# user. As this feature is a prerequisite for all other features, it cannot
# be disabled.
player-user-linking:
  # If you want the player disconnected from the game if they haven't
  # registered, then set this to true. They will be shown the
  # link-unregistered-player message (in message.yml) on the disconnected
  # screen.
  kick-unregistered: false

  # This is a timer that will notify unregistered users to register every few
  # minutes.
  auto-remind: true

  # The interval for the auto-remind timer, this should be no less than
  # 5-10 minutes. Note that this uses the units specified above in
  # auto-every-unit.
  auto-remind-every: 10

  # Set these to true to inform players when they log in if they're linked to
  # the web application. These correspond to the link-unregistered-player and
  # link-registered-player messages in messages.yml.
  notify-registered-player: true
  notify-unregistered-player: true

  # This is where we specify how to associate a Minecraft player with a web
  # application user. We do this by matching up the player name with a name
  # stored in the web application's database.
  #
  # If you want your players to use the same name on both the server and the
  # web application, then you will need to provide the table and column
  # information where the web application relates its user IDs with the user's
  # name/login ID/login/username.
  #
  # If you want to allow them to use a different name on the web application,
  # you need to add a custom field or column to your web application's database,
  # and provide that table and column information here.

  # The name of the table which contains the columns:
  table-name:
```

```
# Column on the table that contains the user ID. Typically something like
# user_id or member_id
user-id-column:

# If the player name is stored in a key-value pair of columns instead of
# its own column, set this to true:
uses-key: false

# If you set 'linking-uses-key' to false, then set this to the column that
# the playername is stored in. Otherwise, leave it empty.
playername-column:

# If you set 'linking-uses-key' to true, then set the key column, value column,
# and the key-name here. Otherwise, leave these fields empty.
key-name:
key-column:
value-column:
```

3.2 messages.yml Section for Linking Feature

Messages related to the linking feature in the messages.yml.

```
#                              Player Linking Messages
# ------------------------------------------------------------------------------
link-registered-player: Registered Account, Linked to Forums.
link-registered-player-group: Registered Account, Linked to Forums. You have been place
link-unregistered-player: Unregistered Account - Please register at ~APPURL~ for full ac
link-unregistered-reminder: Just a reminder to visit ~APPURL~ and register today!
```

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...

4.1 Proposed config.yml Section

```
# Set this to true if you want to require the player to confirm that they're
# the player that is registered on the web application by providing their
# web application password.
authentication-required: false
```

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

5.1 Proposed config.yml Section

```
#                              Group Synchronization Settings
# ------------------------------------------------------------------------------
# Settings to control group synchronization features.
group-synchronization:
  # This subsection contains settings for primary group synchronization. If your
  # web application has an equivalent to a PRIMARY group, you can enable this
  # feature. Do not enable this feature if your web application only has
  # "multiple" group functionality.
  primary:
    enabled: false

    # Notify player that they've been placed in a primary group.
    notify-player: false

    # The table that contains the primary group ID.
    table-name:

    # The column on the table that contains the user ID.
    user-id-column:

    # If the primary group is stored in a key-value pair, set this to true.
    uses-key: false

    # If you set uses-key to false, then set this to the column that the group
    # ID is in on the table. Otherwise, this setting is ignored.
    group-id-column:

    # If you set uses-key to true, then set the following three settings:
    # Key name for the key-value pair.
    key-name:
    # Column that the key name is in
    key-column:
    # Column that the value is in
    value-column:

    # ***** PRIMARY Group Rules *****
    # This subsection is where you tell Community Bridge how you want it to
    # synchronize the primary groups. These settings can be per-world or apply
    # to all worlds (which could be only one world). If you have only one world
    # or you want one or more synchronization rules to apply to all worlds,
    # place those rules in 'all-worlds'. Otherwise, place rules under a section
    # label with the same name as the Minecraft world name.
```

```
    # * webapp-id is the group IDs that the web application uses.
    #   Often, these are numbers, but not always. It is recommended, but not
    #   required, to set up a rule for each possible web application group ID.
    # * permissions-group is the name of the permissions group that the
    #   group ID should be synchronized with.
    # * direction controls which way the synchronization occurs. Possible
    #   options are:
    #   - webapp: Assigns web application user to the specified group ID, based
    #             on the permissions group name.
    #   - minecraft: Assigns the Minecraft player to the specified permissions
    #                group, based on the web application group ID.
    groups:
      all-worlds:
        1:
          webapp-id: 15
          permissions-group: groupname
          direction: webapp
        2:
          webapp-id: 5
          permissions-group: groupname
          direction: minecraft
#     myworld1:
#       1:
#         webapp-id: 15
#         permissions-name: groupname
#         direction: webapp
#     myworld2:
#       1:
#         webapp-id: 15
#         permissions-name: groupname
#         direction: webapp
  # This subsection contains settings for multiple (secondary) group
  # synchronization. Some web applications will have this either addition to
  # or instead of, primary groups. Do not enable this if your web application
  # does not have a technique for handling multiple groups.
  multiple:
    enabled: false

    # Notify player that they've been placed in groups.
    notify-player: false

    # The table that contains the columns.
    table-name:

    # The column on the table that contains the user ID.
```

```
user-id-column:

# There are three ways of storing group information supported.
# 1) delimited: The group IDs are all in the same column, separated by a
#    comma or some other delimiter.
# 2) junction: The group IDs are on a special table (a junction table) that
#    contains user_ids and group_ids.
# 3) key-value: The group IDs are stored in key-value pairs on a key-value
#    pair table.
# This setting tells CommunityBridge which technique your web application
# uses. Choose either: delimited, junction, or key-value.
storage-technique:

# If your web application uses 'delimited' or 'junction', enter the column
# that contains the group IDs here:
group-id-column:

# If your web application uses 'key-value', enter the following three
# settings:
# Key name for the key-value pair.
key-name:
# Column that the key name is in
key-column:
# Column that the value is in
value-column:

# ***** MULTIPLE/SECONDARY Group Rules *****
# This subsection is where you tell Community Bridge how you want it to
# synchronize the groups. These settings can be per-world or apply
# to all worlds (which could be only one world). If you have only one world
# or you want one or more synchronization rules to apply to all worlds,
# place those rules in 'all-worlds'. Otherwise, place rules under a section
# label with the same name as the Minecraft world name.

# * webapp-id is the group IDs that the web application uses.
#   Often, these are numbers, but not always. It is recommended, but not
#   required, to set up a rule for each possible web application group ID.
# * permissions-group is the name of the permissions group that the
#   group ID should be synchronized with.
# * direction controls which way the synchronization occurs. Possible
#   options are:
#   - webapp: Assigns web application user to the specified group ID, based
#             on the permissions group name.
#   - minecraft: Assigns the Minecraft player to the specified permissions
#                group, based on the web application group ID.
```

```
    groups:
      all-worlds:
        1:
          webapp-id: 15
          permissions-group: groupname
          direction: webapp
        2:
          webapp-id: 5
          permissions-group: groupname
          direction: minecraft
#     myworld1:
#       1:
#         webapp-id: 15
#         permissions-name: groupname
#         direction: webapp
#     myworld2:
#       1:
#         webapp-id: 15
#         permissions-name: groupname
#         direction: webapp
```

5.2 Proposed messages.yml Section

```
group-synchronization-primary-notify-player: You have been placed in the primary group '
group-synchronization-multiple-notify-player: You have been placed in the following grou
```

A Full Proposed config.yml

```
#                  Community Bridge Configuration File
# ------------------------------------------------------------------------------

#                              General Settings
# ------------------------------------------------------------------------------
# Log level controls the degree of detail that is sent to the console/log
# The possible settings are (in order of quietest to noisiest:
#   info, config, fine, finer, finest, all
# - During configuration, I recommend using 'config'.
# - During normal operation, I recommend using 'info'.
# - If you want to see the notifications that the synchronize and reminder
#   notices have been sent, use 'fine'
# - During troubleshooting of problems, set this to either finest or all.
log-level: config

# Allow plugin metrics to start up for this plugin.
# Disable for all plugins by editing the plugins/PluginMetrics/config.yml file.
# They can be viewed at: http://mcstats.org
plugin-metrics: true

# Set the unit of measurement for sync and reminder scheduling. Options are:
# ticks, seconds, minutes, or hours. Note that this applies to both
# auto-sync-every and auto-remind-every.
auto-every-unit: minutes

# This is a timer that will check and sync all data with your database at a set
# interval. Otherwise group and player statistics will only be synchronized when
# they join and leave the server.
auto-sync: true

# The interval for the auto-sync timer, this should not be set any lower than
# 20 minutes.
auto-sync-every: 20


#                              Database Settings
# ------------------------------------------------------------------------------
# Please ensure if you run your Minecraft server on a remote host
# that your MySQL server allows REMOTE connections. This WILL NOT WORK unless
# your MySQL server is configured to allow connections from the machine that
# your Minecraft server is running on. If you don't know what that means,
# consult your hosting provider on how to configure your MySQL server correctly.
# All of this information should be the same server and database used by your
```

```
# web-application...
database:
  # Hostname for your MySQL Server
  hostname: localhost

  # MySQL Port
  port: 3306

  # Database Name
  name: databasename

  # Database Username
  username: username

  # Database Password
  password: password


#                              Player Linking Settings
# ------------------------------------------------------------------------------
# Settings associated with linking a Minecraft player with a web application's
# user. As this feature is a prerequisite for all other features, it cannot
# be disabled.
player-user-linking:
  # If you want the player disconnected from the game if they haven't
  # registered, then set this to true. They will be shown the
  # link-unregistered-player message (in message.yml) on the disconnected
  # screen.
  kick-unregistered: false

  # This is a timer that will notify unregistered users to register every few
  # minutes.
  auto-remind: true

  # The interval for the auto-remind timer, this should be no less than
  # 5-10 minutes. Note that this uses the units specified above in
  # auto-every-unit.
  auto-remind-every: 10

  # Set these to true to inform players when they log in if they're linked to
  # the web application. These correspond to the link-unregistered-player and
  # link-registered-player messages in messages.yml.
  notify-registered-player: true
  notify-unregistered-player: true
```

```
  # This is where we specify how to associate a Minecraft player with a web
  # application user. We do this by matching up the player name with a name
  # stored in the web application's database.
  #
  # If you want your players to use the same name on both the server and the
  # web application, then you will need to provide the table and column
  # information where the web application relates its user IDs with the user's
  # name/login ID/login/username.
  #
  # If you want to allow them to use a different name on the web application,
  # you need to add a custom field or column to your web application's database,
  # and provide that table and column information here.

  # The name of the table which contains the columns:
  table-name:

  # Column on the table that contains the user ID. Typically something like
  # user_id or member_id
  user-id-column:

  # If the player name is stored in a key-value pair of columns instead of
  # its own column, set this to true:
  uses-key: false

  # If you set 'linking-uses-key' to false, then set this to the column that
  # the playername is stored in. Otherwise, leave it empty.
  playername-column:

  # If you set 'linking-uses-key' to true, then set the key column, value column,
  # and the key-name here. Otherwise, leave these fields empty.
  key-name:
  key-column:
  value-column:

#                              Group Synchronization Settings
# ------------------------------------------------------------------------------
# Settings to control group synchronization features.
group-synchronization:
  # This subsection contains settings for primary group synchronization. If your
  # web application has an equivalent to a PRIMARY group, you can enable this
  # feature. Do not enable this feature if your web application only has
  # "multiple" group functionality.
  primary:
    enabled: false
```

```
# Notify player that they've been placed in a primary group.
notify-player: false

# The table that contains the primary group ID.
table-name:

# The column on the table that contains the user ID.
user-id-column:

# If the primary group is stored in a key-value pair, set this to true.
uses-key: false

# If you set uses-key to false, then set this to the column that the group
# ID is in on the table. Otherwise, this setting is ignored.
group-id-column:

# If you set uses-key to true, then set the following three settings:
# Key name for the key-value pair.
key-name:
# Column that the key name is in
key-column:
# Column that the value is in
value-column:

# ***** PRIMARY Group Rules *****
# This subsection is where you tell Community Bridge how you want it to
# synchronize the primary groups. These settings can be per-world or apply
# to all worlds (which could be only one world). If you have only one world
# or you want one or more synchronization rules to apply to all worlds,
# place those rules in 'all-worlds'. Otherwise, place rules under a section
# label with the same name as the Minecraft world name.

# * webapp-id is the group IDs that the web application uses.
#   Often, these are numbers, but not always. It is recommended, but not
#   required, to set up a rule for each possible web application group ID.
# * permissions-group is the name of the permissions group that the
#   group ID should be synchronized with.
# * direction controls which way the synchronization occurs. Possible
#   options are:
#   - webapp: Assigns web application user to the specified group ID, based
#             on the permissions group name.
#   - minecraft: Assigns the Minecraft player to the specified permissions
#                group, based on the web application group ID.
groups:
  all-worlds:
```

```
        1:
          webapp-id: 15
          permissions-group: groupname
          direction: webapp
        2:
          webapp-id: 5
          permissions-group: groupname
          direction: minecraft
#     myworld1:
#       1:
#         webapp-id: 15
#         permissions-name: groupname
#         direction: webapp
#     myworld2:
#       1:
#         webapp-id: 15
#         permissions-name: groupname
#         direction: webapp
  # This subsection contains settings for multiple (secondary) group
  # synchronization. Some web applications will have this either addition to
  # or instead of, primary groups. Do not enable this if your web application
  # does not have a technique for handling multiple groups.
  multiple:
    enabled: false

    # Notify player that they've been placed in groups.
    notify-player: false

    # The table that contains the columns.
    table-name:

    # The column on the table that contains the user ID.
    user-id-column:

    # There are three ways of storing group information supported.
    # 1) delimited: The group IDs are all in the same column, separated by a
    #    comma or some other delimiter.
    # 2) junction: The group IDs are on a special table (a junction table) that
    #    contains user_ids and group_ids.
    # 3) key-value: The group IDs are stored in key-value pairs on a key-value
    #    pair table.
    # This setting tells CommunityBridge which technique your web application
    # uses. Choose either: delimited, junction, or key-value.
    storage-technique:
```

```
    # If your web application uses 'delimited' or 'junction', enter the column
    # that contains the group IDs here:
    group-id-column:

    # If your web application uses 'key-value', enter the following three
    # settings:
    # Key name for the key-value pair.
    key-name:
    # Column that the key name is in
    key-column:
    # Column that the value is in
    value-column:

    # ***** MULTIPLE/SECONDARY Group Rules *****
    # This subsection is where you tell Community Bridge how you want it to
    # synchronize the groups. These settings can be per-world or apply
    # to all worlds (which could be only one world). If you have only one world
    # or you want one or more synchronization rules to apply to all worlds,
    # place those rules in 'all-worlds'. Otherwise, place rules under a section
    # label with the same name as the Minecraft world name.

    # * webapp-id is the group IDs that the web application uses.
    #   Often, these are numbers, but not always. It is recommended, but not
    #   required, to set up a rule for each possible web application group ID.
    # * permissions-group is the name of the permissions group that the
    #   group ID should be synchronized with.
    # * direction controls which way the synchronization occurs. Possible
    #   options are:
    #   - webapp: Assigns web application user to the specified group ID, based
    #             on the permissions group name.
    #   - minecraft: Assigns the Minecraft player to the specified permissions
    #                group, based on the web application group ID.
    groups:
      all-worlds:
        1:
          webapp-id: 15
          permissions-group: groupname
          direction: webapp
        2:
          webapp-id: 5
          permissions-group: groupname
          direction: minecraft
#     myworld1:
#       1:
#         webapp-id: 15
```

```
#         permissions-name: groupname
#         direction: webapp
#     myworld2:
#       1:
#         webapp-id: 15
#         permissions-name: groupname
#         direction: webapp
```

B Full Proposed messages.yml

```
#                              Player Linking Messages
# ------------------------------------------------------------------------------
link-registered-player: Registered Account, Linked to Forums.
link-registered-player-group: Registered Account, Linked to Forums. You have been place
link-unregistered-player: Unregistered Account - Please register at ~APPURL~ for full ac
link-unregistered-reminder: Just a reminder to visit ~APPURL~ and register today!


group-synchronization-primary-notify-player: You have been placed in the primary group '
group-synchronization-multiple-notify-player: You have been placed in the following grou
```