# Table of Contents

**User Authentication**....................1
- Overview....................1
- Password Management....................1
- User Mapping....................1
- User Registration....................2
- Login Management....................2
  - No Login (select none)....................2
  - Template Login (select Foswiki::LoginManager::TemplateLogin)....................2
    - Enabling Template Login....................2
  - Apache Login (select Foswiki::LoginManager::ApacheLogin)....................3
    - Enabling Apache Login using mod_auth....................3
    - Logons via bin/logon....................4
- Sessions....................4
  - Getting, Setting, and Clearing Session Variables....................5
  - Cookies and Transparent Session IDs....................5
- Username vs. Login Username....................5
- Changing Passwords....................6
- Changing E-mail Addresses....................6
- Controlling access to individual scripts....................6
- How to choose an authentication method....................6

# User Authentication

*Controlling who can access your site*

## Overview

Authentication, or "login", is the process by which a user lets Wiki know who they are.

Authentication isn't just to do with access control. Wiki uses authentication to keep track of who made changes, and manage a wide range of personal settings. With authentication enabled, users can personalise Wiki and contribute as recognised individuals, instead of shadows.

Wiki authentication is very flexible, and can either stand alone or integrate with existing authentication schemes. You can set up Wiki to require authentication for every access, or only for changes. Authentication is also essential for access control.

**Quick Authentication Test** - Use the %USERINFO% macro to return your current identity:

- You are guest, WikiGuest,

Wiki user authentication is split into four sections; password management, user mapping, user registration, and login management. Password management deals with how users personal data is stored. Registration deals with how new users are added to the wiki. Login management deals with how users log in.

Once a user is logged on, they can be remembered using a *Client Session* stored in a cookie in the browser (or by other less elegant means if the user has disabled cookies). This avoids them having to log on again and again.

Wiki user authentication is configured through the Security Settings pane in the configure interface.

Please note FileAttachments are not protected by Wiki User Authentication.

## Password Management

As shipped, Wiki supports the Apache 'htpasswd' password manager. This manager supports the use of `.htpasswd` files on the server. These files can be unique to Wiki , or can be shared with other applications (such as an Apache webserver). A variety of password encodings are supported for flexibility when re-using existing files. See the descriptive comments in the Security Settings section of the configure interface for more details.

You can easily plug in alternate password management modules to support interfaces to other third-party authentication databases.

The password manager is selected using the {PasswordManager} setting in `configure`.

## User Mapping

Usually when you are using an external authentication method, you want to map from an unfriendly "login name" to a more friendly WikiName. Also, an external authentication database may well have user information you want to import to Wiki , such as user groups.

By default, Wiki supports mapping of usernames to wikinames, and supports Wiki groups internal to Wiki . If you want, you can plug in an alternate user mapping module to support import of groups etc.

The user mapping manager is selected using the {UserMappingManager} setting in `configure`.

## User Registration

New user registration uses the password manager to set and change passwords and store email addresses. It is also responsible for the new user verification process. the registration process supports **single user registration** via the UserRegistration page, and **bulk user registration** via the BulkRegistration page (for admins only).

The registration process is also responsible for creating user topics, and setting up the mapping information used by the User Mapping support.

## Login Management

Login management controls the way users have to log in. There are three basic options; no login, login via a Wiki login page, and login using the webserver authentication support. the login manager is selected using the {LoginManager} setting in `configure`.

### No Login (select `none`)

Does exactly what it says on the tin. Forget about authentication to make your site completely public - anyone can browse and edit freely, in classic Wiki style. All visitors are given the WikiGuest default identity, so you can't track individual user activity.

⚠ **Note:** This setup is **not** recommended on public websites for security reasons; anyone would be able to change system settings and perform tasks usually restricted to administrators.

### Template Login (select `Foswiki::LoginManager::TemplateLogin`)

Template Login asks for a username and password in a web page, and processes them using whatever Password Manager you choose. Users can log in and log out. Client Sessions are used to remember users. Users can choose to have their session remembered so they will automatically be logged in the next time they start their browser.

#### Enabling Template Login

1. Use the configure interface to
    1. select the `Foswiki::LoginManager::TemplateLogin` login manager (on the Security Settings pane).
    2. select the appropriate password manager for your system, or provide your own.
    3. ℹ there is also an EXPERT configure setting `{TemplateLogin}{PreventBrowserRememberingPassword}` that you can set to prevent Browsers from remembering username and passwords if you are concerned about public terminal usage.
2. Register yourself in the UserRegistration topic.
   ℹ Check that the password manager recognises the new user. If you are using `.htpasswd` files,

check that a new line with the username and encrypted password is added to the `.htpasswd` file. If not, you probably got a path wrong, or the permissions may not allow the webserver user to write to that file.
3. Create a new topic to check if authentication works.
4. **Edit the AdminGroup topic in the Main web to include users with system administrator status.** **This is a very important step**, as users in this group can access *all* topics, independent of Wiki access controls.

AccessControl has more information on setting up access controls.

At this time AccessControls cannot control access to files in the `pub` area, unless they are only accessed through the `viewfile` script. If your `pub` directory is set up in the webserver to allow open access you may want to add `.htaccess` files in there to restrict access.

You can create a custom version of the UserRegistration form by copying the topic, and then deleting or adding input tags in your copy. The `name=""` parameter of the input tags must start with: `"Twk0..."` (if this is an optional entry), or `"Twk1..."` (if this is a required entry). This ensures that the fields are carried over into the user home page correctly. Do **not** modify the version of UserRegistration shipped with Wiki , as your changes will be overwritten next time you upgrade.

The default new user template page is in System.NewUserTemplate. The same macros get expanded as in the template topics. You can create a custom new user home page by creating the Main.NewUserTemplate? topic, which will then override the default.

## Apache Login (select `Foswiki::LoginManager::ApacheLogin`)

Using this method Wiki does not authenticate users internally. Instead it depends on the `REMOTE_USER` environment variable, which is set when you enable authentication in the webserver.

The advantage of this scheme is that if you have an existing website authentication scheme using Apache modules such as `mod_auth_ldap` or `mod_auth_mysql` you can just plug in directly to them.

The disadvantage is that because the user identity is cached in the browser, you can log in, but you can't log out again unless you restart the browser.

Wiki maps the `REMOTE_USER` that was used to log in to the webserver to a WikiName using the table in WikiUsers. This table is updated whenever a user registers, so users can choose not to register (in which case their webserver login name is used for their signature) or register (in which case that login name is mapped to their WikiName).

The same private `.htpasswd` file used in Wiki Template Login can be used to authenticate Apache users, using the Apache Basic Authentication support.

**Warning:** Do **not** use the Apache `htpasswd` program with `.htpasswd` files generated by Wiki ! `htpasswd` wipes out email addresses that Wiki plants in the info fields of this file.

### Enabling Apache Login using `mod_auth`

You can use any other Apache authentication module that sets REMOTE_USER.

1. Use configure to select the `Foswiki::LoginManager::ApacheLogin` login manager.
2. Use configure to set up Wiki to create the right kind of `.htpasswd` entries.

3. Create a `.htaccess` file in the `bin` directory.
   🛈 There is an template for this file in `bin/.htaccess.txt` that you can copy and change. The comments in the file explain what need to be done.
   🛈 If you got it right, the browser should now ask for login name and password when you click on the Edit. If `.htaccess` does not have the desired effect, you may need to "AllowOverride All" for the directory in `httpd.conf` (if you have root access; otherwise, e-mail web server support)
   ⚠ At this time AccessControls do not control access to files in the `pub` area, unless they are only accessed through the `viewfile` script. If your `pub` directory is set up to allow open access you may want to add `.htaccess` files in there as well to restrict access
4. You can create a custom version of the UserRegistration form by copying the default topic, and then deleting or adding input tags in your copy. The `name=""` parameter of the input tags must start with: `"Twk0..."` (if this is an optional entry), or `"Twk1..."` (if this is a required entry). This ensures that the fields are carried over into the user home page correctly. Do **not** modify the version of UserRegistration shipped with Wiki , as your changes will be overwritten next time you upgrade. The default new user template page is in System.NewUserTemplate. The same macros get expanded as in the template topics. You can create a custom new user home page by creating the Main.NewUserTemplate? topic, which will then override the default.
5. Register yourself in the UserRegistration topic.
   🛈 Check that a new line with the username and encrypted password is added to the `.htpasswd` file. If not, you may have got a path wrong, or the permissions may not allow the webserver user to write to that file.
6. Create a new topic to check if authentication works.
7. **Edit the AdminGroup topic in the Main web to include users with system administrator status.**
   ⚠ **This is a very important step**, as users in this group can access *all* topics, independent of Wiki access controls.

AccessControl has more information on setting up access controls.

### Logons via bin/logon

Any time a user requests a page that needs authentication, they will be forced to log on. It may be convenient to have a "logon" link as well, to give the system a chance to identify the user and retrieve their personal settings. It may be convenient to force them to log on.

The **`bin/logon`** script enables this. If you are using Apache Login, the **`bin/logon`** script must be setup in the **`bin/.htaccess`** file to be a script which requires a `valid user`. Once authenticated, it will redirect the user to the view URL for the page from which the `logon` script was linked.

## Sessions

Wiki uses the CPAN:CGI::Session and CPAN:CGI::Cookie modules to track sessions. These modules are de facto standards for session management among Perl programmers. If you can't use Cookies for any reason, CPAN:CGI::Session also supports session tracking using the client IP address.

You don't *have* to enable sessions to support logins in Wiki . However it is **strongly** recommended. Wiki needs some way to remember the fact that you logged in from a particular browser, and it uses sessions to do this. If you don;t enable sessions, Wiki will try hard to remember you, but due to limitations in the browsers it may also forget you (and then suddenly remember you again later!). So for the best user experience, you should enable sessions.

There are a number of macros available that you can use to interrogate your current session. You can even add your own session variables to the Wiki cookie. Session variables are referred to as "sticky" variables.

## Getting, Setting, and Clearing Session Variables

You can get, set, and clear session variables from within Wiki web pages or by using script parameters. This allows you to use the session as a personal "persistent memory space" that is not lost until the web browser is closed. Also note that if a session variable has the same name as a Wiki preference, the session variables value takes precedence over the Wiki preference. **This allows for per-session preferences.**

To make use of these features, use the tags:

```
%SESSION_VARIABLE{ "varName" }%
%SESSION_VARIABLE{ "varName" set="varValue" }%
%SESSION_VARIABLE{ "varName" clear="" }%
```

Note that you **cannot** override access controls preferences this way.

## Cookies and Transparent Session IDs

Wiki normally uses cookies to store session information on a client computer. Cookies are a common way to pass session information from client to server. Wiki cookies simply hold a unique session identifier that is used to look up a database of session information on the Wiki server.

For a number of reasons, it may not be possible to use cookies. In this case, Wiki has a fallback mechanism; it will automatically rewrite every internal URL it sees on pages being generated to one that also passes session information.

# Username vs. Login Username

This section applies only if you are using authentication with existing login names (i.e. mapping from login names to WikiNames).

Wiki internally manages two usernames: Login Username and Foswiki Username.

- **Login Username:** When you login to the intranet, you use your existing login username. This name is normally passed to Foswiki by the **`REMOTE_USER`** environment variable, and used internally. Login Usernames are maintained by your system administrator.

- **Foswiki Username:** Your name in WikiNotation, ex: **`JohnSmith`**, is recorded when you register using UserRegistration; doing so also generates a personal home page in the Main web.

Foswiki can automatically map an Intranet (Login) Username to a Foswiki Username if the {AllowLoginName} is enabled in configure. The default is to use your WikiName as a login name.

> **NOTE: To correctly enter a WikiName** - your own or someone else's - be sure to include the Main web name in front of the Wiki username, followed by a period, and no spaces, for example **`Main.WikiUsername`** or **`%USERSWEB%.WikiUsername`**. This points **`WikiUsername`** to the Main web, where user home pages are located, no matter which web it's entered in. Without the web prefix, the name appears as a NewTopic? everywhere but in the Main web.

## Changing Passwords

If your {PasswordManager} supports password changing, you can change and reset passwords using forms on regular pages.

- The ChangePassword form ( **`Foswiki/ChangePassword`** )
- The ResetPassword form ( **`Foswiki/ResetPassword`** )

## Changing E-mail Addresses

If the active {PasswordManager} supports storage and retrieval of user e-mail addresses, you can change your e-mail using a regular page. As shipped, this is true only for the Apache 'htpasswd' password manager.

- The ChangeEmailAddress form ( **`Foswiki/ChangeEmailAddress`** )

## Controlling access to individual scripts

You may want to add or remove scripts from the list of scripts that require authentication. The method for doing this is different for each of Template Login and Apache Login.

- For Template Login, update the {AuthScripts} list using configure
- For Apache Login, add/remove the script from `.htaccess`

## How to choose an authentication method

One of the key features of Foswiki is that it is possible to add HTML to topics. No authentication method is 100% secure on a website where end users can add HTML, as there is always a risk that a malicious user can add code to a topic that gathers user information, such as session IDs. The Foswiki developers have been forced to make certain tradeoffs, in the pursuit of efficiency, that may be exploited by a hacker.

This section discusses some of the known risks. You can be sure that any potential hackers have read this section as well!

At one extreme, the most secure method is to use Foswiki via SSL (Secure Sockets Layer), with a login manager installed and Client Sessions turned **off**.

Using Foswiki with sessions turned off is a pain, though, as with all the login managers there are occasions where Foswiki will forget who you are. The best user experience is achieved with sessions turned **on**.

As soon as you allow the server to maintain information about a logged-in user, you open a door to potential attacks. There are a variety of ways a malicious user can pervert Foswiki to obtain another users session ID, the most common of which is known as a cross-site scripting attack. Once a hacker has an SID they can pretend to be that user.

To help prevent these sorts of attacks, Foswiki supports **IP matching**, which ensures that the IP address of the user requesting a specific session is the same as the IP address of the user who created the session. This works well as long as IP addresses are unique to each client, and as long as the IP address of the client can't be faked.

Session IDs are usually stored by Foswiki in cookies, which are stored in the client browser. Cookies work well, but not all environments or users permit cookies to be stored in browsers. So Foswiki also supports two other methods of determining the session ID. The first method uses the client IP address to determine the session ID. The second uses a rewriting method that rewrites local URLs in Foswiki pages to include the session ID in the URL.

The first method works well as long as IP addresses are **unique** to each individual client, and client IP addresses can't be faked by a hacker. If IP addresses are unique and can't be faked, it is almost as secure as cookies + IP matching, so it ranks as the **fourth most secure method**.

If you have to turn IP matching off, and cookies can't be relied on, then you may have to rely on the second method, URL rewriting. This method exposes the session IDs very publicly, so should be regarded as "rather dodgy".

Most Foswiki sites don't use SSL, so, as is the case with **most** sites that don't use SSL, there is always a possibility that a password could be picked out of the aether. Browsers do not encrypt passwords sent over non-SSL links, so using Apache Login is no more secure than Template Login.

Of the two shipped login managers, Apache Login is probably the most useful. It lets you do this sort of thing:

```
wget --http-user=RogerRabbit --http-password=i'mnottelling
http://www.example.com/bin/save/Sandbox/StuffAUTOINC0?text=hohoho,%20this%20is%
```

i.e. pass in a user and password to a request from the command-line. However it doesn't let you log out.

Template Login degrades to url re-writing when you use a client like dillo that does not support cookies. However, you can log out and back in as a different user.

Finally, it would be really neat if someone was to work out how to use certificates to identify users.....

See Foswiki:Support.SupplementalDocuments for more information.

---

**Related Topics:** AdminDocumentationCategory, AccessControl
Edit | Attach | Print version | History: %REVISIONS% | Backlinks | Raw View | More topic actions
Topic revision: r1 - 12 Sep 2009 - 04:03:24 - ProjectContributor

- System

- Log In

- **Toolbox**
- Users
- Groups
- Index
- Search
- Changes
- Notifications
- RSS Feed
- Statistics
- Preferences

- **User Reference**
- BeginnersStartHere
- TextFormattingRules
- Macros
- FormattedSearch

- QuerySearch
- DocumentGraphics
- SkinBrowser
- InstalledPlugins

- **Admin Maintenance**
- Reference Manual
- AdminToolsCategory
- InterWikis
- ManagingWebs
- SiteTools
- DefaultPreferences
- WebPreferences

- **Categories**
- Admin Documentation
- Admin Tools
- Developer Doc
- User Documentation
- User Tools

- **Webs**
- Public
- System

-
-
-