From EULER Project

PmWiki: Passwords Admin

PmWiki has built-in support for password-protecting various areas of the wiki site. Passwords can be applied to individual pages, to Wiki Groups, or to the entire wiki site. Note that the password protection mechanisms described here are only a small part of overall system (and wiki) security, see PmWiki.Security for more discussion of this.

Authors can use PmWiki to add passwords to individual pages and WikiGroups as described in Passwords. However, WikiAdministrators can also set passwords in *local/config.php* as described below. (Please note that one cannot set passwords reliably in per group or per page customization files. See the FAQ section for details.)

Password basics

PmWiki supports several levels of access to wiki pages, known as authorisation level:

- **`read`** passwords allow viewing the contents of wiki pages
- **`edit`** passwords control editing and modification of wiki pages
- **`attr`** passwords control who is able to set passwords on pages (and potentially other future attributes)
- **`upload`** password, if uploads are enabled, controls uploading of files and attachments
- in addition all available actions can be password authorised
- **`admin`** password allows an administrator to override the passwords set for any individual page or group.

By default, PmWiki has the following password settings:

- The `admin` and `upload` passwords are locked by default.
- The Main and PmWiki groups have a locked `attr` password (in their respective GroupAttributes pages).
- The pages in the Site group except Site.SideBar are locked against editing; by default the Site.SideBar page requires the admin or the site-wide edit password.

An `admin` password can be used to overcome "locked" passwords, other than that, no password will allow access.

See Passwords for information about setting per-page and per-group passwords. The remainder of this page describes setting site-wide passwords from the *local/config.php* file.

Setting site-wide passwords

One of the first things an admin should do is set an `admin` password for the site. This is done via a line like the following in the *local/config.php* file:

`$DefaultPasswords`['admin'] = crypt('secret_password');

Note that the crypt() call is required for this -- PmWiki stores and processes all passwords internally as encrypted strings. See the crypt section below for details about eliminating the cleartext password from the configuration file.

To set the entire site to be editable only by those who know an "edit" password, add a line like the following to *local/config.php*:

`$DefaultPasswords`['edit'] = crypt('edit_password');
Similarly, you can set a password for any available action, viz `$DefaultPasswords['read']`, `$DefaultPasswords['edit']`, and `$DefaultPasswords['upload']` to control default `read`, `edit`, and `upload` passwords for the entire site. The default passwords are used only for pages and groups which do not have passwords set. Also, each of the `$DefaultPasswords` values may be arrays of encrypted passwords:

`$DefaultPasswords`['read'] = array(crypt('alpha'), crypt('beta'));
`$DefaultPasswords`['edit'] = crypt('beta');
This says that either "alpha" or "beta" can be used to read pages, but only the "beta" password will allow someone to edit a page. Since PmWiki remembers any passwords entered during the current session, the "beta" password will allow both reading and writing of pages, while the "alpha" password allows reading only. A person without either password would be unable to view pages at all.

Identity-based authorization (username/password logins, AuthUser)

Unlike many systems which have **identity-based** systems for controlling access to pages (e.g., using a separate *username* and *password* for each person), PmWiki defaults to a *password-based* system as described above. In general password-based systems are often easier to maintain because they avoid the administrative overheads of creating user accounts, recovering lost passwords, and mapping usernames to permitted actions.

However, PmWiki's *authuser.php* script augments the password-based system to allow access to pages based on a username and password combination. See AuthUser for more details on controlling access to pages based on user identity.

Security holes ...

Administrators need to carefully plan where passwords are applied to avoid opening inadvertent security holes. If your wiki is open (anyone can read and edit), this would not seem to be a concern, **except**, a malicious or confused user could apply a read password to a group and make the group completely unavailable to all other users. At the very least, even an open wiki should have a site-wide "admin" password and a site-wide "attr" password set in config.php. The *sample-config.php* file distributed with PmWiki indicates that the PmWiki and Main groups have "attr" locked by default, but if anyone creates a new group, "attr" is unlocked. Administrators must remember to set "attr" passwords for each new group (if desired) in this case. An easier solution is to include these lines in *config.php* :

```
$DefaultPasswords['admin'] = crypt('youradminpassword');
$DefaultPasswords['attr'] = crypt('yourattrpassword');
```

Encrypting passwords in *config.php*

One drawback to using the crypt() function directly to set passwords in *config.php* is that anyone able to view the file will see the unencrypted password. For example, if *config.php* contains

`$DefaultPasswords`['admin'] = crypt('mysecret');
then the "mysecret" password is in plain text for others to see. However, a wiki administrator can obtain and use an encrypted form of the password directly by using `?action=crypt` on any PmWiki url (or just jump to PasswordsAdmin?action=crypt). This action presents a form that generates encrypted versions of passwords for use in the *config.php* file. For example, when `?action=crypt` is given the password "`mysecret`", PmWiki will return a string like

`$1$hMMhCdfT$mZSCh.BJOidMRn4SOUUSi1`
The string returned from `?action=crypt` can then be placed directly into config.php, as in:

`$DefaultPasswords`['admin'] = '$1$hMMhCdfT$mZSCh.BJOidMRn4SOUUSi1';
Note that in the encrypted form the *crypt* keyword and parentheses are removed, since the password is already encrypted. Also, the encrypted password must be in single quotes. In this example the password is still "`mysecret`", but somebody looking at *config.php* won't be able to see that just from looking at the encrypted form. *Crypt* may give you different encryptions for the same password--this is normal (and makes it harder for someone else to determine the original password).

Removing passwords

To remove a site password entirely, such as the default locked password for uploads, just set it to empty:

`$DefaultPasswords`['upload'] = '';
You can also use the special password "@nopass" via `?action=attr` to have a non-password protected page within a password-protected group, or a non-password protected group with a site-wide default password set.

Revoking or invalidating passwords

If a password is compromised and the wiki administrator wants to quickly invalidate all uses of that password on a site, a quick solution is the following in *local/config.php*:

```
$ForbiddenPasswords = array('secret', 'tanstaafl');
if (in_array(@$_POST['authpw'], $ForbiddenPasswords))
  unset($_POST['authpw']);
```

This prevents "secret" and "tanstaafl" from ever being accepted as a valid authorization password, regardless of what pages may be using it.

See Also

- The `$HandleAuth` array, which sets the required authentication level that is necessary to perform an action.
- Cookbook:RequireAuthor

Protecting actions (example)

Each action can be password protected. Cookbook authors providing scripts with own actions can use this also, but I'll limit the example to a (by default) not protected `?action=source`. This action shows the wikisource of the actual page. Sometimes you don't want that especially to Cookbook:protect email or when using some conditional markup which should not be discovered easily or only by persons that are allowed to edit the page.

There are several solutions for that:

1. Limit "source" only to editors add the following to your *local/config.php*:
   `$HandleAuth['source'] ='edit';`
2. For using "source" with an own password, then add:
   `$HandleAuth['source'] ='source';`
   `$DefaultPasswords['source'] = crypt(secret);` # *see above*

If you additionally want to set the password in the attributes page add:

```
$PageAttributes['passwdsource'] = "$['Set new source password']";
```

In general, adding the prefix 'passwd' to an action name in the `$PageAttributes` array indicates that you wish for the given field to be encrypted when saved to disk.

The full set of steps to add new password handling for an action such as "diff" would be:

```
# add a new (encrypted) field to the attr page
$PageAttributes['passwddiff'] = '$[Set new history password]';

# clear the default password for 'diff'
$DefaultPasswords['diff'] = '';

# Tell PmWiki that the 'diff' password allows action 'diff'.
$HandleAuth['diff'] = 'diff';

# Tell PmWiki that a 'read' password
# (or optionally the 'edit') password
# is also sufficient to enable 'diff'.
# Of course, the 'admin' password will work too.
$AuthCascade['diff'] = 'read';    ## or 'edit'
```

There seems to be a default password. What is it?

There isn't any valid password until you set one. Passwords admin describes how to set one.

PmWiki comes "out of the box" with `$DefaultPasswords`['admin'] set to '*'. This doesn't mean the password is an asterisk, it means that default admin password has to be something that encrypts to an asterisk. Since it's impossible for the crypt() function to ever return a 1-character encrypted value, the admin password is effectively locked until the admin sets one in config.php.

How do I use passwd-formatted files (like .htpasswd) for authentication?

See AuthUser, Cookbook:HtpasswdForm or Cookbook:UserAuth.

Is there anything I can enter in a GroupAttributes field to say 'same as the admin password'? If not, is there anything I can put into the config.php file to have the same effect?

Enter '@lock' in GroupAttributes?action=attr to require an admin password for that group.

How do I edit protect, say, all RecentChanges pages?

see Security#wikivandalism.

How can I read password protect all pages in a group except the HomePage using configuration files?

As described in PmWiki.PerGroupCustomizations per-group or per-page configuration files should not be used for defining passwords. The reason is that per-group (or per-page) customization files are only loaded for the current page. So, if `$DefaultPasswords['read']` is set in *local/GroupA.php*, then someone could use a page in another group to view the contents of pages in GroupA. For example, Main.WikiSandbox could contain:

    (:include GroupA.SomePage:)

and because the *GroupA.php* file wasn't loaded (we're looking at Main.WikiSandbox --> *local/Main.php*), there's no read password set.

How can I password protect the creation of new pages?

See Cookbook:LimitWikiGroups, Cookbook:NewGroupWarning, Cookbook:LimitNewPagesInWikiGroups.

How do I change the password prompt screen?

If your question is about how to make changes to that page... edit Site.AuthForm. If your question is about how to change which page you are sent to when prompted for a password, you might check out the Cookbook:CustomAuthForm for help.

I get http error 500 "Internal Server Error" when I try to log in. What's wrong?

This can happen if the encrypted passwords are not created on the web server that hosts the PmWiki.
The crypt function changed during the PHP development, e.g. a password encrypted with PHP 5.2 can not be decrypted in PHP 5.1, but PHP 5.2 can decrypt passwords created by PHP 5.1.
This situation normally happens if you prepare everything on your local machine with the latest PHP version and you upload the passwords to a webserver which is running an older version.
The same error occurs when you add encrypted passwords to local/config.php.

Solution: Create the passwords on the system with the oldest PHP version and use them on all other systems.

I only want users to have to create an 'edit' password, which is automatically used for their 'upload' & 'attr' passwords (without them having to set those independently). How do I do this?

By setting `$HandleAuth` like so:

```
$HandleAuth['upload'] = 'edit';
      // And to prevent a WikiSandbox from having it's 'attr' permissions changed
      // except by the admin (but allowing editors to change it on their own pages/group)
      if(($group=="Site") || ($group=="Main") || ($group=="Category") ||
             ($group=="SiteAdmin") || ($group=="PmWiki") ) {
$HandleAuth['attr'] = 'admin';  // for all main admin pages, set 'attr' to 'admin' password
      } else {
$HandleAuth['attr'] = 'edit';  // if you can edit, then you can set attr
      }
```

Retrieved from https://www-sop.inria.fr/mascotte/EULER/wiki/pmwiki.php/PmWiki/PasswordsAdmin
Page last modified on July 10, 2009, at 04:10 PM