Opinion: This is too complex and unclear for Tiki users. I think it should be removed.

Permission Check

Tiki Permission Check (TPC): An additional source of information if Tiki install fails.
Since Oct. 2012, standalone (attached below to this page), and included also as a feature in Tiki10.

1. Introduction

What is Tiki Permission Check?

If Tiki installer and tiki-check.php fail, Tiki Permission Check can be used to figure out some details about filesystem permissions needed by the webserver to make those ones work. Permissions for Tiki Permission Check can be fixed manually (only a few files via FTP or SSH or other shell access) or by an included script. Then Tiki Permission Check can be used to figure out which filesystem permissions are needed to run Tiki on the webserver. This permission setting needn't be unique, so you may choose more or less restrictions depending on the security level you want and/or need.

Tiki Permission Check can be used to figure out file permission problems independent from Tiki itself. In those cases file permissions have to be adjusted manually or by whatsoever method appropriate to that alien project.

A convenient way to use the information for installing Tiki is by setup.sh modification (upcoming soon).

Predefined Permission Models

Due to several webserver settings und use cases TPC offers several predefined models to check whether an installation will work with them or not. Once the correct model(s) is (are) known, they can be used to run the setup script `sh setup.sh foo` with the appropriate command. Builtin permission models (Effective 2012-11-11):

- insane
- mixed
- morepain
- moreworry
- pain
- paranoia
- paranoia-suphp
- risky
- sbox
- worry

Without shell access the permissions need to be set by FTP or another method. This may be a lot of work, since some files and subdirs need to be writable, but those wo needn't be writable shouldn't be given write permission. Hopefully those installations (e.g. shared hosting) use SuPHP webserver protection, which makes it easier to set permissions to a useful level (even if not theoretical optimum).

Where is it available?

Tiki Permission Check is available in trunk and as standalone download. It will be available in Tiki10 (Effective 2012-10-22). Standalone version will (at least: should) work with any version of Tiki. At the bottom of this page there is an alpha version attached.

Bleeding edge code is at:

- https://svn.code.sf.net/p/tikiwiki/code/trunk/permissioncheck/
- https://svn.code.sf.net/p/tikiwiki/code/branches/12.x/permissioncheck/

- https://svn.code.sf.net/p/tikiwiki/code/branches/11.x/permissioncheck/
- https://svn.code.sf.net/p/tikiwiki/code/branches/10.x/permissioncheck/

2. Enable and Disable Tiki Permission Check

An important issue is to enable and disable Tiki Permission Check. Once you've got the information you need it is strongly recommended to disable it, because some subdirectories are set world writable for testing purposes and ownership (*user*/*group*) of files is exposed to the public (but you may use htaccess protection if possible, this depends on webserver's settings).

2.1. Usage of Tiki Permission Check with Shell

Use a shell like *sh* (or *bash*, *dash*) to run the script `prepare_permissioncheck.sh` in Tiki's document root.

2.1.1. Set Script Permissions

In Tiki's document root:
`chmod 600 prepare_permissioncheck.sh`

2.1.2. Set htaccess Permissions

If you need htaccess protection, copy `permissioncheck/_htaccess` to `permissioncheck/.htaccess`.

In Tiki's document root:
`chmod 644 permissioncheck/.htaccess`
If **.htaccess** exists but is not readable by webserver, problems may occur.

This is done each time Tiki Permission Check is enabled or disabled by shell script. So in most cases you needn't do this manually.

You cannot use htaccess password protection when TPC is enabled, because TPC uses http requests to figure out the possible models. You need htaccess password protection only if you are beyond paranoia. In this case, you may delete the complete folder `permissioncheck/` once Tiki is installed and runs. This deletion is without impact to Tiki. The shipped `.htpasswd` contains pre-encoded user `foo` with password `bar`.

2.1.3. Enable via Shell

In Tiki's document root:
`sh prepare_permissioncheck.sh enable`

2.1.4. Disable via Shell

In Tiki's document root:
`sh prepare_permissioncheck.sh disable`

2.2. Usage of Tiki Permission Check with FTP

Setting permissions without shell access is kind of annoying. You may set local file permissions and upload them or you set file permissions via FTP (in both cases: **enable**/**disable**). In addition, you need to copy `permissioncheck/yes.bin` to `permissioncheck/permission_granted.bin` to **enable** and copy `permissioncheck/no.bin` to `permissioncheck/permission_granted.bin` to **disable** Tiki Permission Check (in both cases: then FTP upload `permissioncheck/permission_granted.bin`).

2.2.1. General Settings via FTP chmod

`chmod 755 permissioncheck`
`chmod 644 permissioncheck/check.php`
`chmod 644 permissioncheck/functions.php.inc`
`chmod 600 permissioncheck/_htaccess`
`chmod 644 permissioncheck/.htaccess` if it exists
`chmod 600 permissioncheck/_htpasswd`

`chmod 600 permissioncheck/.htpasswd`
`chmod 644 permissioncheck/index.php`
`chmod 444 permissioncheck/no.bin`
`chmod 444 permissioncheck/permission_print.php.inc`
`chmod 644 permissioncheck/permission_granted.bin`
`chmod 644 permissioncheck/permission_granted.php.inc`
`chmod 644 permissioncheck/usecases.php.inc`
`chmod 644 permissioncheck/usecases.bin`
`chmod 644 permissioncheck/usecases.txt` (will become obsolete)
`chmod 444 permissioncheck/yes.bin`

Set htaccess Permissions
If Tiki Permission Check is protected by an existing **.htaccess** in *permissioncheck/*, in Tiki's document root:
`chmod 644 permissioncheck/.htaccess`
If **.htaccess** exists but is not readable by webserver, problems may occur. You don't really need `.htaccess` and Tiki Permission Check does not work properly with password protection when it is enabled.

2.2.2. Enable via FTP chmod

create `permissioncheck/new_htaccess` (with arbitrary content), copy `permissioncheck/yes.bin` to `permissioncheck/permission_granted.bin` and upload both, change file permissions at FTP server as follows:
`chmod 644 permissioncheck/create_new_htaccess.php`
`chmod 777 permissioncheck/insane`
`chmod 777 permissioncheck/insane/check.php`
`chmod 700 permissioncheck/mixed`
`chmod 660 permissioncheck/mixed/check.php`
`chmod 705 permissioncheck/morepain`
`chmod 606 permissioncheck/morepain/check.php`
`chmod 705 permissioncheck/moreworry`
`chmod 604 permissioncheck/moreworry/check.php`
`chmod 666 permissioncheck/new_htaccess`
`chmod 701 permissioncheck/pain`
`chmod 606 permissioncheck/pain/check.php`
`chmod 770 permissioncheck/paranoia`
`chmod 600 permissioncheck/paranoia/check.php`
`chmod 701 permissioncheck/paranoia-suphp`
`chmod 600 permissioncheck/paranoia-suphp/check.php`
`chmod 775 permissioncheck/risky`
`chmod 664 permissioncheck/risky/check.php`
`chmod 500 permissioncheck/sbox`
`chmod 600 permissioncheck/sbox/check.php`
`chmod 701 permissioncheck/worry`
`chmod 604 permissioncheck/worry/check.php`

2.2.3. Disable via FTP chmod

copy `permissioncheck/no.bin` to `permissioncheck/permission_granted.bin` and upload, change file permissions at FTP server as follows:
`chmod 000 permissioncheck/create_new_htaccess.php`
`chmod 700 permissioncheck/insane`
`chmod 600 permissioncheck/insane/check.php`
`chmod 700 permissioncheck/mixed`
`chmod 600 permissioncheck/mixed/check.php`
`chmod 700 permissioncheck/morepain`
`chmod 600 permissioncheck/morepain/check.php`
`chmod 700 permissioncheck/moreworry`
`chmod 600 permissioncheck/moreworry/check.php`
`chmod 600 permissioncheck/new_htaccess`
`chmod 700 permissioncheck/pain`

```
chmod 600 permissioncheck/pain/check.php
chmod 700 permissioncheck/paranoia
chmod 600 permissioncheck/paranoia/check.php
chmod 700 permissioncheck/paranoia-suphp
chmod 600 permissioncheck/paranoia-suphp/check.php
chmod 700 permissioncheck/risky
chmod 600 permissioncheck/risky/check.php
chmod 700 permissioncheck/sbox
chmod 600 permissioncheck/sbox/check.php
chmod 700 permissioncheck/worry
chmod 600 permissioncheck/worry/check.php
```

3. Permission Overview: Examples

http://example.org/permissioncheck/
http://demo.tiki.org/pd/permissioncheck/
http://demo.tiki.org/10x/permissioncheck/
http://demo.tiki.org/trunk/permissioncheck/

3.1. Use Tiki Permission Check

Visit your own Tiki installation's path /permissioncheck/ and you will see the TPC main page. You can replace the domain example.com with your own domain in the example above. When the page tells you it's disabled, you need to enable it. You should see all permission models, user, group and file permissions for those models and a hint whether this model works or not. Note the model names which seem to work, you'll need them.

4. Use Information From Tiki Permission Check

4.1. Shell Access

Go to your Tiki root directory and run the setup script with one of the model names which seem to work:
`sh setup.sh $model`
where you replace `$model` with the name you noted before. If this model still does not work, try another one. If no model works, try all predefined models one by one and see what happens.

5. Commands of setup.sh

5.1. General Commands

- default
- menu
- nothing

5.1.1. Composer

- composer

5.2. Classic Commands

- fix
- open

5.3. Predefined Models

- insane
- morepain
- moreworry
- pain
- paranoia
- paranoia-suphp

- risky
- sbox
- worry

5.4. Fine Grained Permission Bits

5.4.1. Complete Tiki Tree

- gmr
- gmw
- gmx
- gpr
- gpw
- gpx
- omr
- omw
- omx
- opr
- opw
- opx
- umr
- umw
- umx
- upr
- upw
- upx

5.4.2. Special Directories (Webserver Write Access)

- sdgmw
- sdgpw
- sdomw
- sdopw
- sdumw
- sdupw

6. Customized Use Cases

Warning: not recommended for beginners

Arbitrary use cases can be added, this is quickly and easily done by three steps:
1) Define a use case's name, add a directory with this name in subdir `permissioncheck/` and copy `permissioncheck/check.php` into the new use case subdir.
2) Define octal subdir read (default) permission, octal file read (default) permission, octal subdir write permission and octal file write permission. Add name and permissions to `permissioncheck/usecases.bin` seperated by colon. Be careful with line endings, Apple (CR) and Windows (CR+LF) are not tested yet.
3) Add the use case to `setup.sh` in Tiki's main directory. At the end of the script in main program: copy the line `php) permission_via_php_check ;;` and replace (*in the new line only*) the beginning `php)` with `name)`, where *name* is the one you've got chosen at 1) before.

7. Related Content

- https://dev.tiki.org/Permission+Check
- Server check
- htaccess
- https://dev.tiki.org/How+to+avoid+direct+access+of+a+file