Wikiprint Book

Title: Fine grained permissions

Subject: Ecopath Developer Site - TracFineGrainedPermissions

Version: 2

Date: 2020-01-18 00:30:15

Table of Contents

Fine grained permissions

Before Trac 0.11, it was only possible to define fine-grained permissions checks on the repository browser sub-system.

Since 0.11, there's a general mechanism in place that allows custom permission policy plugins to grant or deny any action on any kind of Trac resources, even at the level of specific versions of such resources.

Note that for Trac 0.12, `authz_policy` has been integrated in trunk branch as `tracopt.perm.authz_policy.*`

Permission Policies

AuthzPolicy

An example policy based on an Authz-style system has been added. See ?authz_policy.py for details (current version requires >= Python 2.4). (See also ?sample-plugins/permissions for more samples.)

- Install ?ConfigObj (required).
- Copy authz_policy.py into your plugins directory.
- Put a ?authzpolicy.conf file somewhere, preferably on a secured location on the server, not readable for others than the webuser. If the file contains non-ASCII characters, the UTF-8 encoding should be used.

  Update your `trac.ini`:

  modify the permission_policies entry in the `[trac]` section

  ```
  [trac]
  ...
  permission_policies = AuthzPolicy, DefaultPermissionPolicy, LegacyAttachmentPolicy
  ```

  add a new `[authz_policy]` section

  ```
  [authz_policy]
  authz_file = /some/trac/env/conf/authzpolicy.conf
  ```

iii. enable the single file plugin

```
[components]
...
authz_policy.* = enabled
```

Note that the order in which permission policies are specified is quite critical, as policies will be examined in the sequence provided.

A policy will return either `True`, `False` or `None` for a given permission check. Only if the return value is `None` will the *next* permission policy be consulted. If no policy explicitly grants the permission, the final result will be `False` (i.e. no permission).

For example, if the `authz_file` contains:

```
[wiki:WikiStart@*]
* = WIKI_VIEW

[wiki:PrivatePage@*]
john = WIKI_VIEW
* =
```

and the default permissions are set like this:

```
john           WIKI_VIEW
jack           WIKI_VIEW
# anonymous has no WIKI_VIEW
```

Then:

- All versions of WikiStart will be viewable by everybody (including anonymous)
- PrivatePage will be viewable only by john
- other pages will be viewable only by john and jack

mod_authz_svn-like permission policy

At the time of this writing, the old fine grained permissions system from Trac 0.10 and before used for restricting access to the repository has not yet been converted to a permission policy component, but from the user point of view, this makes little if no difference.

That kind of fine-grained permission control needs a definition file, which is the one used by Subversion's mod_authz_svn. More information about this file format and about its usage in Subversion is available in the ?Path-Based Authorization section in the Server Configuration chapter of the svn book.

Example:

```
[/]
* = r

[/branches/calc/bug-142]
harry = rw
sally = r

[/branches/calc/bug-142/secret]
harry =
```

- **/** = *Everyone has read access by default*
- **/branches/calc/bug-142** = *harry has read/write access, sally read only*
- **/branches/calc/bug-142/secret** = *harry has no access, sally has read access (inherited as a sub folder permission)*

Trac Configuration

To activate fine grained permissions you must specify the `authz_file` option in the `[trac]` section of trac.ini. If this option is set to null or not specified the permissions will not be used.

```
[trac]
authz_file = /path/to/svnaccessfile
```

If you want to support the use of the `[`*modulename*`:/`*some*`/`*path*`]` syntax within the `authz_file`, add

```
authz_module_name = modulename
```

where *modulename* refers to the same repository indicated by the `repository_dir` entry in the `[trac]` section. As an example, if the `repository_dir` entry in the `[trac]` section is `/srv/active/svn/blahblah`, that would yield the following:

```
[trac]
authz_file = /path/to/svnaccessfile
authz_module_name = blahblah
...
repository_dir = /srv/active/svn/blahblah
```

where the svn access file, `/path/to/svnaccessfile`, contains entries such as `[blahblah:/some/path]`.

Note: Usernames inside the Authz file must be the same as those used inside trac.

Subversion Configuration

The same access file is typically applied to the corresponding Subversion repository using an Apache directive like this:

```
<Location /repos>
 DAV svn
 SVNParentPath /usr/local/svn
```

```
 # our access control policy
 AuthzSVNAccessFile /path/to/svnaccessfile
</Location>
```

For information about how to restrict access to entire projects in a multiple project environment see ?wiki:TracMultipleProjectsSVNAccess

Getting TracFineGrainedPermissions to work

Don't forget to restart Trac engine to apply new configuration if you are running tracd standalone server.

See also: TracPermissions, ?TracHacks:FineGrainedPageAuthzEditorPlugin for a simple editor plugin.