Tracd

Tracd is a lightweight standalone Trac web server. In most cases it's easier to setup and runs faster than the CGI script.

Pros

- Fewer dependencies: You don't need to install apache or any other web-server.
- Fast: Should be as fast as the mod_python version (and much faster than the CGI).

Cons

- Less features: Tracd implements a very simple web-server and is not as configurable as Apache HTTPD.
- No native HTTPS support: ?sslwrap can be used instead, or ?STUNNEL.

Usage examples

A single project on port 8080. (?http://localhost:8080/)

```
$ tracd -p 8080 /path/to/project
```

With more than one project. (?http://localhost:8080/project1/ and ?http://localhost:8080/project2/)

```
$ tracd -p 8080 /path/to/project1 /path/to/project2
```

You can't have the last portion of the path identical between the projects since that's how trac keeps the URLs of the different projects unique. So if you use /project1/path/to and /project2/path/to, you will only see the second project.

Using Authentication

Tracd provides support for both Basic and Digest authentication. The default is to use Digest; to use Basic authentication, replace `--auth` with `--basic-auth` in the examples below, and omit the realm.

If the file `/path/to/users.htdigest` contain user accounts for project1 with the realm "mycompany.com", you'd use the following command-line to start tracd:

```
$ tracd -p 8080 --auth project1,/path/to/users.htdigest,mycompany.com /path/to/project1
```

Note that the project ?name? passed to the `--auth` option is actually the base name of the project environment directory.""

Of course, the digest file can be be shared so that it is used for more than one project:

```
$ tracd -p 8080 \
  --auth project1,/path/to/users.htdigest,mycompany.com \
  --auth project2,/path/to/users.htdigest,mycompany.com \
  /path/to/project1 /path/to/project2
```

Generating Passwords Without Apache

If you don't have Apache available, you can use this simple Python script to generate your passwords:

```
from optparse import OptionParser
import md5

# build the options
usage = "usage: %prog [options]"
parser = OptionParser(usage=usage)
parser.add_option("-u", "--username",action="store", dest="username", type = "string",
                  help="the username for whom to generate a password")
parser.add_option("-p", "--password",action="store", dest="password", type = "string",
                  help="the password to use")
(options, args) = parser.parse_args()

# check options
if (options.username is None) or (options.password is None):
   parser.error("You must supply both the username and password")

# Generate the string to enter into the htdigest file
realm = 'trac'
kd = lambda x: md5.md5(':'.join(x)).hexdigest()
print ':'.join((options.username, realm, kd([options.username, realm, options.password])))
```

See also: TracInstall, TracCgi, TracModPython, TracGuide