Trac with FastCGI

Since version 0.9, Trac supports being run through the ?FastCGI interface. Like mod_python, this allows Trac to remain resident, and is faster than external CGI interfaces which must start a new process for each request. However, unlike mod_python, it is able to support ?SuEXEC. Additionally, it is supported by much wider variety of web servers.

Note for Windows: Trac's FCGI does not run under Windows, as Windows does not implement Socket.fromfd, which is used by _fcgi.py

Simple Apache configuration

There are two FastCGI modules commonly available for Apache: `mod_fastcgi` and `mod_fcgid`. The `FastCgiIpcDir` and `FastCgiConfig` directives discussed below are `mod_fastcgi` directives; the `DefaultInitEnv` is a `mod_fcgid` directive.

For `mod_fastcgi`, add the following to an appropriate Apache configuration file:

```
# Enable fastcgi for .fcgi files
# (If you're using a distro package for mod_fcgi, something like
# this is probably already present)
<IfModule mod_fastcgi.c>
   AddHandler fastcgi-script .fcgi
   FastCgiIpcDir /var/lib/apache2/fastcgi
</IfModule>
LoadModule fastcgi_module /usr/lib/apache2/modules/mod_fastcgi.so
```

Setting `FastCgiIpcDir` is optional if the default is suitable. Note that the `LoadModule` line must be after the `IfModule` group.

Configure `ScriptAlias` or similar options as described in TracCgi, but calling `trac.fcgi` instead of `trac.cgi`.

You can set up the `TRAC_ENV` as an overall default:

```
FastCgiConfig -initial-env TRAC_ENV=/path/to/env/trac
```

Or you can serve multiple Trac projects in a directory like:

```
FastCgiConfig -initial-env TRAC_ENV_PARENT_DIR=/parent/dir/of/projects
```

But neither of these will work for `mod_fcgid`. A similar but partial solution for `mod_fcgid` is:

```
DefaultInitEnv TRAC_ENV /path/to/env/trac/
```

But this cannot be used in `Directory` or `Location` context, which makes it difficult to support multiple projects.

A better method which works for both of these modules (and for ?lighttpd and CGI as well), because it involves no server configuration settings for environment variables, is to set one of the variables in `trac.fcgi`, e.g.:

```
import os
os.environ['TRAC_ENV'] = "/path/to/projectenv"
```

or

```
import os
os.environ['TRAC_ENV_PARENT_DIR'] = "/path/to/project/parent/dir"
```

Using this method, different projects can be supported by using different `.fcgi` scripts with different `ScriptAliases`, copying and appropriately renaming `trac.fcgi` and adding the above code to create each such script.

See ?this fcgid example config which uses a ScriptAlias directive with trac.fcgi with a trailing / like this:

```
ScriptAlias / /srv/tracsite/cgi-bin/trac.fcgi/
```

Simple Cherokee Configuration

Configuration wanted.

Simple Lighttpd Configuration

The FastCGI front-end was developed primarily for use with alternative webservers, such as ?lighttpd.

lighttpd is a secure, fast, compliant and very flexible web-server that has been optimized for high-performance environments. It has a very low memory footprint compared to other web servers and takes care of CPU load.

For using `trac.fcgi` with lighttpd add the following to your lighttpd.conf:

```
fastcgi.server = ("/trac" =>
                    ("trac" =>
                      ("socket" => "/tmp/trac-fastcgi.sock",
```

```
                    "bin-path" => "/path/to/cgi-bin/trac.fcgi",
                    "check-local" => "disable",
                    "bin-environment" =>
                      ("TRAC_ENV" => "/path/to/projenv")
                   )
                  )
                 )
```

Note that you will need to add a new entry to `fastcgi.server` for each separate Trac instance that you wish to run. Alternatively, you may use the `TRAC_ENV_PARENT_DIR` variable instead of `TRAC_ENV` as described above, and you may set one of the two in `trac.fcgi` instead of in `lighttpd.conf` using `bin-environment` (as in the section above on Apache configuration).

For using two projects with lighttpd add the following to your `lighttpd.conf`:

```
fastcgi.server = ("/first" =>
                   ("first" =>
                    ("socket" => "/tmp/trac-fastcgi-first.sock",
                     "bin-path" => "/path/to/cgi-bin/trac.fcgi",
                     "check-local" => "disable",
                     "bin-environment" =>
                       ("TRAC_ENV" => "/path/to/projenv-first")
                    )
                   ),
                   "/second" =>
                     ("second" =>
                     ("socket" => "/tmp/trac-fastcgi-second.sock",
                      "bin-path" => "/path/to/cgi-bin/trac.fcgi",
                      "check-local" => "disable",
                      "bin-environment" =>
                        ("TRAC_ENV" => "/path/to/projenv-second")
                     )
                   )
                 )
```

Note that field values are different. If you prefer setting the environment variables in the `.fcgi` scripts, then copy/rename `trac.fcgi`, e.g., to `first.fcgi` and `second.fcgi`, and reference them in the above settings. Note that the above will result in different processes in any event, even if both are running from the same `trac.fcgi` script.

Note from c00i90wn: It's very important the order on which server.modules are loaded, if mod_auth is not loaded **BEFORE** mod_fastcgi, then the server will fail to authenticate the user.

For authentication you should enable mod_auth in lighttpd.conf 'server.modules', select auth.backend and auth rules:

```
server.modules              = (
...
  "mod_auth",
...
)

auth.backend               = "htpasswd"

# Separated password files for each project
# See "Conditional Configuration" in
# http://trac.lighttpd.net/trac/file/branches/lighttpd-merge-1.4.x/doc/configuration.txt

$HTTP["url"] =~ "^/first/" {
  auth.backend.htpasswd.userfile = "/path/to/projenv-first/htpasswd.htaccess"
}
$HTTP["url"] =~ "^/second/" {
  auth.backend.htpasswd.userfile = "/path/to/projenv-second/htpasswd.htaccess"
}

# Enable auth on trac URLs, see
# http://trac.lighttpd.net/trac/file/branches/lighttpd-merge-1.4.x/doc/authentication.txt

auth.require = ("/first/login" =>
                ("method"  => "basic",
                 "realm"   => "First project",
                 "require" => "valid-user"
                ),
                "/second/login" =>
                ("method"  => "basic",
                 "realm"   => "Second project",
                 "require" => "valid-user"
                )
               )
```

Note that lighttpd (I use version 1.4.3) stopped if password file doesn't exist.

Note that lighttpd doesn't support 'valid-user' in versions prior to 1.3.16.

Conditional configuration is also useful for mapping static resources, i.e. serving out images and CSS directly instead of through FastCGI:

```
# Aliasing functionality is needed
server.modules += ("mod_alias")

# Setup an alias for the static resources
alias.url = ("/trac/chrome/common" => "/usr/share/trac/htdocs")

# Use negative lookahead, matching all requests that ask for any resource under /trac, EXCEPT
# /trac/chrome/common, and use FastCGI for those
$HTTP["url"] =~ "^/trac(?!/chrome/common)" {
# Even if you have other fastcgi.server declarations for applications other than Trac, do NOT
fastcgi.server = ("/trac" =>
                   ("trac" =>
                     ("socket" => "/tmp/trac-fastcgi.sock",
                      "bin-path" => "/path/to/cgi-bin/trac.fcgi",
                      "check-local" => "disable",
                      "bin-environment" =>
                        ("TRAC_ENV" => "/path/to/projenv")
                     )
                   )
                 )
}
```

The technique can be easily adapted for use with multiple projects by creating aliases for each of them, and wrapping the fastcgi.server declarations inside conditional configuration blocks. Also there is another way to handle multiple projects and it's to use TRAC_ENV_PARENT_DIR instead of TRAC_ENV and use global auth, let's see an example:

```
#  This is for handling multiple projects
  alias.url       = ( "/trac/" => "/path/to/trac/htdocs/" )

  fastcgi.server += ("/projects"  =>
                      ("trac" =>
                        (
                          "socket" => "/tmp/trac.sock",
                          "bin-path" => "/path/to/cgi-bin/trac.fcgi",
                          "check-local" => "disable",
                          "bin-environment" =>
                            ("TRAC_ENV_PARENT_DIR" => "/path/to/parent/dir/of/projects/" )
                        )
                      )
                    )
#And here starts the global auth configuration
  auth.backend = "htpasswd"
  auth.backend.htpasswd.userfile = "/path/to/unique/htpassword/file/trac.htpasswd"
  $HTTP["url"] =~ "^/projects/.*/login$" {
    auth.require = ("/" =>
                     (
```

```
                        "method"  => "basic",
                        "realm"   => "trac",
                        "require" => "valid-user"
                      )
                    )
  }
```

Changing date/time format also supported by lighttpd over environment variable LC_TIME

```
fastcgi.server = ("/trac" =>
                    ("trac" =>
                      ("socket" => "/tmp/trac-fastcgi.sock",
                       "bin-path" => "/path/to/cgi-bin/trac.fcgi",
                       "check-local" => "disable",
                       "bin-environment" =>
                         ("TRAC_ENV" => "/path/to/projenv",
                         "LC_TIME" => "ru_RU")
                      )
                    )
                  )
```

For details about languages specification see TracFaq? question 2.13.

Other important information like ?this updated TracInstall page, and this are useful for non-fastcgi specific installation aspects.

If you use trac-0.9, read ?about small bug

Relaunch lighttpd, and browse to `http://yourhost.example.org/trac` to access Trac.

Note about running lighttpd with reduced permissions:

> If nothing else helps and trac.fcgi doesn't start with lighttpd settings server.username = "www-data", server.groupname = "www-data", then in the `bin-environment` section set `PYTHON_EGG_CACHE` to the home directory of `www-data` or some other directory accessible to this account for writing.

Simple LiteSpeed? Configuration

The FastCGI front-end was developed primarily for use with alternative webservers, such as ?LiteSpeed.

LiteSpeed? web server is an event-driven asynchronous Apache replacement designed from the ground-up to be secure, scalable, and operate with minimal resources. LiteSpeed? can operate directly from an Apache config file and is targeted for business-critical environments.

Setup

1) Please make sure you have first have a working install of a Trac project. Test install with ?tracd? first.

2) Create a Virtual Host for this setup. From now on we will refer to this vhost as TracVhost?. For this tutorial we will be assuming that your trac project will be accessible via:

```
http://yourdomain.com/trac/
```

3) Go ?TracVhost? ? External Apps? tab and create a new ?External Application?.

```
Name: MyTracFCGI
Address: uds://tmp/lshttpd/mytracfcgi.sock
Max Connections: 10
Environment: TRAC_ENV=/fullpathto/mytracproject/ <--- path to root folder of trac project
Initial Request Timeout (secs): 30
Retry Timeout (secs): 0
Persistent Connection   Yes
Connection Keepalive Timeout: 30
Response Bufferring: No
Auto Start: Yes
Command: /usr/share/trac/cgi-bin/trac.fcgi  <--- path to trac.fcgi
Back Log: 50
Instances: 10
```

4) Optional. If you need to use htpasswd based authentication. Go to ?TracVhost? ? Security? tab and create a new security ?Realm?.

```
DB Type: Password File
Realm Name: MyTracUserDB               <--- any name you wish and referenced later
User DB Location: /fullpathto/htpasswd <--- path to your htpasswd file
```

If you don?t have a htpasswd file or don?t know how to create the entries within one, go to ?http://sherylcanter.com/encrypt.php, to generate the user:password combos.

5) Go to ?PythonVhost? ? Contexts? and create a new ?FCGI Context?.

```
URI: /trac/                              <--- URI path to bind to python fcgi app we created
Fast CGI App: [VHost Level] MyTractFCGI  <--- select the trac fcgi extapp we just created
```

```
Realm: TracUserDB                        <--- only if (4) is set. select realm created in (4)
```

6) Modify /fullpathto/mytracproject/conf/trac.ini

```
#find/set base_rul, url, and link variables
base_url = http://yourdomain.com/trac/ <--- base url to generate correct links to
url = http://yourdomain.com/trac/      <--- link of project
link = http://yourdomain.com/trac/     <--- link of graphic logo
```

7) Restart LiteSpeed?, ?lswsctrl restart?, and access your new Trac project at:

```
http://yourdomain.com/trac/
```

See also TracCgi, TracModPython, TracInstall, TracGuide