Trac と mod_python

Trac では mod_python を利用可能です。 mod_python は Trac のレスポンスタイムを飛躍的に向上し、特に CGI と比べて、 tracd/mod_proxy では使用できない多くの Apache 機能を使えるようにします。

以下の説明は Apache2 のためのものです; まだ Apache1.3 を使用しているなら、 TracModPython2.7 にいくつか情報があります。

警告

2010 年 6 月 16 日に、 mod_python プロジェクトが正式に終了しました。もし mod_python を新しいインストールで使用することを考えているならば、 お願いだからしないで下さい!
解決されない既知の課題がありますし、今ではより良い代替手段もあります。詳細については、インストールしようとしているバージョンの TracInstall ページをチェックして下さい。

シンプルなコンフィグレーション

mod_python をインストールしたら、 Apache の設定ファイルに以下の一行を追加してモジュールをロードしなければなりません:

```
LoadModule python_module modules/mod_python.so
```

Note: モジュールがインストールされている正しいパスは HTTPD をどこにインストールしたかによって変わります。

Debian で apt-get を使用する場合

```
apt-get install libapache2-mod-python libapache2-mod-python-doc
```

(Debian の続き) mod_python をインストールした後に、apache2 (上の Load Module に相当するもの) のモジュールを有効にしなければなりません:

```
a2enmod python
```

Fedora で yum を使用する場合:

```
yum install mod_python
```

httpd.conf に以下を加えることで、 mod_python がインストールされたかテストすることができます。セキュリティ上の理由から、テストが終わった時点で以下のコンフィグは削除するべきです。
Note: mod_python.testhandler は mod_python 3.2+ で利用可能です。

```
<Location /mpinfo>
  SetHandler mod_python
  PythonInterpreter main_interpreter
  PythonHandler mod_python.testhandler
  Order allow,deny
  Allow from all
</Location>
```

mod_python を使用した簡単な Trac のセットアップ方法は以下のようになります:

```
<Location /projects/myproject>
  SetHandler mod_python
  PythonInterpreter main_interpreter
  PythonHandler trac.web.modpython_frontend
  PythonOption TracEnv /var/trac/myproject
  PythonOption TracUriRoot /projects/myproject
  Order allow,deny
  Allow from all
</Location>
```

TracUriRoot オプションは不要な場合もあります。 TracUriRoot オプションを付けずに試し、 Trac が正しく URL を生成できないか、 "No handler matched request to..." というエラーが出るようであれば TracUriRoot を追加して下さい。 Location と TracUriRoot が同じパスになるようにしてください。

PythonOption の一覧は以下の通りです。

```
    # For a single project
   PythonOption TracEnv /var/trac/myproject
   # For multiple projects
   PythonOption TracEnvParentDir /var/trac/myprojects
   # For the index of multiple projects
   PythonOption TracEnvIndexTemplate /srv/www/htdocs/trac/project_list_template.html
   # A space delimitted list, with a "," between key and value pairs.
   PythonOption TracTemplateVars key1,val1 key2,val2
   # Useful to get the date in the wanted order
   PythonOption TracLocale en_GB.UTF8
   # See description above
   PythonOption TracUriRoot /projects/myproject
```

Python Egg Cache

Genshi のように圧縮された Python egg は通常、実行するユーザのホームディレクトリ配下の .python-eggs ディレクトリに展開されます。 Apache のホームディレクトリは多くの場合、書き込みできないようになっているので、他のディレクトリを egg cache として指定しなければなりません:

```
PythonOption PYTHON_EGG_CACHE /var/trac/myprojects/egg-cache
```

または Genshi の egg を解凍して展開することで、この問題を回避できます。

認証設定

パスワードファイルを作成して、認証を構成するには CGI と同じように行います。

```
<Location /projects/myproject/login>
 AuthType Basic
 AuthName "myproject"
 AuthUserFile /var/trac/myproject/.htpasswd
 Require valid-user
</Location>
```

Apache の mod_ldap 認証のコンフィグは少し扱いにくいです。(httpd 2.2.x と OpenLDAP: slapd 2.3.19)

1. Apache の httpd.conf に以下のモジュールをロードする必要があります

```
LoadModule ldap_module modules/mod_ldap.so
LoadModule authnz_ldap_module modules/mod_authnz_ldap.so
```

1. httpd.conf は以下のような感じになります:

```
<Location /trac/>
 SetHandler mod_python
 PythonInterpreter main_interpreter
 PythonHandler trac.web.modpython_frontend
 PythonOption TracEnv /home/trac/
 PythonOption TracUriRoot /trac/
 Order deny,allow
 Deny from all
 Allow from 192.168.11.0/24
 AuthType Basic
 AuthName "Trac"
```

```
 AuthBasicProvider "ldap"
 AuthLDAPURL "ldap://127.0.0.1/dc=example,dc=co,dc=ke?uid?sub?(objectClass=inetOrgPerson)"
 authzldapauthoritative Off
 require valid-user
</Location>
```

Microsoft Active Directory の LDAP インタフェースを使用する場合:

```
<Location /trac/>
 SetHandler mod_python
 PythonInterpreter main_interpreter
 PythonHandler trac.web.modpython_frontend
 PythonOption TracEnv /home/trac/
 PythonOption TracUriRoot /trac/
 Order deny,allow
 Deny from all
 Allow from 192.168.11.0/24
 AuthType Basic
 AuthName "Trac"
 AuthBasicProvider "ldap"
 AuthLDAPURL "ldap://adserver.company.com:3268/DC=company,DC=com?sAMAccountName?sub?(objectClass=user)"
 AuthLDAPBindDN       ldap-auth-user@company.com
 AuthLDAPBindPassword "the_password"
 authzldapauthoritative Off
 # require valid-user
 require ldap-group CN=Trac Users,CN=Users,DC=company,DC=com
</Location>
```

Note 1: このケースでは LDAP 検索で複数の OU をまとめて取得するために、 AD のグローバルカタログサーバ (Global Catalog Server) に接続しています (ポート番号が通常 LDAP で使用される 389 ではなく 3268 であることに注意してください) 。 GCS は基本的に "平らな" ツリーであり、ユーザが、どの OU に属するか不明な場合でも検索することができます。

Note 2: Active Directory は、レコードにアクセスするために user/password (AuthLDAPBindDN と AuthLDAPBindPassword) による認証を必要とします。 (訳注: GC ではなく、通常の LDAP であれば Active Directory の ACL に認証なしユーザからの読み取り許可を設定すれば不要です)

Note 3: "require ldap-group ..." ディレクティブはメンバのアクセスが許可されている AD のグループを指定します。

Python Egg Cache を設定する

Web サーバが Egg Cache に書き込みできない場合、パーミッションを変更するか、 Apache が書き込み可能な場所を指定する必要があります。設定しないと 500 Internal Server Error や syslog へのエラー出力が発生します。

```
<Location /projects/myproject>
 ...
 PythonOption PYTHON_EGG_CACHE /tmp
 ...
</Location>
```

PythonPath を設定する

もし Trac のインストールが、通常の Python ライブラリのパスの中に無い場合、 Apache が Trac の mod_python ハンドラを見つけられるように PythonPath ディレクティブで指定しなければなりません:

```
<Location /projects/myproject>
 ...
 PythonPath "sys.path + ['/path/to/trac']"
 ...
</Location>
```

PythonPath ディレクティブを使用するときは気をつけてください。そして、 SetEnv PYTHONPATH は動かないので 使用しない で下さい。

マルチプロジェクトのセットアップ

Trac の mod_python ハンドラには Subversion の SvnParentPath とよく似た TracEnvParentDir というコンフィグレーションオプションがあります。

```
<Location /projects>
 SetHandler mod_python
 PythonInterpreter main_interpreter
 PythonHandler trac.web.modpython_frontend
 PythonOption TracEnvParentDir /var/trac
 PythonOption TracUriRoot /projects
</Location>
```

/projects の URL をリクエストすると、 TracEnvironment の親ディレクトリ TracEnvParentDir として設定したディレクトリ配下のサブディレクトリー覧が表示されます。その一覧から何かプロジェクトを選択するとそれに該当する TracEnvironment を開くことができます。

あなたのプロジェクトのホームページとして、サブディレクトリのリストが必要ないならば、以下のようにすることができます

```
<LocationMatch "/.+/">
```

これは DocumentRoot フォルダの直下にカスタムホームページとして配置されていない場合には、すべてのロケーションで代わりに mod_python を使用することを Apache に教えます。

すべてのプロジェクトに対して、 <LocationMatch> ディレクティブを使用することによって同じ認証の仕組みを使用することができます。

```
<LocationMatch "/projects/[^/]+/login">
 AuthType Basic
 AuthName "Trac"
 AuthUserFile /var/trac/.htpasswd
 Require valid-user
</LocationMatch>
```

仮想ホストの設定

以下に示す例は Trac を仮想サーバーとしてセットアップするときに必要な設定です。 (例えば、http://trac.mycompany.com といった URL でアクセスすることができます):

```
<VirtualHost * >
   DocumentRoot /var/www/myproject
   ServerName trac.mycompany.com
   <Location />
       SetHandler mod_python
       PythonInterpreter main_interpreter
       PythonHandler trac.web.modpython_frontend
       PythonOption TracEnv /var/trac/myproject
       PythonOption TracUriRoot /
   </Location>
   <Location /login>
       AuthType Basic
       AuthName "MyCompany Trac Server"
       AuthUserFile /var/trac/myproject/.htpasswd
       Require valid-user
   </Location>
</VirtualHost>
```

この設定は全てのケースでうまく動くわけではありません。動かない場合は以下を試してください:

- <Location> の代わりに <LocationMatch> を使用する。

- <Location /> はサーバの設定によっては、単にサーバのルートではなく完全なホスト名を参照していることがあります。このような場合、(上記の例では下段にあたるログイン用ディレクトリを含む) 全てのリクエストが Python に送信され、認証が動かなくなります (認証を行おうとすると、認証が設定されていないというエラー画面が表示されます)。 URL を変更できるのであれば (/, /login の代わりに /web/, /web/login などのように) ルートではなくサブディレクトリを使ってみてください。
- Apache の `NameVirtualHost` を設定している場合、 `<VirtualHost *>` ではなく `<VirtualHost *:80>` を使用せねばならないかもしれません。

複数のプロジェクトをサポートする仮想ホストの設定では、 "TracEnv" /var/trac/myproject を "TracEnvParentDir" /var/trac/ に置き換えて下さい。

Note: DocumentRoot は TracEnvironment と同じディレクトリにしないでください。 何かのバグがあった場合に TracEnvironment の内容が外部からアクセス可能になってしまうおそれがあります。

トラブルシューティング

サーバエラーのページがでたときには、 まずは Apache のエラーログを確認するか、 `PythonDebug` オプションを有効にして下さい:

```
<Location /projects/myproject>
 ...
 PythonDebug on
</Location>
```

複数プロジェクトの場合は、全てのプロジェクトでサーバを再起動してみてください。

Expat-related のセグメンテーションフォルト

この問題は Unix 上で Python 2.4 を使用するとき、ほぼ確実に発生します。 Python 2.4 の使用する Expat (C で書かれた XML パーザライブラリ) と Apache の使用する Expat のバージョンが異なる場合に、セグメンテーションフォルトが発生します。 Trac 0.11 は Genshi (間接的に Expat が使用される) を使用しているため、以前 Trac 0.10 で正常に動いていたとしても、現在のあなたの環境で問題が起こり得ます。

Graham Dumpleton が、この問題について詳しく書いています。問題の 説明と回避方法 を確認してください。

フォームを送信するときの問題

もし、 Trac で何かしらのフォームを送信したときに、トラブルに見舞われたら (送信後にスタートページにリダイレクトされてしまう、などがよくある問題です) `DocumentRoot` の中に mod_python をマッピングしたパスと同じフォルダやファイルが存在しないか確認してください。どういうわけか、 mod_python は静的リソースと同じところにマッピングされると混乱してしまいます。

仮想ホストの設定においての問題

<Location /> ディレクティブが使用されている場合に `DocumentRoot` を設定すると 403 (Forbidden) エラーになることがあります。 `DocumentRoot` ディレクティブを削除するか、アクセスが許されているディレクトリに設定されているかどうかを確認して下さい (対応する `<Directory>` ブロックにて)

<Location /> で `SetHandler` を使用すると、すべてを mod_python でハンドルすることになりますが、いかなる CSS も image/icons もダウンロードできなくなります。この問題を回避するために、われわれは <Location /trac> で `SetHandler None` を使用しています。しかし、この方法がエレガントな解決方法だとは思っていません。

Zip された egg での問題

mod_python のバージョンによっては Zip された egg ファイルからモジュールを import できないことがあります。 Apache のログに `ImportError: No module named trac` が出力される場合、問題が発生している原因であると考えられます。 Python の site-packages ディレクトリを見てみてください; Trac のモジュールが ディレクトリ ではなく ファイル として配置されている場合、問題の原因と考えられます。解決するためには、下記の上に `--always-unzip` オプションと伴に Trac をインストールしてみてください。

```
easy_install --always-unzip Trac-0.12.zip
```

.htaccess ファイルを使用する

ディレクトリの設定をほんのちょっと修正するには .htaccess ファイルを使用すればいいかもしれませんが、これは動作しません。 Apache が Trac の URL に "/"（スラッシュ）を追加すると、正しい動作を妨げてしまいます。

それでは、 mod_rewrite を使用すればいいように見えますが、これも動作しません。とにかく、百害あって一利なしです。指示に従ってください。 :)

成功した事例: 以下の設定値で成功した事例があります:

```
SetHandler mod_python
PythonInterpreter main_interpreter
PythonHandler trac.web.modpython_frontend
PythonOption TracEnv /system/path/to/this/directory
PythonOption TracUriRoot /path/on/apache

AuthType Basic
AuthName "ProjectName"
AuthUserFile /path/to/.htpasswd
Require valid-user
```

TracUriRoot は Web ブラウザが取得する Trac のパスを明示的に設定するのに使用します。 (例: domain.tld/projects/trac)

.htaccess 使用時の特記事項

.htaccess を使用している場合、 Trac のディレクトリが他のディレクトリで設定されたた .htaccess
ディレクティブを継承し、問題を生じることがあります。このような場合、以下のように .htaccess ファイルに設定してみて下さい:

```
<IfModule mod_rewrite.c>
 RewriteEngine Off
</IfModule>
```

Win32 での特記

Windows 上で mod_python 3.2 より前のバージョンで Trac を動かしている場合、添付ファイルのアップロードが 動かない でしょう。この問題は 3.1.4 以降で解決されました。 mod_python をアップグレードしてこの問題を解決してください。

OS X での特記

OS X で mod_python を使用するとき、 apachectl restart コマンドで Apache の再起動ができないでしょう。これは、 mod_python 3.2 でおそらく修正されるでしょう。しかし、 ここ にあるパッチを適用すれば、 3.2 以前のバージョンでもこの問題を回避できます。

SELinux での特記

もし、 Trac が Cannot get shared lock on db.lock というようなメッセージが出力したら、 リポジトリに
セキュリティコンテキストを設定する必要があるでしょう:

```
chcon -R -h -t httpd_sys_content_t PATH_TO_REPOSITORY
```

http://subversion.tigris.org/faq.html#reposperms も参考にして下さい

FreeBSD での特記

mod_python と sqlite パッケージのインストールバージョンに注意して下さい。 Ports
には両パッケージともいろいろなバージョンがありますが、初期の pysqlite と mod_python は組み合わせることができません。前者は python のスレッド機能サポートが必要ですし、 後者 は python のスレッド機能なしのインストールが必要です。

apache2 を普通にコンパイルしてインストールした場合、 apache はスレッドのサポートなしになります（これが FreeBSD 上であまりよく動かない原因)。 --enable-threads を使用して ./configure を実行することで apache にスレッドのサポートありにすることができますが、これはお勧めできません。 最良のオプションは /usr/local/apache2/bin/ennvars に下記の一行を追加することだと 考えられます。

```
export LD_PRELOAD=/usr/lib/libc_r.so
```

Subversion での特記

コマンドラインや TracStandalone で使用しているときは動くのに、 mod_python を使用しているときのみ、 `Unsupported version control system "svn"` というエラーが出力される場合、 PythonPath ディレクティブに Python bindings
へのパスを追加するのを忘れている可能性があります。 (Python の `site-packages` ディレクトリに Python binding へのリンクを追加するか、`.pth` ファイルを作っておくのがベターです。)

これに当てはまらない場合、使用している Subversion のライブラリが Apache が使用しているバージョンと適合性がないかもしれません。 (たいていは `apr` ライブラリの不適合性が原因になります。) その場合、 Apache の svn モジュール (`mod_dav_svn`) も使用できないでしょう。

また、ランタイムエラー (`argument number 2: a 'apr_pool_t *' is expected`)
を抑止するためにも、複数のサブインタプリタを使用できる最近のバージョンの `mod_python` が必要になります。 3.2.8 では たぶん 動きますが、#3371 に記述されている通り、メインインタプリタを使用するように強制するワークアラウンドを使用する方がおそらく良いでしょう:

```
PythonInterpreter main_interpreter
```

これは、よく知られている mod_python と Subversion の Python バインディングの他の問題 (#2611, #3455)
について推奨しているワークアラウンドです。 #3455 Graham Dumpleton のコメントに問題点が指摘されています。

ページレイアウトの問題

Trac のページフォーマットが奇妙に見えるなら、ページレイアウトを管理するスタイルシートが Web
サーバによって適切に扱われていない可能性が考えられます。 Apache のコンフィグに以下を追加してみてください:

```
Alias /myproject/css "/usr/share/trac/htdocs/css"
<Location /myproject/css>
   SetHandler None
</Location>
```

Note: 上記のコンフィグが効果を発揮するためには、プロジェクトの root 位置のコンフィグ後に追加しなければなりません。つまり `<Location /myproject />` 以降です。

また `PythonOptimize On`
が設定されている場合、ページのヘッダとフッタの表示が乱れたり、マクロやプラグインのドキュメンテーションが表示されないことがあります (#8956 参照)。オプションの設定によって影響を受ける箇所について充分考慮できない場合は `Off` に設定する方がよいでしょう。

HTTPS の問題

Trac を完全に https で実行したいにも関わらず、プレーンな http にリダイレクトされる場合、 Apache のコンフィグに以下を追加してください:

```
<VirtualHost * >
   DocumentRoot /var/www/myproject
   ServerName trac.mycompany.com
   SetEnv HTTPS 1
   ....
</VirtualHost>
```

Fedora 7 の問題

必ず 'python-sqlite2' をインストールしてください。 TracModPython では必須です (tracd では必須ではありません)。

php5-mhash または その他の php5 モジュールのセグメンテーションフォルト

php5-mhash モジュールがインストールされている場合、 (debian etch について報告された) セグメンテーションフォルトに遭遇するでしょう。
php-mhash を削除して、問題が解決するかを確かめてみてください。 debian のバグレポート
http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=411487 を参考にして下さい。

システムライブラリの代わりに、サードパーティのライブラリでコンパイルされた php5
を使用する一部の人々にもトラブルが発生します。ここを確認してください
http://www.djangoproject.com/documentation/modpython/#if-you-get-a-segmentation-fault

---

See also: TracGuide, TracInstall, ModWSGI, FastCGI, TracNginxRecipe