pyramid_auth Documentation

Release 0.4

Aurélien Matouillot

April 02, 2015

Contents

1 Introduction 1

2 Cookie policy 3
- 2.1 Installation 3
- 2.2 Configuration 3

3 Remote_user policy 7
- 3.1 Installation 7
- 3.2 Configuration 7

4 ldap policy 9
- 4.1 Installation 9
- 4.2 Configuration 9

CHAPTER 1

Introduction

This is a plugin for pyramid which provides a simple authentication system. The idea was to use exising authentication's policies to provide multiple support. Currently this plugin support cookie, remote_user and ldap policies.

By default the cookie and ldap policies generate the form and the urls automatically:

- /login: display the login form
- /logout: logout the user
- /forbidden: the user is redirected to this page when he is logged but doesn't have the right permission to see a page.

If you want to generate your own urls you can set the following parameter

`pyramid_auth.no_routes`

If set in your config, no routes will be added automatically. It's usefull when you use an API for authentication.

Note: If you want to change the rendering of the template to include your design you can:

- Create a template in the folder templates/auth of your project named base.mak. Each templates (login, forbidden) inherit from it.
- Create the login.mak and/or forbidden.mak templates in the folder templates/auth to overwrite the default ones.

CHAPTER 2

Cookie policy

This policy uses pyramid.authentication.AuthTktAuthenticationPolicy. When a user wants to login, it displays a login/password form to process to the authentication.

2.1 Installation

In your .ini file add *pyramid_auth* to *pyramid.includes* like this:

```
pyramid.includes =
    pyramid_auth
    ...
```

Also you need to add *pyramid_auth* in *setup.py* in install_requires:

```
install_requires=[
    ...
    'pyramid_auth'
]
```

2.2 Configuration

You need to set some options in your .ini file. See this example for the required ones:

```
pyramid_auth.policy = cookie
pyramid_auth.cookie.secret = mysecret
pyramid_auth.cookie.validate_function = validate_function
```

2.2.1 Options

pyramid_auth.cookie.validate_function

Function to validate the credential. It can make some call in your DB or make some static verification. Here is a small example:

```
def validate(request, login, password):
    if login == 'Bob' and password == 'bobpwd':
        return True
    return False
```

Required.

`pyramid_auth.cookie.secret`

The secret (a string) used for auth_tkt cookie signing. Required.

`pyramid_auth.cookie.callback`

Default: `None`. A callback passed the userid and the request, expected to return `None` if the userid doesn't exist or a sequence of principal identifiers (possibly empty) if the user does exist. If `callback` is `None`, the userid will be assumed to exist with no principals. Optional.

`pyramid_auth.cookie.cookie_name`

Default: `auth_tkt`. The cookie name used (string). Optional.

`pyramid_auth.cookie.secure`

Default: `False`. Only send the cookie back over a secure conn. Optional.

`pyramid_auth.cookie.include_ip`

Default: `False`. Make the requesting IP address part of the authentication data in the cookie. Optional.

For IPv6 this option is not recommended. The `mod_auth_tkt` specification does not specify how to handle IPv6 addresses, so using this option in combination with IPv6 addresses may cause an incompatible cookie. It ties the authentication ticket to that individual's IPv6 address.

`pyramid_auth.cookie.timeout`

Default: `None`. Maximum number of seconds which a newly issued ticket will be considered valid. After this amount of time, the ticket will expire (effectively logging the user out). If this value is `None`, the ticket never expires. Optional.

`pyramid_auth.cookie.reissue_time`

Default: `None`. If this parameter is set, it represents the number of seconds that must pass before an authentication token cookie is automatically reissued as the result of a request which requires pyramid_auth. The duration is measured as the number of seconds since the last auth_tkt cookie was issued and 'now'. If this value is `0`, a new ticket cookie will be reissued on every request which requires authentication.

A good rule of thumb: if you want auto-expired cookies based on inactivity: set the `timeout` value to 1200 (20 mins) and set the `reissue_time` value to perhaps a tenth of the `timeout` value (120 or 2 mins). It's nonsensical to set the `timeout` value lower than the `reissue_time` value, as the ticket will never be reissued if so. However, such a configuration is not explicitly prevented.

Optional.

`pyramid_auth.cookie.max_age`

Default: `None`. The max age of the auth_tkt cookie, in seconds. This differs from `timeout` inasmuch as `timeout` represents the lifetime of the ticket contained in the cookie, while this value represents the lifetime of the cookie itself. When this value is set, the cookie's `Max-Age` and `Expires` settings will be set, allowing the auth_tkt cookie to last between browser sessions. It is typically nonsensical to set this to a value that is lower than `timeout` or `reissue_time`, although it is not explicitly prevented. Optional.

`pyramid_auth.cookie.path`

Default: `/`. The path for which the auth_tkt cookie is valid. May be desirable if the application only serves part of a domain. Optional.

`pyramid_auth.cookie.http_only`

Default: `False`. Hide cookie from JavaScript by setting the HttpOnly flag. Not honored by all browsers. Optional.

`pyramid_auth.cookie.wild_domain`

Default: `True`. An auth_tkt cookie will be generated for the wildcard domain. If your site is hosted as `example.com` this will make the cookie available for sites underneath `example.com` such as `www.example.com`. Optional.

`pyramid_auth.cookie.parent_domain`

Default: `False`. An auth_tkt cookie will be generated for the parent domain of the current site. For example if your site is hosted under `www.example.com` a cookie will be generated for `.example.com`. This can be useful if you have multiple sites sharing the same domain. This option supercedes the `wild_domain` option. Optional.

`pyramid_auth.cookie.domain`

Default: `None`. If provided the auth_tkt cookie will only be set for this domain. This option is not compatible with `wild_domain` and `parent_domain`. Optional.

`pyramid_auth.cookie.hashalg`

Default: `sha512` (the literal string).

Any hash algorithm supported by Python's `hashlib.new()` function can be used as the `hashalg`.

Cookies generated by different instances of AuthTktAuthenticationPolicy using different `hashalg` options are not compatible. Switching the `hashalg` will imply that all existing users with a valid cookie will be required to re-login.

Optional.

`pyramid_auth.cookie.debug`

Default: `False`. If `debug` is `True`, log messages to the Pyramid debug logger about the results of various authentication steps.

Optional.

CHAPTER 3

Remote_user policy

This policy uses `pyramid.authentication.RemoteUserAuthenticationPolicy`. The user is authenticated by the http server which provides in the environ a key with the login.

3.1 Installation

In your .ini file add *pyramid_auth* to *pyramid.includes* like this:

```
pyramid.includes =
    pyramid_auth
    ...
```

Also you need to add *pyramid_auth* in *setup.py* in install_requires:

```
install_requires=[
    ...
    'pyramid_auth'
]
```

3.2 Configuration

You need to set some options in your .ini file. See this example for the required ones:

```
pyramid_auth.policy = remote_user
```

3.2.1 Options

`environ_key` Default: `REMOTE_USER`. The key in the WSGI environ which provides the userid. Optional.

`callback` Default: `None`. A callback passed the userid and the request, expected to return None if the userid doesn't exist or a sequence of principal identifiers (possibly empty) representing groups if the user does exist. If `callback` is None, the userid will be assumed to exist with no group principals. Optional.

`debug` Default: `False`. If `debug` is `True`, log messages to the Pyramid debug logger about the results of various authentication steps. Optional.

CHAPTER 4

ldap policy

This policy uses `pyramid_ldap`. Basically the same logic than the cookie policy but we just validate the login/password with the ldap. As you will see in the configuration, it's possible to get the ldap user's groups. In this way, you will be able to set some permissions in your pyramid project according to the ldap configuration.

4.1 Installation

You need to have openldap header installed. For example on centos/fedora:

```
yum install openldap-devel
```

In your .ini file add *pyramid_ldap* and *pyramid_auth* to *pyramid.includes* like this:

```
pyramid.includes =
    pyramid_ldap
    pyramid_auth
    ...
```

Warning: the order is important, you need to include pyramid_ldap before pyramid_auth

Also you need to add *pyramid_ldap* and *pyramid_auth* in *setup.py* in install_requires:

```
install_requires=[
    ...
    'pyramid_ldap'
    'pyramid_auth'
]
```

Note: *pyramid_ldap* is not installed in pyramid_auth since we don't want to force the installation of ldap if we don't want to use it!

4.2 Configuration

You need to set some options in your .ini file. See this example for the required ones:

```
pyramid_auth.policy = ldap
pyramid_auth.ldap.cookie.secret = mysecret
pyramid_auth.ldap.setup.uri = http://ldap.lereskp.fr
pyramid_auth.ldap.setup.passwd = myldappasswd

pyramid_auth.ldap.login.base_dn = CN=Users,DC=lereskp,DC=fr
pyramid_auth.ldap.login.filter_tmpl = (sAMAccountName=$login)
```

If you want to put some permissions according to the ldap groups, you have to give the parameters to be able to query the ldap:

```
pyramid_auth.policy = ldap
pyramid_auth.ldap.cookie.secret = mysecret
pyramid_auth.ldap.setup.uri = http://ldap.lereskp.fr
pyramid_auth.ldap.setup.passwd = myldappasswd

pyramid_auth.ldap.login.base_dn = CN=Users,DC=lereskp,DC=fr
pyramid_auth.ldap.login.filter_tmpl = (sAMAccountName=$login)

pyramid_auth.ldap.groups.base_dn = CN=Users,DC=lereskp,DC=fr
pyramid_auth.ldap.groups.filter_tmpl = (&(objectCategory=group)(member=$userdn))
```

4.2.1 Options

Cookie

`pyramid_auth.ldap.cookie.secret`

The secret (a string) used for auth_tkt cookie signing. Required.

`pyramid_auth.ldap.cookie.callback`

Default: `None`. A callback passed the userid and the request, expected to return `None` if the userid doesn't exist or a sequence of principal identifiers (possibly empty) if the user does exist. If `callback` is `None`, the userid will be assumed to exist with no principals. Optional.

`pyramid_auth.ldap.cookie.cookie_name`

Default: `auth_tkt`. The cookie name used (string). Optional.

`pyramid_auth.ldap.cookie.secure`

Default: `False`. Only send the cookie back over a secure conn. Optional.

`pyramid_auth.ldap.cookie.include_ip`

Default: `False`. Make the requesting IP address part of the authentication data in the cookie. Optional.

For IPv6 this option is not recommended. The `mod_auth_tkt` specification does not specify how to handle IPv6 addresses, so using this option in combination with IPv6 addresses may cause an incompatible cookie. It ties the authentication ticket to that individual's IPv6 address.

`pyramid_auth.ldap.cookie.timeout`

Default: `None`. Maximum number of seconds which a newly issued ticket will be considered valid. After this amount of time, the ticket will expire (effectively logging the user out). If this value is `None`, the ticket never expires. Optional.

`pyramid_auth.ldap.cookie.reissue_time`

Default: `None`. If this parameter is set, it represents the number of seconds that must pass before an authentication token cookie is automatically reissued as the result of a request which requires authentication. The duration is measured as the number of seconds since the last auth_tkt cookie was issued and 'now'. If this value is `0`, a new ticket cookie will be reissued on every request which requires authentication.

A good rule of thumb: if you want auto-expired cookies based on inactivity: set the `timeout` value to 1200 (20 mins) and set the `reissue_time` value to perhaps a tenth of the `timeout` value (120 or 2 mins). It's nonsensical to set the `timeout` value lower than the `reissue_time` value, as the ticket will never be reissued if so. However, such a configuration is not explicitly prevented.

Optional.

`pyramid_auth.ldap.cookie.max_age`

Default: `None`. The max age of the auth_tkt cookie, in seconds. This differs from `timeout` inasmuch as `timeout` represents the lifetime of the ticket contained in the cookie, while this value represents the lifetime of the cookie itself. When this value is set, the cookie's `Max-Age` and `Expires` settings will be set, allowing the auth_tkt cookie to last between browser sessions. It is typically nonsensical to set this to a value that is lower than `timeout` or `reissue_time`, although it is not explicitly prevented. Optional.

`pyramid_auth.ldap.cookie.path`

Default: `/`. The path for which the auth_tkt cookie is valid. May be desirable if the application only serves part of a domain. Optional.

`pyramid_auth.ldap.cookie.http_only`

Default: `False`. Hide cookie from JavaScript by setting the HttpOnly flag. Not honored by all browsers. Optional.

`pyramid_auth.ldap.cookie.wild_domain`

Default: `True`. An auth_tkt cookie will be generated for the wildcard domain. If your site is hosted as `example.com` this will make the cookie available for sites underneath `example.com` such as `www.example.com`. Optional.

`pyramid_auth.ldap.cookie.parent_domain`

Default: `False`. An auth_tkt cookie will be generated for the parent domain of the current site. For example if your site is hosted under `www.example.com` a cookie will be generated for `.example.com`. This can be useful if you have multiple sites sharing the same domain. This option supercedes the `wild_domain` option. Optional.

`pyramid_auth.ldap.cookie.domain`

Default: `None`. If provided the auth_tkt cookie will only be set for this domain. This option is not compatible with `wild_domain` and `parent_domain`. Optional.

`pyramid_auth.ldap.cookie.hashalg`

Default: `sha512` (the literal string).

Any hash algorithm supported by Python's `hashlib.new()` function can be used as the `hashalg`.

Cookies generated by different instances of AuthTktAuthenticationPolicy using different `hashalg` options are not compatible. Switching the `hashalg` will imply that all existing users with a valid cookie will be required to re-login.

Optional.

`pyramid_auth.ldap.cookie.debug`

Default: `False`. If `debug` is `True`, log messages to the Pyramid debug logger about the results of various authentication steps.

Optional.

Setup

`pyramid_auth.ldap.setup.uri`

ldap server uri. Required.

`pyramid_auth.ldap.setup.bind`

Default `None`. Bind that will be used to bind a connector. Optional.

`pyramid_auth.ldap.setup.passwd`

Default `None`. Password that will be used to bind a connector. Optional.

`pyramid_auth.ldap.setup.size`

Default `10`. pool size. Optional.

`pyramid_auth.ldap.setup.retry_max`

Default `3`. Number of attempts when a server is down. Optional.

`pyramid_auth.ldap.setup.retry_delay`

Default: `.1`. Delay in seconds before a retry. Optional.

`pyramid_auth.ldap.setup.use_tls`

Default `False`. Activate TLS when connecting. Optional.

`pyramid_auth.ldap.setup.timeout`

Default `-1`. Connector timeout. Optional.

`pyramid_auth.ldap.setup.use_pool`

Default `True`. Activates the pool. If False, will recreate a connector each time. Optional.

Login

`pyramid_auth.ldap.login.base_dn`

is the DN at which to begin the search.

`pyramid_auth.ldap.login.filter_tmpl`

is a string which can be used as an LDAP filter: it should contain the replacement value %(login)s.

`pyramid_auth.ldap.login.scope`

is any valid LDAP scope value (e.g. ldap.SCOPE_ONELEVEL).

`pyramid_auth.ldap.login.cache_period`

is the number of seconds to cache login search results; if it is 0, login search results will not be cached.

Groups

`pyramid_auth.ldap.groups.base_dn`

is the DN at which to begin the search.

`pyramid_auth.ldap.groups.filter_tmpl`

is a string which can be used as an LDAP filter: it should contain the replacement value %(userdn)s.

Important: In pyramid_ldap userdn represent the user distinguished name. In pyramid_auth it represents the user uid. So you should make your filter_tmpl according to the user uid.

`pyramid_auth.ldap.groups.scope`

is any valid LDAP scope value (e.g. ldap.SCOPE_SUBTREE). cache_period is the number of seconds to cache groups search results; if it is 0, groups search results will not be cached.

Extra

`pyramid_auth.ldap.validate_function`

Default: `None`. You can set a function to validate the ldap login/password it you want to be more specific. Optional.

`pyramid_auth.ldap.callback`

Default: `None`. A callback passed the userid and the request to extend the groups found by the ldap groups query. Optional.