zope.password Documentation

Release 4.1

Zope Foundation and Contributors

Sep 01, 2017

Contents

Contents:

CHAPTER 1

Using `zope.password`

This package provides a password manager mechanism. Password manager is an utility object that can encode and check encoded passwords. Beyond the generic interface, this package also provides eight implementations:

`zope.password.password.PlainTextPasswordManager`

The most simple and the less secure one. It does not do any password encoding and simply checks password by string equality. It's useful in tests or as a base class for more secure implementations.

`zope.password.password.MD5PasswordManager`

A password manager that uses MD5 algorithm to encode passwords. It's generally weak against dictionary attacks due to a lack of a salt.

`zope.password.password.SMD5PasswordManager`

A password manager that uses MD5 algorithm, together with a salt to encode passwords. It's better protected against against dictionary attacks, but the MD5 hashing algorithm is not as strong as the SHA1 algorithm.

`zope.password.password.SHA1PasswordManager`

A password manager that uses SHA1 algorithm to encode passwords. It has the same weakness as the MD5PasswordManager.

`zope.password.password.SSHAPasswordManager`

A password manager that is strong against dictionary attacks. It's basically SHA1-encoding password manager which also incorporates a salt into the password when encoding it.

`zope.password.password.CryptPasswordManager`

A manager implementing the crypt(3) hashing scheme. Only available if the python crypt module is installed. This is a legacy manager, only present to ensure that zope.password can be used for all schemes defined in RFC 2307 (LDAP).

`zope.password.password.MySQLPasswordManager`

A manager implementing the digest scheme as implemented in the MySQL PASSWORD function in MySQL versions before 4.1. Note that this method results in a very weak 16-byte hash.

`zope.password.password.BCRYPTPasswordManager`

A manager implementing the bcrypt hashing scheme. Only available if the bcrypt module is installed. This manager is considered one of the most secure.

`zope.password.password.BCRYPTKDFPasswordManager`

A manager implementing the bcrypt_kdf hashing scheme. Only available if the bcrypt module is installed. This manager is considered one of the most secure.

The `Crypt`, `MD5`, `SMD5`, `SHA` and `SSHA` password managers are all compatible with RFC 2307 LDAP implementations of the same password encoding schemes.

Note: It is strongly recommended to use the BCRYPTPasswordManager or BCRYPTKDFPasswordManager, as they are the most secure.

The package also provides a script, **`zpasswd`**, to generate principal entries in typical `site.zcml` files.

Password Manager Interfaces

The `zope.password.interfaces.IPasswordManager` interface defines only two methods:

```
    def encodePassword(password):
        """Return encoded data for the given password

        Return encoded bytes.
        """
```

```
    def checkPassword(encoded_password, password):
        """Does the encoded password match the given password?

        Return True if they match, else False.
        """
```

An extended interface, `zope.password.interfaces.IMatchingPasswordManager`, adds one additional method:

```
    def match(encoded_password):
        """Was the given data was encoded with this manager's scheme?

        Return True when the given data was encoded with the scheme
        implemented by this password manager.
        """
```

Looking Up Password Managers via a Vocabulary

The `zope.password.vocabulary` module provides a vocabulary of registered password manager utility names. It is typically registered as an `zope.schema.interfaces.IVocabularyFactory` utility named "Password Manager Names".

It's intended to be used with `zope.component` and `zope.schema`, so you need to have them installed and the utility registrations needs to be done properly. The `configure.zcml` file contained in `zope.password` does the registrations, as well as in `zope.password.testing.setUpPasswordManagers()`.

Encrypting Passwords with `zpasswd`

zpasswd is a script to generate principal entries in typical `site.zcml` files.

You can create a **zpasswd** script in your buildout by adding a section like this to your `buildout.cfg`:

```
[zpasswd]
recipe = z3c.recipe.dev:script
eggs = zope.password
module = zope.password.zpasswd
method = main
```

This will generate a script **zpasswd** next time you run **buildout**.

When run, the script will ask you for all parameters needed to create a typical principal entry, including the encrypted password.

Use:

```
$ bin/zpasswd --help
```

to get a list of options.

Using

```
$ bin/zpasswd -c some/site.zcml
```

the script will try to lookup any password manager you defined and registered in your environment. This is lookup is not necessary if you go with the standard password managers defined in `zope.password`.

A typical **zpasswd** session might look like:

```
$ ./bin/zpasswd

Please choose an id for the principal.

Id: foo


Please choose a title for the principal.

Title: The Foo


Please choose a login for the principal.

Login: foo

Password manager:

 1. Plain Text
 2. MD5
 3. SMD5
 4. SHA1
 5. SSHA
 6. BCRYPT

Password Manager Number [6]:
BCRYPT password manager selected
```

```
Please provide a password for the principal.

Password:
Verify password:

Please provide an optional description for the principal.

Description: The main foo

================================================
Principal information for inclusion in ZCML:

  <principal
    id="foo"
    title="The Foo"
    login="foo"
    password="{BCRYPT}$2b$12$ez4eHl6W1PfAWix5bPIbe.drdnyqjpuT1Cp0N.xcdxkAEbA7K6AHK"
    description="The main foo"
    password_manager="BCRYPT"
    />
```

CHAPTER 2

zope.password API

Interfaces

Password Manager Implementations

Deprecated Implementations

Warning: The following password managers are deprecated, because they produce unacceptably-weak password hashes. They are only included to allow apps which previously used them to migrate smoothly to a supported implementation.

Vocabulary

CHAPTER 3

Indices and tables

- genindex
- modindex
- search