SecureMongoEngine Documentation

Release 0.1.1

Juan Urrego

August 29, 2014

Contents

It is a library for MongoEngine(https://github.com/MongoEngine) that you can use to encrypt certain fields of your models. Currently is in Beta Version and it only supports AES for Symmetric Encryption and SHA for hashing.

CHAPTER 1

Usage

To install SecureMongoeEngine just execute:

```
$ pip install securemongoengine
```

And now you can create a docummnent with some encrypted fields, you just need a key an thats all!!!

```
from mongoengine import *
from securemongoengine.fields import *

_key = 'workingWithAES256AlgorithmKey32B'

class User(Document):
    name = StringField(max_length=40, required=True)
    lastname = StringField(max_length=40, required=True)
    email = EmailField(required=True, unique=True)
    password = EncryptedStringField(key=_key,max_length=40, required=True)

user = User(name='Juan',lastname='Urrego',email='js.urrego@novcat.co',password = '123456')

connect('test', host='127.0.0.1')
user.save()
```

In your Mongo database you will see something like this:

```
{ "_id" : ObjectId("5400c7205f9370f0603c3cfa"), "name" : "Juan", "lastname" : "Urrego",
"email" : "js.urrego@novcat.co", "password" : "@::::@685f0500d7b99a59c9d6c496184a65bc" }
```

CHAPTER 2

Contents:

Tutorial A quick tutorial building a tumblelog to get you up and running with SecureMongoEngine.

API The complete API documentation

CHAPTER 3

Offline Reading

Download the docs in pdf or epub formats for offline reading.

3.1 Tutorial

Here we will introduce the main features of SecureMongoEngine, and how to use it in a real environment.

3.1.1 Installation

SecureMongoEngine uses MongoEngine and pyCrypto, so it is able to encrypt all the data that you need. It is available on PyPI, so to use it you can use pip:

```
$ pip install securemongoengine
```

Or you can also use easy_install:

```
$ easy_install securemongoengine
```

Alternatively, if you don't have setuptools installed, download it from PyPi or Github and run:

```
$ python setup.py install
```

3.1.2 Encrypted Fields

The main idea of SecureMongoEngine is that you can transparently use encryption using MongoEngine models. By default all the fields will encrypt your data with AES, nevertheless you can change the encryption algorithm. The library has differents fields, so even when the data is encrypted you will see the plain values in python. Next we will present the differents fields:

- `EncryptedEmailField`
- `EncryptedStringField`
- `EncryptedIntField`
- `EncryptedDecimalField`
- `EncryptedFloatField`
- `EncryptedLongField`

Field arguments

Each field type can be customized by keyword arguments. The following keyword arguments can be set on all fields:

`algorithm` (Default: 'AES') Defines which encryption algorithm is going to be used. By default, SecureMongoEngine uses AES and the size of the encryption key will define if it will AES256, AES192 or AES128. More details next.

`key` (Default: None) Is the value needed to make the encryption. The key size used for an AES cipher specifies the number of repetitions of transformation rounds that convert the input, called the plaintext, into the final output, called the ciphertext. The number of cycles of repetition will be:

- 10 cycles of repetition for 128-bit (16 Bytes) keys.
- 12 cycles of repetition for 192-bit (24 Bytes) keys.
- 14 cycles of repetition for 256-bit (32 Bytes) keys.

In few words, bigger keys will give you strongers encryptions. Remember that your keys must have exactly those lengths, it means that *'Example Key'* (length in bytes = 11) will raise you an exception.

`iv` (Default: None) The Initialization Vector (IV) will produce you a higher entropy in your encryption, especially when we work with block ciphers. A single invocation of the AES algorithm transforms a 128-bit plaintext block into a ciphertext block of 128 bits in size, it means that block ciphers need the same block size in our plain data. Obviously, most of the cases, this does not happen. For that reason we need to fill those empty spaces with something random, and that is the IV (16 Bytes). We always recommend to use an IV for any Block Cipher. For more information Block Cipher Mode Operation.

3.2 API

3.2.1 Fields

class `securemongoengine.base.fields.`**`EncryptedField`**(*algorithm=None*, *key=None*, *iv=None*, ***kwargs*)

Constants

class `securemongoengine.fields.`**`EncryptedEmailField`**(*algorithm=None*, *key=None*, *iv=None*, ***kwargs*)

class `securemongoengine.fields.`**`EncryptedStringField`**(*algorithm=None*, *key=None*, *iv=None*, ***kwargs*)

class `securemongoengine.fields.`**`EncryptedIntField`**(*algorithm=None*, *key=None*, *iv=None*, ***kwargs*)

class `securemongoengine.fields.`**`EncryptedDecimalField`**(*algorithm=None*, *key=None*, *iv=None*, ***kwargs*)

class `securemongoengine.fields.`**`EncryptedFloatField`**(*algorithm=None*, *key=None*, *iv=None*, ***kwargs*)

class `securemongoengine.fields.`**`EncryptedLongField`**(*algorithm=None*, *key=None*, *iv=None*, ***kwargs*)

CHAPTER 4

Indices and tables

- *genindex*
- *modindex*
- *search*