django-sage-encrypt

Sage Team

Jul 18, 2021

CONTENTS

This app supports the following combinations of Django and Python:

Django	Python
3.1	3.6, 3.7, 3.8, 3.9
3.2	3.6, 3.7, 3.8, 3.9

CHAPTER

ONE

FUNCTIONALITY

Django Sage Encrypt is a tool for using pgcrypto in django.

pgcrypto is an extension for encrypting PostgreSQL at rest, With this module you can increase the security of your sensitive data.

Sage Encrypt supports:

- all database lookups (search, filter, . . .)
- symmetric & asymmetric encryption algorithms
- multiple scenarios for preventing data losing

CHAPTER

TWO

DOCUMENTATION

2.1 Installation

First install package

```
$ pip install django-sage-encrypt
```

Then add 'sage_encrypt' to INSTALLED_APPS in settings.py

```
INSTALLED_APPS = [
  ...
  'sage_encrypt',
  ...
]
```

Also you need to install `pgcrypto` extension in your database:

```
sudo -u postgres psql <db_name>
```

```
CREATE EXTENSION pgcrypto;
```

2.2 Fields

For encrypting each row of your database there are multiple ways:

1. use `encrypt_field` function in your models.py

```
from django.db import models
from sage_encrypt.services.encrypt import encrypt_field

# symmetric encryption
title = encrypt_field(models.CharField(max_length=255))

# asymmetric encryption
title = encrypt_field(models.CharField(max_length=255), algorithm='asymmetric')
```

2. use field directly

```
# symmetric encryption
from sage_encrypt.fields.symmetric import EncryptedCharField

title = EncryptedCharField(max_length=255)


# asymmetric encryption
from sage_encrypt.fields.asymmetric import EncryptedCharField

title = EncryptedCharField(max_length=255)
```

If you want to use `symmetric encryption` you don't need to generate secret keys default is SECRET_KEY

But if you want to use `asymmetric encryption` you have to generate private key & public key

2.3 Generate secret key

```
# generate private & public key
gpg --gen-key # in password section do not enter password

gpg --list-keys
# output
pub   rsa3072 2021-06-20 [SC] [expires: 2023-06-20]
      <test_token_generated>
uid           [ultimate] Test <test@gmail.com>
sub   rsa3072 2021-06-20 [E] [expires: 2023-06-20]

gpg -a --export <test_token_generated> > public.key
gpg -a --export-secret-keys <test_token_generated> > private.key
```

2.4 Management Commands

sage_encrypt provides 2 management commands:

1. `encryptdb`

```
python manage.py encryptdb --table <table_name> --column <col_name> --cast <field_
↪previous_cast_type> --algorithm <algorithm> #(symmetric/asymmetric)
```

Options:

1. –database (if you have multiple db's specify for your database)
2. –table (table name in your database not django model title)
3. –column (col name in the specified table)
4. –algorithm (symmetric/asymmetric)
5. –cast (field previous cast that you want to encrypt from that)

Usage:

When you want to add encryption on a row and there is valuable data in you db, you can encrypt the data to be compatible with Encrypted Field.

2. `decryptdb`

```
python manage.py decryptdb --table <table_name> --column <col_name>
```

Options:

1. –database (if you have multiple db's specify for your database)
2. –table (table name in your database not django model title)
3. –column (col name in the specified table)

Usage:

When your data is encrypted in db and you want to remove encryption from a row, for getting back data you can use this command, it decrypts data and replaces in your db.

2.5 Settings

Here are the parameters that you can set from setting:

Parameter	Description
ENCRYPT_KEY	Secret key that using for symmetric encryption. default: SECRET_KEY
ENCRYPT_PRIVATE_KEY	Private key for asymmetric encryption. default: None
ENCRYPT_PUBLIC_KEY	Private key for asymmetric encryption. default: None

CHAPTER

THREE

ISSUES

If you have questions or have trouble using the app please file a bug report at:

https://github.com/sageteam-org/django-sage-encrypt/issues

CHAPTER

FOUR

INDICES AND TABLES

- search