Flask Api Sign Documentation

Release 0.1.0

juforg

Aug 02, 2021

CONTENTS

1 Flask Api Sign Verification **3**
- 1.1 Features . . . 3
- 1.2 Quickstart . . . 3
- 1.3 Links . . . 4

2 Installation **5**
- 2.1 Stable release . . . 5
- 2.2 From sources . . . 5

3 Usage **7**
- 3.1 Configuring flask-api-sign . . . 7

4 Contributing **9**
- 4.1 Types of Contributions . . . 9
- 4.2 Get Started! . . . 10
- 4.3 Pull Request Guidelines . . . 11
- 4.4 Tips . . . 11

5 Credits **13**
- 5.1 Development Lead . . . 13
- 5.2 Contributors . . . 13

6 History **15**
- 6.1 0.1.0 (2021-07-27) . . . 15

7 Indices and tables **17**

Contents:

CHAPTER

ONE

FLASK API SIGN VERIFICATION

1.1 Features

- Testing setup with `unittest` and `python setup.py test` or `py.test`
- Command line interface using Click

1.2 Quickstart

Install the latest Cookiecutter if you haven't installed it yet

```
pip install -U flask-api-sign
```

Then:

```
from flask import Flask
from flask_api_sign import ApiSignManager
from flask_api_sign import verify_sign

app = Flask(__name__)

api_sign_mgr = ApiSignManager()
api_sign_mgr.init_app(app)
@app.route("/")
@verify_sign
def index():
    pass
```

1.3 Links

- Documentation: https://flask-api-sign.readthedocs.io/en/latest/index.html
- Changes: https://flask-api-sign.readthedocs.io/en/latest/history.html
- PyPI Releases: https://pypi.org/project/flask-api-sign/
- Source Code: https://github.com/juforg/flask-api-sign/
- Issue Tracker: https://github.com/juforg/flask-api-sign/issues/

CHAPTER

TWO

INSTALLATION

2.1 Stable release

To install Flask Api Sign, run this command in your terminal:

```
$ pip install flask_api_sign
```

This is the preferred method to install Flask Api Sign, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

2.2 From sources

The sources for Flask Api Sign can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/juforg/flask-api-sign
```

Or download the tarball:

```
$ curl  -OL https://github.com/juforg/flask-api-sign/tarball/master
```

Once you have a copy of the source, you can install it with:

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

CHAPTER

THREE

USAGE

To use Flask Api Sign in a project:

```
import flask_api_sign
```

3.1 Configuring flask-api-sign

flask-api-sign is configured through the standard Flask config API. These are the available options (each is explained later in the documentation):

- **SIGN_LOCATION** : default **query_string**
- **SIGN_TIMESTAMP_EXPIRATION** : default **30**
- **SIGN_APP_IDS** : default ``**{'testapp': 'testsecret'}**``

verification is managed through a `ApiSignManager` instance:

```
from flask import Flask
from flask_api_sign import ApiSignManager

app = Flask(__name__)
api_sign_mgr = ApiSignManager(app)
```

In this case all verification using the configuration values of the application that was passed to the `ApiSignManager` class constructor.

Alternatively you can set up your `ApiSignManager` instance later at configuration time, using the **init_app** method:

```
from flask import Flask
api_sign_mgr = ApiSignManager()
app = Flask(__name__)
api_sign_mgr.init_app(app)
```

In this case verification will use the configuration values from Flask's `current_app` context global. This is useful if you have multiple applications running in the same process but with different configuration options.

3.1.1 Flask Api Sign Verification

To generate a serial number first create a `ApiSignManager` instance:

```
from flask import Flask
from flask_api_sign import ApiSignManager
from flask_api_sign import verify_sign

app = Flask(__name__)

api_sign_mgr = ApiSignManager()
api_sign_mgr.init_app(app)
@app.route("/")
@verify_sign
def index():
    pass
```

you can write a java client with the demo to generate the x-sign.

NOTE: Remember to set the secret key of the application, and ensure that no one else is able to view it. The request are signed with the secret key, so if someone gets that, they can create arbitrary request.

CHAPTER

FOUR

CONTRIBUTING

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/juforg/flask-api-sign/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

4.1.4 Write Documentation

Flask Api Sign could always use more documentation, whether as part of the official Flask Api Sign docs, in docstrings, or even on the web in blog posts, articles, and such.

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/juforg/flask-api-sign/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

4.2 Get Started!

Ready to contribute? Here's how to set up *flask-api-sign* for local development.

1. Fork the *flask-api-sign* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/flask-api-sign.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv flask-api-sign
$ cd flask-api-sign/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 flask_api_sign tests
$ python setup.py test or py.test
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.3, 3.4, 3.5, 3.7, 3.8 and 3.9 and for PyPy. Check https://travis-ci.org/juforg/flask-api-sign/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ py.test tests.test_flask_api_sign
```

CHAPTER

FIVE

CREDITS

5.1 Development Lead

- juforg <juforg@gmail.com>

5.2 Contributors

None yet. Why not be the first?

CHAPTER

SIX

HISTORY

6.1 0.1.0 (2021-07-27)

- First release on PyPI.

CHAPTER

SEVEN

INDICES AND TABLES

- genindex
- modindex
- search