cipherkit

Release 0.0.0

Apr 25, 2019

Contents

Cipherkit: a cryptography toolbox made with Python

<img align="left" src="docs/img/logo_cipherkit.png">

A Python package holding classical ciphers and different cryptanalysis tools. For the moment, this package is under heavy development. Some of the features it will include are the following:

- Classical ciphers: atbash, caesar, vigenere, affine...
- Crytanalysis tools: brute force and frequancy analysis
- Language identifier based on dictionaries

Once the package is complete and first versions start releasing, some future implementation would be:

- Work with .txt files
- Steganography algorithms
- Modern non-symmetric ciphers

[![made-with-python](https://img.shields.io/badge/Made%20with-Python-1f425f.svg){}](https://www.python.org/) [![Documentation Status](https://readthedocs.org/projects/cipherkit/badge/?version=latest){}](https://cipherkit.readthedocs.io/en/latest/?badge=latest) [![Build Status](https://travis-ci.org/jorgepiloto/cipherkit.svg?branch=master){}](https://travis-ci.org/jorgepiloto/cipherkit)

Why cryptography?

Cryptography is one of the most used sciencies today due to its strong relation with cryptocurrencies, money transactions, online business and others. Most of moder ciphers try to solve the key-exchange problem by using a public/private keys.

Elliptic curves, hash functions, RSA, AES and others may be familiar when we talk about modern cryptography. Their implementation is really smart and at the same time really counter-intuitive. It is not only the process of designing the cipher but also to make as 'hard' as possible for cryptanalysists.

Although classical ciphers are just obsolete in termns of security, this package objective is just to have fun with their implementation in a beautiful programming language.

Books and sources about cryptography

There exist really good books on cryptography history, algorithms and cryptanalysis techniques. Some of them, which were used consulted during the coding phase of this package ares listed down:

- Hacking secret ciphers with Python, AI Sweigart
- The Code Book, Simon Singh
- Boletín Enigma, Arturo Quirantes Sierra
- Cripgrafía sin secretos con Python, David Arboledas

Installation

```bash pip install cipherkit ```

Documentation

https://cipherkit.readthedocs.io/

Development

Bug solving, new features and improves are always welcome. This is an opensource project: we want to have fun, learn and improve. For that reason if you have doubts, issues with the package or just want to add a new cipher please follow the following steps:

- Fork cipherkit
```

- Download and install with *pip install -e cipherkit*. This wll install the package in editable mode, which will enable you to make code changes and continue while using cipherkit.
- Create a new branch in your local repo, improve the code and start a pull request.
- Celebrate your first contribution.

CHAPTER 1

Installation

At the command line:

```
pip install cipherkit
```

CHAPTER 2

Usage

To use cipherkit in a project:

```
import cipherkit
```

CHAPTER 3

Reference

3.1 cipherkit

CHAPTER 4

Contributing

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

4.1 Bug reports

When 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.

4.2 Documentation improvements

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

4.3 Feature requests and feedback

The best way to send feedback is to file an issue at https://github.com/jorgepiloto/cipherkit/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 code contributions are welcome :)

4.4 Development

To set up *cipherkit* for local development:

1. Fork cipherkit (look for the "Fork" button).
2. Clone your fork locally:

```
git clone git@github.com:your_name_here/cipherkit.git
```

3. Create a branch for local development:

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

   Now you can make your changes locally.

4. When you're done making changes, run all the checks, doc builder and spell checker with tox one command:

```
tox
```

5. 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
```

6. Submit a pull request through the GitHub website.

4.4.1 Pull Request Guidelines

If you need some code review or feedback while you're developing the code just make the pull request.

For merging, you should:

1. Include passing tests (run `tox`)[1].
2. Update documentation when there's new API, functionality etc.
3. Add a note to `CHANGELOG.rst` about the changes.
4. Add yourself to `AUTHORS.rst`.

4.4.2 Tips

To run a subset of tests:

```
tox -e envname -- pytest -k test_myfeature
```

To run all the test environments in *parallel* (you need to `pip install detox`):

```
detox
```

[1] If you don't have all the necessary python versions available locally you can rely on Travis - it will run the tests for each change you add in the pull request.
It will be slower though ...

CHAPTER 5

Authors

- Jorge Martínez - https://github.com/jorgepiloto

CHAPTER 6

Changelog

6.1 0.0.0 (2019-04-17)

- First release on PyPI.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

C

Index

C