Welcome to `pyca/cryptography`

`cryptography` includes both high level recipes and low level interfaces to common cryptographic algorithms such as symmetric ciphers, message digests, and key derivation functions. For example, to encrypt something with `cryptography`'s high level symmetric encryption recipe:

```
>>> from cryptography.fernet import Fernet
>>> # Put this somewhere safe!
>>> key = Fernet.generate_key()
>>> f = Fernet(key)
>>> token = f.encrypt(b"A really secret message. Not for prying eyes.")
>>> token
b'...'
>>> f.decrypt(token)
b'A really secret message. Not for prying eyes.'
```

If you are interested in learning more about the field of cryptography, we recommend Crypto 101, by Laurens Van Houtven and The Cryptopals Crypto Challenges.

Installation

You can install `cryptography` with `pip`:

```
$ pip install cryptography
```

See Installation for more information.

Layout

`cryptography` is broadly divided into two levels. One with safe cryptographic recipes that require little to no configuration choices. These are safe and easy to use and don't require developers to make many decisions.

The other level is low-level cryptographic primitives. These are often dangerous and can be used incorrectly. They require making decisions and having an in-depth knowledge of the cryptographic concepts at work. Because of the potential danger in working at this level, this is referred to as the "hazardous materials" or "hazmat" layer. These live in the `cryptography.hazmat` package, and their documentation will always contain an admonition at the top.

We recommend using the recipes layer whenever possible, and falling back to the hazmat layer only when necessary.

The recipes layer

- Fernet (symmetric encryption)
    - `Fernet`
    - `MultiFernet`
    - `InvalidToken`
    - Using passwords with Fernet
    - Implementation
    - Limitations
- X.509
    - Tutorial
    - Certificate Transparency
    - OCSP
    - X.509 Reference

The hazardous materials layer

- Primitives
    - Authenticated encryption
    - Asymmetric algorithms
    - Constant time functions
    - Key derivation functions
    - Key wrapping
    - Message authentication codes
    - Message digests (Hashing)
    - Symmetric encryption
    - Symmetric Padding
    - Two-factor authentication
- Exceptions
    - `UnsupportedAlgorithm`
    - `AlreadyFinalized`
    - `InvalidSignature`
    - `NotYetFinalized`
    - `AlreadyUpdated`
    - `InvalidKey`
- Random number generation

The cryptography open source project

- Installation
  - Supported platforms
  - Building cryptography on Windows
  - Building cryptography on Linux
  - Building cryptography on macOS
  - Rust
- Changelog
  - 39.0.0 - main
  - 38.0.4 - 2022-11-27
  - 38.0.3 - 2022-11-01
  - 38.0.2 - 2022-10-11 (YANKED)
  - 38.0.1 - 2022-09-07
  - 38.0.0 - 2022-09-06
  - 37.0.4 - 2022-07-05
  - 37.0.3 - 2022-06-21 (YANKED)
  - 37.0.2 - 2022-05-03
  - 37.0.1 - 2022-04-27
  - 37.0.0 - 2022-04-26
  - 36.0.2 - 2022-03-15
  - 36.0.1 - 2021-12-14
  - 36.0.0 - 2021-11-21
  - 35.0.0 - 2021-09-29
  - 3.4.8 - 2021-08-24
  - 3.4.7 - 2021-03-25
  - 3.4.6 - 2021-02-16
  - 3.4.5 - 2021-02-13
  - 3.4.4 - 2021-02-09
  - 3.4.3 - 2021-02-08
  - 3.4.2 - 2021-02-08
  - 3.4.1 - 2021-02-07
  - 3.4 - 2021-02-07
  - 3.3.2 - 2021-02-07
  - 3.3.1 - 2020-12-09
  - 3.3 - 2020-12-08
  - 3.2.1 - 2020-10-27
  - 3.2 - 2020-10-25
  - 3.1.1 - 2020-09-22
  - 3.1 - 2020-08-26
  - 3.0 - 2020-07-20
  - 2.9.2 - 2020-04-22
  - 2.9.1 - 2020-04-21
  - 2.9 - 2020-04-02
  - 2.8 - 2019-10-16

  - 2.7 - 2019-05-30
  - 2.6.1 - 2019-02-27
  - 2.6 - 2019-02-27
  - 2.5 - 2019-01-22
  - 2.4.2 - 2018-11-21
  - 2.4.1 - 2018-11-11
  - 2.4 - 2018-11-11
  - 2.3.1 - 2018-08-14
  - 2.3 - 2018-07-18
  - 2.2.2 - 2018-03-27
  - 2.2.1 - 2018-03-20
  - 2.2 - 2018-03-19
  - 2.1.4 - 2017-11-29
  - 2.1.3 - 2017-11-02
  - 2.1.2 - 2017-10-24
  - 2.1.1 - 2017-10-12
  - 2.1 - 2017-10-11
  - 2.0.3 - 2017-08-03
  - 2.0.2 - 2017-07-27
  - 2.0.1 - 2017-07-26
  - 2.0 - 2017-07-17
  - 1.9 - 2017-05-29
  - 1.8.2 - 2017-05-26
  - 1.8.1 - 2017-03-10
  - 1.8 - 2017-03-09
  - 1.7.2 - 2017-01-27
  - 1.7.1 - 2016-12-13
  - 1.7 - 2016-12-12
  - 1.6 - 2016-11-22
  - 1.5.3 - 2016-11-05
  - 1.5.2 - 2016-09-26
  - 1.5.1 - 2016-09-22
  - 1.5 - 2016-08-26
  - 1.4 - 2016-06-04
  - 1.3.4 - 2016-06-03
  - 1.3.3 - 2016-06-02
  - 1.3.2 - 2016-05-04
  - 1.3.1 - 2016-03-21
  - 1.3 - 2016-03-18
  - 1.2.3 - 2016-03-01
  - 1.2.2 - 2016-01-29
  - 1.2.1 - 2016-01-08
  - 1.2 - 2016-01-08
  - 1.1.2 - 2015-12-10
  - 1.1.1 - 2015-11-19
  - 1.1 - 2015-10-28

  - 1.0.2 - 2015-09-27
  - 1.0.1 - 2015-09-05
  - 1.0 - 2015-08-12
  - 0.9.3 - 2015-07-09
  - 0.9.2 - 2015-07-04
  - 0.9.1 - 2015-06-06
  - 0.9 - 2015-05-13
  - 0.8.2 - 2015-04-10
  - 0.8.1 - 2015-03-20
  - 0.8 - 2015-03-08
  - 0.7.2 - 2015-01-16
  - 0.7.1 - 2014-12-28
  - 0.7 - 2014-12-17
  - 0.6.1 - 2014-10-15
  - 0.6 - 2014-09-29
  - 0.5.4 - 2014-08-20
  - 0.5.3 - 2014-08-06
  - 0.5.2 - 2014-07-09
  - 0.5.1 - 2014-07-07
  - 0.5 - 2014-07-07
  - 0.4 - 2014-05-03
  - 0.3 - 2014-03-27
  - 0.2.2 - 2014-03-03
  - 0.2.1 - 2014-02-22
  - 0.2 - 2014-02-20
  - 0.1 - 2014-01-08
- Frequently asked questions
  - I cannot suppress the deprecation warning that `cryptography` emits on import
  - `cryptography` failed to install!
  - How does `cryptography` compare to NaCl (Networking and Cryptography Library)?
  - Why use `cryptography`?
  - Why does `cryptography` require Rust?
  - Installing `cryptography` produces a `fatal error: 'openssl/opensslv.h' file not found` error
  - `cryptography` raised an `InternalError` and I'm not sure what to do?
  - error: `-Werror=sign-conversion`: No option `-Wsign-conversion` during installation
  - Installing cryptography with OpenSSL 0.9.8, 1.0.0, 1.0.1, 1.0.2, 1.1.0 fails
  - Installing `cryptography` fails with `error: Can not find Rust compiler`
  - I'm getting errors installing or importing `cryptography` on AWS Lambda
  - Why are there no wheels for my Python3.x version?
  - Why can't I import my PEM file?
  - What happened to the backend argument?
  - Will you upload wheels for my non-x86 non-ARM64 CPU architecture?
- Development
  - Getting started

  - Submitting patches
  - Reviewing and merging patches
  - Test vectors
  - C bindings
- Use of OpenSSL
  - `cryptography.hazmat.backends.openssl.backend`
  - `cryptography.hazmat.backends.openssl.openssl_version_text()`
  - `openssl_version_number()`
  - `activate_osrandom_engine()`
  - `osrandom_engine_implementation()`
  - `activate_builtin_random()`
  - Legacy provider in OpenSSL 3.x
  - OS random engine
  - OS random sources
- Security
  - Infrastructure
  - What is a security issue?
  - Reporting a security issue
  - Supported Versions
  - New releases for OpenSSL updates
  - Disclosure Process
- Known security limitations
  - Secure memory wiping
  - RSA PKCS1 v1.5 constant time decryption
- API stability
  - What does this policy cover?
  - What does “stable” mean?
  - What doesn’t this policy cover?
  - Versioning
- Doing a release
  - Security Releases
  - Verifying OpenSSL version
  - Upgrading OpenSSL
  - Bumping the version number
  - Performing the release
  - Verifying the release
  - Post-release tasks
- Community
- Glossary

Note

`cryptography` has not been subjected to an external audit of its code or documentation. If you're interested in discussing an audit please get in touch.