rehash

Aug 28, 2022

Contents

Rehash is a resumable interface to the OpenSSL-based hashers (message digest objects) in the CPython `hashlib` standard library. Rehash provides hashers that can be pickled, persisted and reconstituted from their `repr()`, and otherwise serialized. The rest of the Rehash API is identical to `hashlib`.

Rehash hashers can be used to checkpoint and restore progress when hashing large byte streams:

```
import pickle, rehash
hasher = rehash.sha256(b"foo")
state = pickle.dumps(hasher)

hasher2 = pickle.loads(state)
hasher2.update(b"bar")

assert hasher2.hexdigest() == rehash.sha256(b"foobar").hexdigest()
```

CHAPTER 1

Installation

```
pip install rehash
```

1.1 Applications

Rehash is useful in any situation when your VM is short-lived or preemptible, and the object you're hashing is huge. For example, Rehash can be used to hand off the hashing state of large objects between AWS Lambda functions or Google Cloud Functions, which have runtime limits of 15 and 9 minutes, respectively.

Non-openssl hashers

`sha3` and `blake2` hash algorithms in Python 3.6 are not OpenSSL-based and not supported by rehash.

PyPy

PyPy uses its own hasher implementations. Those are not serializable using rehash.

Security note

By default, rehash objects present themselves with a `repr()` that exposes their internal state. This allows one to resume the hashing from the point where it stopped. If exposed through an untrusted channel under specific conditions, this could potentially allow an attacker to use an extension attack. If you are unsure about the implications of this, set `rehash.opaque_repr = True` after importing rehash.

CHAPTER 2

Links

- Project home page (GitHub)
- Documentation (Read the Docs)
- Package distribution (PyPI)
- Change log

2.1 Bugs

Please report bugs, issues, feature requests, etc. on GitHub.

CHAPTER 3

License

Licensed under the terms of the Apache License, Version 2.0.