s3sfe Documentation

Release 0.1.1

Jason Antman

Dec 24, 2017

Contents

s3sfe (S3 Sync Filelist Encrypted) Sync a list of files to S3, using server-side encryption with customer-provided keys.

CHAPTER 1

Introduction

This is a quick script I wrote for my own purposes. It's not terribly well tested, and it serves a small niche use case. If you're looking to securely sync your backups to S3 or another offsite storage, I'd highly encourage you to look into the other options.

My use case is relatively simple:

- I want to sync just some files from my backups to S3; a specific whitelist of files and directories.
- I don't want to keep history, I just want the latest versions somewhere offsite.
- I want to use S3 Server-Side Encryption with Customer-Provided Encryption Keys (SSE-C); I'm fine keeping the key on my computer, because if someone can get it, they can get the original files too. I'm not worried about Amazon snooping on my data. I'm not concerned with anyone being able to access the filenames or metadata. All I'm really concerned about is that if a malicious party gets access to my AWS account, they don't also implicitly get the file contents.

This tool takes a list of files or directories on the local filesystem and syncs them to S3, using server-side encryption. It uses the files' md5sums to only upload files that differ from what's already in S3.

CHAPTER 2

Requirements

- Python 2.7 or 3.4+ (currently tested with 2.7, 3.4+ and developed with 3.6)
- Python VirtualEnv and `pip` (recommended installation method; your OS/distribution should have packages for these)

CHAPTER 3

Installation

It's recommended that you install into a virtual environment (virtualenv / venv). See the virtualenv usage documentation for information on how to create a venv.

```
pip install s3sfe
```

CHAPTER 4

Configuration

s3sfe takes all of its configuration via command-line options. It does, however, expect a few elements of configuration to be present on the system:

- Your AWS Credentials must be available to the program in one of the methods supported by boto3, typically either environment variables or one of the supported credentials files (`~/.aws/credentials` or `~/.aws/config`) or boto configuration files (`~/.boto` or `/etc/boto.cfg`).
- Your encryption key for S3 Server-Side Encryption with Customer-Provided Encryption Keys (SSE-C) must be stored in a file readable by this program. This must be a 256-bit AES256 key, stored in binary format.

CHAPTER 5

Usage

To backup: `s3sfe --help`

To restore: `s3sfe-restore --help`

CHAPTER 6

Bugs and Feature Requests

Bug reports and feature requests are happily accepted via the GitHub Issue Tracker. Pull requests are welcome. Issues that don't have an accompanying pull request will be worked on as my time and priority allows.

CHAPTER 7

Development

To install for development:

1. Fork the s3sfe repository on GitHub
2. Create a new branch off of master in your fork.

```
$ virtualenv s3sfe
$ cd s3sfe && source bin/activate
$ pip install -e git+git@github.com:YOURNAME/s3sfe.git@BRANCHNAME#egg=s3sfe
$ cd src/s3sfe
```

The git clone you're now in will probably be checked out to a specific commit, so you may want to `git checkout BRANCHNAME`.

7.1 Guidelines

- pep8 compliant with some exceptions (see pytest.ini)
- 100% test coverage with pytest (with valid tests)

7.2 Testing

Testing is done via pytest, driven by tox.

- testing is as simple as:
  - `pip install tox`
  - `tox`
- If you want to pass additional arguments to pytest, add them to the tox command line after "–". i.e., for verbose pytext output on py27 tests: `tox -e py27 -- -v`

7.3 Release Checklist

1. Open an issue for the release; cut a branch off master for that issue.
2. Confirm that there are CHANGES.rst entries for all major changes.
3. Ensure that Travis tests passing in all environments.
4. Ensure that test coverage is no less than the last release (ideally, 100%).
5. Increment the version number in s3sfe/version.py and add version and release date to CHANGES.rst, then push to GitHub.
6. Confirm that README.rst renders correctly on GitHub.
7. Upload package to testpypi:
   - Make sure your ~/.pypirc file is correct (a repo called `test` for https://testpypi.python.org/pypi)
   - `rm -Rf dist`
   - `python setup.py register -r https://testpypi.python.org/pypi`
   - `python setup.py sdist bdist_wheel`
   - `twine upload -r test dist/*`
   - Check that the README renders at https://testpypi.python.org/pypi/s3sfe
8. Create a pull request for the release to be merged into master. Upon successful Travis build, merge it.
9. Tag the release in Git, push tag to GitHub:
   - tag the release. for now the message is quite simple: `git tag -s -a X.Y.Z -m 'X.Y.Z released YYYY-MM-DD'`
   - push the tag to GitHub: `git push origin X.Y.Z`
11. Upload package to live pypi:
   - `twine upload dist/*`
10. make sure any GH issues fixed in the release were closed.

CHAPTER 8

Contents

8.1 s3sfe

8.1.1 s3sfe package

8.1.1.1 Submodules

s3sfe.filesyncer module

s3sfe.restorer module

s3sfe.runner module

s3sfe.runstats module

s3sfe.s3 module

s3sfe.utils module

s3sfe.utils.**dtnow**()
Helper for testing; just returns datetime.datetime.now()

Returns current datetime - `datetime.datetime.now()`

Return type datetime.datetime

s3sfe.utils.**md5_file**(*path*)
Return the MD5 sum of the contents of the file at `path`.

Parameters **path** (*str*) – path to the file

Returns md5sum of the file at the given path, as a hex digest

Return type str

s3sfe.utils.**read_filelist**(*path*)
Given the path to the filelist, read the file and return a list of all paths contained in it.

Parameters **path** (*str*) – path to filelist

Returns list of files and directories to back up

Return type list

s3sfe.utils.**read_keyfile**(*path*)
Read the AES256 key from the file.

Parameters **path** (*str*) – path to the key file

Returns 32-bit AES256 key

Return type bytes

s3sfe.utils.**set_log_debug**(*_logger*)
set logger level to DEBUG, and debug-level output format

s3sfe.utils.**set_log_info**(*_logger*)
set logger level to INFO

s3sfe.utils.**set_log_level_format**(*_logger*, *level*, *format*)
Set logger level and format.

Parameters

- **level** (*int*) – logging level; see the logging constants.
- **format** (*str*) – logging formatter format string

s3sfe.version module

8.2 Changelog

8.2.1 Unreleased Changes

- Stop testing and supporting py33

8.2.2 0.1.1 (2017-03-17)

- Fix fatal bug in listing items from S3

8.2.3 0.1.0 (2017-03-17)

- Initial version

CHAPTER 9

Indices and tables

- genindex
- modindex
- search

Python Module Index

S

Index