dmarc_report

Release 6.0.0

Gene C

May 06, 2025

CONTENTS:

CHAPTER

ONE

DMARC_REPORT

1.1 Overview

Generate a human readable report from 1 or more standard DMARC and TLS-RPT xml email reports . DMARC reports are made using *dmarc-rpt* while TLS-RPTs use *tls-rpt*

Note:

> All git tags are signed by <arch@sapience.com>. Public key is available via WKD or download from website: https://www.sapience.com/tech After key is on keyring use the PKGBUILD source line ending with *?signed* or manually verify using *git tag -v <tag-name>*

1.2 New / Interesting

New

- Tidy ups: PEP-8, PEP-257, PEP-484 PEP-561
- And Reorganize code especially for PEP-561 (type hints)
- Has passed all tests here, so hopefully no problems. But, always some risk cleaning up code - please let me know if something is not right.

Interesting

- New config file format using single config file. Older 2 fille configs will be automatically converted to the new version 2 format. See *config_files_section* section and *configs* directory for sample config.
- Switch to *py-cidr* package for handling IPs instead of own versions.
- **Available**
  - github <https://github.com/gene-git/py-cidr>
  - AUR <https://aur.archlinux.org/packages/py-cidr>
- Now use python 3's ipaddress module instead of netaddr. Its faster and we no longer require 3rd party library
- Require python version 3.11 or later
- Switch to lxml for better handling of xml namespaces found in some reports
- Add support for handling mbox file with multiple emails containing reports. While some clients save multiple emails in separate *.eml* files, others, like evolution, save them all in a single *.mbox* file. Add support for this.
- tls-rpt

  New tool to generate report for TLS reports for MTA-STS or DANE. See README-tls.md This report has been updated - see Changelog for details.

CHAPTER

TWO

GETTING STARTED

2.1 Applications

Save all DMARC or TLS-RPT reports into a directory. These are typically compressed xml/json files sent as email attachments. The saved reports can be :

- individual email files each with a compressed xml/json attachment. Thunderbird saves them this way. These are saved with a *.eml* extension.
- one single file with several emails, each with the attachment. Evolution saves this way. These are saved with *.mbox* extension.
- Individual compressed, or uncmompressed, xml reports created by saving the attachments from each email.

dmarc-rpt and *tls-rpt* will extract the actual **xml** (*dmarc*) or **json** (tls-rpt) data from all of the above.

2.1.1 Quick start

Save all emails with DMARC or TLS-RPT attachments to a directory, change into that directory and run either dmarc-rpt or tls-rpt as appropriate.

It is generally more convenient to use a config file explained below.

2.1.2 Config Files

Config files are read, in order, from directories :

```
/etc/dmarc_report/
~/.config/dmarc_report/
```

with the settings in latter *~/.config/...* overriding any found in */etc/...*.

There are 2 config file formats supported. The older version 1 format uses 2 separate files:

- *config* - for dmarc-rpt
- *tls-config* - for tls-rpt

New version 2 format uses a single file, *config.v2*. Version 2 config will be used if its found. If only version 1 configs are found they will be automatically converted to version 2, which will then be used going forward.

All config files use standard TOML format. Config files use 3 sections. A global section and one each for dmarc and tls-rpt.

Available config values are set using:

```
command_line_long_opt_name = xxx
```

e.g. to set data report dir use:

```
dir = "/foo/goo/dmarc_reports"
```

A sample config is available in the *conf.d* directory. A typical config might be of the form:

```
# comment
[global]
    theme = 'dark'
    inp_files_disp = "save"
    inp_files_save_dir = "../saved"

[dmarc]
    dom_ips = ['1.1.1.1', '1.2.2.0/24']
    dir = "~/mail-reports/dmarc/xml"

[tls]
    dir = "~/mail-reports/tls/xml"
```

Variables set in *[dmarc]* or *[tls]* sections override any correspodning global ones.

This sample config says to read all the saved dmarc email reports from *~/mail-reports/dmarc/xml* and the tls reports from *~/mail-reports/tls/xml*.

And to save the raw files after processing report by moving them to *~/mail-reports/dmarc/saved* or *~/mail-reports/tls/saved*.

For dmarc it says that ips listed in *dom_ips* are for your own domains.

Command line options override the corresponding config setting. See *Options* section for more detail.

2.1.3 dmarc-rpt Usage

Change to the directory containing the one or more dmarc report files and simply run

```
dmarc-rpt
```

When using the *–dir* option (or config setting *dir*) it is not necessary to change directories before running the report.

Any email files, those ending with *.eml* will be processed first. These are assumed to contain the report as a mime attachment. The attachment is extracted from any such email files. Some mail clients save multiple emails as a single mbox file. Each email in the mbox file will be similarly processed and have the attached report extracted.

Then all remaining files are read and processed. The tool processes all xml and gzip/zip compressed xml dmarc report files and generates a human readable report.

We follow Postel's law and try to be liberal in what we accept as input. To that end we accept the dmarc XML report file, a gzip/zip compressed version of same or a saved email file text file with the report itself being a mime attachment.

Any file with extension *.eml* is treated as an email file.

To avoid line wrapping, the report should be viewed on wide enough terminal; roughly 112 or chars or more.

For convenience after report is generated, the input files can be automatically moved to a save direcory, left where they are or removed. A typical sequents of events is to save the email reports, run dmarc-rpt. By auto moving (or removing) the input files, makes it simpler when doing the next batch of dmarc reports.

Then save all the raw .eml files into ~/dmarc/reports and run before running the report

```
dmarc-rpt
```

All attachments from dmarc email reports would be saved into "~/dmarc/saved/2023-01" in this example.

2.1.4 tls-rpt Usage

tls-rpt works in a similar way to dmarc-rpt, except it operates on TLS-RPT (compressed) xml inputs.

Command line options are shown first in parens below, followed by the corresponding config version in square brackets, if available.

2.1.5 Common Options

These apply to both dmarc-rpt and tls-rpt

- (*-h, –help*) Help for command line options.
- (*-d, –dir*) [*dir = /path/xxx/*]

  Allows specifying the directory with the dmarc report files to be processed. The directory holding the report files (.eml, .xml, .gz or .zip) By default, dir is the current directory.

- (*-k, –keep*)

  Prevent the *.eml* being removed after the attached xml reports are extracted.

- (*-thm, –theme*)

  Report is now in color. Default theme is 'dark'. Theme can be 'light' 'dark' or 'none', which turns off color report.

- (*-v, –verb*)

  More verbose output

- (*-ifd, –inp_file_disp*)

  Input file disposition options one of : none,save,delete If set to save then all input files (xml, compressed xml and any kept eml files) are moved to directory specified by *inp_files_save_dir*.

- (*-ifsd, –inp_files_save_dir*)

  When *inp_file_disp* is set, then input files are moved to this directory after report is generated. Files are saved by year-month under the save directory

2.1.6 dmarc-rpt Specific Options

These are only applicable for dmarc-rpt.

- (*-ips, –dom_ips*) [*dom_ips = [ip, cidr, . . .]*]

  Set the ips for your own domain(s), which will then be colored to make them easy to spot. Command line option is a comma separated list of IPs. e.g.:

  ```
  --dom_ips "1.1.1.0/24,2.2.2.16/29"
  ```

  When used in config file format as array of IP stringsC. e.g.:

  ```
  dom_ips = ['1.1.1.0/24', '2.2.2.16/29']
  ```

- (*fdm, –dmarc_fails*)

  Only include dmarc failures in report

- (*fdk, –dkim_fails*)

  Only include dkim failures in report

- (*fsp, –spf_fails*)

  Only include spf failures in report

2.2 Saving Email Reports From Email Client

In most mail clients, such as thunderbird, one can select multiple email reports and then use *File -> Save As* to save the email files into a directory of your choosing. Each email gets saved with a *.eml* extension.

CHAPTER

THREE

APPENDIX

3.1 Dependencies

- Run Time : * python (3.13 or later) * python-dateutil * python-lxml * py-cidr (2.7.0 or later) * tomli-w (for writing version 2 configs converted from version 1)
- Building Package: * git * wheel (aka python-wheel) * build (aka python-build) * installer (aka python-installer) * poetry (aka python-poetry) - rsync
- Optional for building docs:
  - sphinx
  - texlive-latexextra (archlinux packaguing of texlive tools)

3.2 Installation

Available on

- Github
- Archlinux AUR

On Arch you can build using the PKGBUILD provided in packaging directory or from the AUR package. To build manually, clone the repo and

```
rm -f dist/*
python -m build --wheel --no-isolation
root_dest="/"
./scripts/do-install $root_dest
```

When running as non-root then set root_dest a user writable directory

3.3 Philosophy

We follow the *live at head commit* philosophy. This means we recommend using the latest commit on git master branch. We also provide git tags.

This approach is also taken by Google[1][2].

[1] https://github.com/google/googletest
[2] https://abseil.io/about/philosophy#upgrade-support

3.4 License

Created by Gene C. and licensed under the terms of the MIT license.

- SPDX-License-Identifier: MIT
- Copyright (c) 2023, Gene C

CHAPTER

FOUR

SMTP TLS-RPT

4.1 Overview

Generate a human readable tls report from one or more standard tls report files. These reports are used for a email domain with support for either DANE or MTA-STS or both.

4.1.1 Usage

Run from command line: .. code-block:: bash

    tls-rpt

Generates reports from one or more emailed tls reports. Similar to dmarc-rpt, the tool can consume email files (.eml) or the json attachments (plain or compressed) delivered as part of the usual mts-sts reports - and in directory specified by *inp_files_save_dir*.

tls-rpt is provided as part of the dmarc_report package

Background

TLS Reports are oprionally generated for a mail domaain when so requested by a TXT record in the domain's DNS. The tool parses and summarizes such email reports for human consumption.

SMTP TLS reporting is described by [RFC 8460][1] where it summarizes:

```
A number of protocols exist for establishing encrypted channels
between SMTP Mail Transfer Agents (MTAs), including STARTTLS, DNS-
Based Authentication of Named Entities (DANE) TLSA, and MTA Strict
Transport Security (MTA-STS).
```

MTA-STS, is explained by [RFC 8641][2] where it is summarized:

```
SMTP MTA Strict Transport Security (MTA-STS) is a mechanism enabling
mail service providers (SPs) to declare their ability to receive
Transport Layer Security (TLS) secure SMTP connections and to specify
whether sending SMTP servers should refuse to deliver to MX hosts
that do not offer TLS with a trusted server certificate.
```

while DANE is documented in [RFC 6698][3], [RFC 7671][4] and [RFC 7672][5]

[1] TLS Report [RFRC 8460] https://www.rfc-editor.org/rfc/rfc8460.txt
[2] MTA-STS [RFC 8461] https://www.rfc-editor.org/rfc/rfc8461.txt
[3] DANE [RFC 6698] https://www.rfc-editor.org/rfc/rfc6698.txt
[4] DANE [RFC 7671] https://www.rfc-editor.org/rfc/rfc7671.txt
[5] DANE SMTP [RFC 7672] https://www.rfc-editor.org/rfc/rfc7672.txt

```
Encrypted communication on the Internet often uses Transport Layer
Security (TLS), which depends on third parties to certify the keys
used.  This document improves on that situation by enabling the
administrators of domain names to specify the keys used in that
domain's TLS servers.  This requires matching improvements in TLS
client software, but no change in TLS server software
```

Discussion

To receive TLS reports requires a DNS record requesting a TLS report along with either a DANE TLSA record or MTA-STS. MTA-STS requires both a policy and and a DNS record.

4.1.2 TLS Report DNS Record

Example

_smtp._tls.example.org IN TXT "v=TLSRPTv1;rua=mailto:tlsrpt@example.com"

The TLS reports will be sent to the email provided by the string following *rua=*. In this example reports would be sent to *tlsrpt@example.com*.

4.1.3 MTA-STS

Requieres both a DNS record and a policy file available from the email's domain web server.

Policy file example to be provided by web server:

```
https://mta-sts.example.com/.well-known/mta-sts.txt
```

The policy mode can be set to *enforce* or *testing*. Example *mta-sts.txt* file:

```
version: STSv1
mode: enforce
mx: example.com
mx: \*.example.com
max_age: 1296000
```

DNS TXT record example:

```
_mta-sts.example.org.  IN TXT "v=STSv1; id=202301011200;"
```

4.1.4 DANE TLSA

DNS record example:

```
_25._tcp.example.com. TLSA 3 1 1 (xxx)
```

where xxx would be the appropriate public key hash.

Using tls-rpt

Save all tls email reports into a directory. Change to the directory containing one or more dmarc report files and simply run .. code-block:: back

tls-rpt

Using the –dir option (or setging the config option *dir*) makes unnecessary to change directories before running the report.

Any email files, those ending with *.eml* will be processed first. These are assumed to contain the dmarc report as a mime attachment. The attachment is extracted from such email files.

Subsequently, all remaining files are read and processed. The tool processes all json and gzip/zip compressed json tls report files and produces a human readable report.

Any file with extension *.eml* is treated as an email file.

For convenience after report is generated, the input files can be automatically moved to a save direcory, left where they are or removed. A typical sequents of eveents is to save the email reports, run dmarc-rpt. By auto moving (or removing) the input files, makes it simpler when doing the next batch of dmarc reports.

For example, you might save all .eml files in same directory and with config settings:

```
dir = "~/tlsrpt/reports"
inp_files_disp = "save"
inp_files_save_dir = "../saved"
```

Then save all the raw .eml files into ~/tlsrpt/reports and run

```
tls-rpt
```

All attachments from email reports would be saved into "~/tlsrpt/saved/2023-01" in this example.

4.1.5 tls-rpt Options

Options are read first from config files then command line. Config files are read from */etc/dmarc_report/config-tls* then *~/.config/dmarc_report/config-tls*. Config files are in standard TOML format.

Config settings use corresponding command line option:

```
long-option = xxx.
```

e.g. to set data report dir in config use

```
dir = /foo/goo/other
```

The command line options are shown first in parens followed by corresponding config in square brackets if available.

- (*-d, –dir*) [*dir = /some/path*]

  Allows specifying the directory with the dmarc report files to be processed. The directory holding the report files (.eml, .json, .gz or .zip) By default, dir is the current directory.

- (*-k, –keep*) [*keep = true*]

  Prevent the *.eml* being removed after the attached xml reports are extracted.

- (*-thm, –theme*)

  Report is now in color. Default theme is 'dark'. Theme can be 'light' 'dark' or 'none', which turns off color report.

- (*-ifd, –inp_file_disp*)

  Input file disposition options one of : none,save,delete If set to save then all input files (xml, compressed xml and any kept eml files) are moved to directory specified by *inp_files_save_dir*.

- (*-ifsd, –inp_files_save_dir*)

  When *inp_file_disp* is set, then input files are moved to this directory after report is generated. Files are saved by year-month under the save directory

- (*-h, –help*)

  Help for command line options.

4.1.6 Saving Email Reports From Email Client

In most mail clients, such as thunderbird, one can select multiple email reports and then use *File -> Save As* to save the email files into a directory of your choosing. Each email gets saved with a *.eml* extension.

4.1.7 License

CHAPTER

FIVE

CHANGELOG

5.1 Tags

```
0.6.0 (2023-01-01) -> 6.0.0 (2025-05-05)
136 commits.
```

5.2 Commits

- 2025-05-05 : **6.0.0**

```
              Tidy ups: PEP-8, PEP-257, PEP-484 PEP-561
              Reorganize code especially for PEP-561 (type hints)
2025-03-15    update Changelog
              update Changelog
```

- 2025-03-15 : **5.1.4**

```
Pre build PDF doc file.
> PKGBUILD includes short changelog (pacman -Qc dmarc_report)
update Changelog.rst
```

- 2025-03-15 : **5.1.3**

```
              Improve README and add theme to sample config
2025-03-14    update Changelog.rst
```

- 2025-03-14 : **5.1.2**

```
              Just update PKGBUILD version
2025-03-13    update Changelog.rst
```

- 2025-03-13 : **5.1.1**

```
update Changelog.rst
```

- 2025-03-13 : **5.1.0**

```
              Add missing config sample file
              update Changelog.rst
              Bug fix by @rikyborg : Typo in ConfData class
2025-03-11    update Changelog.rst
```

- 2025-03-11 : **5.0.2**

```
              Update Readme
2025-03-10    update Changelog.rst
```

- 2025-03-10 : **5.0.1**

```
Require py-cidr >= 2.7.0 which has ip sort fix
update Changelog.rst
```

- 2025-03-10 : **5.0.0**

```
              remove files no longer being used
              New config file format using single config file shared by dmarc and tls
              report generators
                 version 1 configs with 2 files will be automatically converted and␣
↪saved
                 as config.v2
                 Auto conversion pulls in new dependency on tomli-w to write the config
                 file.
              Reorg and simplify config and options code.
2025-02-25    update Changelog.rst
```

- 2025-02-25 : **4.13.2**

```
              Small update to README
2025-02-23    Add html and pdf docs to repo
              update Changelog.rst
```

- 2025-02-23 : **4.13.1**

```
              Change to py-cidr package for network tools.
              Update README
2025-01-11    update Changelog.rst
```

- 2025-01-11 : **4.12.5**

```
              Ensure python version requirement is consistent (README, pyproject,
              PKGBUILD, requirements)
2024-12-31    update Changelog.rst
```

- 2024-12-31 : **4.12.4**

```
Add git signing key to Arch Package
update Changelog.rst
```

- 2024-12-31 : **4.12.3**

```
typo
update Changelog.rst
```

- 2024-12-31 : **4.12.2**

```
Add validpgpkeys to PKGBUILD
update Changelog.rst
```

- 2024-12-31 : **4.12.1**

```
              All git tags are now signed.
              Update SPDX tags
2024-11-28    update Changelog.rst
```

- 2024-11-28 : **4.12.0**

```
              Handle another seconds format in xml file
2024-10-22    update Changelog.rst
```

- 2024-10-22 : **4.11.0**

```
Additional input protections in cidr utils
update Changelog.rst
```

- 2024-10-22 : **4.10.0**

```
              Bug fix when no "dom_ips" set. Resolves issue #2 reported by @g4242
2024-10-20    update Changelog.rst
```

- 2024-10-20 : **4.9.0**

```
remove dead code
update Changelog.rst
```

- 2024-10-20 : **4.8.0**

```
              For completeness, Handle ip address of form ip/prefix
2024-10-19    update Changelog.rst
```

- 2024-10-19 : **4.7.0**

```
              Now use python 3s ipaddress module instead of netaddr.
                Its faster and we no longer require 3rd party module
              Require python version 3.11 or later
2024-08-29    update Changelog.rst
```

- 2024-08-29 : **4.6.0**

```
              Switch to lxml for better handling of namespaces found in some reports
              Now handle namespaces (e.g. GMX uses them)
2023-12-26    update Changelog.rst
```

- 2023-12-26 : **4.3.1**

```
              Add missing dateutil to depends in PKGBUILD
2023-12-10    update Changelog.rst
```

- 2023-12-10 : **4.3.0**

```
              Add support for extracting reports from multiple emails saved into an mbox
              file - evolution saves emails this way
2023-11-28    update Changelog.rst
```

- 2023-11-28 : **4.2.0**

```
               Handle badly formed dmarc report with missing date range
               Switch python build backend to hatch (was poetry)
2023-10-29     update CHANGELOG.md
```

- 2023-10-29 : **4.0.0**

```
               Improve tls-rpt
                     Show policy name (tlsa, sts, none)
                     Show count of each failure result type
                     Now checks all "policies" returned in the json report.
                     Add date ranges to report
2023-09-27     update CHANGELOG.md
```

- 2023-09-27 : **3.10.0**

```
               Reorganize documentation under Docs and migrate to restructured text
2023-07-14     Nicer formatting in README-tls.rst
               update CHANGELOG.md
```

- 2023-07-14 : **3.9.2**

```
Change to 3.9.2
update CHANGELOG.md
```

- 2023-07-14 : **3.9.1**

```
With updated README-tls.rst this time
update CHANGELOG.md
```

- 2023-07-14 : **3.9.0**

```
               Update README with better description of TLS Report and use rst
2023-07-09     update CHANGELOG.md
```

- 2023-07-09 : **3.8.0**

```
               Add any failure details to tls report
2023-05-18     update CHANGELOG.md
```

- 2023-05-18 : **3.7.1**

```
Update build info in README
update CHANGELOG.md
```

- 2023-05-18 : **3.7.0**

```
install: switch from pip to python installer package. This adds optimized
bytecode
update CHANGELOG.md
```

- 2023-05-18 : **3.6.3**

```
PKGBUILD: add python-build to makedepends
update CHANGELOG.md
```

- 2023-05-18 : **3.6.2**

```
              PKGBUILD: build wheel back to using python -m build instead of poetry
2023-05-17    update CHANGELOG.md
```

- 2023-05-17 : **3.6.1**

```
              Simplify Arch PKGBUILD and more closely follow arch guidelines
2023-04-29    update CHANGELOG.md
```

- 2023-04-29 : **3.6.0**

```
              Handle exceptions from bad XML report files
2023-01-21    update CHANGELOG.md
```

- 2023-01-21 : **3.5.0**

```
              Remove duplicate line in options class - has no effect
2023-01-17    update CHANGELOG.md
```

- 2023-01-17 : **3.4.0**

```
              Turn off debug - accidently left on with last release! So sorry
2023-01-12    typo in README-mta-sts.md
2023-01-09    update CHANGELOG.md
```

- 2023-01-09 : **3.3.0**

```
More info about selectors including missing ("-")
update CHANGELOG.md
```

- 2023-01-09 : **3.2.0**

```
Add more info about dkim selectors typically from forwarded mail
update CHANGELOG.md
```

- 2023-01-09 : **3.1.0**

```
              Sort short dkim selector tags before printing
2023-01-07    tweak readme for new tls-rpt tool
              update CHANGELOG.md
```

- 2023-01-07 : **3.0.0**

```
Refactor code some.
Add new tls-rpt to generate reports for MTA-STS TLS reports
update CHANGELOG.md
```

- 2023-01-07 : **2.3.0**

```
              Bug fix - clean up went too far added silly print bug - so sorry
2023-01-06    tidy README, add SPDX license line to missed file
              update CHANGELOG.md
```

- 2023-01-06 : **2.2.1**

```
               Use SPDX licensing.
               Lint and tidy
2023-01-05     Fix description of input file disposition to show none,save,delete
               update CHANGELOG.md
```

- 2023-01-05 : **2.2.0**

```
               Add option for disposition of input files after report is generated.
                  --inp_files_disp can be none, save or delete.  Default is none.
                  --inp_files_save_dir specifies where to save input files when␣
↪disposition
                  is "save"
2023-01-03     update CHANGELOG.md
```

- 2023-01-03 : **2.1.0**

```
Right align numbers
small tweak to README
update CHANGELOG.md
```

- 2023-01-03 : **2.0.0**

```
Fix bug where grand total missed orgs with 1 IP
Add color report, default theme is dark. Can be light, dark or none to turn
color off
Add support for config files: /etc/dmarc_report/config -
~.config/dmarc_report/config
  Config file is TOML format where each variable is the long_option name:
  e.g. dir = "/a/b/dmarc_stuff"
Add new option to set your IP or CIDR blocks - this will allow your own IPs
to be colored
  Makes it easy to spot mail generated from your own IP vs mail lists etc
update CHANGELOG.md
```

- 2023-01-03 : **1.3.1**

```
               Improve report format a bit
2023-01-02     typo
               small README tweak
               update CHANGELOG.md
```

- 2023-01-02 : **1.3.0**

```
silly bug with multipart accidenlty ignoring report file
update CHANGELOG.md
```

- 2023-01-02 : **1.2.1**

```
remove reference to ripmime - no longer needed now that we handle mime
attachments ourselves
update CHANGELOG.md
```

- 2023-01-02 : **1.2.0**

```
Fix bug with some multipart mime email from some reporters
update CHANGELOG.md
```

- 2023-01-02 : **1.1.0**

```
*.eml* files are now removed after the dmarc report is extracted.
   Use option *-k, --keep* to prevent the *.eml* being removed
update CHANGELOG.md
```

- 2023-01-02 : **1.0.0**

```
Added support to extract dmarc reports from mime attachments in email files
    Added option *-d, --dir* to specify the directory containing report
    files
more readme tweaks
tweak readme
update CHANGELOG.md
```

- 2023-01-02 : **0.9.1**

```
               Add note on handling email reports efficiently to README
2023-01-01     remove unused file
               update CHANGELOG.md
```

- 2023-01-01 : **0.9.0**

```
Small tweak to report output
fix typo
update CHANGELOG.md
```

- 2023-01-01 : **0.8.1**

```
update readme
update CHANGELOG.md
```

- 2023-01-01 : **0.8.0**

```
bump vers to 0.8.0
update CHANGELOG.md
```

- 2023-01-01 : **0.7.0**

```
prep for release
```

- 2023-01-01 : **0.6.0**

```
initial commit
```

CHAPTER

SIX

MIT LICENSE

CHAPTER

SEVEN

HOW TO HELP WITH THIS PROJECT

Thank you for your interest in improving this project. This project is open-source under the MIT license.

7.1 Important resources

- Git Repo

7.2 Reporting Bugs or feature requests

Please report bugs on the issue tracker in the git repo. To make the report as useful as possible, please include

- operating system used
- version of python
- explanation of the problem or enhancement request.

7.3 Code Changes

If you make code changes, please update the documentation if it's appropriate.

CHAPTER

EIGHT

CONTRIBUTOR COVENANT CODE OF CONDUCT

8.1 Our Pledge

In the interest of fostering an open and welcoming environment, we as contributors and maintainers pledge to making participation in our project and our community a harassment-free experience for everyone, regardless of age, body size, disability, ethnicity, sex characteristics, gender identity and expression, level of experience, education, socio-economic status, nationality, personal appearance, race, religion, or sexual identity and orientation.

8.2 Our Standards

Examples of behavior that contributes to creating a positive environment include:

- Using welcoming and inclusive language
- Being respectful of differing viewpoints and experiences
- Gracefully accepting constructive criticism
- Focusing on what is best for the community
- Showing empathy towards other community members

Examples of unacceptable behavior by participants include:

- The use of sexualized language or imagery and unwelcome sexual attention or advances
- Trolling, insulting/derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as a physical or electronic address, without explicit permission
- Other conduct which could reasonably be considered inappropriate in a professional setting

8.3 Our Responsibilities

Maintainers are responsible for clarifying the standards of acceptable behavior and are expected to take appropriate and fair corrective action in response to any instances of unacceptable behavior.

Maintainers have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, or to ban temporarily or permanently any contributor for other behaviors that they deem inappropriate, threatening, offensive, or harmful.

8.4 Scope

This Code of Conduct applies both within project spaces and in public spaces when an individual is representing the project or its community. Examples of representing a project or community include using an official project e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event. Representation of a project may be further defined and clarified by project maintainers.

8.5 Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported by contacting the project team at <arch@sapience.com>. All complaints will be reviewed and investigated and will result in a response that is deemed necessary and appropriate to the circumstances. The Code of Conduct Committee is obligated to maintain confidentiality with regard to the reporter of an incident. Further details of specific enforcement policies may be posted separately.

8.6 Attribution

This Code of Conduct is adapted from the Contributor Covenant, version 1.4, available at https://www.contributor-covenant.org/version/1/4/code-of-conduct.html

8.7 Interpretation

The interpretation of this document is at the discretion of the project team.

CHAPTER

NINE

INDICES AND TABLES

- genindex
- modindex
- search