onionrouter Documentation

Release 0.4.0

Ehlo Onion

May 02, 2017

Contents

Contents:

CHAPTER 1

OnionRouter

Python Onion Routed Mail Deliveries

- Free software: GNU General Public License v3
- Documentation: https://onionrouter.readthedocs.io.

What is this ?

This python script implements dynamic SRV lookups to be fed to postfix as TCP transport map replies. More information on mail delivery over onion services can be found at https://github.com/ehloonion/onionmx.

An alternative implementation in Go can be found at https://git.autistici.org/ale/postfix-onion-transport

Features

- Interactive/Client/Daemon/Debug modes
- Configurable SRV lookup string
- Domain whitelisting
- Lazy rerouting using static mappings

How to run

There are two ways to install onionrouter, automatically using pip or cloning the repository and manually installing the needed packages on Debian. Currently onionrouter has only been tested on Debian Jessie.

Using pip

```
$ sudo pip install onionrouter
```

Test functionality

```
$ onionrouter --help
```

Manual installation on Debian Jessie

Clone repository

```
$ git clone https://github.com/ehloonion/onionrouter.git
```

Install Debian packages

onionrouter has only been tested on Debian Jessie. Install the following packages:

```
$ sudo apt install python-dnspython python-yaml
```

Test functionality

```
$ python onionrouter_run.py --help
```

Configuration and other options

- Copy or update the onionrouter.ini file and with your settings (reference file is in onionrouter/configs folder if you cloned the git repo or in /etc/onionrouter/ if you installed the package)
- **Edit the config**
  - Under the DOMAIN section replace the value of the **hostname** key with your local domain to be whitelisted from lookups.
  - Under the RESOLVER section put in the **resolver_ip** field your preferred resolver (default is 127.0.0.1). To use multiple resolvers, seperate them with comma ‘,’
  - Under the RESOLVER section put in the **resolver_port** field the port that your resolver listens to (default is 53)

onionrouter by default queries the destination domain for a specific SRV record, *_onion-mx._tcp.* and if it finds a .onion address in the reply it gives it back to postfix to be used by the smtptor service defined in master.cf. If no valid SRV record is found the mail is passed to smtp service. This gives us dynamic SRV lookups that lead to SMTP over onion addresses!

- To change the SRV record the scripts looks for, edit the config file mentioned above and change under the DNS section the srv_record field with the SRV record you have setup (default is _onion-mx._tcp.)

- To change the service that will be used when a .onion address is found, edit the config file mentioned above and change under the REROUTE section the onion_transport field with the service you want to be used (default is smtptor)

Execution options

onionrouter by default runs in server mode and acts as a daemon waiting for connections.

Daemon mode can be configured with the following options:

- **–port PORT** or **-p PORT** to define port for daemon to listen (default 23000)
- **–host HOST** or **-l HOST** to define host for daemon to listen (default 127.0.0.1)

Other options are supported as well:

- **–mappings MAPPINGS** to define absolute path to static mappings folder (everything inside will be parsed as a yaml file) or yaml file
- **–config CONFIG** to define the absolute path to config folder (must contain a onionrouter.ini file inside) or config file
- **–client** or **-c** to connect to the daemon and ineract with. Use the host and port options to define the options for the connection to the daemon
- **–debug** or **-d** to start the daemon in debug mode. In this mode, daemon will also print (besides replying) the queries and answers Use the host and port options to define the options for the daemon
- **–interactive** or **-i** to run onionrouter in interactive input mode for debugging or testing purposes without daemon

How to run

Currently onionrouter runs in the foreground, so you need to either run it via a systemd unit file or through some other daemonizing methond (eg screen/tmux/etc). An example systemd unit is included in the *contrib* directory, modify it to your liking.

```
$ python onionrouter_run.py --config /srv/onionrouter/onionrouter/configs/onionrouter.
↪ini --mappings /srv/onionrouter/onionrouter/configs/map.yml -p 23002 --debug
```

There are two ways to install onionrouter, automatically using pip or cloning the repository and manually installing the needed packages on Debian. Currently onionrouter has only been tested on Debian Jessie.

CHAPTER 2

Using pip

```
$ sudo pip install onionrouter
```

Test functionality

```
$ onionrouter --help
```

CHAPTER 3

Manual installation on Debian Jessie

Clone repository

```
$ git clone https://github.com/ehloonion/onionrouter.git
```

Install Debian packages

onionrouter has only been tested on Debian Jessie. Install the following packages:

```
$ sudo apt install python-dnspython python-yaml
```

Test functionality

```
$ python onionrouter_run.py --help
```

CHAPTER 4

Contributing

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

You can contribute in many ways:

Types of Contributions

Report Bugs

Report bugs at https://github.com/ehloonion/onionrouter/issues.

If you are 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.

Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

Write Documentation

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

Submit Feedback

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

Get Started!

Ready to contribute? Here's how to set up *onionrouter* for local development.

1. Fork the *onionrouter* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/onionrouter.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv onionrouter
$ cd onionrouter/
$ python setup.py develop
```

4. Create a branch for local development:

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

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 onionrouter tests
$ python setup.py test or py.test
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. 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
```

7. Submit a pull request through the GitHub website.

Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.7, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/ehloonion/onionrouter/pull_requests and make sure that the tests pass for all supported Python versions.

Tips

To run a subset of tests:

```
$ py.test tests.test_onionrouter
```

CHAPTER 5

Indices and tables

- genindex
- modindex
- search