Contribution and Compliance Guidelines

- Two-factor authentication (2FA)
- License
  - Code License identification
- Copyright Notices
  - Example of the SPDX short-form license identifiers and copyright notice in a source file
  - Get your file headers in order using REUSE
- Contribution sign off
  - Useful tools to make doing DCO signoffs easier
  - Signoff for commits where the DCO signoff was missed
  - Handling DCO errors using GitHub website commits
- Dependency Management

This document captures the general guidelines for contributing to open source projects hosted by LF Energy Foundation.

Note that each hosted project may adopt its own guidelines, which would supersede these provisions in the case of conflict.

This page contains some tips on how to get a grip on copyright and licensing for new and existing projects. Many great tools and guides already exist, this is just an LF Energy specific selection based on our experiences.

Two-factor authentication (2FA)

To enable stronger security for hosted projects, LF Energy Foundation TAC requires all hosted projects to require Two-factor authentication (2FA) on those accounts with privileges in accessing code repositories. Instructions for GitHub are below...

- Configuring 2FA for your GitHub account
- Accessing GitHub using 2FA
- Recovering your account if you lose your 2FA credentials

License

Generally open source projects at LF Energy Foundation that have not previously selected an open source license leverage the Apache License, Version 2.0 for their codebase, a Community Data License Agreement (CDLA) license for any data sets, and the Creative Commons Attribution 4.0 International License for all documentation and non-code assets. These licenses are widely used and understood by both developers and organizations alike, providing flexibility for downstream usage and patent protection for those contributing code.

LF Energy Foundation provides a service for it's projects to manage license compliance, which you can read more about here.

Code License identification

Each repository must contain a license file. Include the plain-text version of the license as a LICENSE file in the top-level directory of the repository.

All source files need to include a header to clearly show the license. LF Energy Foundation recommends the use of SPDX short-form license identifiers in source code files, which vastly reduces errors in copy and pasting license text and enables the headers to be machine-readable. Examples of the use of SPDX short-form license identifiers can be found at https://spdx.org/ids.

Copyright Notices

LF Energy Foundation TAC has been recommending that contributors to a new project establish a common format for copyright notices in their own code. This can help minimize compliance burdens that might otherwise require downstream distributors to reproduce a large number of variations in years, entity names, and formats for notices. We recommend a common copyright notice in a form similar to the following, which does not refer to years or specific contributing entities:

```
Copyright Contributors to the __________ Project.
```

For clarity, we would not recommend removing a third party's license or copyright notice in any circumstance. If a third-party dependency is added to a repository, its license and copyright notices should be preserved and should not be modified or removed.

More insight and background on this recommendation can be found in this blog post.

Example of the SPDX short-form license identifiers and copyright notice in a source file

Assumes [Apache License, Version 2.0] and Foo project name.

```
    # SPDX-License-Identifier: Apache-2.0
# Copyright Contributors to the Foo Project.
```

Get your file headers in order using REUSE

It is best practice to track copyright- and license information on a per-file basis. Writing this down per file allows you to be more precise about copyright which helps people interested in using or contributing.

The REUSE project defines a specification that ensures that copyright information of the project is clear and can be analyzed in an automated fashion. This specification is based on other standards like the use of SPDX identifiers and community best practices. So it is likely you are already mostly complying with the REUSE specification.

Apart from the specification, REUSE is a small command line utility that can be used to properly add copyright and license information and verify against the specification. Some example commands are provided here for quick reference. More documentation can be found on the REUSE getting started page.

Install REUSE and check if your project is REUSE compliant

```
    $ pip install reuse
$ reuse lint
```

Add copyright and license information to all files that are automatically recognised

```
    $ git ls-files | xargs reuse addheader -c "Alliander N.V." -l MPL-2.0 -y 2020 --skip-unrecognised
```

Add copyright and license information to specific files that were not recognised

```
    $ reuse addheader -c "Alliander N.V." -l MPL-2.0 -y 2019-2021 --template=MPL-2.0-full --explicit-license app
/resources/testfile.json
```

Download all licenses for all recognised licenses in the project

```
    $ reuse download --all
```

To ensure that your project stays REUSE compliant you can include a check in your build pipeline. There is a dedicated GitHub Action you can run to achieve this (just copy the template).

REUSE also offers a badge service to show off your compatibility.

Contribution sign off

Ensuring a clean code pedigree and lineage is critical to downstream adoption of open source code in the industry.

LF Energy Foundation requires the use of the Developer's Certificate of Origin 1.1 (DCO), which is the same mechanism that the Linux® Kernel and many other communities use to manage code contributions. The DCO is considered one of the simplest tools for sign-offs from contributors as the representations are meant to be easy to read and indicating signoff is done as a part of the commit message.

Here is an example Signed-off-by line, which indicates that the submitter accepts the DCO:

```
    Signed-off-by: John Doe <john.doe@example.com>
```

You can include this automatically when you commit a change to your local git repository using `git commit -s`.

Useful tools to make doing DCO signoffs easier

There are a number of great tools out there to manage DCO signoffs for developers to make it much easier to do signoffs.

- DCO command line tool, which lets you do a single signoff for an entire repo (https://github.com/coderanger/dco)
- GitHub UI integrations for adding the signoff automatically (https://github.com/scottrigby/dco-gh-ui)
    - Chrome - https://chrome.google.com/webstore/detail/dco-github-ui/onhgmjhnaeipfgacbglaphlmllkpoijo
    - Firefox - https://addons.mozilla.org/en-US/firefox/addon/scott-rigby/?src=search

Additionally, it is possible to use shell scripting to automatically apply the sign-off. Here is an example for bash, to be put into a .bashrc file:

```
    git() {
    if [[ $1 == "commit" ]]; then
        shift
        echo "Executing git commit -s $@"
        command git commit -s "$@"
    else
        command git "$@"
    fi
}
```

Signoff for commits where the DCO signoff was missed

When bringing in a code repository for the first time, or commits done before the DCO checks are enabled, there would be a series of commits that don't include the sign-off statement. You can retroactively signoff commits you've made by making a commit with your DCO signoff that contains a new text file (the suggested name is past_commits.txt) with the following contents:

```
    The following commits were made pursuant to the Developer Certificate of Origin, even though a Signed-off-
by: was not included in the commit message.

<COMMIT HASH> <COMMIT MSG>
...
```

Each user who has made the past commits should have their own `Signed-off-by:` line in the commit message.

This process can be automated using the Contrib check tool.

Handling DCO errors using GitHub website commits

The Probot: DCO app requires that the email address and name specified in the DCO Signoff match that of the current information from the user making the commit. Generally, this is handled automatically when using a local git client, but when making contributions from the GitHub website directly this needs to be aligned manually.

If you are using one of the recommended GitHub UI integrations for adding the signoff automatically, you will want to ensure that the name and email listed there match that which is in your GitHub profile.

Examples of the UI elements to match are below

DCO GitHub UI Configuration blocked URL

blocked URL

GitHub user profile (https://github.com/settings/profile)

Dependency Management

A project typically includes external dependencies. As these dependencies have their own licenses, it is important to keep an eye on these licenses. These licenses should be compatible among the dependencies and between the dependencies and your project code. As you select other dependencies or different versions of dependencies, these relations can change. Therefore it is important to continuously monitor the relevant licenses. If the dependency is bundled in the code base, you should add a THIRD_PARTY file that identifies the components and licenses for those components.

The Linux Foundation provides services for their open source projects through the LFX Tools. One of those tools LFX Security provides insight into security issues as well as license details of project dependencies. You need a login to be able to view this information. The LFX Security documentation provides some example overviews of scan results and how to configure this for your repositories.