APERTIS

Open Source License Expectations

Contents

The license is an important element in open source projects as the license define acceptable use cases, user rights, and contribution guidelines. There are different ways to identify the license from the project source code such as SPDX headers, the LICENSE file, and the COPYING file. However an open source project may contain files from other projects and may use different licenses for different files.

Apertis goals

Apertis aims to accomplish the following goals:

- Ensure that all the software shipped in Apertis is open source or at least freely distributable, so that downstreams are entitled to use, modify and redistribute work derived from our deliverables.
- Ensure that Apertis images targeting devices (such as target and minimal), are not subject to licensing constraints that may conflict with the regulatory requirements of some intended use cases.

In order to reach these goals, the below assumptions are made:

- **Licenses declared by open source projects are correct:** The software authors correctly document the licensing of their released software sources and that they have all the rights to distribute it under the documented terms.
- **Licenses verified by the Debian project are correct:** The package distributors (that is, Debian maintainers and the FTP Masters team) check that the licensing terms provided by the software authors are open source using the definitions in the Debian Free Software Guidelines[1] and ensure those terms are documented in a canonical location (debian/copyright in the package sources).

Licensing constraints

Apertis currently limits the usage of the licenses below:

- GPL-3.0 and derivatives (LGPL-3, AGPL-3)
- BSD-4-Clause

GPL-3 and derivatives

Version 3 of the GPL license[2] was created to address the concern of users who were prevented from running modified code on their device, when the device was shipped with open source software. A common method for preventing users to run their own code is by using signature verification. This practice is known as Tivoization[3]. Those licensing rules are a constraint because in some application domains, it is a regulatory (or safety) requirement to ensure that the hardware runs verified software.

Original 4 clause BSD license

The BSD-4-Clause[4] license still contains the problematic advertisment clause that was dropped in later versions and is thus to be avoided in Apertis.

The original authors of the license retroactively deleted the problematic clause on the software under the University of California copyright, leading to the BSD-4-Clause-UC[5] variant which resolves the issue on the original software, but not on software with different copyright holders.

[1] https://www.debian.org/social_contract#guidelines

[2] https://spdx.org/licenses/GPL-3.0-or-later.html

[3] https://en.wikipedia.org/wiki/Tivoization

[4] https://spdx.org/licenses/BSD-4-Clause.html

[5] https://spdx.org/licenses/BSD-4-Clause-UC.html

Ensuring continuous maintenance of open source licence documentation

Maintaining the open source licenses documentation is an incremental process:

At the time of rebase, licenses are checked manually for all packages involved in the rebase. This covers the whole archive.

During the development, updates are monitored. The integration of a new project in Apertis and the update of source code are the operations that can result in the update of a license. New projects can be integrated at any time in Apertis. If new sources for a project already in Apertis are received: the license of the project can change, or the licensing for some distributables within this project can differ from the prevalent license.

From a project perspective, Apertis teams tries to do a full scan on all projects at each release cycle.

Open source software shipped with devices that users buy adds significant licensing constraints to the software stack of preview and product releases. These constraints do not affect development releases, and it is possible to save some work on those releases.

In an ideal situation, regular checks of the whole archive would be automated to ensure nothing escaped the manual checks. While the Apertis maintainers[6] are already manually checking packages, the automated whole-archive checks are not currently implemented. Future improvements presents a possible solution.

Apertis Licensing expectations

General rules of the Apertis project and their specific constraints

The Debian Free Software Guidelines[7] defines expectations for the licenses of the projects that are integrated in Debian. They serve as a base for Apertis policy. The DFSG can be read in the Appendix section of this document.

On top of the DFSG expectations, Apertis defines additional rules for specific sections of its package repository which are described in Apertis specific rules. In particular, the sections in the Apertis package repository are meant to group the packages that are installed on images for target devices and should thus be free of licensing constraints.

Debian packages in a repository are organized in components. A component is a group of packages sharing a common policy. A single image can incorporate

[6] https://sjoerd.pages.apertis.org/apertis-website/policies/contributions/#the-role-of-maintainers

[7] https://www.debian.org/social_contract#guidelines

packages from different components.

Apertis Repository component specific rules

The canonical source of Licensing information is this document. Each repository is listed here, with the rules that apply.

Each component contains several source packages, and each source package can generate multiple binary packages. For example, in a client server project, it's possible for a source package to generate two binary packages: one for the server side of a project, and one for the client side. Each binary package can have a different license.

For current apertis releases, the following components exist:

- target: contains packages for the final devices,
- hmi: contains user interfaces packages,
- sdk: contains packages specific to SDK
- development: contains packages useful for developers

The license expectations for each of those components are defined below. Any package outside these expectations should be documented as license exceptions[8].

target

This component ships source packages producing binary packages used in images deployable on target devices. For a file in a binary package to be considered an artifact, the file must have been generated/compiled/translated from a source package. An artifact can be an executable, a library, or any other file that is subject to a license. Specifically, the binary packages installed on those images should not be affected by licensing constraints. This does not mean that every source or binary package in the component must be completely unrestricted:

- source packages may contain restricted build scripts, provided that the license does not affect generated artifacts
- source packages may contain restricted tests or utilities, provided that they are not shipped in the same package as the unrestricted artifacts installed on target images
- binary packages may contain restricted artifacts, provided that they are built from a source package also producing unrestricted packages that are shipped on target images
- binary packages may contain restricted artifacts with added exceptions. The GCC Runtime Library Exception[9] covering `libgcc` is the main example. Those exceptions should be documented as license exceptions[10].

[8] https://sjoerd.pages.apertis.org/apertis-website/policies/license-exceptions/

[9] https://www.gnu.org/licenses/gcc-exception-3.1-faq.html

[10] https://sjoerd.pages.apertis.org/apertis-website/policies/license-exceptions/

hmi

This component has the same usage and constraints as the target component.

sdk

This component ships source packages producing binary packages suitable for images deployable on SDK images. Since the packages hosted in this component are only meant for development purposes, no further requirement is imposed other than the DFSG ones.

development

This component provides the packages needed to build the packages in the `target` repository component but that are not meant to be installed on target devices. Build tools like GNU binutils, the GNU Autotools, or Meson are hosted in this component.

Dependencies of packages in the target component that are not meant to be installed on target images are also hosted in this component. For instance, many source package in the target component also build a binary package containing their tests which are not intended to be part of the target images: the extra dependencies required by the test package but not by the main package are hosted in the development component.

The development component also host development tools that are not part of the target images by default, but that may be useful to install manually on target devices during development. Tools like `strace`, `tcpdump` or `bash` belong to this category.

Since those packages are exclusively intended for a development purpose within the Apertis development team no further requirement is imposed other than the DFSG ones.

Auditing the license of a project

Auditing the license of an imported package depends of the type of the project.

For debian packages, the Debian licensing informations gives a good indication if a project can be integrated in Apertis. Debian maintainers take extreme precaution to ensure that what they redistribute is redistributable. Using the Debian licensing informations provides many benefits:

- vetting licensing terms to ensure they are open source (in particular, as defined in the DFSG)
- ensuring that non DFSG-compliant items are excluded from the source code
- a standardized location for the licensing information (that is, `debian/copyright` in the package source)

- an ongoing effort to make the provided licensing information machine-readable (DEP-5)

Some projects may not be packaged by Debian. In this case, the project source code should contain a document stating the license. Any project that do not provide license information should not be redistributed.

Documenting exceptions

For Apertis, the list of exceptions should mention:

- The project location in Apertis mainly gitlab or OBS.
- The project source package name
- The project component
- The rule the project does not meet that requires the exception
- The reason behind the exception
- The date at which the exception was made
- The name of the person who validated the exception

The canonical source of Licensing exceptions is the license exceptions[11] document.

Apertis derived projects should provide an equivalent location for their specific exceptions.

Future improvements

Manually checking licenses will not scale and may not be done in a deterministic way. Introducing automation is a key.

FOSSology is a license reporting tool. It is described in the Automated License Compliance[12] document along with an approach to enable end-to-end tracking of licensing information. Although we trust the developer to check license, the use of FOSSology could help ensure correct identification.

Appendix

The Debian Free Software Guidelines (DFSG)

1. Free Redistribution

The license of a Debian component may not restrict any party from selling or giving away the software as a component of an aggregate software distribution

[11] https://sjoerd.pages.apertis.org/apertis-website/policies/license-exceptions/

[12] https://sjoerd.pages.apertis.org/apertis-website/concepts/automated-license-compliance/

containing programs from several different sources. The license may not require a royalty or other fee for such sale.

2. Source Code

The program must include source code, and must allow distribution in source code as well as compiled form.

3. Derived Works

The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

4. Integrity of The Author's Source Code

The license may restrict source-code from being distributed in modified form only if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software. (This is a compromise. The Debian group encourages all authors not to restrict any files, source or binary, from being modified.)

5. No Discrimination Against Persons or Groups

The license must not discriminate against any person or group of persons.

6. No Discrimination Against Fields of Endeavor

The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

7. Distribution of License

The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

8. License Must Not Be Specific to Debian

The rights attached to the program must not depend on the program's being part of a Debian system. If the program is extracted from Debian and used or distributed without Debian but otherwise within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the Debian system.

9. License Must Not Contaminate Other Software

The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be free software.

10. Example Licenses

The “GPL”, “BSD”, and “Artistic” licenses are examples of licenses that we consider “free”.