Software Freedom
Law Center

1995 Broadway, 17th Floor
New York, NY 10023–5882
tel +1–212–580–0800
fax +1–212–580–0898
www.softwarefreedom.org

A Practical Guide to GPL Compliance

Bradley M. Kuhn
Aaron Williamson
Karen M. Sandler

August 26, 2008

1 Executive Summary

This is a guide to effective compliance with the GNU General Public License (GPL) and related licenses. In accordance with the Software Freedom Law Center's (SFLC's) philosophy of assisting the community with GPL compliance cooperatively, this guide focuses on avoiding compliance actions and minimizing the negative impact when enforcement actions occur. It introduces and explains basic legal concepts related to the GPL and its enforcement by copyright holders. It also outlines business practices and methods that lead to better GPL compliance. Finally, it recommends proper post-violation responses to the concerns of copyright holders.

2 Background

Early GPL enforcement efforts began soon after the GPL was written by Richard Stallman in 1989, and consisted of informal community efforts, often in public Usenet discussions.[1] Over the next decade, the Free Software Foundation (FSF), which holds copyrights in many GNU programs, was the only visible entity actively enforcing its GPL'd copyrights on behalf of the community of Free/Libre and Open Source Software (FOSS) developers. FSF's enforcement was generally a private process; the FSF contacted violators confidentially and helped them to comply with the license. Most violations were pursued this way until the early 2000's.

By that time, Linux-based systems had become very common, particularly in embedded devices such as wireless routers. During this period, public ridicule of violators in the press and on Internet fora supplemented ongoing private enforcement and increased pressure on businesses to comply. In 2003, the FSF formalized its

[1]One example is the public outcry over NeXT's attempt to make the Objective-C front-end to GCC proprietary.

efforts into the GPL Compliance Lab, increased the volume of enforcement, and built community coalitions to encourage copyright holders to together settle amicably with violators. Beginning in 2004, Harald Welte took a more organized public enforcement approach and launched `gpl-violations.org`, a website and mailing list for collecting reports of GPL violations. On the basis of these reports, Welte successfully pursued many enforcements in Europe, including formal legal action.

In 2007, the SFLC filed the first U.S. copyright infringement lawsuit based on a violation of the GPL. While the lawsuits filed by SFLC on behalf of its clients have been quite public, SFLC resolves the vast majority of enforcement actions privately via cooperative communications with violators. As we have worked to bring individual companies into compliance, we have encountered numerous violations resulting from preventable problems such as inadequate attention to licensing of upstream software, misconceptions about the GPL's terms, and poor communication between software developers and their management. In this document, we highlight these problems and describe best practices to encourage corporate users of FOSS to reevaluate their approach to GPL'd software and avoid future violations.

SFLC continues to conduct GPL enforcement and compliance efforts for many of its clients who release their software under the GPL, the GNU Lesser Public License (LGPL) and other copyleft licenses. In doing so, we have found that most violations stem from a few common mistakes that can be, for the most part, easily avoided. We hope to educate the community of commercial distributors, redistributors, and resellers on how to avoid violations in the first place, and to respond adequately and appropriately when a violation occurs.

3 Best Practices to Avoid Common Violations

Unlike highly permissive FOSS licenses (such as the ISC license), which typically only require preservation of copyright notices, the GPL places a number of important requirements upon licensees. These requirements are carefully designed to uphold certain values and standards of the software freedom community. While the GPL's requirements may appear initially counter-intuitive to those more familiar with proprietary software licenses, by comparison its terms are in fact clear and favorable to licensees. The terms of the GPL actually simplify compliance when violations occur.

GPL violations are often caused or compounded by a failure to adopt sound practices for the incorporation of GPL'd components into a company's internal development environment. In this section, we introduce some best practices for software tool selection, integration and distribution, inspired by and congruent with FOSS methodologies. We suggest companies establish such practices before building a product based on GPL'd software.[2]

3.1 Evaluate License Applicability

Political discussion about the GPL often centers around the "copyleft" requirements of the license. Indeed, the license was designed primarily to embody this licensing feature. Most companies adding non-trivial features (beyond mere porting and bug-fixing) to GPL'd software, and thereby implicating these requirements, are already well aware of their more complex obligations under the license.[3]

However, in our experience with GPL enforcement, few redistributors' compliance challenges relate directly to the copyleft provisions; this is doubly true for most embedders. Instead, the distributions of GPL'd systems that we encounter typically consist of a full operating system including components under the GPL (e.g.,

[2]This document addresses compliance with GPLv2, GPLv3, LGPLv2, and LGPLv3. Advice on avoiding the most common errors differs little for compliance with these four licenses. § 7.1 discusses the key differences between GPL and LGPL compliance.

[3]There has been much legal discussion regarding copyleft and derivative works. In practical reality, this issue is not relevant to the vast majority of companies distributing GPL'd software.

Linux, BusyBox) and components under the LGPL (e.g., the GNU C Library). Sometimes, these programs have been patched or slightly improved by direct modification of their sources, resulting unequivocally in a derivative work. Alongside these programs, companies often distribute fully independent, proprietary programs, developed from scratch, which are designed to run on the FOSS operating system but do not combine with, link to, modify, or otherwise derive from the GPL'd components.[4] In the latter case, where the work is unquestionably a separate work of creative expression, no derivative work has been created. The tiny minority of situations which lie outside these two categories, and thus involve close questions about derivative works, require a highly fact-dependent analysis and cannot be addressed in a general-purpose document.

Most companies accused of violations, however, lack a basic understanding of how to comply even in the straightforward scenario. This document provides that fundamental and generally applicable prerequisite knowledge. For answers to rarer and more complicated legal questions, such as whether your software is a derivative work of some copylefted software, consult with an attorney.[5]

For this discussion, we will assume that you have already identified the "work" covered by the license, and that any components not under the GPL (e.g., applications written entirely by your developers that merely happen to run on a Linux-based operating system) distributed in conjunction with those works are separate works within the meaning of copyright law. In such a case, the GPL requires you to provide complete and corresponding source for the GPL'd components and your modifications thereto, but not for independent proprietary applications. The procedures described in this document address this typical scenario.

3.2 Monitor Software Acquisition

Software engineers should have the freedom to innovate and import useful software components to improve your product. However, along with that freedom should come rules and reporting procedures to make sure that you are aware of what software is being tested or included with your product.

The companies we contact about GPL violations often respond with: "We didn't know there was GPL'd stuff in there". This answer indicates a failure in the software acquisition and procurement process. Integration of third-party proprietary software typically requires a formal arrangement and management/legal oversight before the developers incorporate the software. By contrast, your developers often obtain and integrate FOSS without intervention. The ease of acquisition, however, does not mean the oversight is any less necessary. Just as your legal and/or management team negotiates terms for inclusion of any proprietary software, they should be involved in all decisions to bring FOSS into your product.

Simple, engineering-oriented rules help provide a stable foundation for FOSS integration. Ask your software developers to send an email to a standard place describing each new FOSS component they add to the system, and have them include a brief description of how they will incorporate it into the product. Make sure they use a revision control system, and have store the upstream versions of all software in a "vendor branch" or similar mechanism, whereby they can easily track and find the main version of the software and local changes made.

Such procedures are best instituted at your project's launch. Once a chaotic and poorly-sourced development process has begun, the challenges of determining and cataloging the presence of GPL'd components is difficult. If you are in that situation, we recommend the Fossology system, which analyzes a source-code base and produces a list of FOSS licenses that may apply to the code. Fossology can help you build a catalog of the sources you have already used to build your product. You can then expand that into a more structured inventory and process.

[4]However, these programs do often combine with LGPL'd libraries. This is discussed in detail in § 7.1.

[5]If you would like more information on the application of derivative works doctrine to software, a detailed legal discussion is presented in our colleague Dan Ravicher's article, *Software Derivative Work: A Circuit Dependent Determination.*

3.3 Track Your Changes and Releases

As we will explain in further detail below, the most important component to maintaining GPL compliance is inclusion of the complete and corresponding source code in any distributions that you make of GPL'd software. Knowing at all times what sources generated a given binary distribution is paramount.

In an unfortunately large number of our enforcement cases, the violating company's engineering team had difficulty reconstructing the precise sources for a given binary distributed by the company. Ensure that your developers are using revision control systems properly. Have them mark or tag the full source tree corresponding to builds distributed to customers. Finally, check that your developers store all parts of the software development in the revision control system, including READMEs, build scripts, engineers' notes, and documentation. Your developers will also benefit from a system that tracks the precise version of source that corresponds to any deployed binary.

3.4 Avoid the "Build Guru"

Too many software projects rely on only one or a very few team members who know how to build and assemble the final released product. Such knowledge centralization not only creates engineering redundancy issues, but it also endangers GPL compliance, which requires you to provide build scripts.

Avoid relying on a "build guru", a single developer who is the only one who knows how to produce your final product. Make sure the build process is well defined. Train every developer on the build process for the final binary distribution, including (in the case of embedded software) generating a final firmware image suitable for distribution to the customer. Require developers to use revision control for build processes. Make a rule that adding new components to the system without adequate build instructions (or better yet, scripts) is unacceptable engineering practice.

4 Details of Compliant Distribution

In this section, we explain the specific requirements placed upon distributors of GPL'd software. Note that this section refers heavily to specific provisions and language in GPLv2 and GPLv3. It may be helpful to have a copy of each license open while reading this section.

4.1 Binary Distribution Permission

The various versions of the GPL are copyright licenses that grant permission to make certain uses of software that are otherwise restricted by copyright law. This permission is conditioned upon compliance with the GPL's requirements.[6] This section walks through the requirements (of both GPLv2 and GPLv3) that apply when you distribute GPL'd programs in binary (i.e., executable or object code) form, which is typical for embedded applications. Because a binary application derives from a program's original sources, you need permission from the copyright holder to distribute it. § 3 of GPLv2 and § 6 of GPLv3 contain the permissions and conditions related to binary distributions of GPL'd programs.[7]

[6]For a full discussion of this concept, please see the chapter entitled "Common Copyright Questions" in SFLC's publication, *A Legal Issues Primer for Open Source and Free Software Projects*.

[7]These sections cannot be fully understood in isolation; read the entire license thoroughly before focusing on any particular provision. However, once you have read and understood the entire license, look to these sections to guide compliance for binary distributions.

GPL's binary distribution sections offer a choice of compliance methods, each of which we consider in turn. Each option refers to the "Corresponding Source" code for the binary distribution, which includes the source code from which the binary was produced. This abbreviated and simplified definition is sufficient for the binary distribution discussion in this section, but you may wish to refer back to this section after reading the thorough discussion of "Corresponding Source" that appears in § 4.2.

4.1.1 Option (a): Source Alongside Binary

GPLv2 § 3(a) and v3 § 6(a) embody the easiest option for providing source code: including Corresponding Source with every binary distribution. While other options appear initially less onerous, this option invariably minimizes potential compliance problems, because when you distribute Corresponding Source with the binary, *your GPL obligations are satisfied at the time of distribution.* This is not true of other options, and for this reason, we urge you to seriously consider this option. If you do not, you may extend the duration of your obligations far beyond your last binary distribution.

Compliance under this option is straightforward. If you ship a product that includes binary copies of GPL'd software (e.g., in firmware, or on a hard drive, CD, or other permanent storage medium), you can store the Corresponding Source alongside the binaries. Alternatively, you can include the source on a CD or other removable storage medium in the box containing the product.

GPLv2 refers to the various storage mechanisms as "medi[a] customarily used for software interchange". While the Internet has attained primacy as a means of software distribution where super-fast Internet connections are available, GPLv2 was written at a time when downloading software was not practical (and was often impossible). For much of the world, this condition has not changed since GPLv2's publication, and the Internet still cannot be considered "a medium customary for software interchange". GPLv3 clarifies this matter, requiring that source be "fixed on a durable physical medium customarily used for software interchange". This language affirms that option (a) requires binary redistributors to provide source on a physical medium.

Please note that while selection of option (a) requires distribution on a physical medium, voluntary distribution via the Internet is very useful. This is discussed in detail in § 4.1.2.

4.1.2 Option (b): The Offer

Many distributors prefer to ship only an offer for source with the binary distribution, rather than the complete source package. This option has value when the cost of source distribution is a true per-unit cost. For example, this option might be a good choice for embedded products with permanent storage too small to fit the source, and which are not otherwise shipped with a CD but *are* shipped with a manual or other printed material.

However, this option increases the duration of your obligations dramatically. An offer for source must be good for three full years from your last binary distribution (under GPLv2), or your last binary or spare part distribution (under GPLv3). Your source code request and provisioning system must be designed to last much longer than your product life cycle.

In addition, if you are required to comply with the terms of GPLv2, you **cannot** use a network service to provide the source code. For GPLv2, the source code offer is fulfilled only with physical media. This usually means that you must continue to produce an up-to-date "source code CD" for years after the product's end-of-life.

Under GPLv2, it is acceptable and advisable for your offer for source code to include an Internet link for downloadable source *in addition* to offering source on a physical medium. This practice enables those with fast network connections to get the source more quickly, and typically decreases the number of physical media fulfillment requests. (GPLv3 § 6(b) permits provision of source with a public network-accessible distribution only and no physical media. We discuss this in detail at the end of this section.)

The following is a suggested compliant offer for source under GPLv2 (and is also acceptable for GPLv3) that you would include in your printed materials accompanying each binary distribution:

> The software included in this product contains copyrighted software that is licensed under the GPL. A copy of that license is included in this document on page X. You may obtain the complete Corresponding Source code from us for a period of three years after our last shipment of this product, which will be no earlier than 2011-08-01, by sending a money order or check for $5 to:
> GPL Compliance Division
> Our Company
> Any Town, US 99999
>
> Please write "source for product Y" in the memo line of your payment.
>
> You may also find a copy of the source at `http://www.example.com/sources/Y/`.
>
> This offer is valid to anyone in receipt of this information.

There are a few important details about this offer. First, it requires a copying fee. GPLv2 permits "a charge no more than your cost of physically performing source distribution". This fee must be reasonable. If your cost of copying and mailing a CD is more than around $10, you should perhaps find a cheaper CD stock and shipment method. It is simply not in your interest to try to overcharge the community. Abuse of this provision in order to make a for-profit enterprise of source code provision will likely trigger enforcement action.

Second, note that the last line makes the offer valid to anyone who requests the source. This is because v2 § 3(b) requires that offers be "to give any third party" a copy of the Corresponding Source. GPLv3 has a similar requirement, stating that an offer must be valid for "anyone who possesses the object code". These requirements indicated in v2 § 3(c) and v3 § 6(c) are so that non-commercial redistributors may pass these offers along with their distributions. Therefore, the offers must be valid not only to your customers, but also to anyone who received a copy of the binaries from them. Many distributors overlook this requirement and assume that they are only required to fulfill a request from their direct customers.

The option to provide an offer for source rather than direct source distribution is a special benefit to companies equipped to handle a fulfillment process. GPLv2 § 3(c) and GPLv3 § 6(c) avoid burdening noncommercial, occasional redistributors with fulfillment request obligations by allowing them to pass along the offer for source as they received it.

Note that commercial redistributors cannot avail themselves of the option (c) exception, and so while your offer for source must be good to anyone who receives the offer (under v2) or the object code (under v3), it *cannot* extinguish the obligations of anyone who commercially redistributes your product. The license terms apply to anyone who distributes GPL'd software, regardless of whether they are the original distributor. Take the example of Vendor V, who develops a software platform from GPL'd sources for use in embedded devices. Manufacturer M contracts with V to install the software as firmware in M's device. V provides the software to M, along with a compliant offer for source. In this situation, M cannot simply pass V's offer for source along to its customers. M also distributes the GPL'd software commercially, so M too must comply with the GPL and provide source (or M's *own* offer for source) to M's customers.

This situation illustrates that the offer for source is often a poor choice for products that your customers will likely redistribute. If you include the source itself with the products, then your distribution to your customers is compliant, and their (unmodified) distribution to their customers is likewise compliant, because both include source. If you include only an offer for source, your distribution is compliant but your customer's distribution does not "inherit" that compliance, because they have not made their own offer to accompany their distribution.

The terms related to the offer for source are quite different if you distribute under GPLv3. Under v3, you may make source available only over a network server, as long as it is available to the general public and remains active for three years from the last distribution of your product or related spare part. Accordingly, you may satisfy your fulfillment obligations via Internet-only distribution. This makes the "offer for source" option less troublesome for v3-only distributions, easing compliance for commercial redistributors. However, before you switch to a purely Internet-based fulfillment process, you must first confirm that you can actually distribute *all* of the software under GPLv3. Some programs are indeed licensed under "GPLv2, *or any later version*" (often abbreviated "GPLv2-or-later"). Such licensing gives you the option to redistribute under GPLv3. However, a few popular programs are only licensed under GPLv2 and not "or any later version" ("GPLv2-only"). You cannot provide only Internet-based source request fulfillment for the latter programs.

If you determine that all GPL'd works in your whole product allow upgrade to GPLv3 (or were already GPLv3'd to start), your offer for source may be as simple as this:

> The software included in this product contains copyrighted software that is licensed under the GPLv3. A copy of that license is included in this document on page X. You may obtain the complete Corresponding Source code from us for a period of three years after our last shipment of this product and/or spare parts therefor, which will be no earlier than 2011-08-01, on our website at `http://www.example.com/sources/productnum/`.

Under both GPLv2 and GPLv3, source offers must be accompanied by a copy of the license itself, either electronically or in print, with every distribution.

Finally, it is unacceptable to use option (b) merely because you do not have Corresponding Source ready. We find that some companies chose this option because writing an offer is easy, but producing a source distribution as an afterthought to a hasty development process is difficult. The offer for source does not exist as a stop-gap solution for companies rushing to market with an out-of-compliance product. If you ship an offer for source with your product but cannot actually deliver *immediately* on that offer when your customers receive it, you should expect an enforcement action.

4.1.3 Option (c): Noncommercial Offers

As discussed in the last section, GPLv2 § 3(c) and GPLv3 § 6(c) apply only to noncommercial use. These options are not available to businesses distributing GPL'd software. Consequently, companies who redistribute software packaged for them by an upstream vendor cannot merely pass along the offer they received from the vendor; they must provide their own offer or corresponding source to their distributees. We talk in detail about upstream software providers in § 7.2.

4.1.4 Option 6(d) in GPLv3: Internet Distribution

Under GPLv2, your formal provisioning options for Corresponding Source ended with § 3(c). But even under GPLv2, pure Internet source distribution was a common practice and generally considered to be compliant. GPLv2 mentions Internet-only distribution almost as aside in the language, in text at the end of the section after the three provisioning options are listed. To quote that part of GPLv2 § 3:

> If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

When that was written in 1991, Internet distribution of software was the exception, not the rule. Some FTP sites existed, but generally software was sent on magnetic tape or CDs. GPLv2 therefore mostly assumed that binary distribution happened on some physical media. By contrast, GPLv3 § 6(d) explicitly gives an option for this practice that the community has historically considered GPLv2-compliant.

Thus, you may fulfill your source-provision obligations by providing the source code in the same way and from the same location. When exercising this option, you are not obligated to ensure that users download the source when they download the binary, and you may use separate servers as needed to fulfill the requests as long as you make the source as accessible as the binary. However, you must ensure that users can easily find the source code at the time they download the binary. GPLv3 § 6(d) thus clarifies a point that has caused confusion about source provision in v2. Indeed, many such important clarifications are included in v3 which together provide a compelling reason for authors and redistributors alike to adopt GPLv3.

4.1.5 Option 6(e) in GPLv3: Software Torrents

Peer-to-peer file sharing arose well after GPLv2 was written, and does not easily fit any of the v2 source provision options. GPLv3 § 6(e) addresses this issue, explicitly allowing for distribution of source and binary together on a peer-to-peer file sharing network. If you distribute solely via peer-to-peer networks, you can exercise this option. However, peer-to-peer source distribution *cannot* fulfill your source provision obligations for non-peer-to-peer binary distributions. Finally, you should ensure that binaries and source are equally seeded upon initial peer-to-peer distribution.

4.2 Preparing Corresponding Source

Most enforcement cases involve companies that have unfortunately not implemented procedures like our § 3 recommendations and have no source distribution arranged at all. These companies must work backwards from a binary distribution to come into compliance. Our recommendations in § 3 are designed to make it easy to construct a complete and Corresponding Source release from the outset. If you have followed those principles in your development, you can meet the following requirements with ease. If you have not, you may have substantial reconstruction work to do.

4.2.1 Assemble the Sources

For every binary that you produce, you should collect and maintain a copy of the sources from which it was built. A large system, such as an embedded firmware, will probably contain many GPL'd and LGPL'd components for which you will have to provide source. The binary distribution may also contain proprietary components which are separate and independent works that are covered by neither the GPL nor LGPL.

The best way to separate out your sources is to have a subdirectory for each component in your system. You can then easily mark some of them as required for your Corresponding Source releases. Collecting subdirectories of GPL'd and LGPL'd components is the first step toward preparing your release.

4.2.2 Building the Sources

Few distributors, particularly of embedded systems, take care to read the actual definition of Corresponding Source in the GPL. Consider carefully the definition, from GPLv3:

> The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities.

and the definition from GPLv2:

> The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable.

Note that you must include "scripts used to control compilation and installation of the executable" and/or anything "needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities". These phrases are written to cover different types of build environments and systems. Therefore, the details of what you need to provide with regard to scripts and installation instructions vary depending on the software details. You must provide all information necessary such that someone generally skilled with computer systems could produce a binary similar to the one provided.

Take as an example an embedded wireless device. Usually, a company distributes a firmware, which includes a binary copy of Linux[8] and a filesystem. That filesystem contains various binary programs, including some GPL'd binaries, alongside some proprietary binaries that are separate works (i.e., not derived from, nor based on FOSS sources). Consider what, in this case, constitutes adequate "scripts to control compilation and installation" or items "needed to generate, install and run" the GPL'd programs.

Most importantly, you must provide some sort of roadmap that allows technically sophisticated users to build your software. This can be complicated in an embedded environment. If your developers use scripts to control the entire compilation and installation procedure, then you can simply provide those scripts to users along with the sources they act upon. Sometimes, however, scripts were never written (e.g., the information on how to build the binaries is locked up in the mind of your "build guru"). In that case, we recommend that you write out build instructions in a natural language as a detailed, step-by-step README.

No matter what you offer, you need to give those who receive source a clear path from your sources to binaries similar to the ones you ship. If you ship a firmware (kernel plus filesystem), and the filesystem contains binaries of GPL'd programs, then you should provide whatever is necessary to enable a reasonably skilled user to build any given GPL'd source program (and modified versions thereof), and replace the given binary in your filesystem. If the kernel is Linux, then the users must have the instructions to do the same with the kernel. The best way to achieve this is to make available to your users whatever scripts or process your engineers would use to do the same.

These are the general details for how installation instructions work. Details about what differs when the work is licensed under LGPL is discussed in § 7.1, and specific details that are unique to GPLv3's installation instructions are in § 7.3.

[8] "Linux" refers only to the kernel, not the larger system as a whole.

4.2.3 What About the Compiler?

The GPL contains no provision that requires distribution of the compiler used to build the software. While companies are encouraged to make it as easy as possible for their users to build the sources, inclusion of the compiler itself is not normally considered mandatory. The Corresponding Source definition – both in GPLv2 and GPLv3 – has not been typically read to include the compiler itself, but rather things like makefiles, build scripts, and packaging scripts.

Nonetheless, in the interest of goodwill and the spirit of the GPL, most companies do provide the compiler itself when they are able, particularly when the compiler is based on GCC or another FOSS compiler. If you have a GCC-based system, it is your prerogative to redistribute that GCC version (binaries plus sources) to your customers. We in the FOSS community encourage you to do this, since it often makes it easier for users to exercise their software freedom. However, if you chose to take this recommendation, ensure that your GCC distribution is itself compliant.

If you have used a proprietary, third-party compiler to build the software, then you probably cannot ship it to your customers. We consider the name of the compiler, its exact version number, and where it can be acquired as information that *must* be provided as part of the Corresponding Source. This information is essential to anyone who wishes to produce a binary. It is not the intent of the GPL to require you to distribute third-party software tools to your customer (provided the tools themselves are not based on the GPL'd software shipped), but we do believe it requires that you give the user all the essential non-proprietary facts that you had at your disposal to build the software. Therefore, if you choose not to distribute the compiler, you should include a README about where you got it, what version it was, and who to contact to acquire it, regardless of whether your compiler is FOSS, proprietary, or internally developed.

4.3 Best Practices and Corresponding Source

§ 3 and § 4.2 above are closely related. If you follow the best practices outlined above, you will find that preparing your Corresponding Source release is an easier task, perhaps even a trivial one.

Indeed, the enforcement process itself has historically been useful to software development teams. Development on a deadline can lead organizations to cut corners in a way that negatively impacts its development processes. We have frequently been told by violators that they experience difficulty when determining the exact source for a binary in production (in some cases because their "build guru" quit during the release cycle). When management rushes a development team to ship a release, they are less likely to keep release sources tagged and build systems well documented.

We suggest that, if contacted about a violation, product builders use GPL enforcement as an opportunity to improve their development practices. No developer would argue that their system is better for having a mysterious build system and no source tracking. Address these issues by installing a revision system, telling your developers to use it, and requiring your build guru to document his or her work!

5 When The Letter Comes

Unfortunately, many GPL violators ignore their obligations until they are contacted by a copyright holder or the lawyer of a copyright holder. You should certainly contact your own lawyer if you have received a letter alleging that you have infringed copyrights that were licensed to you under the GPL. This section outlines a typical enforcement case and provides some guidelines for response. These discussions are generalizations and do not all apply to every alleged violation.

5.1 Communication Is Key

GPL violations are typically only escalated when a company ignores the copyright holder's initial communication or fails to work toward timely compliance. We urge accused violators to respond very promptly to the initial request. As the process continues, follow up weekly with the copyright holders to make sure everyone agrees on targets and deadlines for resolving the situation.

Ensure that any staff who might receive communications regarding alleged GPL violations understands how to channel the communication appropriately within your organization. Often, initial contact is addressed for general correspondence (e.g., by mail to corporate headquarters or by e-mail to general informational or support-related addresses). Train the staff that processes such communications to escalate them to someone with authority to take action. An unknowledgable response to such an inquiry (e.g., from a first-level technical support person) can cause negotiations to fail prematurely.

Answer promptly by multiple means (paper letter, telephone call, and email), even if your response merely notifies the sender that you are investigating the situation and will respond by a certain date. Do not let the conversation lapse until the situation is fully resolved. Proactively follow up with synchronous communication means to be sure communications sent by non-reliable means (such as email) were received.

Remember that the FOSS community generally values open communication and cooperation, and these values extend to GPL enforcement. You will generally find that FOSS developers and their lawyers are willing to have a reasonable dialogue and will work with you to resolve a violation once you open the channels of communication in a friendly way.

5.2 Termination

Many redistributors overlook GPL's termination provision (GPLv2 § 4 and GPLv3 § 8). Under v2, violators forfeit their rights to redistribute and modify the GPL'd software until those rights are explicitly reinstated by the copyright holder. In contrast, v3 allows violators to rapidly resolve some violations without consequence.

If you have redistributed an application under GPLv2[9], but have violated the terms of GPLv2, you must request a reinstatement of rights from the copyright holders before making further distributions, or else cease distribution and modification of the software forever. Different copyright holders condition reinstatement upon different requirements, and these requirements can be (and often are) wholly independent of the GPL. The terms of your reinstatement will depend upon what you negotiate with the copyright holder of the GPL'd program.

Since your rights under GPLv2 terminate automatically upon your initial violation, *all subsequent distributions* are violations and infringements of copyright. Therefore, even if you resolve a violation on your own, you must still seek a reinstatement of rights from the copyright holders whose licenses you violated, lest you remain liable for infringement for even compliant distributions made subsequent to the initial violation.

GPLv3 is more lenient. If you have distributed only v3-licensed programs, you may be eligible under v3 § 8 for automatic reinstatement of rights. You are eligible for automatic reinstatement when:

- you correct the violation and are not contacted by a copyright holder about the violation within sixty

[9]This applies to all programs licensed to you under only GPLv2 ("GPLv2-only"). However, most so-called GPLv2 programs are actually distributed with permission to redistribute under GPLv2 *or any later version of the GPL* ("GPLv2-or-later"). In the latter cases, the redistributor can choose to redistribute under GPLv2, GPLv3, GPLv2-or-later or even GPLv3-or-later. Where the redistributor has chosen v2 explicitly, the v2 termination provision will always apply. If the redistributor has chosen v3, the v3 termination provision will always apply. If the redistributor has chosen GPLv2-or-later, then the redistributor may want to narrow to GPLv3-only upon violation, to take advantage of the termination provisions in v3.

days after the correction, or

- you receive, from a copyright holder, your first-ever contact regarding a GPL violation, and you correct that violation within thirty days of receipt of copyright holder's notice.

In addition to these permanent reinstatements provided under v3, violators who voluntarily correct their violation also receive provisional permission to continue distributing until they receive contact from the copyright holder. If sixty days pass without contact, that reinstatement becomes permanent. Nonetheless, you should be prepared to cease distribution during those initial sixty days should you receive a termination notice from the copyright holder.

Given that much discussion of v3 has focused on its so-called more complicated requirements, it should be noted that v3 is, in this regard, more favorable to violators than v2.

6 Standard Requests

As we noted above, different copyright holders have different requirements for reinstating a violator's distribution rights. Upon violation, you no longer have a license under the GPL. Copyright holders can therefore set their own requirements outside the license before reinstatement of rights. We have collected below a list of reinstatement demands that copyright holders often require.

- **Compliance on all FOSS copyrights**. Copyright holders of FOSS often want a company to demonstrate compliance for all GPL'd software in a distribution, not just their own. A copyright holder may refuse to reinstate your right to distribute one program unless and until you comply with the licenses of all FOSS in your distribution.
- **Notification to past recipients**. Users to whom you previously distributed non-compliant software should receive a communication (email, letter, bill insert, etc.) indicating the violation, describing their rights under GPL, and informing them how to obtain a gratis source distribution. If a customer list does not exist (such as in reseller situations), an alternative form of notice may be required (such as a magazine advertisement).
- **Appointment of a GPL Compliance Officer.** The FOSS community values personal accountability when things go wrong. Copyright holders often require that you name someone within the violating company officially responsible for FOSS license compliance, and that this individual serve as the key public contact for the community when compliance concerns arise.
- **Periodic Compliance Reports.** Many copyright holders wish to monitor future compliance for some period of time after the violation. For some period, your company may be required to send regular reports on how many distributions of binary and source have occurred.

These are just a few possible requirements for reinstatement. In the context of a GPL violation, and particularly under v2's termination provision, the copyright holder may have a range of requests in exchange for reinstatement of rights. These software developers are talented professionals from whose work your company has benefited. Indeed, you are unlikely to find a better value or more generous license terms for similar software elsewhere. Treat the copyright holders with the same respect you treat your corporate partners and collaborators.

7 Special Topics in Compliance

There are several other issues that are less common, but also relevant in a GPL compliance situation. To those who face them, they tend to be of particular interest.

7.1 LGPL Compliance

GPL compliance and LGPL compliance mostly involve the same issues. As we discussed in § 3.1, questions of modified versions of software are highly fact-dependant and cannot be easily addressed in any overview document. The LGPL adds some additional complexity to the analysis. Namely, the various LGPL versions permit proprietary licensing of certain types of modified versions. These issues are well beyond the scope of this document, but as a rule of thumb, once you have determined (in accordance with LGPLv3) what part of the work is the "Application" and what portions of the source are "Minimal Corresponding Source", then you can usually proceed to follow the GPL compliance rules that we discussed, replacing our discussion of "Corresponding Source" with "Minimal Corresponding Source".

LGPL also requires that you provide a mechanism to combine the Application with a modified version of the library, and outlines some options for this. Also, the license of the whole work must permit "reverse engineering for debugging such modifications" to the library. Therefore, you should take care that the EULA used for the Application does not contradict this permission.

7.2 Upstream Providers

With ever-increasing frequency, software development (particularly for embedded devices) is outsourced to third parties. If you rely on an upstream provider for your software, note that you *cannot ignore your GPL compliance requirements* simply because someone else packaged the software that you distribute. If you redistribute GPL'd software (which you do, whenever you ship a device with your upstream's software in it), you are bound by the terms of the GPL. No distribution (including redistribution) is permissible absent adherence to the license terms.

Therefore, you should introduce a due diligence process into your software acquisition plans. This is much like the software-oriented recommendations we make in § 3. Implementing practices to ensure that you are aware of what software is in your devices can only improve your general business processes. You should ask a clear list of questions of all your upstream providers and make sure the answers are complete and accurate. The following are examples of questions you should ask:

- What are all the licenses that cover the software in this device?
- From which upstream vendors, be they companies or individuals, did *you* receive your software from before distributing it to us?
- What are your GPL compliance procedures?
- If there is GPL'd software in your distribution, we will be redistributors of this GPL'd software. What mechanisms do you have in place to aid us with compliance?
- If we follow your recommended compliance procedures, will you formally indemnify us in case we are nonetheless found to be in violation of the GPL?

This last point is particularly important. Many GPL enforcements are escalated because of petty finger-pointing between the distributor and its upstream. In our experience, agreements regarding GPL compliance issues and procedures are rarely negotiated up front. However, when they are, violations are resolved much more smoothly (at least from the point of view of the redistributor).

Consider the cost of potential violations in your acquisition process. Using FOSS allows software vendors to reduce costs significantly, but be wary of vendors who have done so without regard for the licenses. If your vendor's costs seem "too good to be true," you may ultimately bear the burden of the vendor's inattention to GPL compliance. Ask the right questions, demand an account of your vendors' compliance procedures, and seek indemnity from them.

7.3 User Products and Installation Information

GPLv3 requires you to provide "Installation Information" when v3 software is distributed in a "User Product." During the drafting of v3, the debate over this requirement was contentious. However, the provision as it appears in the final license is reasonable and easy to understand.

If you put GPLv3'd software into a User Product (as defined by the license) and *you* have the ability to install modified versions onto that device, you must provide information that makes it possible for the user to install functioning, modified versions of the software. Note that if no one, including you, can install a modified version, this provision does not apply. For example, if the software is burned onto an non-field-upgradable ROM chip, and the only way that chip can be upgraded is by producing a new one via a hardware factory process, then it is acceptable that the users cannot electronically upgrade the software themselves.

Furthermore, you are permitted to refuse support service, warranties, and software updates to a user who has installed a modified version. You may even forbid network access to devices that behave out of specification due to such modifications. Indeed, this permission fits clearly with usual industry practice. While it is impossible to provide a device that is completely unmodifiable[10], users are generally on notice that they risk voiding their warranties and losing their update and support services when they make modifications.[11]

GPLv3 is in many ways better for distributors who seek some degree of device lock-down. Technical processes are always found for subverting any lock-down; pursuing it is a losing battle regardless. With GPLv3, unlike with GPLv2, the license gives you clear provisions that you can rely on when you are forced to cut off support, service or warranty for a customer who has chosen to modify.

8 Conclusion

GPL compliance need not be an onerous process. Historically, struggles have been the result of poor development methodologies and communications, rather than any unexpected application of the GPL's source code disclosure requirements.

Compliance is straightforward when the entirety of your enterprise is well-informed and well-coordinated. The receptionists should know how to route a GPL source request or accusation of infringement. The lawyers should know the basic provisions of FOSS licenses and your source disclosure requirements, and should explain those details to the software developers. The software developers should use a version control system that allows them to associate versions of source with distributed binaries, have a well-documented build process that anyone skilled in the art can understand, and inform the lawyers when they bring in new

[10]Consider that the iPhone, a device designed primarily to restrict users' freedom to modify it, was unlocked and modified within 48 hours of its release.

[11]A popular t-shirt in the FOSS community reads: "I void warranties.". Our community is well-known for modifying products with full knowledge of the consequences. GPLv3's "Installation Instructions" section merely confirms that reality, and makes sure GPL rights can be fully exercised, even if users exercise those rights at their own peril.

software. Managers should build systems and procedures that keep everyone on target. With these practices in place, any organization can comply with the GPL without serious effort, and receive the substantial benefits of good citizenship in the FOSS community, and lots of great code ready-made for their products.