A Survival Guide To Using GPL

Agenda

Sometimes you find the exact open source component you need only to find out that it is a GPL-licensed component.

We all know that GNU GPL is a copyleft license, which has the potential to be problematic when used in commercial software products. Most companies try to avoid using a copyleft licensed component, so what should you do?

This paper was written by David Mirchin.

David is the head of Meitar's Technology Group. He is a leading expert on software, licensing and intellectual property transactions, as well as on privacy and open source issues. He represents technology companies, including software developers, internet search engines, information aggregators and life sciences companies, and advises on internet, copyright and intellectual property issue in M&A and other transactions.
http://meitar.com/
dmirchin@meitar.com

In this white paper, we will explain the basic terms and conditions of the GNU GPL family license and guide you how to use the GPL properly.

Introduction to Open Source Licenses

When your team uses third party components, they need to make sure your company holds a license that permits use and distribution of it. The component may be under a commercial license, which allows your team to use it, but usually not to modify or distribute it. Of course, your open source components have open source licenses, which do allow you to distribute the software you build using the component – but under certain conditions.

Generally speaking, the importance of holding a license only comes into play if you plan on distributing your software. As long as your developers are only using a component for internal purposes or as an operating system on a back office server, then you probably aren't risking any potential compliance issues.

There are over 70 different open source licenses approved by the Open Source Initiative, with many more open source licenses approved by other organizations. Each one has its own scope of terms and conditions that apply. Some of the common terms addressed by an open source license include:

- What sort of recognition must you give the third party author?
- What statements explaining users' rights to your work you have to include in the license to your software
- How must the source code be distributed and made available to other developers?
- The conditions, if any, under which you don't have to distribute source code
- Under what conditions you can commercially distribute your software

If you are going to distribute your software, then you do need to start thinking about how to do that in a way that complies with any open source license applicable to the open source components you've used. If you've used multiple open source components under different open source licenses, you need to comply with all of them. To add one more wrinkle – you also need to comply with the open source licenses of any dependencies, direct or indirect, to which any of your components is linking.

The GNU GPL License Family

There actually isn't a single type of GNU GPL license. The most recent version of the GPL, version 3, was written mostly as a clarification to the rights and obligations set out in GPL version 2. Version 3 also expands the scope of GPL-compatible licenses and makes it easier to comply with the requirements to provide the source code. Both GPL versions are currently used. In fact, GPLv2 and GPLv3 are estimated to account for up to 35% of open source licenses used in open source projects.

The GNU GPL license family also includes the Lesser GPL (LGPL), which also has two versions in active use. The code sharing requirements under the LGPLs aren't as strict as the GPL requirements. While the GPL requires you to share the code of all 'derivative work', the LGPL requires you to share only the source code of the work 'based' on the LGPL-licenses components and not of the work which 'uses' it. So only "using" an LGPL-licensed component means you are not obligated to share the code that is designed to work with the library by being complied or linked with it - static or dynamic.

Last, is the Affero GPL. The Affero GPL was created to close the major loophole in the GPL

blasted open by the Software-as-a-Service (SaaS) model. The requirement under a GPL to make the source code available under only applies if the licensee *distributes* the code to third parties. Technically, the software being used in a SaaS infrastructure isn't downloaded or distributed. It's running back on licensee's servers, not on their end user's device. Under the traditional GPL, no distribution means no need to make your source code available downstream. Pretty big loop hole undermining the entire open source principle.

So GNU developed the Affero GPL, which imposes the same copyleft rules as a GPL, but specifies that making software available over a server is a form of software distribution. Loophole closed. Under an Affero GPL, the source code of the program must be made available to downstream users, just as the plain GPL requires.

Clarifying the Risks of Using GPL Components

As noted earlier, a GPL requires the release of the component's full source code, along with broad rights to modify and distribute it. When you build software using GPL-licensed components, you must release your work under a GPL compatible license. This requirement applies regardless of the percentage of GPL-licensed code within the program's entire body of code. That means you have to release your source code as well.

Having to release your source code can produce some undesirable results. Anyone, including competitors, can learn how your software is structured and designed. Now they can modify your program, perhaps building out new, high-demand functionality before you can roll it out yourself. Releasing your source code also means that other companies can provide software support and maintenance, seriously diverting a highly profitable revenue stream away from your company.

This is why using a GPL-licensed component presents a higher risk than simple copyright law. Authors own the copyright only to components they've created. Yet, making a component available under a GPL allows them to put conditions on what you can do with the software you create using that GPL-licensed component. No wonder you need to stay on top of what open source components your team links to or combines with your software.

Identifying the Risk

We already talked about distribution as a necessary condition to trigger the GPL code-sharing requirements. The other necessary trigger is when the use of a GPL component "contaminates" your software. Yes, "contamination" is the legal term here. Remember, use of the GPL component in your distributed program requires you to make the source of code of your entire program – not just the GPL components – open and available.

That's the scary news. The good news is that there are license restrictions within the GPL itself that you can use to avoid having to share your proprietary code, which you are not willing to reveal.

The GPL requirement to share source code applies to all 'derivative work' based on the GPL-licensed component. However, this requirement doesn't extend to programs that are "separate and independent" from the GPL-licensed work. Unfortunately, despite the fact that the GPL has been in use for more than 25 years, there is no clear legal definition anywhere in the world that precisely sets out when proprietary software is "separate and independent" from any GPL component used in its building.

Even so, there are ways to reduce your exposure to the risk that use of a GPL component obligates you to release your proprietary source code:

- Use a dynamic link, rather than a static link, between the GPL module and the proprietary software
- Use separate names for the GPL and proprietary modules
- Include separate copyright notices for the GPL and proprietary modules
- Price the software with and without the GPL module, if practical
- Provided the GPL and proprietary modules are downloaded separately
- Design the GPL and proprietary software as separate executables
- Prepare separate documentation for the proprietary and GPL modules

Creating a Company Policy Regarding Use of GPL Components

You've probably already asked yourself whether your company should use GPL components at all. The benefit is clear – access to a never-ending array of free, stable, highly-vetted components that reduce the time to market and costs of developing your own program. But the risks are there, which is why the decision whether to use GPL components can't be left to developers. The majority of them will use whatever is the fastest, easiest means to solve the problem at hand without considering the legal implications.

Every company needs a clear policy regarding how, or if, open source licensed components can be used by your developers. Your "Open Source Policy" needs to be put together by the CTO, CSO, product managers, as well as legal counsel. At a minimum, it must cover the following issues:

- Will the company permit any GPL and Affero GPL components in their products that get distributed? Or can they only be used as a back-end tool that's not part of a distributed product?
- If the use of open source licensed components in a deliverable is allowed, which GPL versions are permitted or prohibited?
- Who is responsible for reviewing and approving whether the open source licensed component can be used before it's actual inclusion in any product?
- Who is responsible for maintaining and updating the record of all open source components used, including a link to their license, and notes whether the component has been modified?

Recent studies have shown that the vast majority of companies (anywhere from 78% to 85% of companies) do use open source software and that GPL is the most widely-used license for open source projects. So I encourage you not to be hasty and decide that avoiding the risk of "contamination" from using GPL components is worth giving up on this endless pool of possibilities.

Using GPL components in a legal and fiscally responsible way is possible. With so many other companies using them, it's also worth considering what sort of disadvantages barring their use throws up to your team by comparison.

Steps to Take When Distributing Software with GPL Components

When you are distributing any software based on a GPL component, there are two key steps you must take to comply with the GPL:

1. Prepare a Notice File that complies with GPL requirements
2. Make all the source code available

Preparing a Notice File

When you distribute a product that contains any open source components, a notice file needs to accompany it. Sometimes called a "header file," the notice files must also be updated as new releases of your program come out.

The notice file must contain the disclosures required by each open source license used in your program. Remember, use of a GPL component means that you're bound by all the open source licenses that apply to all components on which that GPL component is based. They may all be under the same GPL, or the component your team used may be based on a component made available under one of the other 70 types of open source licenses. Your notice file must comply with all the disclosure requirements of all the licenses used.

Typically, a notice file includes:

- Standard copyright notice
- Relevant disclaimers
- Specific license text required by any open source licenses used. In the case of GPL, Affero GPL, or LGPL, a company must include the entire license text, which is available here.

Make Source Code Available

You can distribute the GPL-covered software in object code form, as long as you make the corresponding source code available. You can meet the requirement to make source code available by making it downloadable via a server.

Quantifying the Risk of Violating a GPL

You should do your best to avoid GPL violation, but where are the risks for violating it?

The legal risks are rather low. You've probably read reports about threats stemming from potential GPL violations in the tech media and blogger community. What you haven't seen are many reports of massive legal damages owed, forced market withdrawals, or even formal settlements being enforced. There's a good reason for this.

The goal of the GNU, Free Software Foundation and other open source organizations is to reach full GPL compliance through non-legal actions. Their intention isn't to help developers recover legal damages and take programs off the market. The purposes of these organizations to encourage the free flow of code to inspire creativity and innovation. Heavy-handed legal threats and penalties produce the very opposite results.

No, instead when a violation is discovered, the negotiations begin to bring the company into compliance. Only when it becomes clear a company has no interest in complying with the GPL are legal actions taken.

Consider one of the most well-known legal suits regarding GPL non-compliance: the VMware lawsuit. VMware used a version of Christoph Hellwig's BusyBox on a line of its products. Hellwig made BusyBox available under a GPLv2. When Hellwig found out that VMware wasn't making its source code openly available, as required under the GPL v2 license, the Software Freedom Conservancy opened negotiations with VMware on Hellwig's behalf.

After two years of negotiating, VMware's legal counsel stated outright that VMware had no intention of ceasing its distribution of proprietary-licensed works derived from Hellwig's

and other kernel developers' copyrights, while not releasing its own code. It was only at this point that Hellwig and the Conservancy moved the compliance dispute to court.

The moral of this story is that any company making a good faith error in its use of a GPL component has little to fear from lawsuits. Missing disclosures can be provided. Source code can be made available. If necessary and possible, back-end changes can be made to create the "separate and independent" conditions that protect your proprietary code.

So hopefully it's clear now that the business risks, disclosing a GPL violation in the middle of a due diligence process, are larger than the legal risks. However, even if back-end rework is possible, it's certainly undesirable. This is why enforcing your company's open source use policy and maintaining detailed inventory records from the start of a project are your best protection against any potential risk – business or legal - to using a GPL component.

Preparing for a Software Due Diligence Process (due to investment rounds, acquisitions or public offering)

The due diligence process required when a company looks for funding is exploring combining with another company, or going public has the potential to unveil all manner of non-compliance issues that may have otherwise remained unknown. We offer you this list of guidelines to follow that will help you keep a record of any third party and open source components used and avoid due diligence surprises:

1. Don't allow people to download software, including open source software, onto company computers. It doesn't matter if the downloading is done by employees or contractors. Nor does it matter whether the third party software is proprietary, freeware, shareware, or open source software. It all has to clear your company's license review process, conducted by whoever is designated responsible for your company's open source policy.
2. As part of the license review process, the responsible reviewer should meet with the developers to understand how they intended to use software, and for what purposes. Is this just a back-end tool? Will it be part of deliverable? Do they want to modify it?

3. Categorize each item of software used by your company by its license type, along with its permitted uses. For example, is it approved for general use in executable form only, or can it be used at the source code level in specialized applications or modified applications? Has it been prohibited from any sort of use?
4. As part of your software documentation, include what release version your company is using, and link it to the full text of whatever license applies to that version.
5. Keep in mind any specific restrictions within a proprietary third party software license. A typical proprietary restriction prohibits on reverse engineering or modification, while some proprietary vendors may permit such activities under a special development license or a community source code license. Use your categorization process to document whether your company has access to a proprietary program under one of these broader proprietary licenses or a standard commercial license.
6. Ensure that the source code version is available for download for any GPL components, as well as any code based off the GPL components that aren't "separate and independent" from it.

Having a complete, well-maintained inventory list of your software components and their licenses, open source and otherwise, sends a strong and positive impression of your company's organization and maturity to potential investors or those considering your company for acquisition. If you are a small company with just a handful of developers, you can track your usage manually. However, larger companies find it impossible to accurately track their third party components usage manually and are using automated tool, like WhiteSource, by integrating it into their software development lifecycle (SDLC).

In addition, it will make future transactions, such additional investment rounds or public offering, less expensive and less problematic. Your company doesn't want to get surprised by what's uncovered during due diligence any more than the company that's doing the investigating. Head off any unpleasant surprises by creating and enforcing your software license and usage policy, including having all the necessary documentation that verifies your compliance easily accessible.

Conclusion

Using GPL components in your company's software does require planning and oversight. With the right knowledge and tools, you can mitigate any potential risk of using a GPL component in order to take advantage of the many benefits having access to the wide world of GPL libraries. Hopefully, we've given you a useful overview here. Here's a quick checklist of some of the action points we shared throughout this ebook.

- Assess risk
    - Is the component being used to build a distributed software or just as a back-end tool?
    - If it's used to build a deliverable, have you taken steps to keep your company's propriety work "separate and independent" from the GPL component?
- Develop a company policy that's communicated to developers and makes clear
    - What sort of open source licenses are acceptable and under what circumstances?
    - What sort of communication developers needs to provide to you regarding the use of any open source component?
    - Documentation procedures that must be followed regarding their use
- When using a GPL component
    - Prepare a notice file
    - Make the source code available

If you have questions or suggestions how to improve this to be useful for others, feel free to contact info@whitesourcesoftware.com.

You can also sign up for a free trial of WhiteSource to get a full inventory of all open source components currently being used by your company's software within minutes. Understanding the current scope of your open licensing use is a good first step to building your own company policy on how to manage these valuable resources while minimizing any risk to your company.