WHITEPAPER

Mitigating IP Risk: Three Strategies to Ensure Open Source Compliance

FOSSA

TL;DR

Open Source is free, but there are responsibilities inherent to using open source as it comes with a set of obligations — as well as risks. As open source software continues to be adopted at an increasing rate compliance with open source licenses becomes a more pressing initiative.

Irresponsible usage of open source could result in litigation risk, lowered valuation, loss of market opportunity and damaged reputation which could impact sales and the recruitment of top talent.

Three approaches to mitigating risk in using open source include manual audits, semi-automated compliance and continuous compliance. There are pros and cons to each approach, but continuous compliance fits best for companies leveraging agile development methodologies, DevOps and CI / CD technology tools.

Open source brings about several questions regarding intellectual property (IP). Most of the risk inherent to using open source components involves violating IP law or losing the rights to your IP through the terms of the license (see: copyleft licenses).

For more information about licenses and their implications visit tldrlegal.com.

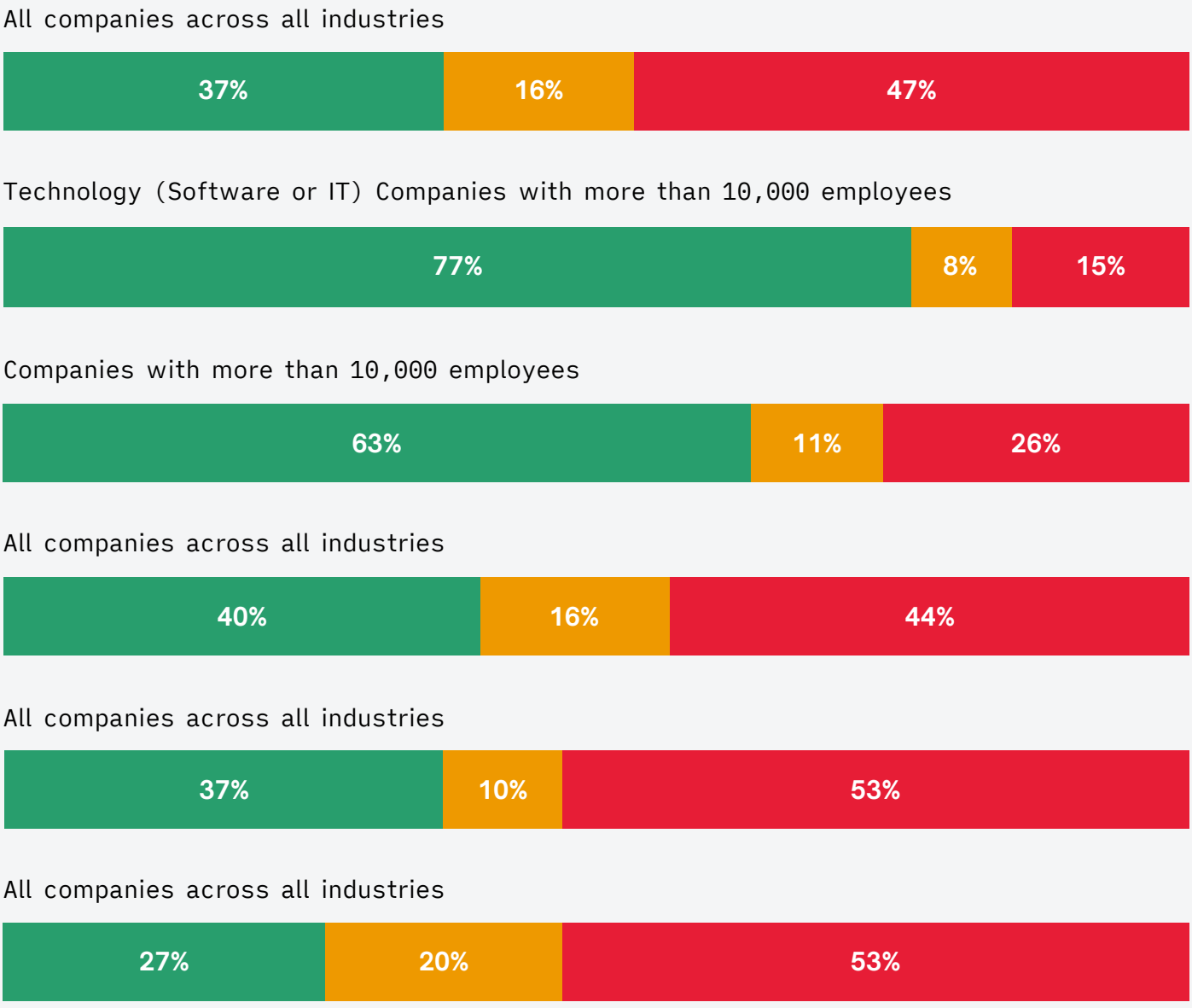
Overview

Most companies today leverage open source software to accelerate product development, reduce total cost of ownership, increase software stability, and enhance software security posture. In fact, according to the Linux Foundation's most recent report, 72% of enterprise companies cite using open source frequently[1]. While open source has many merits (such as the ones mentioned above), companies should not view open source as completely free. It's more like "free puppy" free — great joy that comes with responsibility. Using open source software comes with a set of obligations and responsibilities as well as risks. As open source software continues to be adopted at an increasing rate compliance with open source licenses becomes a more pressing initiative.

1 "Corporate Open Source Programs Are on the Rise as Shared Software Development Becomes Mainstream for Businesses." The Linux Foundation, 11 Sept. 2018, www.linuxfoundation.org/uncategorized/2018/08/corporate- open-source-programs-are-on-the-rise-as-shared-software-development-becomes-mainstream-for-businesses/.

Industry Adoption of Open Source

The enterprise has embraced open source, regardless of industry.



Source: Todo Group. “Open Source Programs Survey.” GitHub Todogroup Survey: Open Source Programs Survey, Todo Group, github.com/todogroup/survey.

Open Source License Risk

Litigation Risk

An open source license has been recognized as a legally bind- ing contract[2] in federal courts. Violating the terms of a license can expose your company, giving the owner of the open source project grounds to sue. Similar to those trolling companies to ensure

GDPR compliance (The EU's data privacy legislation) or patent trolls, there are people who actively look for GPL licenses in order to profit[3].

Lowered Valuation

Open source audits are a standard part of due diligence for M&A as well as preparing to go public (IPO). They are also increasingly included as requirements in fundraising rounds, establishing business lines of credit, and more. Having copy- left components in your distributed software can not only lower the value of your company, but can completely derail a deal due to the fact you may have to share your IP as required by the license. When identified too late in the transaction procedures, changing out an open source component may no longer be a viable option or may derail important engineering efforts.

---

2 Artifex Software, Inc. v. Hancom, Inc., Case No.16-Cv-06982-JSC (N.D. Cal. Sep. 12, 2017.

3 Meeker, Heather. "Patrick McHardy and Copyright Profiteering." Opensource.com, 24 Aug. 2017, opensource.com/article/17/8/patrick-mchardy-and-copyright-profiteering.

Loss of Market Opportunity

One of the more common (and underrated) risks is the loss of revenue or market opportunity. The most obvious events that trigger open source due diligence are during fundraising, acquisition, and IPO events. However, there is an increasing number of companies that require a due diligence report, and sometimes even the continuous availability of a report, before closing a transaction. This is especially prevalent amongst enterprise brands and can also be a frequent requirement for any software deployed on-prem or any products that are part of a manufacturing process (i.e. radio systems embedded within automotive manufacturing). Similarly, some online marketplaces like Google Cloud Platform require due diligence reports before allowing you to deploy your product in their marketplace.

Damaged Reputation

The open source community is all about transparency and collaboration. A big part of this for companies is ensuring you are consuming open source responsibly and fulfilling the obligations as dictated by the license. Another huge facet of the open source community is excellent engineering talent. Violating open source license requirements may not lead to financial implications, but it can hurt the company's brand, the engineering brand, and engineering recruiting / retention efforts.

Top Apporaches to Mitigating Risk

Manual Audits

Although it is both the most time consuming and the least accurate, many companies rely on spreadsheets and forms in their approach to compliance. Generally, this approach is triggered by a compelling event in the form of a sales opportunity or a potential partnership, acquisition, raise, etc. As part of due diligence, the key stakeholders ask for your Bill of Materials or Attribution Report. Suddenly your legal, product, compliance, and engineering teams (or some mix of the above) are thrown into an all- hands-on-deck emergency.

The manual approach general involves several approval processes. The first is typically an open source request form. Once a company has scrambled to produce the proper attribution/bill of material reports they put a policy in place. As the engineers work on developing software for a company, they need to run any open source component by legal to review before integrating it into your company's proprietary software. This then generates a spreadsheet of all requested open source components. Note, this is not the most reliable source of information because it requires both self-reporting (seen as a blocker by most engineers) and it does not include deep dependencies which your company is still liable for.

In many cases, the engineering team is not logging all of the open source components they use in a project.

Manual Audtis Pros:

- Awareness of open source policy

Manual Audtis Cons:

- Missing product deadlines
- Prolonging a sales cycle for a large client, or worse losing them
- Delayed launch into an ecosystem
- Lost engineering productivity
- Inaccurate reports
- Difficult to scale

When the code is deployed, the engineering team scrambles to track down all of the dependencies. This might even require a code freeze in order to have a static body of code to analyze. In a fire drill your most knowledgeable, connected engineers will be leading the charge. They will have the best context on what open source components have been brought in, or at least how to navigate the codebase.

Next, the dependencies and their licenses are compiled in a list that legal must pour through to ensure that there is no legal liability to using each of these components. Any issues require tracking down the engineers to get a better understanding of why the software was used so context can be applied when resolving the issue.

Finally, key reports must be generated based on the given criteria. Manual processes are in direct conflict with modern development practices which advocate for an agile method and continuous integration and continuous deployment/continuous delivery (CI/CD). Essentially, this process means that engineering teams are continuously adding new functionality and making changes to the production codebase.

Semi-Automated Compliance

As a company matures and scales it may find a manual process is not sufficient due to the number of engineers, the number of open source components used, and the frequency of required reports.
These companies start to have more regular audit processes in place which allow them to bake the cost of developing a due diligence report into your

Semi-Automated Compliance Pros:

- Planned audits
- Lower exposure to risk
- Possible to outsource portion

Semi-Automated Compliance Cons:

- Decreased developer productivity
- High expenses for third-party consultants
- High labor cost for legal teams
- Requires engineering implementation time

engineering cycle.

Traditionally semi-regular audits are performed with legacy code scanning tools like Black Duck or Flexera Code Insight (formerly known as Palamida). There are also a variety of open source tools that help generate a list of dependencies such as FOSSology. These tools require involvement from Engineering or DevOps to integrate (engineering time required varies based on your CI/CD tools and your code repositories). They then generate a list of dependencies as well as the declared licenses associated with each open source component.

The resulting information about the open source components and their licenses then needs to be audited by the legal team to identify any potential conflicts with internal policy, as well as determine steps for resolution. Generally, this requires back and forth with the engineering team to understand how and where the open source component was used and whether it is incorporated into distributed software. Depending on the issues found, resolution can involve anything from upgrading a component to a licensed version, to finding a new component to rebuilding all parts of the software that use the flagged open source component. Once all issues are resolved and rescanned, an attribution report needs to be generated and published.

Continuous Compliance

Continuous compliance is a newer concept. By definition, it means your company maintains compliance with every code commit. Generally, in order to achieve continuous compliance an

investment in third-party software is required. This software should integrate with your developer's build system (Travis, Jenkins, Circle Ci) and/or repository (Bitbucket, Github, etc) so that as new code is committed, new open source dependencies can be evaluated. This allows you to streamline issue management, reducing the time legal teams (and developers) are required to invest.

When evaluating a continuous compliance solution you should ensure that it has:

- **Developer friendly CI/CD integrations**
  Any 3rd party software needs to be easy to integrate in order to maxi- mize both coverage and adoption. It is important that you ensure the software works with the languages, package managers, repositories, and CI/CD tools your teams use. Generally, at large companies, different teams use different toolsets.
- **Intelligent Issue Management**
  Evaluate the issues management workflows within the product. Ensure there are workflows for auto-approvals, manual interventions, and logging issues with you're engineering team's task management tool (JIRA). The issue management UI should also include actionable intelligence to limit the back and forth between legal, DevOps, and engineering, enabling you to promptly resolve any issues.
- **Policy Management**
  At scale, your company needs to ensure the

Continuous Compliance Pros:

- No interruptions
- Always ready to provide due diligence
- Reduced legal costs

Continuous Compliance Cons:

- Generally requires 3rd Party Software
- Requires engineering implementation time

software you choose can apply different policies to different products. Because different types of applications/software etc. require different disclosures/attributions (etc.) based on how the software is used and distributed. Ensuring you can apply different policies to different products will reduce the number of policy violations/ components flagged during the dependency scan.

- **An Attribution Reporting Suite**
  Reporting should be built into your solution. Ideally, the solution you evaluate allows you to customize the format and the information included in your attribution reports and Bill of Materials.
- **Accurate Dependency and License Identification**
  Last (but not least), you want to ensure that the dependency scanning produces accurate results. The solution you evaluate should accurately identify the dependencies as well as the licenses. Ensure the solution is not relying solely on declared licenses, which are often incorrect. False positives create a lot of noise and additional work for both engineering and legal teams.

Why Continuous Compliance puts you at an Advantage

Modern software development has moved from the waterfall (one step at a time) to agile methodology. Complementing the rise in the agile method is the

trend of CI/CD (continuous integration and continuous delivery and/ or continuous deployment). Together, this means that software engineers are moving faster and making traditional compliance methods a barrier to success. Long story short, good software development means continuously committing code to your production codebase.

Continuous delivery means in order to out-innovate (or even keep up with) the market, engineers need to be continuously adding new open source dependencies to their project without major roadblocks. Changes in software delivery practices mean best practices for mon- itoring open source compliance should adapt and mirror the software development practices. By harnessing a continuous compliance process, companies can provide due diligence reports and decreased risk without impinging on developer or legal efficiency For more information about how FOSSA can help you develop a continuous compliance program, contact us or schedule a demo.