KB

OPEN SOURCE SOFTWARE, WITH YOUR EYES WIDE OPEN: UNDERSTANDING YOUR LICENSING-RELATED RISKS

February 2021

Derek E. Brink, CISSP
Vice President and Research Fellow, Information Security and IT GRC

ABERDEEN

To fully realize the *rewards* of using **open source software**, your organization must also understand and manage its associated *risks*. Aberdeen's analysis of the findings from hundreds of **software composition analysis** audits uncovers key insights you need to help *make better-informed business decisions* about how licensing-related risks should be prioritized and managed to an acceptable level.

Open Source Software: Rewards and Risks

In most organizations, application developers are actively taking advantage of **open source software** in their current development projects. Based on Aberdeen's analysis of the findings from more than 200 **software composition analysis** audits performed by a leading solution provider over the past two years, **more than half (median: 54.7%)** of the typical enterprise codebase is comprised of open source software. The trend is sharply upward.

Few would dispute that enterprise use of open source software has gone fully mainstream. The potential *rewards* of incorporating open source into your enterprise application development projects are many, including:

- **Speed** — Developers can get immediate access to the functionality their own projects need, taking full advantage of the capabilities already designed, implemented, and contributed by the open source community.
- **Agility** — Developers can customize and use open source software as needed for their own specific requirements, making enterprise application development projects more independent from the product roadmaps, release schedules, and pricing models of traditional, commercially licensed software.
- **Community** — Developers can gain access to the "force multiplier" of expertise and use cases from across the entire open source community, not only in the addition of new functionality but also in the identification and remediation of defects and vulnerabilities.

At the same time, incorporating open source software into your enterprise application development projects can also introduce significant *risks*, both **licensing-related** and **security-related**:

Open source software refers to publicly accessible source code which can be copied, inspected, modified, enhanced, used, and distributed as part of your own enterprise application development projects — subject to the terms of its software license agreement.

Software composition analysis refers to the process of automating an audit of your codebase to provide critical visibility into the scale and scope of open source usage, the number and types of software licenses, an accurate inventory of open source components, and other information to help understand and manage licensing-related risks.

- **Licensing-related risks** range from relatively simple *compliance requirements* (e.g., the obligation to include a copyright statement from the open source software with your own code), to more complex and potentially costly *intellectual property issues* (e.g., the possibility of violating third-party intellectual property rights, or creating an unintended obligation to release your own proprietary code as open source software).

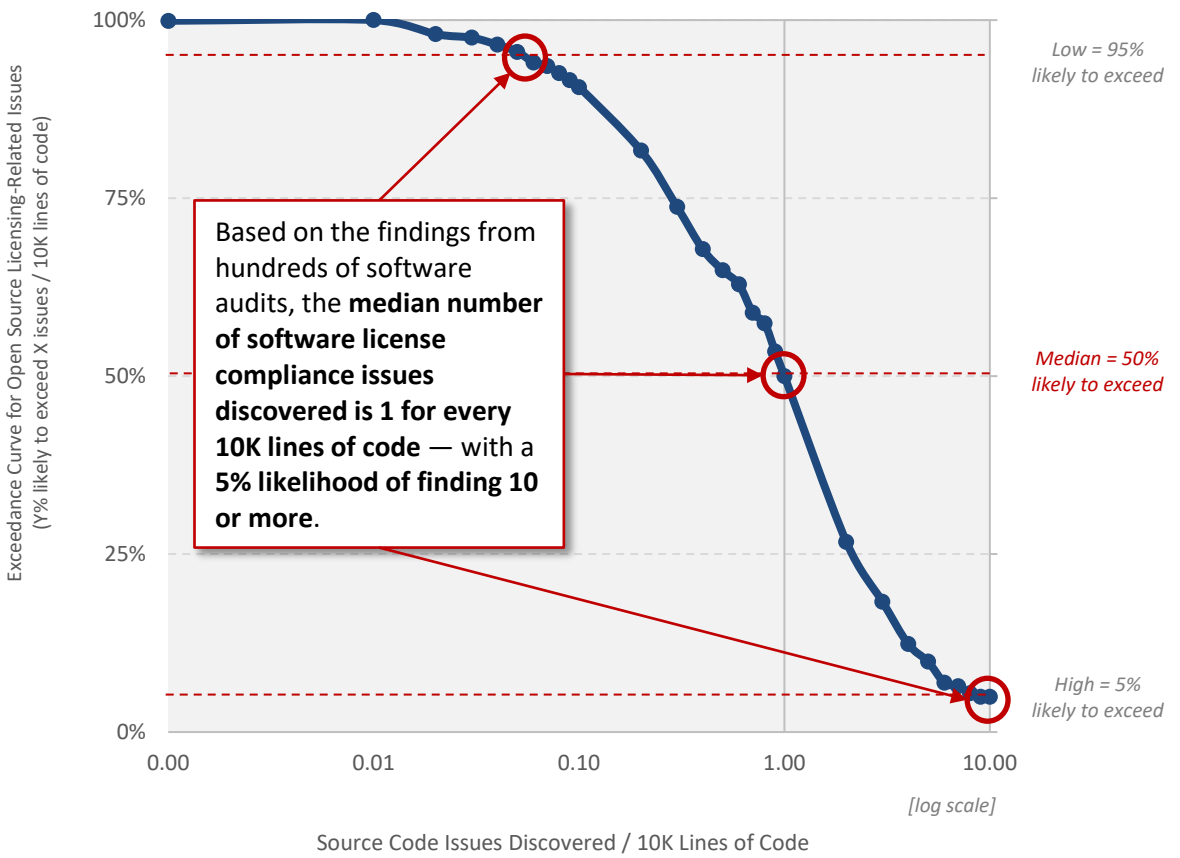
  Aberdeen's analysis of the empirical audit data shows that the number of software licensing issues discovered by a software composition analysis is significant: from **a median of 1 for every 10K lines of code**, to a **5% likelihood of finding more than 10**. See Figure 1.

- **Security-related risks** stem from *vulnerabilities* in open source software which have the potential to be exploited by attackers, resulting in material *data breaches* or disruptions to *availability*.

In *about 1 out of 4 (median: 24.8%)* organizations, software composition analysis audits discovered **security-related vulnerabilities** in addition to software licensing issues.

Nearly half (46.0%) of those discovered have a severity rating of **high** to **critical**.

Figure 1: Open Source Licensing-Related Issues "Exceedance Curve"



Source: Empirical data adapted from Revenera audit services, 2019-2020 (N = 202 client software composition analyses); Aberdeen, February 2021

Understanding Your Licensing-Related Risks

Understanding your licensing-related risks from the use of open source software in your enterprise applications begins by framing the issue in quantitative terms: How many open source software licensing-related issues are currently hidden in your organization's codebase, waiting to be discovered and remediated before they result in negative business impact?

As noted above, Aberdeen's analysis of the findings from more than 200 software composition analysis audits — across a diverse range of organizations and their respective codebases — reveals several key insights:

- **The total number of software licensing issues discovered by an audit is surprisingly high:** For every ten thousand lines of code, performing an audit is likely to discover *between 0.5 to 10 total issues*, with a *median of 1***.** Use your own codebase size, and do the math.
- **The percentage of total software licensing issues that organizations already know about is shockingly low:** Prior to an audit, 8 out of 9 (88.1%) organizations were not aware of *any* software licensing issues in their codebase. Of the 1 in 9 (11.9%) who had identified at least *some* of the existing issues, what they did know represented a *median of just 9.5%* of the total issues discovered.

  Said another way: Most organizations have *virtually no visibility* into the licensing-related risks from their use of open source software.
- **Priority 1 (P1) issues represent a significant proportion of the total:** Among all software licensing issues discovered by an audit, those deemed to be P1 issues represent *between 0% to 90%*, with a *median of 12.5%*. The P1 category refers to a critical licensing-related risk, which should be remediated faster and with greater urgency than lower-priority issues (P2, P3, P4). See Table 1.

Readers should note that these empirically-derived insights are presented as **a range of possible values** (i.e., *lower bound*, *upper bound*) with an associated **shape** (i.e., as driven by the *median* value). This approach provides valuable insights about the risks, properly expressed in terms of both "how likely" and "how much."

Understanding your organization's risks from open source software is key to **making better-informed business decisions** for what to do about them. Investing in a software composition analysis of your own "crown jewels" codebases is a logical place to start.

The empirical data from software composition analysis audits makes it clear that most organizations have *virtually no visibility* into the licensing-related risks from their use of open source software.

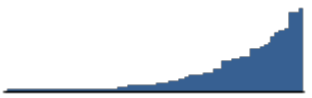
Understanding your organization's risks from open source software is key to *making better-informed business decisions* for what to do about them. Investing in a software composition analysis of your own "crown jewels" codebases is a logical place to start.

Which Types of Open Source Software Licenses in Your Enterprise Codebase Represent the Greatest Risk?

How likely is it that open source software licensing issues are present in your enterprise application codebase, and to what degree? How much business impact could there be, if these issues are not identified and remediated?

Unlike **proprietary** software licenses — which generally provide the *right to use*, but not the *right to modify* or the *right to distribute for commercial purposes* — **open source** software licenses universally provide the rights to *use*, *modify*, and *distribute for commercial purposes* as part of your own enterprise application development projects (see Table 1).

Table 1: What Open Source Licenses (and Legal Obligations) Are Currently Hidden in Your Organization's "Crown Jewels" Codebases?

Types of Software Licenses		Most Prevalent Examples Discovered	Right to Use	Right to Modify	Right to Distribute for Commercial Purposes	Source Code Must Be Distributed	Persistent	Viral	Types of Open Source Software Licenses Discovered (% of Items Identified in a Software Composition Analysis)
Proprietary			Yes	No	No	No	n/a	n/a	n/a
Open Source	Permissive	Apache, BSD, MIT, Python, Ruby	Yes	Yes	Yes	No	No	No	 0% to 100% (median: 72.6%)
	(Weak) Copyleft	EPL, LGPL, MPL	Yes	Yes	Yes	Yes	Yes	No	 0% to 93.8% (median: 7.6%)
	Strong Copyleft	CC BY-SA, GPL	Yes	Yes	Yes	Yes	Yes	Yes	 0% to 35.6% (median: 12.4%)

Source: Empirical data adapted from Revenera audit services, 2020 (N = 84 client software composition analyses); Aberdeen, February 2021

Unfortunately, the obligations of open source software licenses can quickly get more complex than that. In addition to the rights to use, modify, and distribute for commercial purposes, it's essential to understand at least three other license attributes offered by different types of open source licenses:

- **Whether the source code for any derivative work *must* also be distributed.** This question can get complicated depending on how the derivative work software is stored and executed, such as over a network (e.g., software-as-a-service), on a server or endpoint (e.g., traditional installed applications), or accessed over a network and run on an endpoint (e.g., web browser extensions). But in general, some open source software licenses create a requirement for your organization to distribute the source code for any derivative work.
- **Whether the terms of the open source software license are *persistent*,** meaning that any derivative work must be licensed and distributed *under the same license* as the original open source code.
- **Whether the terms of the open source software license are *viral*,** a term which means that when open source code under this type of license is combined with any other code, the *entire combined codebase* (not just the derivative work) also becomes subject to this same type of license.

High-level attributes of the three main categories of open source licenses — *permissive*, (*weak) copyleft*, and *strong copyleft* — along with empirical findings and insights from dozens of software composition analysis audits, are reflected in Table 1.

To tie all this together, Table 1 summarizes the high-level attributes of the three main categories of open source licenses — **permissive**, (**weak) copyleft**, and **strong copyleft** — along with the empirical findings and insights from the software composition analysis audits:

- **Permissive** licenses provide the rights to use, modify, and distribute for commercial purposes — but do not require the source code for derivative works to be distributed, are not persistent, and are not viral.

  By far, permissive licenses were the most prevalent types discovered by the software composition analysis audits, representing a median of 72.6% of all items identified.

  The most frequently occurring examples of permissive licenses discovered by the software composition audits include *Apache*, *BSD*, *MIT*, *Python*, and *Ruby*.

- **(Weak) copyleft** licenses provide the rights to use, modify, and distribute for commercial purposes. In addition, they require the source code for derivative works to be distributed, and that any derivative works must also be licensed and distributed under copyleft license — but they are not viral, i.e., they do not impose copyleft on your entire codebase. (For this reason they are sometimes referred to as “weak” copyleft, as opposed to *strong copyleft* — see below.)

  Copyleft licenses were the least prevalent types discovered by the

software composition analysis audits, representing a median of just 7.6% of all items identified.

The most common examples of copyleft licenses discovered by the software composition audits include *EPL*, *LGPL*, and *MPL*.

- **Strong copyleft** licenses provide the rights to use, modify, and distribute for commercial purposes. In addition, they require the source code for derivative works to be distributed, and that any derivative works must also be licensed and distributed under strong copyleft license — along with *your entire codebase*.

  For your organization's application development projects, the *viral* attribute is what makes the incorporation of open source code which is subject to a strong copyleft license such a significant risk: It creates the obligation to release your entire codebase, including your own proprietary code, as open source software under strong copyleft.

  Strong copyleft licenses represented **up to 35.6%** of the items discovered in the software composition analysis audits, with a **median of 12.4%**. That is: For every thousand software licensing issues discovered by an audit, as many as *356 involve strong copyleft licenses*, with a *median of 124***.** Use your own codebase size, and extend your previous math.

  The most prevalent examples of strong copyleft licenses discovered by the software composition audits include *CC BY-SA* and *GPL*.

The incorporation of open source code which is subject to a *strong copyleft* license creates a significant risk for your enterprise application development projects — it creates the obligation to release your entire codebase, including your own proprietary code, as open source software under strong copyleft.

Towards Quantifying the Potential Business Impact of Your Licensing-Related Risks

Thus far, the software composition analysis audit findings provide empirical insights that help to quantify the *likelihood* aspect of your licensing-related risk from the use of open source software:

- **Between 0.5 and 10 (median: 1) software licensing issues**, for every ten thousand lines of code
- Of the software licensing issues discovered, **between 0% to 35.6% (median: 12.4%) are strong copyleft**, which present the greatest risk

Quantifying the *business impact* aspect of your licensing-related risk is a little more challenging. Should the strong copyleft obligation to release your entire

codebase (including your own proprietary code) as open source software end up being *enforced*, how much business impact could this be?

Keeping in mind that the goal is not to estimate with engineering-like precision — the goal is to *help make better-informed business decisions* about licensing-related risks, in the face of inherent uncertainties — in Aberdeen's view it's possible to make some highly useful estimates:

- **At the lower end of the business impact curve:** Legal case law has established that a legal judgment "can use the value of a commercial license as a basis" for valuing the business impact of a breach of a strong copyleft license. For example, a software provider might use a commercial license for an "enterprise" edition of their software, and use a strong copyleft license such as GPL to offer an open source, "freemium" edition.

  In this case, **commercial pricing for the enterprise edition** can be used to estimate the business impact of a license-related violation of the freemium version — analogous to a judgment to pay all of your back taxes, in addition to paying your annual taxes going forward.

- **At the more extreme, "long tail" end of the business impact curve:** Aberdeen estimates the business impact for the violation of a strong copyleft license by thinking of your codebase in terms of its *intellectual property* value, and thinking of license enforcement as a breach of your IP. This is analogous to falling off the so-called patent cliff, i.e., the sudden loss of revenue experienced on the expiration of patent protections and the entry of lower-priced competitors.

  Based on this simple concept, the cumulative loss of revenue from a strong copyleft license enforcement ranges **up to 440% of the annual revenue** generated by your codebase, in the last full year prior to the enforcement.

  Independently, Aberdeen's analysis of legal judgments in court cases involving a violation of IP — expressed as a percentage of the annual revenue generated by the IP, in the last year before the judgment — yielded a range virtually identical to that of the patent cliff approach.

As previously noted, risk is always expressed properly by combining estimates for both the "how likely" and the "how much impact" aspects — and business decisions about risk are generally made at the "long tail" end of the curve, where the total business impact has the potential to be catastrophic.

Software composition analysis is designed to help your organization more fully realize the *rewards* of using open source software, while keeping your eyes wide open to the associated *risks*.

Yet even under the same circumstances, for different organizations the respective senior leadership teams — e.g., VP of Software Engineering, Chief Information Security Officer, Chief Financial Officer — may have decidedly different *appetites for risk*. The key point is that software composition analysis is designed to help your organization more fully realize the rewards of using open source software, while keeping your eyes wide open to the associated risks.

Summary and Key Takeaways

- To fully realize the *rewards* of using open source software, your organization must also understand and manage its associated *risks*.
- Prior to a **software composition analysis**, most organizations have *virtually no visibility* into the licensing-related risks from their use of open source software.
- By investing in a software composition analysis of their most valuable codebases, organizations get critical visibility into the scale, scope, and type of open source software licenses currently in use.
- Understanding your organization's risks from open source software is essential to *making better-informed business decisions* for how to prioritize and manage them to an acceptable level.

About Aberdeen

Since 1988, Aberdeen has published research that helps businesses worldwide to improve their performance. Our analysts derive fact-based, vendor-neutral insights from a proprietary analytical framework which identifies Best-in-Class organizations from primary research conducted with industry practitioners. Aberdeen also provides intent-based marketing and sales solutions that deliver performance improvements in advertising click-through rates and sales pipelines, resulting in a measurable return on investment. Aberdeen is headquartered in Waltham, Massachusetts, USA.

18289