TIDELIFT

Cover letter

Tidelift appreciates having the opportunity to respond to this RFI from ONCD on a space that our company has been working on for years and our co-founders have been learning in for decades—open source software security. By way of background, Tidelift helps enterprises effectively manage the health and security of the open source software that, as the RFI notes, powers essentially all modern software development. To do this, Tidelift partners with leading independent open source maintainers—and pays them—to ensure their projects follow secure development practices, like those outlined in the U.S. government's NIST Secure Software Development Framework, and document this important work. For enterprises, this human-validated data about secure software development practices used in their open source software supply chain helps reduce risk and increase development velocity. For maintainers, this approach helps them keep their projects healthy and resilient, while providing full recognition and compensation for the value they create.

Our founding team comes from decades of leadership experience in open source communities, at places like Red Hat, Google, Wikimedia, Mozilla, the Open Source Initiative, and Creative Commons. Because of this unique background of corporate and community work, Tidelift is heavily focused on how to make open source work better by understanding and supporting the independent maintainers who make up the backbone of this accidental, but critical, supply chain.

The RFI asks whether it is "appropriate to make open-source software a national public priority." We believe it is not only appropriate, but necessary, and have proved through field testing that a proactive approach to improving open source software security in partnership with open source maintainers is not only possible—but delivering real security outcomes today.

We look forward to sharing our thoughts with you on the following pages.

Response to ONCD's request for information on open source software security

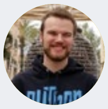
TIDELIFT

Table of contents

Our focus: incentives for securing the open source ecosystem

This response to the ONCD Request for Information will focus specifically on the RFI topic area of "incentives for securing the open source ecosystem." We believe that this is, in many ways, the most important area called out in the RFI. Open source is not magic! While other areas called out in the RFI are important, if incentives and motivations of open source maintainers are not well-understood by policy-makers, those other improvements will not happen, or will happen only slowly. That puts the question of incentives and motivation on the critical path for almost all other improvements to the security of the open source ecosystem.

> **"Being a maintainer of an open source project requires running fast just to stay still. Every project requires security responses with fixes, updates to dependencies, and support for new language versions, features, and platforms. When the amount of work demanded from maintainers becomes too much we lose maintainer time to burnout, disinterest, and frustration."**



> – Seth Michael Larson, maintainer of urllib3, popular Python package with billions of downloads

Core challenge: mismatch of volunteer suppliers and enterprise users

The core challenge of securing the open source ecosystem is that the *composition*, and therefore the *motivation*, of this supply chain is unlike any other supply chain that is so critical to the global economy. This creates a mismatch between the expectations of enterprises at one end of the supply chain, and the motivations of the *largely independent volunteer* developers at the other end of the supply chain.

Let's start with why the *composition* of the supply chain is unlike other supply chains. In short, no other global, economically central supply chain is made up in large part of what we call "accidental" suppliers. These open source maintainers are not, primarily, doing their work because they are employed by a supplier in the supply chain. Rather, they are volunteers, or employed by a company whose profit motives are not aligned with the rest of the supply chain.

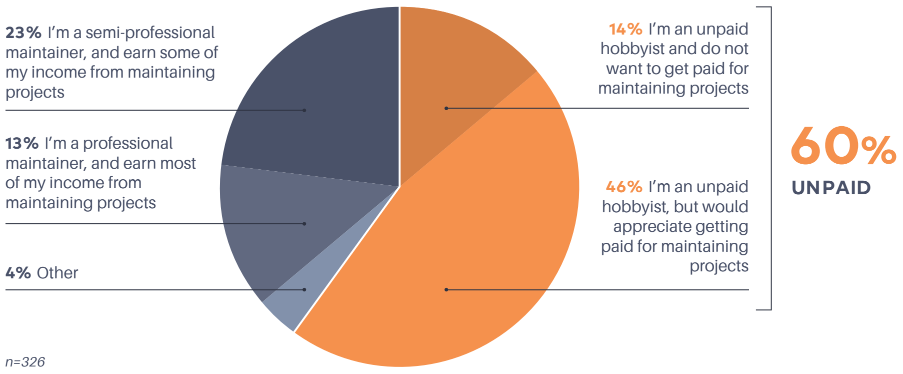
In either case, they can deprioritize the work at any time—not for malign reasons, usually, but for mundane ones like a new pet, new relationship, or simply the need to earn a living. This prioritization—where traditional supplier tasks are often last on the supplier's list of personal priorities, rather than first—is not something that enterprises or governments expect of their suppliers.

Volunteers dominate the open supply chain

Every other year, Tidelift conducts a survey to more deeply understand what motivates open source maintainers, what makes them thrive, and what stands in their way. In Tidelift's latest survey, which came out in spring 2023, 60% of open source maintainers described themselves as unpaid hobbyists, while only 13% said

60% of maintainers describe themselves as unpaid hobbyists

Which of the following phrases best describes how you approach your role as an open source maintainer?

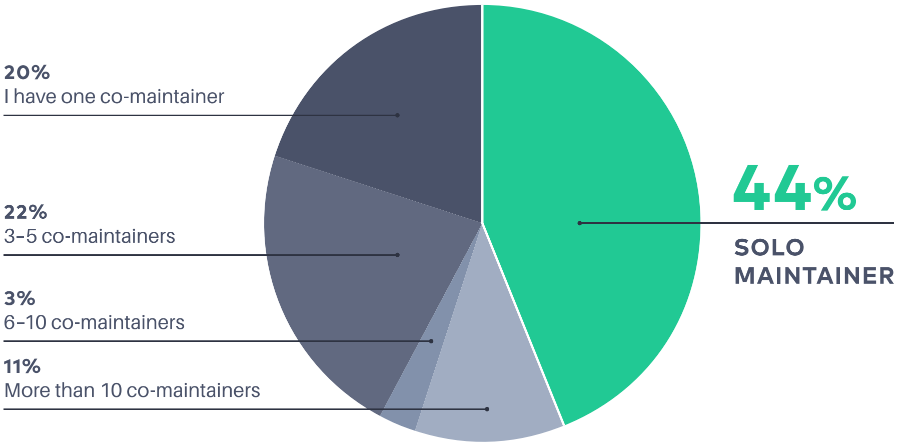


they earn most or all of their income from maintaining open source projects.

Similarly, somewhere around half of open source maintainers do that critical maintenance work alone. In our survey, about 44% of maintainers said they were solo maintainers, and other studies report similar numbers (eg, 57% in this study of *the most popular projects*, and perhaps 93% in this study of Python).

Almost half of all open source maintainers are solo maintainers

Do you have any co-maintainers? If so, how many?



n=326

Note that, contrary to some understandings, this is not a problem limited only to certain programming languages or frameworks. Essentially all open source depends, to some extent or another, on stacks of smaller pieces of work done by others.

The median Tidelift customer, regardless of platform, has more than 3,000 different independent open source projects in their products. Only a handful of them are high-profile enough to be directly backed by for-profit businesses or large industry associations.

This is not to diminish the work of those larger projects, which is important. However, strategies for open source ecosystem security *must* consider both types of projects—the large ones, which get a lot of attention from vendors, and small ones, which are 1-2 orders of magnitude more common, but get much less direct backing.

Volunteers are different from traditional suppliers

Traditional supply chains have traditional motivations: a supplier provides a widget into the supply chain, and the next step in the chain *pays for* that widget. This incentivizes the widget supplier to (among other things) continue existing, so that it can charge for support, and so that it can sell more widgets in the future.

Open source software's volunteer-centered supply chain has different motivations. These motivations can be extremely powerful—many paeans have (correctly!) been written to the diligence, skill, and passion of open source maintainers, and that has resulted in prodigious, innovative, impactful output.

But the motivations necessary to *continue to maintain* that software for decades, in the face of demanding users and a hostile security environment, are different. Open source has had ongoing challenges as a result. This is reflected in a variety of different ways.

Volunteer “lifespan” is different: turnover and burnout

Burnout and turnover are real challenges in open source. In Tidelift’s recent maintainer survey, a majority of maintainers report either having quit or having considered quitting (58%). And these respondents are those who have quit and yet still stayed engaged enough to respond to surveys—suggesting that the actual numbers are probably worse.

The result of this is that the average lifespan of an open source project is not long. Some recent research suggests that there is no language ecosystem where projects have a > 50% chance of surviving for four years.

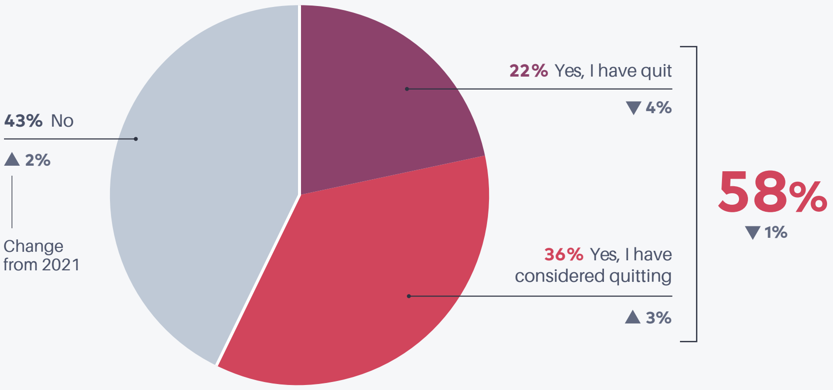
Projects that are unmaintained are more likely to be insecure (along a variety of dimensions, including hijacking), so this burnout problem leads directly to security problems.

Bus factor vs. boss factor

Open source communities have long referred to the problem of disappearing maintainers as “bus factor.” This makes it sound like random bad luck (like, sadly, car crashes) are the primary reason why people leave open source projects. But we now know that the primary reason for departures is more like “boss factor”—when maintainers get new jobs, or new roles at old jobs, they often can’t devote time to their open source projects anymore.

A majority of maintainers have at least considered quitting one of their projects

Have you quit or considered quitting maintaining a project?



2023: n=349; 2021: n=265

This conclusion is both intuitive (job changes happen all the time) and supported by research from Carnegie Mellon. In a paper entitled, “Why Do People Give Up FLOSSing?”, researchers conducted a survey of maintainers who dropped from 300+ contributions over 18 months to < 5 contributions in the following six-month period. The survey shows that “the most common reasons for complete disengagement relate to transitions in employment, such as graduating from academia, changing employers, and changing roles.” The federal government’s approach to security must grapple with these reasons for changing levels of engagement.

Volunteer capacity is different: time matters

Our survey data shows that maintainers who contribute to open source as a "hobby" spend much less time on their software than professional maintainers do—which means that security-related tasks may get deprioritized in favor of more fun or interesting project needs.

For many projects, vulnerabilities are rare, and dealing with them can be time-consuming. That time factor hits in two ways: upfront preparation costs, and at-the-moment prep costs.

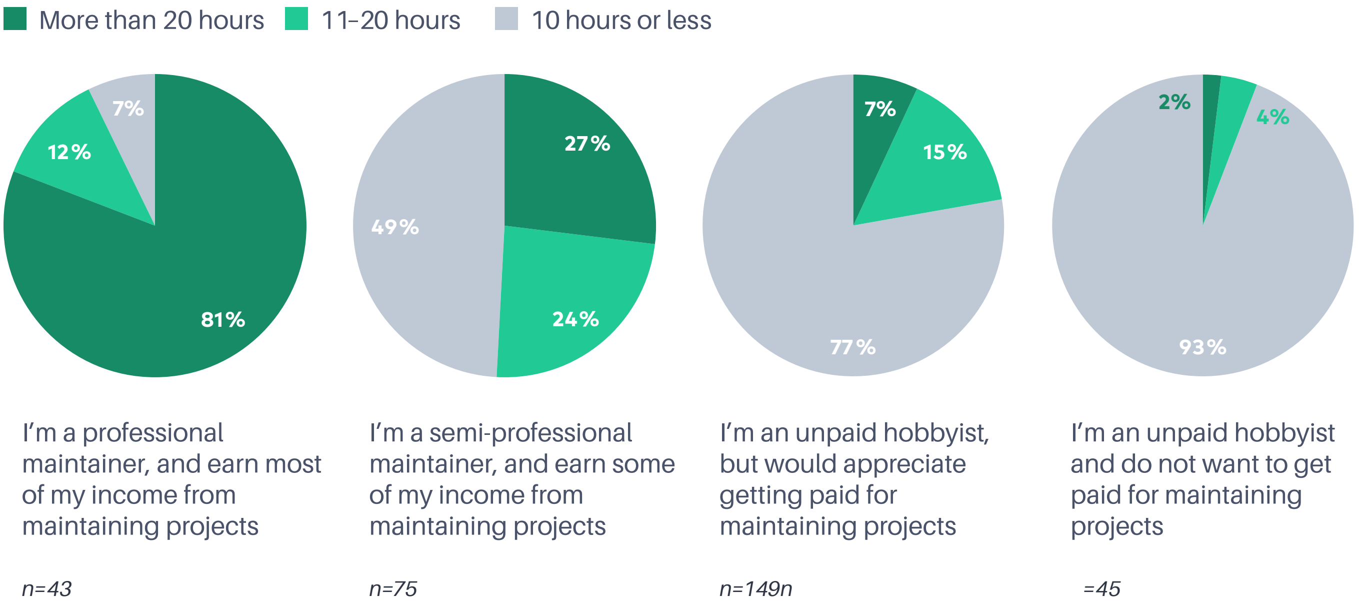
> **"As a maintainer, it's not usually about what I want, it's about what I can do."**
>
> – Thomas Depierre, open source maintainer, in a podcast about the open source "supply chain"

Correctly handling vulnerabilities requires preparation (like creation of a security response policy) that is a large cost, for something that may happen rarely or never. As a result, volunteers may deprioritize that planning. Tidelift incentivizes this by requiring our (paid!) maintainers to set up a security response policy as part of our onboarding, which has helped some projects deal with vulnerability reports that would otherwise have languished.

Similarly, actually handling a vulnerability report when it comes in may be inconvenient. Again, for a Tidelift partnered maintainer, this is a paid task and can be prioritized appropriately, but for a hobbyist, who has very little time to devote to their projects, time-sensitive responses to a vulnerability report may be difficult or simply not possible, depending on demands of the "day job," family, and other draws on their time.

The more maintainers get paid, the more they work on open source

How much time do you spend per week maintaining open source projects?



Urllib3: a case study of paid maintainer incentives

Seth Michael Larson, lead maintainer of urllib3, one of Python's most downloaded packages (250 million downloads a month), is a Tidelift partnered maintainer and has shared how Tidelift makes streamlining processes and staying on top of security easier, allowing his team to free up time to spend on proactive maintenance strategies. The urllib3 team leverages the reliable income from Tidelift and other sources to pay contributors for some of this work.

urllib3 is a critical part of the Python ecosystem, in the same way that Log4j is for Java. To help in zero day vulnerability situations such as the Log4Shell incident , urllib3 partners with Tidelift to handle—and create time for—their coordinated security disclosure process.

Volunteer interests are different: non-performance of security-critical tasks

It's important to say that many volunteer maintainers are interested in security, and we do not intend this to be a critique! However, many security practices are not well-aligned with maintainer motivations and resources.

Some examples:

New processes: For maintainers who are already feeling overworked and underappreciated, the additional time commitment or hardware cost of a new process may be discouraging, and can even be the "last straw," causing them to abandon their projects altogether. This is not hypothetical; one high-profile maintainer—when asked to turn on two-factor authentication—deleted hundreds of widely used projects.

New documentation: Developers are famously averse to "paperwork," and so security attestations and similar requirements (like SBOMs) whose job is primarily to document the outputs of other, more interesting and rewarding tasks, are likely to be completely ignored by maintainers.

To put it another way: If security problems don't align with the interests and time resources of the volunteer supply chain, adding *more* "requirements" is not likely to solve the problem—they will at best not respond, and at worst quit doing other maintenance activities!

Case study: maintainer hero saves day, stops packages from going unmaintained

Jordan Harband, a JavaScript maintainer (and Tidelift partnered maintainer), helped pick up the pieces when another maintainer of widely used JavaScript packages deleted their GitHub account.

“At the time it was a passion project for [the maintainer of the widely used JavaScript packages]. People were getting value from his work for nothing. Over time, the more you have a package that gets heavy usage and adoption, the more burden is placed on you as people complain that things are broken, as people ask you to add features, etc.”

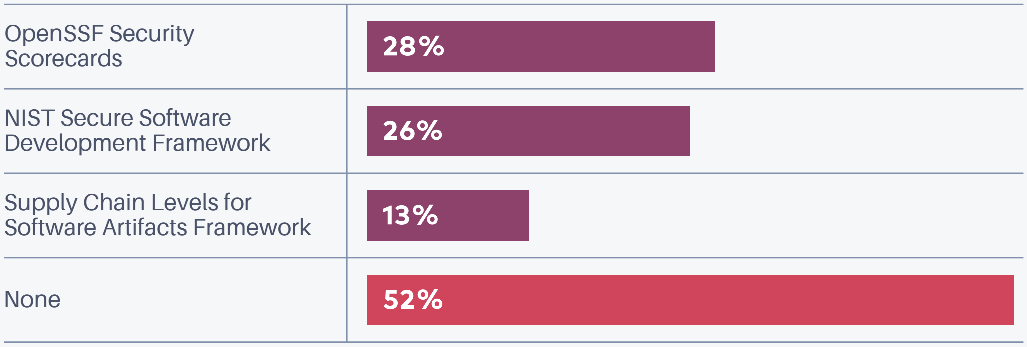
When that maintainer walked away (and in fact deleted many of their packages) Jordan was able to step in and help—in part because Tidelift compensated him for the additional burdens he took on, that benefited the entire open source ecosystem.

Do maintainers know about or care about attesting to industry standards?

In our 2023 Tidelift state of the open source maintainer report we asked respondents to tell us which of the most commonly cited standards initiatives they are aware of today, including the OpenSSF Security Scorecards project, the SLSA framework, and the NIST Secure Software Development Framework (SSDF). We found that over half the respondents were not even aware of the most prominent software security standards.

Over half of maintainers are not aware of prominent software security standards

Which of the following industry standards initiatives are you aware of? (Choose all that apply)



2% responded “Other.”
n=292

What is the impact of this volunteer supply chain on government and industry?

The bottom line for the impact of volunteerism on government and industry is that (1) deep, fundamental security work requires constant, consistent work and (2) uncompensated volunteers don't typically get work done on a constant, consistent basis.

To put it another way: open source security has been seen as "good enough" for many decades, because, *in aggregate*, open source maintainers are conscientious and effective. However, it has become clear that open source security is not just a quality of open source as an aggregate. Instead, it is also a "weakest link" problem: since security is the property of a specific project, and violations can have wide-spread impacts, any one maintainer stopping work on a project can threaten the security of large, security-conscious entities.

- **Across time:** governments and enterprise users need software security work to be done consistently for years, but we know from "boss factor" work and analysis of project abandonment that this is very hard. If an organization has 5,000 packages in its stack, and the median package is only maintained for 5 years (see) or even one year (see), that business has multiple packages in its stack going completely unmaintained every week.

- **Across metrics:** because of the vast scope of dependencies in modern software projects, governments and enterprises need metrics to be consistent—even where there are standard guidelines like the OpenSSF Scorecard, they are not useful at scale unless their application is consistent and standardized.

How do incentives impact other RFI topic areas?

The RFI asks about a number of areas that don't look like incentive problems, until you look more closely.

- **Fostering best practices:** Dissemination of best practices in open source has been a challenging problem for a long time. Some of this is the usual problem of improving standards in any professional culture, but with the additional difficulty that many of the usual tools (primarily, "sticks") aren't available because of the volunteer nature of the labor force—if they don't want to, they simply won't. Money is not the only carrot, of course, but it is a big one—as we've shown with our work on adoption of the OpenSSF Scorecard.

  Tidelift has built a unique software and people model designed to achieve the best results in securing the open source software supply chain. This model provides open source maintainers with a set of clear standards to uphold on a per-package and per-release basis. These standards are a best-in-industry set of development practices grounded in the Center for Internet Security Software Supply Chain Security Guide, OpenSSF Scorecard, and NIST Secure Software Development Framework.

  We have proved that this model is effective at improving the overall security of open source software, as evidenced by the improved OpenSSF Scorecard scores for our focused cohort of packages, which we will share more about in the following section.

- **Attestations—who?** As a general matter, attestations that projects are following secure software development practices work best—the further the attestor is from the facts of the matter, the less accurate and reliable the attestation will be. So, as with many things in the open source supply chain, the more work maintainers can be incentivized to do, the better off we'll be—incentivizing third-parties who have good intentions but lower-quality information will result in lower-quality attestations even when those links in the chain are doing their absolute best.

- **Attestations—why?** Attestations may be important for consumers, but that does not immediately provide an incentive for maintainers to provide them. At best, attestations may be (like SBOMs and other 'scorecards') very dull for maintainers, and hence unlikely to get done, and at worst may be perceived to create liability—leading maintainers to actively avoid doing them if at all possible. In both cases, entities requesting attestations from each point in the supply chain must think carefully about what incentives are available to overcome boredom—or fear of liability.

- **Memory-safe languages:** Memory-safe languages are likely to be an increasingly important part of the security toolkit in coming years, but—assuming that effort succeeds—they will still need a strategy to motivate long-term support and security maintenance, since memory safety solves only one facet of the overall security challenge.

- **AI tools:** Viewed from the perspective of open source maintainers, security tools—AI or otherwise—have often fallen prey to a few key recurring problems. These particularly include steep setup costs (i.e., the work that has to be done to get the tool running the first time) and false positives (i.e., the number of times the tool reports errors that are not errors, or are not relevant). These challenges can, to some extent, be addressed by reducing the time costs/improving the quality of the tools—eg, by building them into commonly used development platforms like GitHub. But the challenges can also be addressed, in at least some cases, by improving the incentives maintainers have to use them.

- **International collaboration:** Because open source is international, we need incentives that work across cultures and borders. Money is quite international—as we've shown at Tidelift by supporting developers from 54 different countries.

Policy choices for open source incentives

The basic story is clear—making significant, meaningful improvements to open source security requires positive incentives for maintainers. This is also not a hypothetical—Tidelift has done this, with clear, measurable outcomes. However, it has not yet been scaled up via policy approaches. In this section we'll discuss some potential approaches that could be used to leverage government capacity to bring incentives (particularly economic ones) to bear on open source's security challenges.

Two traditional policy approaches

Traditional policy approaches to software security (and infrastructure maintenance more generally) have tended towards "stick" rather than carrot. These have not yet been tried substantially in open source, but there is reason to believe that this will be counterproductive.

Liability

The RFI asks about the application of "liability" to the open source ecosystem. This is a complicated area, but worth exploring briefly to highlight the contrasts with a more constructive approach.

The European Union is considering implementing a liability regime for all software, called the Cyber Resilience Act. This act would cover all software, including open source, and open source developers have reacted extremely negatively—for many of the same reasons (time, motivation, and liability) called out in this document. The gist of the critique is that, because so many developers are volunteers or otherwise do not profit from the software, significant costs or risks might deter contribution, reduce innovation, and (unintentionally) decrease security.

It's worth also noting that, since so much open source is produced by individuals, liability may be ineffective for another reason: unlike traditional providers of infrastructure, whose revenue streams allow them to have legal and compliance teams, many providers will simply not know about their liability until after a vulnerability has occurred. This would transform liability, for those providers, from a preventative approach to a purely punitive one.

Regulation

Traditional regulatory channels may prove more fruitful, but still have significant limitations. As already mentioned, many pieces of critical open source software do not have compliance teams, and so would simply not know about applicable regulations.

Even in cases where developers become aware of regulations, enforcement will still be difficult. We've called this an "unfunded mandate"—because the new mandates do not create time, or other resources, to fulfill them. Developers always have the option to walk away, or in non-U.S. jurisdictions, to simply ignore the requirement.

This is not to say that regulation is completely impossible—there may be specific domains where the costs of non-compliance are so high, and the associated revenue streams are large enough, that carefully crafted regulations paired with support of maintainers may be valuable.

Proposed policy approach: directly incentivize maintainers

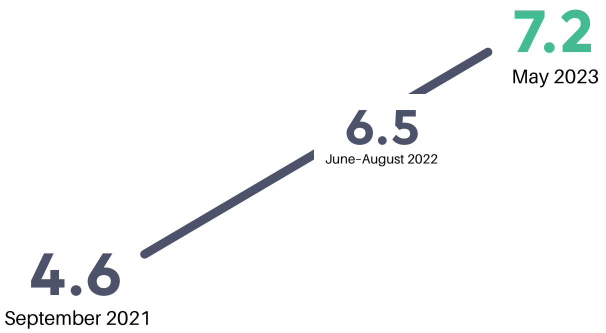
Given the government's many worthwhile proactive goals for open source, and the lack of developer time and resources available to implement them, policymakers must prioritize directly paying independent maintainers or else those goals will almost certainly not be achieved except by the relative handful of large projects that are already widely supported by industry.

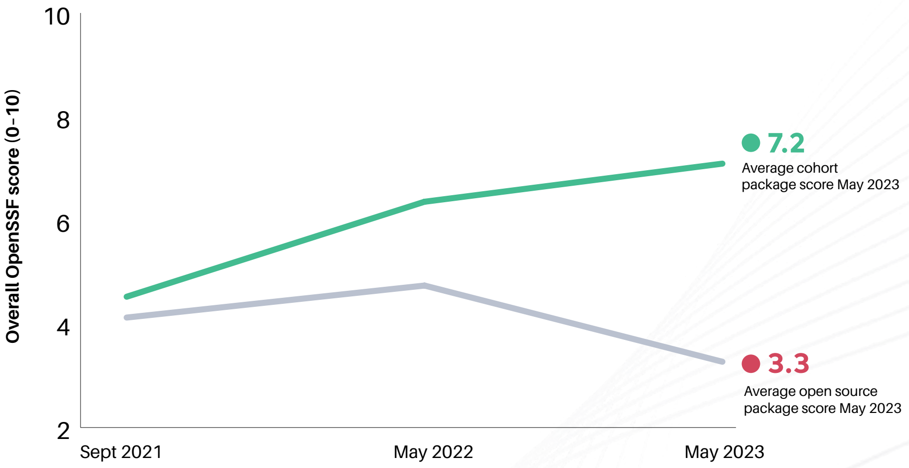
Directly paying independent maintainers to ensure and attest to the secure software development practices followed for their projects, or else those goals—it is an approach with proven impact. Beginning in June of 2022, Tidelift undertook a focused project to incentivize open source maintainers to improve adoption of the OpenSSF Scorecard recommended practices, and overall scores. At the conclusion of this project, Tidelift's paid cohort for this research was outperforming their previous scores from September 2021, as well as their peer open source packages.

Because of this work, Tidelift can now definitively measure investment, outcomes, and impact for open source software supply chain security—and it starts with paying independent maintainers.

Tidelift's multi-year partnership with maintainers has increased overall scores by 57%

Through a focused effort on scorecards starting in June of 2022 to May 2023, Tidelift increased OpenSSF scorecard scores from an average of 6.5 to 7.2 (n=26), increased maintainer engagement with scorecards, and improved how the scoring is assessed.





OpenSSF scorecards over time:
focused cohort vs all asessed opens source packages

Potential mechanisms for government policy

Government support for paying independent maintainers could be handled through a number of potential policy vehicles. Among others that have been suggested include:

- Addition of attested SBOM requirements to other touchpoints that regulate software security or privacy, like HIPAA and SOX
- Inclusion of long-term support incentive schemes in relevant metrics, like the FITARA and FISMA scorecards
- Prioritization of approaches that apply incentives to all steps in the supply chain, not just containerization or other end-product-focused approaches
- Experimentation with Hack the Pentagon-style bounties for developers who need one-time incentives to do the initial investment that brings their projects' SBOMs up to CISA-level standards
- Require testing with, and feedback from, independent maintainers on all government security standards, including upcoming standards like NIST Cybersecurity Framework 2.0, in order to ensure that they are implementable not just by well-resourced federal contractors but also by the individuals who form so much of the supply chain
- Explicit avoidance of "hackathons" and other one-time, volunteer-centric approaches that do not create sustainable incentives for recurring contributions

No one solution is likely to be perfect, or cover all of the many thousands of relevant open source projects used across the government and critical national infrastructure, so we would urge broad experimentation across a number of policy and funding vehicles.

Conclusion

At Tidelift, we've found that open source developers are long on passion, but short on time. The federal government should use its unique purchasing and funding power to attack that problem using a proven tool: paying those volunteers to ensure, and attest to, the secure software development practices followed by their projects.

Appendix

The 2023 Tidelift state of the open source maintainer report, May 2023

On the abandonment and survival of open source projects: An empirical investigation, June 2019

An Empirical Analysis of the Python Package Index (PyPI), July 2019

Bus factor, boss factor, and the economics of disappearing maintainers, August 2019

Open Source Security Podcast: Episode 365 - “I am not your supplier” with Thomas Depierre, March 2023

Why coordinated security vulnerability disclosure policies are important, January 2020

Recap: How the maintainers of urllib3 keep the project secure and healthy, December 2022

Upstream 2023 | Welcome keynote | Luis Villa plus fireside chat with Jordan Harband, June 2023

On the abandonment and survival of open source projects: An empirical investigation, September 2019

Survival of Eclipse third-party plug-ins, September 2012

OpenSSF Scorecard GitHub repository

CIS Software Supply Chain Security Guide, June 2022

Secure Software Development Framework, January 2023

Tidelift 2023 open source maintainer impact report, June 2023

Setting new expectations for open source maintainers, August 2021

Department of Defense Expands ‘Hack the Pentagon’ Crowdsourced Digital Defense Program, October 2018

HSPM: A Better Model to Effectively Preventing Open-Source Projects from Dying, May 2023