2024

REPORT

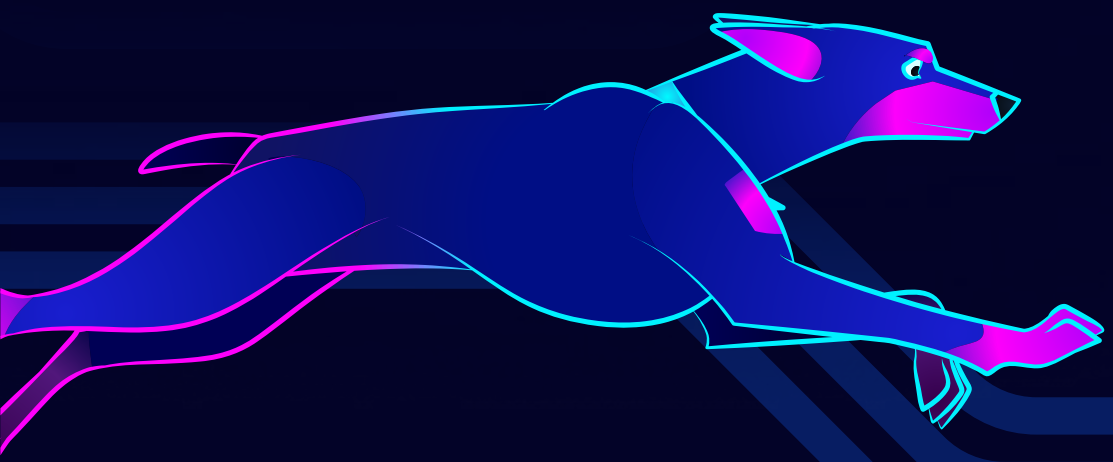
THE STATE OF OPEN SOURCE

snyk

EXECUTIVE SUMMARY

The 2024 Open Source Security Report reveals concerning trends in the software industry's approach to security, particularly in open source software (OSS) and supply chain security. Our research indicates three major findings. There are clear signs of "AppSec exhaustion," with organizations showing diminished engagement in security practices and struggling to meet vulnerability management goals. Open source supply chain security practices remain notably immature, with critical security measures being adopted by less than half of organizations. We continue to see blind trust in AI-generated code, with 77.9% of respondents expressing unwarranted confidence in AI's security capabilities despite evidence of frequent, serious vulnerabilities in AI-generated code.

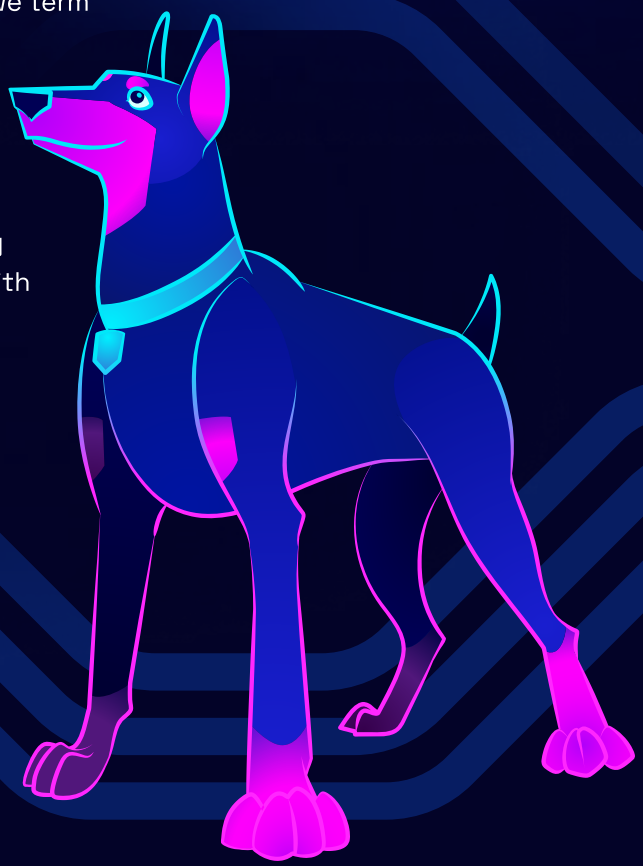
Our findings suggest an industry at a crossroads, facing mounting security challenges while simultaneously showing signs of fatigue in addressing them and potentially misplacing trust in AI solutions. Key findings paint a concerning picture: respondents indicating they had invested in additional security tooling specifically in response to supply chain or open source vulnerabilities dropped from 60.9% to 49.6% year-over-year, while respondents indicating their organizations had invested in additional security training in response to supply chain or open source vulnerabilities plummeted from 53.2% to 35.4%. General application security tooling investments declined more modestly, potentially indicating a focus on maturing the use of tooling already deployed. Despite 74% of organizations setting SLAs of a week or less for high-severity vulnerabilities, 52% regularly fail to meet these targets, with 14.8% reporting frequent failures. The disconnect between perceived and actual AI security is particularly striking. While 77.9% believe AI has improved code security (up from 76.5% last year), only 56.1% expressed concern about AI-introduced vulnerabilities, even as research shows frequent and serious security flaws in AI-generated code.



INTRODUCTION

As open source software continues to grow in popularity and importance, organizations building with and deploying open source software face a growing array of challenges in keeping their open source code secure. Over the past decade, the attack surface of the average enterprise technology stack has grown far more complex and convoluted due to a host of technology trends such as microservices, cloud and serverless computing, growing use of package managers, and more complex deployment and networking environments like Kubernetes. In this new world of increased application risk and challenging application security, the importance of improving security practices and hardening the software supply chain is critical – and even more so for open source, as it is the de facto target for the majority of cyberattacks. This report examines the current state of OSS security practices, supply chain vulnerability management, and the growing but potentially problematic role of AI in code development across the industry.

Our research surveyed developers, architects, application security, security, and DevOps practitioners across a wide variety of verticals to assess their open source and supply chain security practices. We focused on security practice adoption, technology choices, application security deployment, supply chain security maturity, and the impact of nearly ubiquitous coding assistants powered by artificial intelligence. The findings reveal a complex landscape where despite growing security threats, organizations are showing signs of decreased engagement with security practices — a phenomenon we term "AppSec exhaustion." Of particular concern is the continuing pattern of organizations turning to AI solutions with an overconfident belief in their security capabilities, potentially creating new vectors for vulnerability. This report explores these trends along with the current state of supply chain security maturity and the industry's potentially dangerous relationship with AI-generated code security.



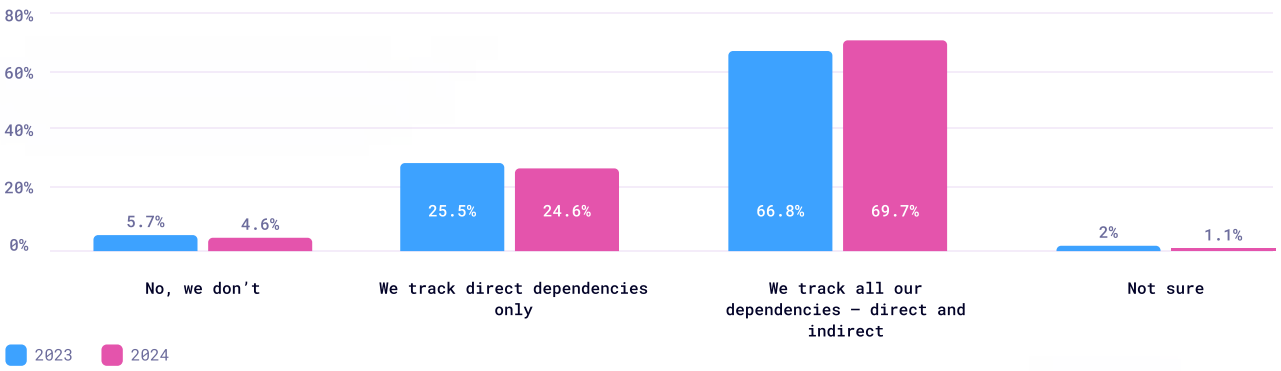
Section 1: Slowing progress in OSS security efforts and signs of AppSec exhaustion

There are signs of slowing progress in OSS security efforts and in DevOps efforts more broadly. Across many questions about supply chain security, we saw either little change year-over-year or surprising declines in adoption and usage.

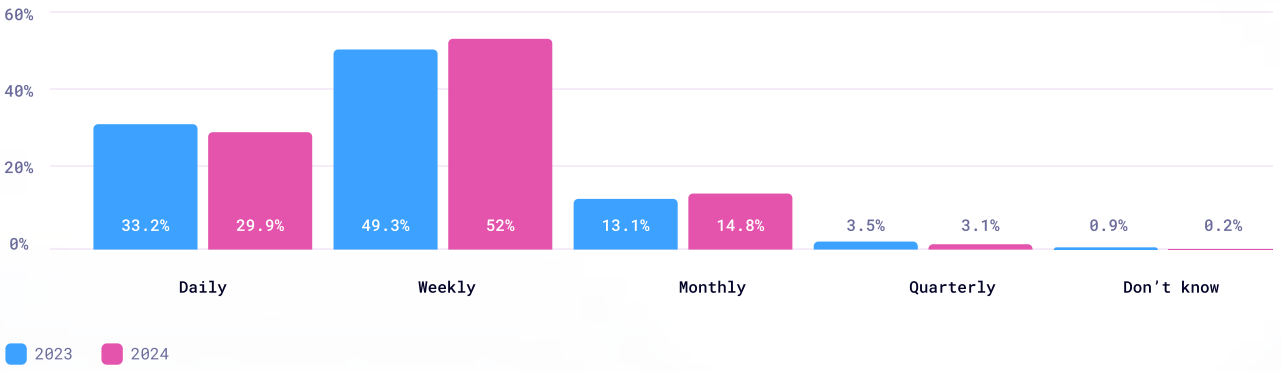
Dependency Tracking and Code Ship Frequency Unchanged

The percentage of respondents that track all dependencies rather than just direct only increased slightly year-over-year. Roughly one-quarter of respondents still only track direct dependencies. Nearly 5% don't track open source dependencies at all, although the majority of those who do not track do run software composition analysis (SCA). This implies that tracking may not be systematic but they do check dependencies and open source components. No change in code ship frequency means we find ourselves at a plateau with existing DevOps and deploy methodologies and that organizations are hitting a wall. In theory, reduced friction from improved tooling and Developer Experience should facilitate faster code iteration. In practice, this does not appear to be happening, likely due to AppSec exhaustion.

Does your company track which open source libraries your applications are using?



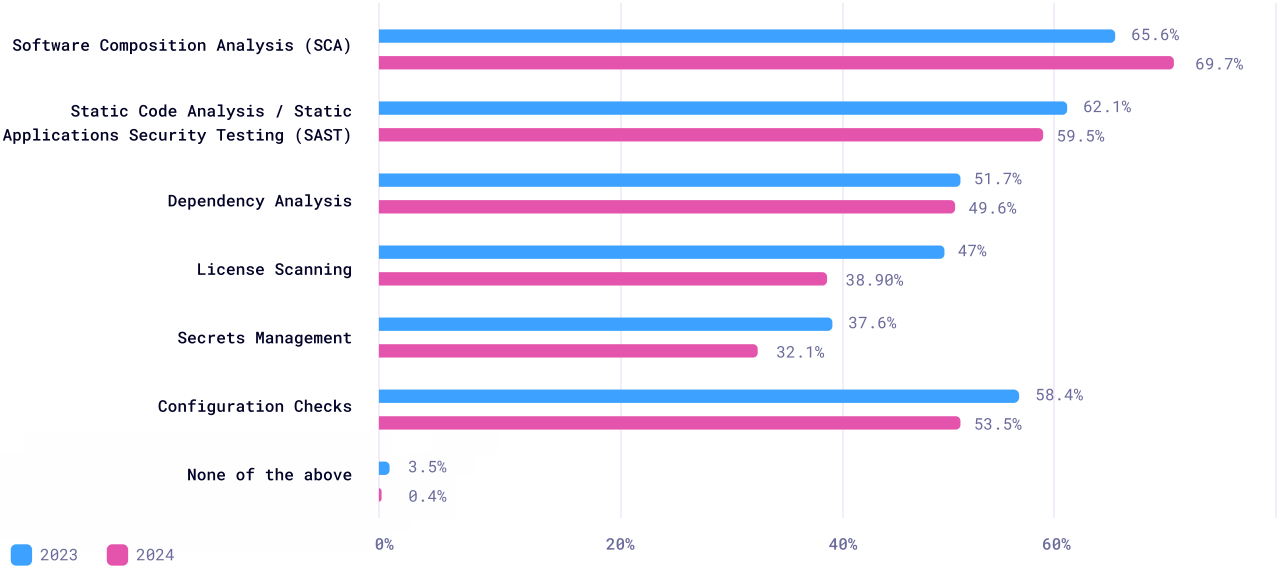
How frequently does your organization ship code?



Signs of Application Security Exhaustion

Signs of application security (AppSec) "exhaustion" are growing, with teams overwhelmed by AppSec requirements and struggling to adopt them. None of the eight AppSec methods surveyed exceeded 70% usage — even SCA (69.7%) and SAST (59.5%) fell short. Four methods — license scanning, secrets scanning, supply chain security, and dependency analysis — were below 50%, with license and secrets scanning under 40%. Year-over-year respondents reported declines in usage of several application security processes. These declines run counter to numerous reports that highlight a significant overall increase in application security tool spending, which likely represents that organizations that have advanced in their use of these tools are expanding their use.

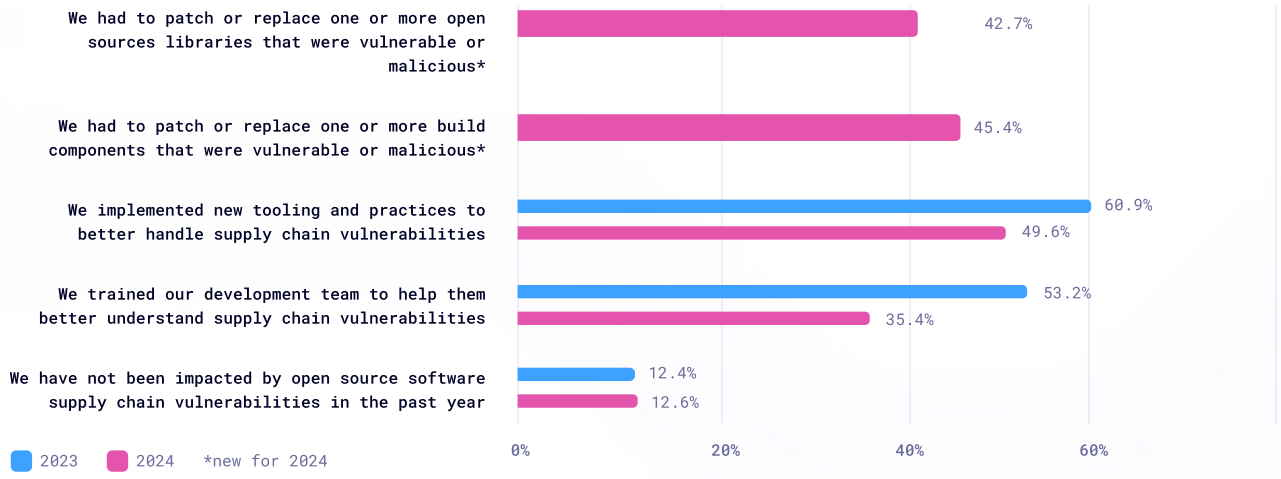
Which of the following application security processes does your organization apply?



Clear Declines in Resources Dedicated to Supply Chain Security

Compared to our research last year, we saw a marked decrease in proactive security measures from 2023 to 2024. The percentage of organizations implementing new tooling and practices to address supply chain vulnerabilities dropped from 60.9% in 2023 to 49.6% in 2024. Similarly, those investing in training their development teams on supply chain vulnerabilities decreased from 53.2% to 35.4% even though more than 40% of respondents indicated that they had to patch or replace vulnerable or malicious packages or build components. These reductions suggest that organizations may be feeling overwhelmed or fatigued by the continuous pressure of supply chain security demands, leading to reduced commitment to preventive actions. This may indicate fatigue, as some may opt to disengage rather than continually invest in complex and evolving security requirements.

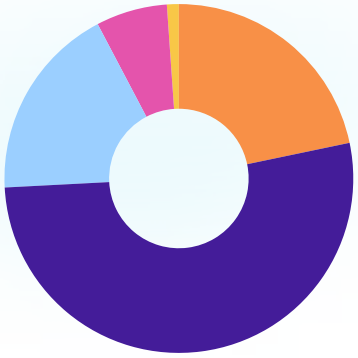
How have you or your organization been impacted by an open source or supply chain security vulnerability in the past year?



52% Fail to Meet Stated SLAs for High-Severity Vulnerability Fixes

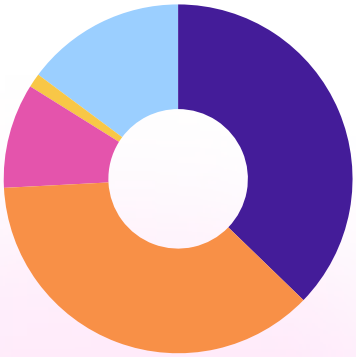
Widespread failure to meet vulnerability mitigation SLAs further highlights AppSec fatigue. Teams struggle to meet these goals, suggesting unrealistic expectations. While 74% have SLAs of a week or less, and 25% a day or less, 52% regularly miss these targets, and 14.8% frequently fail to meet them.

What is your SLA policy for fixing high-severity vulnerabilities?



- Less than a day 21.7%
- More than a month 6.6%
- Less than a week 52.4%
- Don't have one 1.1%
- Less than a month 18.1%

How often do you break your vulnerability SLA?

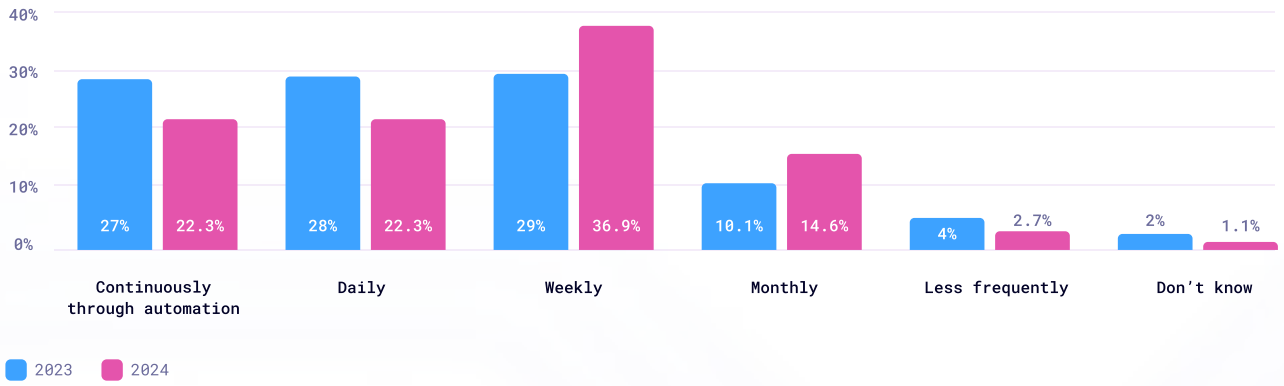


- Often 14.8%
- Never 9.7%
- Sometimes 37.2%
- Don't know 1.3%
- Rarely 36.9%

Teams Auditing Code Less Frequently and Less Continuously

Between 2023 and 2024, there was a noticeable shift toward less frequent code auditing among teams. The percentage of teams auditing code weekly increased from 29% in 2023 to 36.9% in 2024, while continuous auditing through automation decreased from 27% to 22.3%. Additionally, monthly audits saw an increase from 10.1% to 14.6%, indicating a trend toward less frequent and less continuous auditing practices.

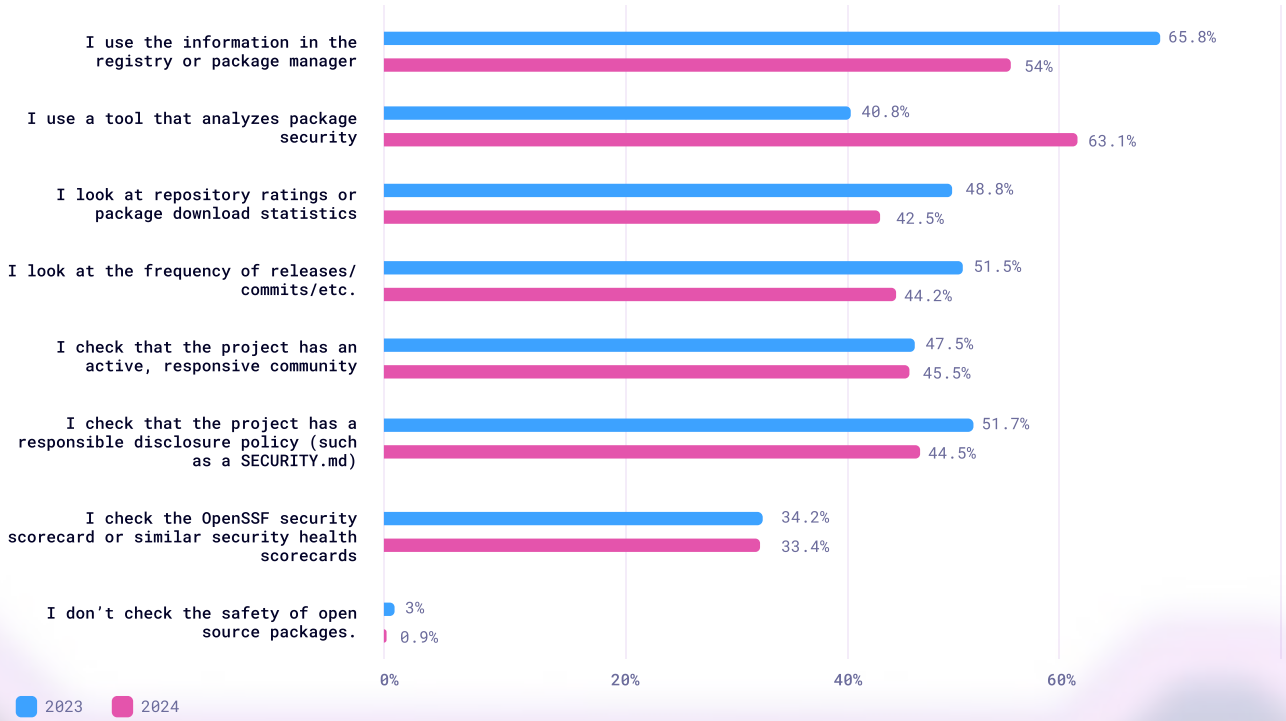
How Often Do You Audit Your Code?



Big Increase In Using Tools That Analyze Package Security

Reliance on tools for package security rose 22.3% YoY, with an 11% drop in manual approaches like checking registry information. This shift may reflect either a proactive move toward automation or a need to manage OSS-related security burdens as teams rely increasingly on tools for package safety analysis.

How do you check the safety of the open source packages used by your software?

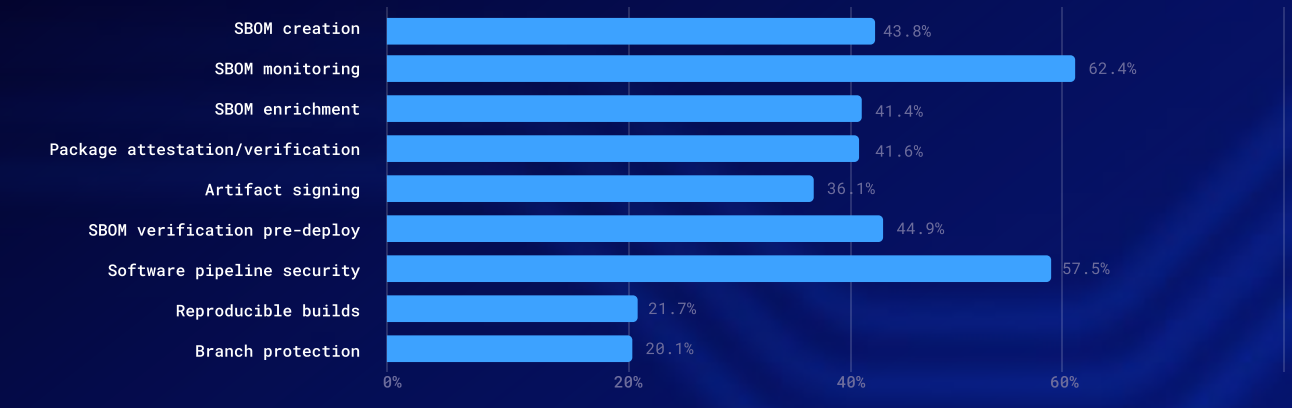


Section 2: Supply Chain Security Remains Immature

Security practices in OSS supply chains lack maturity, with most measures under-adopted and inadequate to meet evolving threats.

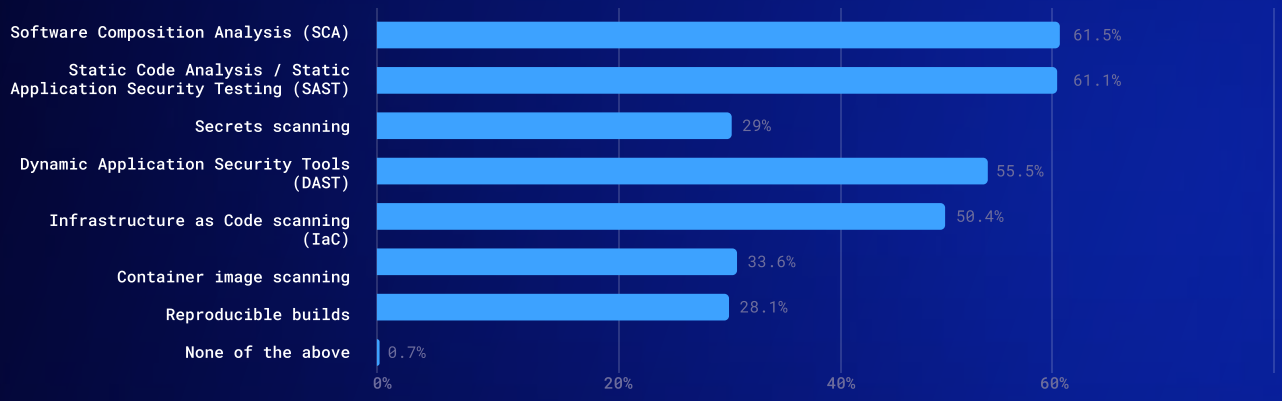
Open source supply chain security remains lightly adopted, with no practice used by more than two-thirds of organizations. SBOM monitoring leads at 62%, and only software pipeline security also surpasses 50% usage. Just 44% verify SBOMs pre-deployment, 41% check for signed artifacts, and only around 20% use protections like reproducible builds or branch protection. This leaves build pipelines vulnerable, as many rely on outdated scanning and lag in adopting cloud-native security.

Which supply chain security practices does your organization follow?

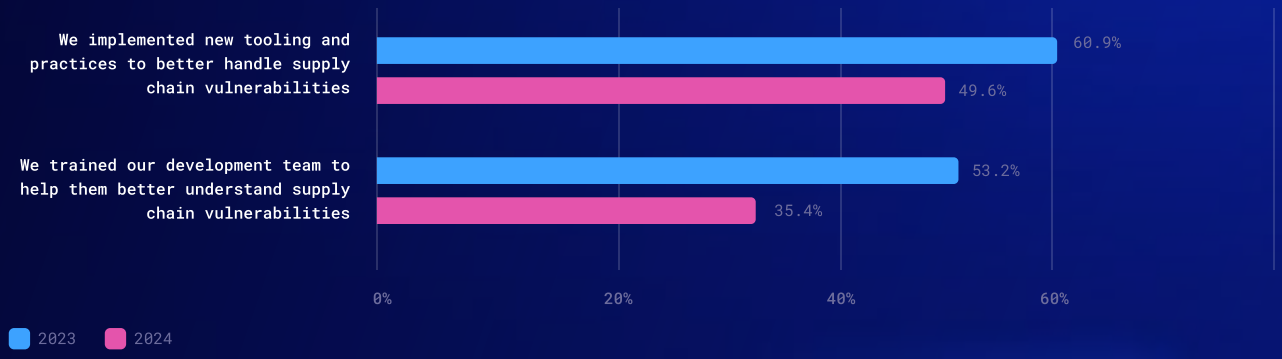


Despite the maturity of SCA and Static Application Security Testing (SAST), adoption is still just over 60%, with container scanning surprisingly low at 35%. Reproducible builds and secrets scanning are around 20%, although supply chain vulnerabilities continue impacting both code and build components. As seen in the previous section, between 2023 and 2024, new tooling adoption went from 61% to 50%, while security training went from 53% to 35%, even as vulnerabilities rose. In 2024, 45% replaced compromised build components, and 42% swapped vulnerable OSS libraries.

What security practices do you use?

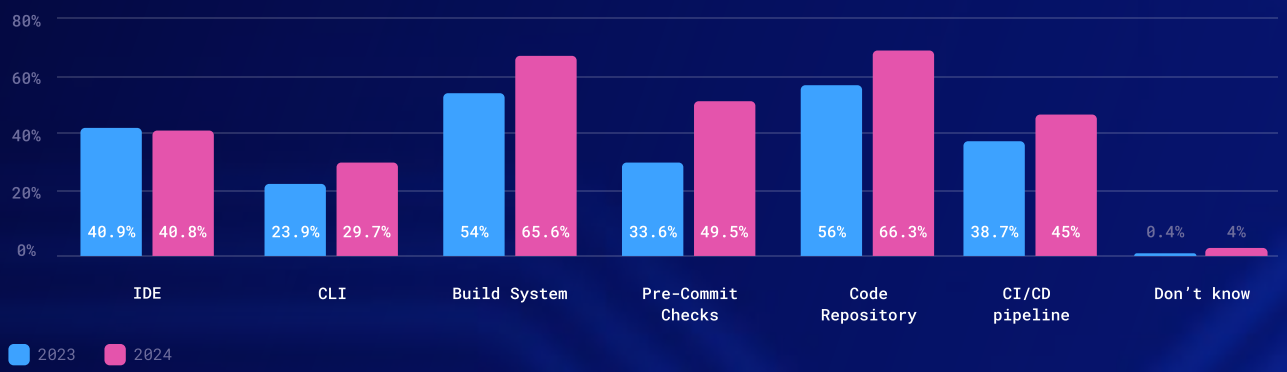


How have you or your organization been impacted by an open source or supply chain security vulnerability in the past year?



One positive trend is the increased distribution of security tooling across development stages. Build systems and pre-commit checks saw notable increases (11.6% and 15.9%, respectively), emphasizing early vulnerability detection. Code repositories (10.3%), CI/CD pipelines (6.3%), and CLI tools (5.8%) also showed growth, while IDE integration slightly declined, suggesting a preference to reduce developer cognitive load by shifting security out of coding environments.

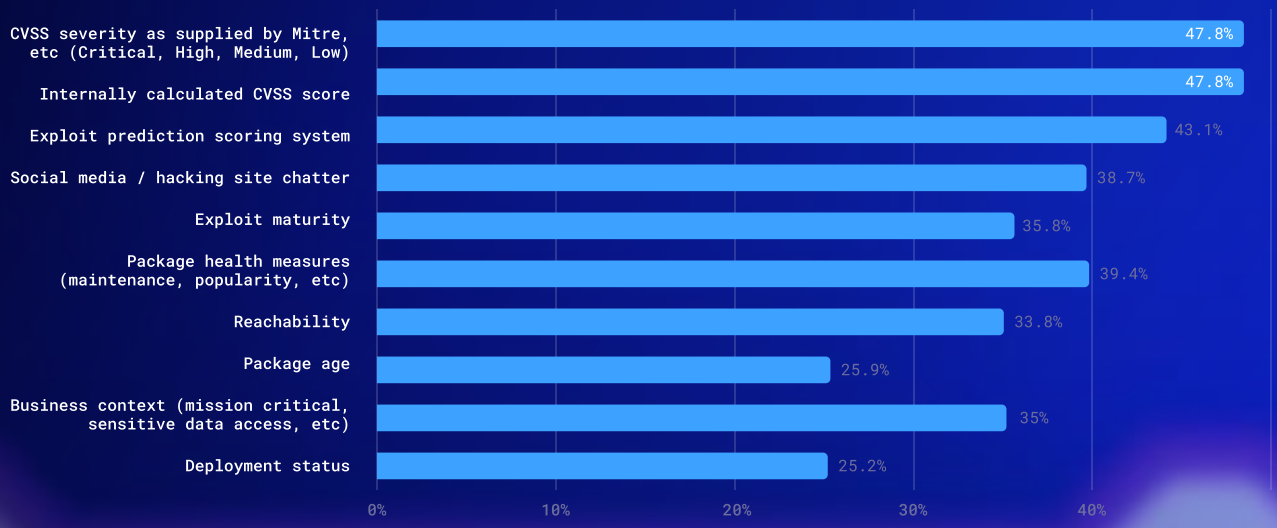
Where are security tools integrated?



Lack of Maturity and Sophistication in Risk Vulnerability Analysis

When determining how severe a vulnerability risk is, the most widely used approaches are traditional scoring systems (CVSS, exploit prediction scoring system (EPSS)). Far less widely used are measures that reflect the actual risk of a vulnerability to an organization (reachability, deployment status, business context). Teams are still struggling to adopt more relevant vulnerability severity rating systems. This implies that they are still struggling to effectively triage vulnerabilities and build risk models that accurately reflect the true business risk of vulnerabilities.

What factors do you use to determine the severity of a vulnerability?



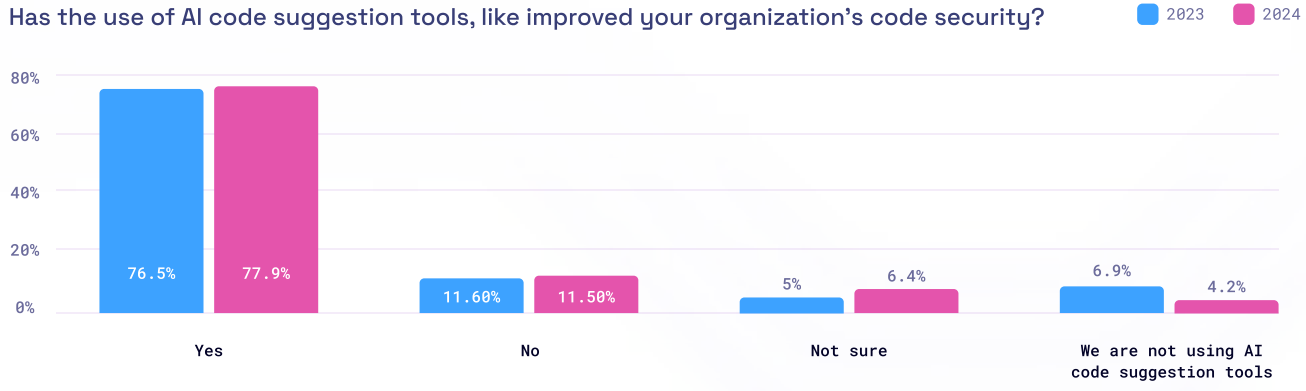
Section 3: Continued Misplaced Confidence in the Security of AI-generated Code

High confidence in AI for secure code persists, despite evidence of vulnerabilities, signaling a need for better education on AI risks.

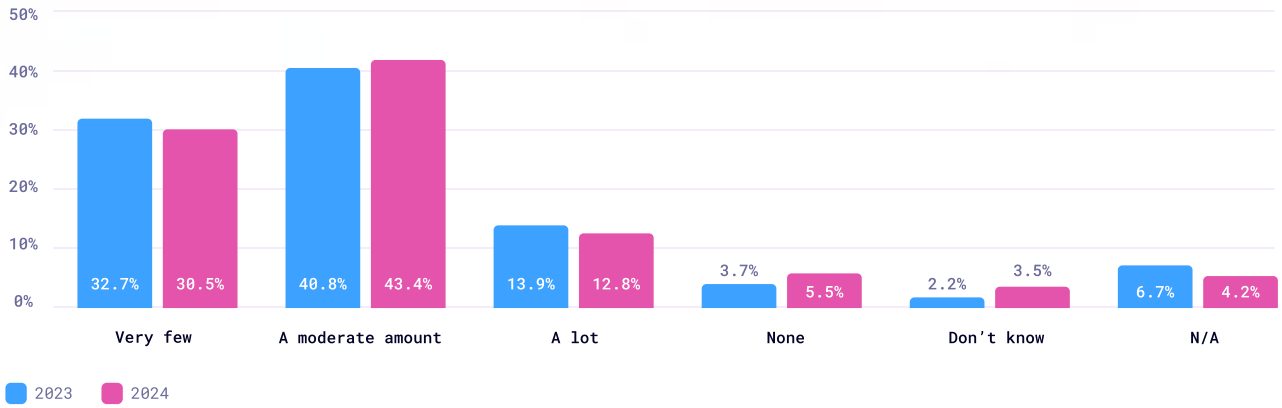
Respondents continued to hold high levels of misplaced confidence in the ability of AI tools to generate secure code. Despite 77.9% of respondents believing AI has improved code security (up slightly from 76.5% last year), Snyk's research shows frequent, serious vulnerabilities in AI-generated code. Meanwhile, 56.1% remain concerned about vulnerabilities introduced by AI — a modest decline in worry, with 38.1% now expressing little or no concern. This disconnect highlights an education gap, as many organizations may be overly trusting of AI's security capabilities. On the positive side, 84.1% of respondents apply the same scrutiny to open source components recommended by AI as they do to human-suggested components, reflecting a mature approach. However, confidence in AI's security contributions remains unaligned with actual risks, emphasizing the need for consistent oversight to prevent a false sense of security as AI adoption grows.

Respondents said AI improved code security and did not introduce vulnerabilities....

Has the use of AI code suggestion tools, like improved your organization's code security?

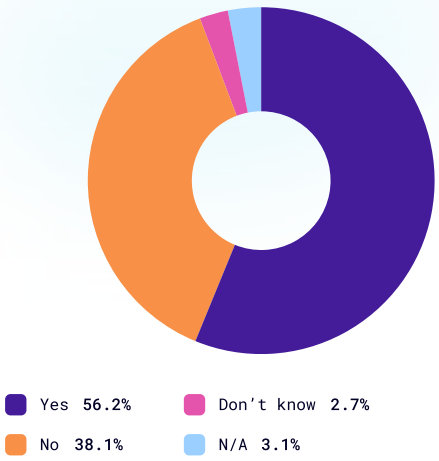


How many vulnerabilities has AI introduced into your code?

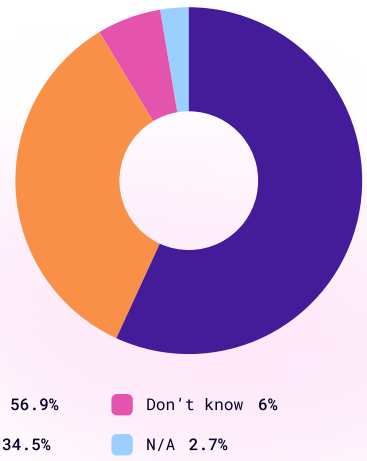


...yet engineers and security teams remain concerned about AI introducing vulnerabilities in code or license and copyright issues....

Are you concerned that using AI coding tools will introduce security vulnerabilities into your applications?

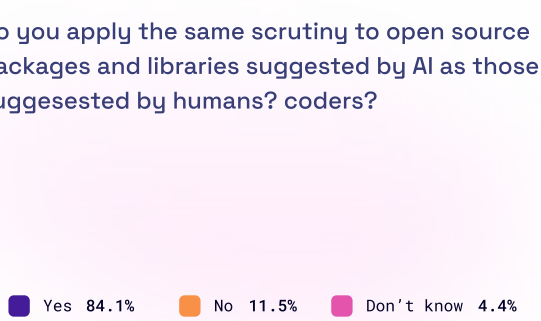


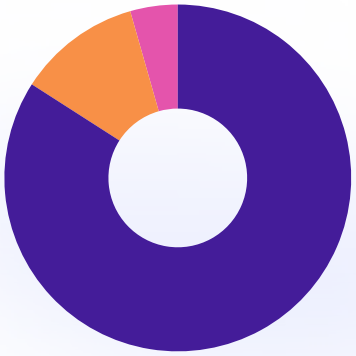
Are you concerned that using AI coding tools will introduce open source licensing and copyright problems into your stack?



...but teams are scrutinizing AI-suggested packages and libraries the same way as human-suggested packages, which is encouraging and implies they understand that AI suggestions present as much risk as human-suggested libraries and packages.

Do you apply the same scrutiny to open source packages and libraries suggested by AI as those suggesested by humans? coders?



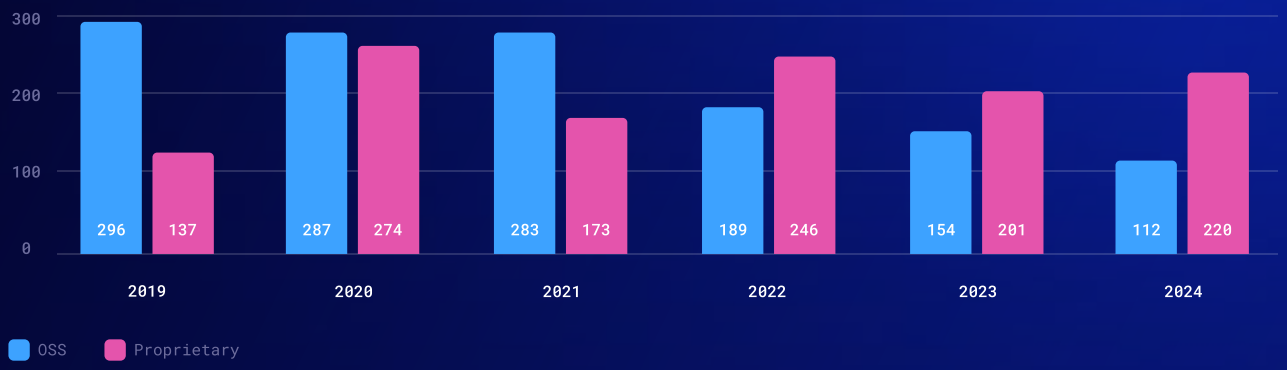


Section 4: Evidence of General Open Source Security Progress

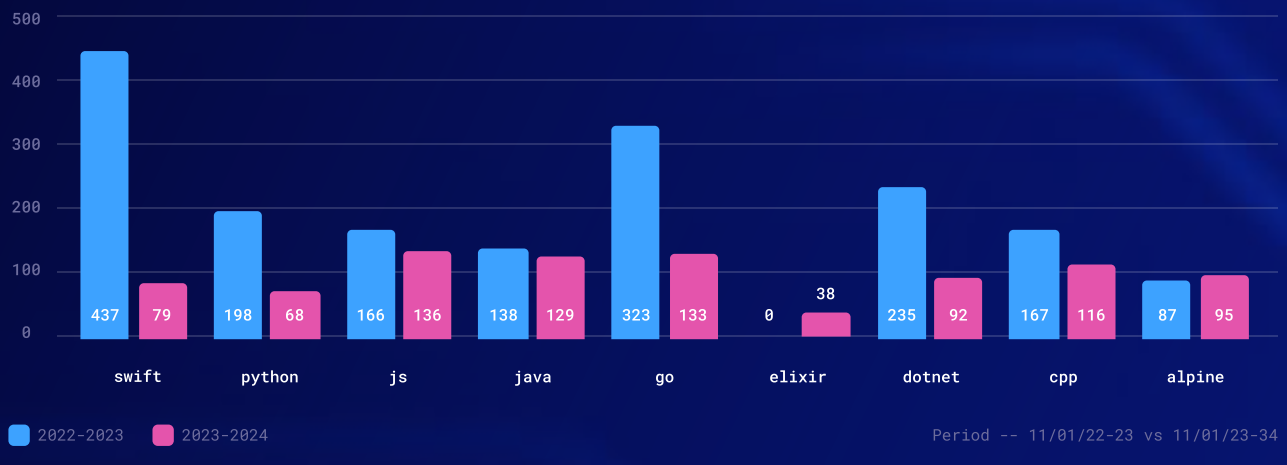
The OSS community has significantly reduced time-to-fix for critical and high severity vulnerabilities in open source projects and continues to outperform proprietary software in response times based on vulnerability database findings.

Even as we see evidence of AppSec exhaustion and slow adoption of supply chain security practices, the open source software (OSS) community has made significant progress in a critical measure reducing the time it takes to fix high and critical severity vulnerabilities present in open source software projects. Over recent years, OSS has consistently shortened its "time-to-fix" for critical bugs, outpacing proprietary software in responsiveness. OSS is also improving time-to-fix across open source projects across popular languages from 2022-2023 and 2023-2024, with a clear reduction in resolution times. These improvements underscore the community's dedication to enhancing security and responsiveness, demonstrating the effectiveness of collaborative, transparent approaches in addressing vulnerabilities quickly at the level of project and project code.

Time-to-Fix of High/Critical Severity Bugs: OSS vs Proprietary



Comparison of Critical/High Severity Time-to-Fix in OSS



CONCLUSION

Room for Improvement in Supply Chain, AppSec with Progress in General OSS Code

The findings from our 2024 research paint a concerning picture of an industry struggling to maintain momentum in security practices while facing evolving challenges. The observed "AppSec exhaustion" phenomenon, evidenced by declining engagement in security measures and widespread failure to meet vulnerability management goals, suggests that current approaches to security may be unsustainable. Encouragingly, we did identify signs of ongoing improvement in the underlying foundations of open source software, with projects turning critical fixes around more quickly and the open source community continuing to distance itself from proprietary code in terms of speed of fixes.

The immaturity of supply chain security practices, combined with decreasing investment in proactive security measures, creates a particularly vulnerable environment. This vulnerability is potentially exacerbated by the industry's overreliance on AI-generated code security. While AI tools offer promising capabilities for code generation, the disconnect between perceived and actual security risks — with 77.9% of respondents expressing confidence despite evidence of serious vulnerabilities — suggests a dangerous trend that could lead to significant security oversights.

Moving forward, organizations need to:

- Reassess their approach to security to prevent burnout and ensure sustainable practices.
- Improve prioritization in vulnerability management and other supply chain risk management tasks.
- Prioritize the adoption of fundamental supply chain security measures and deploy newer supply chain security measures to improve security posture.
- Include more holistic risk analysis as part of SLA determination to ensure security teams can focus more time on risks that matter.
- Take a more cautious and measured approach to AI-generated code, implementing rigorous security reviews rather than assuming inherent security.
- Establish clear guidelines for validating and testing AI-generated code, treating it with the same or greater scrutiny as human-written code.

These findings suggest that the industry must find new ways to balance security requirements with team capacity while maintaining vigilance against emerging threats, including those potentially introduced by overreliance on AI tools. Without addressing these challenges and adjusting attitudes toward AI-generated code security, organizations risk falling further behind in their security posture as threats continue to evolve.

Methodology

We surveyed 453 technologists across application development and security. We used many of the same questions we had asked in the 2023 State of Open Source Security in 2023 and compared to the past results, where applicable. Respondents were located in the United States of America, Canada, and the United Kingdom. The question types included binary responses (only one answer allowed), multi-picks (choose all that apply), and ratings on a scale of 1 to 4, with 1 being 'most concerning'. Respondents came from a wide variety of sectors, including automotive, business services, communications, education, energy and utilities, entertainment/media, financial services, government, and SaaS technology.

snyk