BLACK DUCK®

BALANCING AI USAGE AND RISK IN 2025

THE GLOBAL STATE OF DEVSECOPS

CONTENTS

NAVIGATING SPEED, FRICTION, AND AI IN DEVSECOPS

The goal of DevSecOps has always been to ensure that speed and safety are on equal footing. Black Duck's latest research reveals that although many organizations have successfully built high-velocity development pipelines, **security automation lags far behind.**

Our research also shows that a **proliferation of security tools** intended to identify and manage risk has created the opposite effect: a climate where DevOps teams are overwhelmed by noisy, unhelpful results.

The most transformative challenge facing DevSecOps is artificial intelligence. Our research found that AI is simultaneously perceived as **the most promising tool for embedding security directly into coding** and a **dangerous source of risk.**

This report provides a data-driven deep dive into all three of those challenges for DevSecOps teams: the ongoing tension between development speed and security, the "tool sprawl" crisis, and the double-edged risk/reward sword that is AI.

IT'S TIME TO STOP BUYING MORE TOOLS AND **START OPTIMIZING THE ONES YOU HAVE**

Why You Should Read This Report

This report is a strategic analysis of the forces that are shaping how software is built and secured. We've tailored the data and insights for the people who define the strategy as well as for those who must execute it.

For Executive Leaders

This report is about business risk, investment efficiency, and competitive advantages. Our findings on toolchain inefficiency give you a clear way to evaluate the ROI of your security spending. Our research shows that it's time to stop buying more tools and start optimizing the ones you have.

Our analysis uncovered an AI confidence paradox and shows that the explosion of "shadow AI" is a growing category of unmanaged business risk. For all its promises, unmanaged AI can also be a threat to your IP and your compliance posture that demands a comprehensive governance strategy.

Bottom line: This report gives you the data you need to align your security investments with real business outcomes and build a faster, more resilient organization.

For DevSecOps Professionals

This report is about your daily reality. We measured and quantified the friction you feel every day, from the soul-crushing drag of "tool noise" to the "alert fatigue" that's making you ignore the false positive warnings from the security tools that should be helping you.

Our findings validate the constant battle you fight between speed and security. It gives you the data to start making a business case for the integrated, developer-focused tools and processes you need. Use this data to benchmark your team, understand the essential new skills in an AI-driven world, and get support for the strategic changes required to build secure software.

Survey Methodology

The analysis in this report is grounded in a comprehensive survey of more than 1,000 global software and security professionals conducted by the international market research firm Censuswide in July and August 2025. Throughout the report, data points are cited using the marker [**Q#**] to refer to the specific question in our 15-question survey. The full list of questions can be found in Appendix A.

Our respondent base was designed to provide a holistic view of the DevSecOps ecosystem.

- **Geographies:** Respondents were drawn from key technology hubs across North America, Europe, and Asia, including the U.K., U.S., Singapore, Finland, France, Germany, Japan, and China.
- **Roles:** The survey captured perspectives across the entire organizational chart, from hands-on developers, DevOps engineers, and security architects to senior leadership, including CISOs and directors of security.
- **Companies:** Participants represented organizations of all sizes, from startups to global enterprises with over 100,000 employees, across all major industry verticals, including Technology, Banking/Financial/Insurance, Manufacturing, and Government.

EXECUTIVE SUMMARY: KEY FINDINGS

For senior leadership, the message from our data is clear: The demand for speed, the complexity of your toolchain, and the disruptive force of AI have changed the game. Your ability to navigate these new realities will be the single biggest factor determining your risk posture, your capacity to innovate, and your competitive position.

Here are the six most critical findings from our global survey of more than 1,000 professionals. These findings represent the biggest challenges—and opportunities—in DevSecOps today.

You're Shipping Fast, but You're Building on Sand.

You've achieved incredible speed, with nearly 60% of organizations deploying code daily or even more often [**Q1**]. But that speed is built on a fragile foundation. Security practices are dangerously immature, with 45.56% of companies still relying on manual processes to get new code into the security testing queue [**Q2**]. This automation gap means many businesses are simply unaware of their vulnerabilities, with 61.64% of organizations testing less than 60% of their own applications [**Q3**]. The result is that you're accumulating a massive security debt with every single release.

The "Tool Sprawl" Crisis.

If your organization brought in a multitude of application security testing (AST) tools to deal with a complex threat landscape, you've likely found that the strategy has had unintended consequences. Over 71% of our respondents say a significant chunk of their security alerts is just "noise"—false positives or duplicate findings from different tools [**Q5**]. This flood of useless information is destroying the ROI of your security investments.

QUESTIONS

Q1. On average, how often does your primary team/organization deploy code changes to production for your primary or business-critical application(s)?

Q2. What is the PRIMARY mechanism used to ensure new code sources (e.g., projects, repositories, significant feature branches) are included in your organization's application security testing program?

Q3. Approximately what percentage of your projects, branches, and repositories are being included in your application security testing queue?

Q5. Approximately what percentage of security test results are noise? For example: duplicative results, false positives, conflicting with other tests/tools.

OVER 71% OF RESPONDENTS SAY A SIGNIFICANT CHUNK OF THEIR SECURITY ALERTS IS JUST "NOISE"

Security Is Still a Speed Bump.

The operational drag from all that tool noise is the main reason over 81% of DevSecOps professionals say that security testing slows down development [**Q6**]. This issue creates a toxic tension between development and security teams and actively undermines the entire point of DevSecOps.

AI Is a Double-Edged Sword.

AI is the most disruptive force in development, but the jury is still out if it's a friend or a foe. A clear majority (56.55%) of our respondents find that it introduces novel security risks [**Q8**]. At the same time, an even bigger majority (63.33%) believe it's helping them write more-secure code [**Q11**]. This apparent paradox makes AI governance a high-stakes balancing act that most companies are still trying to master.

Overconfident and Underprepared.

There appears to be a massive disconnect between promised capabilities and reality when it comes to AI. Despite admitting that their existing tools are noisy and have huge coverage gaps, 88.81% of our respondents say their organizations are confident that they can handle the new, more complex risks introduced by AI [**Q9**].

The Unifying Priority Is Development Workflow Integration.

Our survey shows that the problem isn't necessarily the tools themselves, but the painful way developers are forced to use them. When we asked our respondents for their single most important priority for improving their application security testing capabilities, the answer was decisive. It wasn't a new tool or more coverage. The #1 priority is "better development workflow integration" (27.27%) [**Q15**]. The future must be about embedding security seamlessly into the way developers already work.

THE FUTURE MUST BE ABOUT **EMBEDDING SECURITY SEAMLESSLY INTO THE WAY DEVELOPERS ALREADY WORK**

QUESTIONS

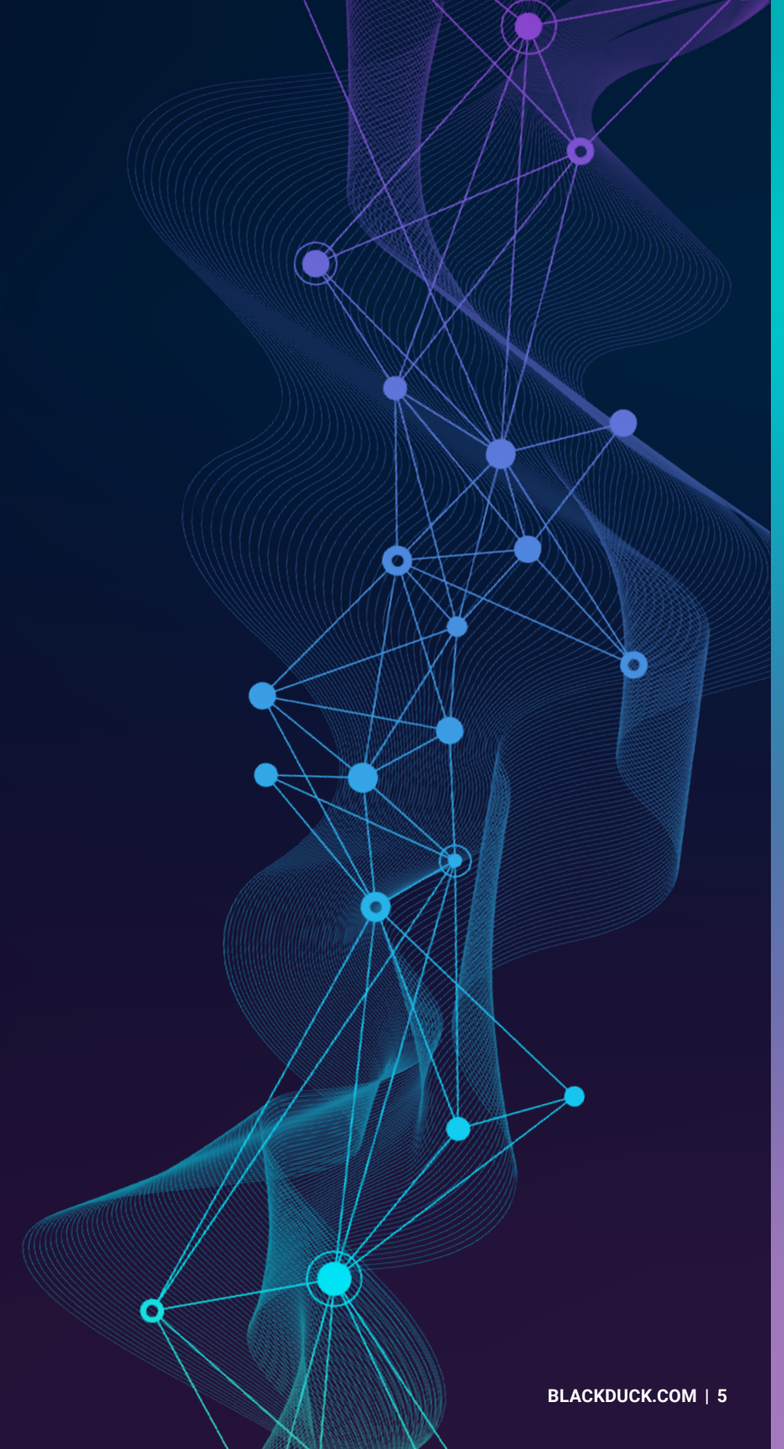
Q6. Which statement best describes the relationship between application security testing and software development/delivery?

Q8. To what extent do you agree or disagree with the following statement: "The use of AI coding assistants has introduced new security risks into, or made it harder to detect issues within, our codebase."

Q9. How confident are you that you have the processes and tools in place to address security issues introduced by AI code generators/coding assistants?

Q11. To what extent do you agree or disagree with the following statement: "The use of AI coding assistants has tangibly improved our ability to write more-secure code and more effectively address security risks within our DevSecOps life cycle (shift left)."

Q15. What is your organization's SINGLE MOST important priority for improving its application security testing capabilities in the next 12 months?



“SEC” LAGS BEHIND “DEV” AND “OPS”

The central problem in DevSecOps today is lopsided maturity. Most organizations in our survey have seemingly mastered the “Dev” and “Ops” parts of the equation. But the “Sec” part is still lagging far behind, stuck with manual processes and incomplete coverage. This has created a fragile system where the speed of development is constantly outrunning security’s ability to keep up. The result? Growing risk.

Velocity as the New Standard

Shipping fast is no longer an aspiration; it’s the standard. A clear majority of companies, 59.74%, are now pushing new code to production for their critical applications every single day, or even multiple times a day [**Q1**].

You would expect that organizations releasing more frequently would also be using automation to detect and add code to the AST queue. Our data shows the converse is true: a heavy prevalence of fully/primarily manual means of detecting and adding projects to the test queue (Figure 1).

Notably, of those shipping code multiple times per day, 47.48% use primarily or fully manual mechanisms for adding projects to security testing queues (compared to 36.33% favoring automation) [**Q1, Q2**]. This skew toward manual practices is also evident for those releasing daily and weekly.

The Automation Maturity Gap

The data is clear: CI/CD is the engine that runs modern software development. But that high-speed engine is bolted to an insecure chassis. Our data shows that security practices are dangerously immature in many organizations.

- **Manual by Default:** For nearly half of all organizations (45.56%), security is still a manual process. The critical first step—getting new code into a security testing program—often depends on someone doing it by hand [**Q2**].
- **A Concerning Coverage Failure: O**ver 61% of respondents admit that their companies are testing 60% or less of their application portfolio [**Q3**]. This means, for most organizations, a huge—and largely unknown—portion of their software is going out the door without proper security vetting. This is the hidden security debt of the DevOps era, and it’s getting bigger with every release.

QUESTIONS

Q1. On average, how often does your primary team/ organization deploy code changes to production for your primary or business-critical application(s)?

Q2. What is the PRIMARY mechanism used to ensure new code sources (e.g., projects, repositories, significant feature branches) are included in your organization’s application security testing program?

Q3. Approximately what percentage of your projects, branches, and repositories are being included in your application security testing queue?

More Tools, More Problems

To keep up with threats, companies have been throwing tools at the problem. A diverse arsenal of AST solutions was supposed to create a layered defense. But this strategy backfired. Instead of a safety net, our modern toolchain has become a primary source of friction, noise, and inefficiency that actively works against that goal.

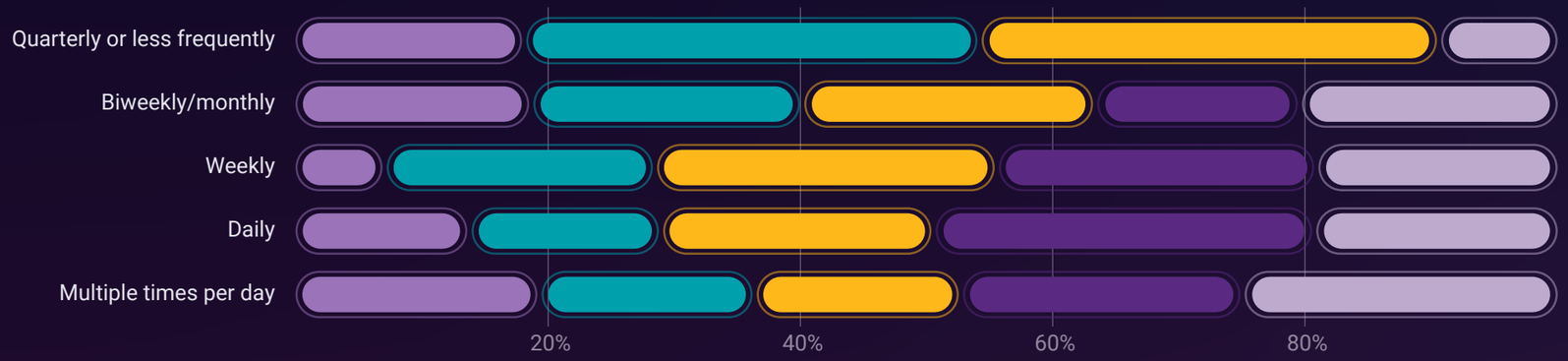


Figure 1. The data shows a heavy prevalence of manual means to detect and add projects to test queues.

- **Fully automated:** Standard new sources are automatically discovered and configured for testing within routine manual steps
- **Primarily automated:** Relies mostly on tools automatically detecting or integrating new sources, with some manual handling for exceptions or specific cases
- **Balanced mix:** Significant use of both manual processes and automated discovery/onboarding
- **Primarily manual:** Mostly manual processes, although some minor automation might assist
- **Fully manual:** Relies entirely on manual declaration or selection by teams (Dev or Sec)

Mapping the AST Ecosystem

Today's application security landscape is defined by tool sprawl. No single technology dominates. Instead, everyone has a portfolio of tools. The top five most common tool types are all used in nearly equal proportion [**Q4**].

Each of these disconnected systems comes with its own overhead, its own APIs, and its own alert formats, creating a fragmented mess that DevOps teams must navigate every day.

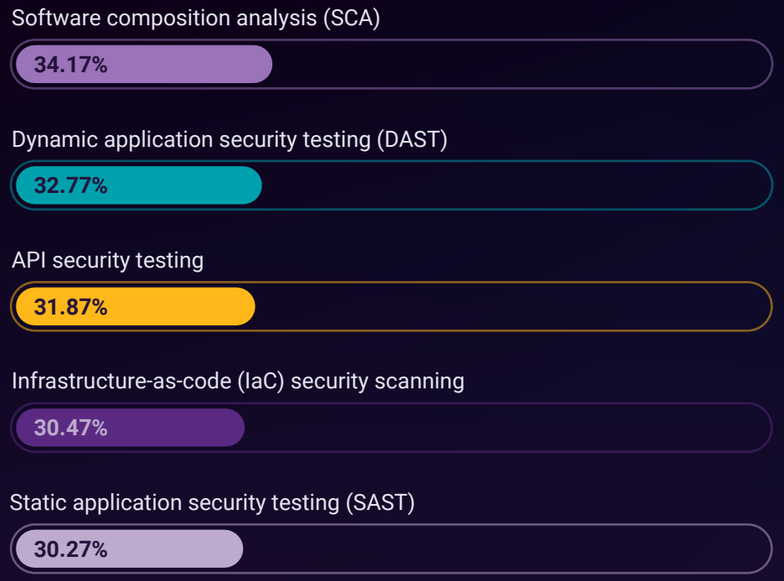


Figure 2. No dominant player: Everyone has a portfolio of disconnected tools.

Drowning in False Positives

The single biggest problem created by a fragmented toolchain is the overwhelming volume of noise, duplicate results from different tools, and conflicting findings that leave developers in the dark about real threats to their code.

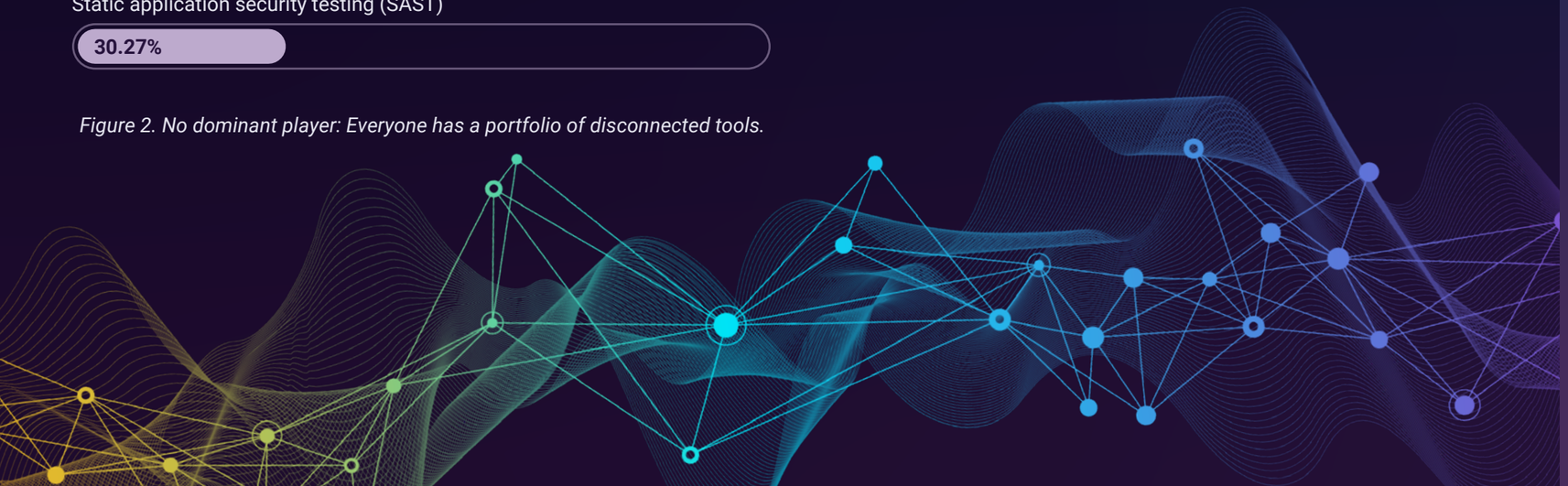
The scale of this problem is staggering. A combined 71.63% of all respondents say that a significant portion—between 21% and 60%—of their security alerts are basically useless [**Q5**].

When most alerts are junk, you're forced to waste time on the low-value, soul-crushing work of triage. This leads directly to alert fatigue, where developers start ignoring all the warnings, making it a matter of when, not if, a critical alert gets lost in the noise.

QUESTIONS

Q4. Which types of application security testing (AST) tools does your organization currently utilize?

Q5. Approximately what percentage of security test results are noise? For example: duplicative results, false positives, conflicting with other tests/tools.



49% OF RESPONDENTS WHO RELY ENTIRELY ON MANUAL PROCESSES FEEL APPLICATION SECURITY TESTING SEVERELY SLOWS DOWN DEVELOPMENT AND DELIVERY

The Persistent “Speed vs. Security” Dilemma

Operational drag from tool sprawl and noise, combined with a reliance on manual processes, has a direct, measurable impact on the main goal of DevOps: speed. An overwhelming 81.22% of professionals say that application security testing slows down their development and delivery life cycle [**Q6**].

Forty-nine percent of respondents who rely entirely on manual processes feel application security testing severely slows down development and delivery (Figure 3). In contrast, 32% to 35% of those who say security testing does not slow development are fully or primarily automated.

For organizations relying heavily on manual processes, the promise of secure, high-velocity DevOps is still unfulfilled. The “Sec” part of DevSecOps is seen as a roadblock, not an enabler. It’s a vicious cycle: You buy more tools, which create more noise, which require more manual triage, which makes developers see security as a speed bump, which makes them resist the very security practices DevSecOps is trying to encourage.

	Fully automated	Primarily automated	Balanced mix	Primarily manual	Fully manual
Application security testing **does not slow down** development/delivery	35%	32%	14%	11%	7%
Application security testing **slightly slows down** development/delivery	11%	23%	29%	26%	12%
Application security testing **moderately slows down** development/delivery	7%	14%	23%	37%	19%
Application security testing **severely slows down** development/delivery	11%	7%	16%	16%	49%

Figure 3. Manual processes affect the perception of slowdowns due to security testing.

QUESTION

Q6. Which statement best describes the relationship between application security testing and software development/delivery?

AI DISRUPTION: A DOUBLE-EDGED SWORD

AI-powered coding assistants and open source AI models are now deeply embedded in the daily life of developers, and our data reveals a huge paradox around its usage. We've found that AI is seen as both a powerful new way to improve security and a significant new source of complex, scalable risk. Figuring out how to navigate the dual nature of AI is the central strategic challenge of every security leader in this new world.

QUESTIONS

Q7. On average, how frequently do you or your team utilize AI coding assistants (e.g., GitHub Copilot, Amazon Q, Gemini Code Assist, Tabnine) during software development?

Q8. To what extent do you agree or disagree with the following statement: "The use of AI coding assistants has introduced new security risks into, or made it harder to detect issues within, our codebase."

Q10. What is your organization's PRIMARY security-related concern regarding the implications of using AI code generators/coding assistants?

Q11. To what extent do you agree or disagree with the following statement: "The use of AI coding assistants has tangibly improved our ability to write more-secure code and more effectively address security risks within our DevSecOps life cycle (shift left)."

Q14. Are you using any open source AI models (e.g., from Hugging Face) in the software you build?

Widespread Adoption of AI and Shadow AI

AI adoption has been less of a curve and more of a vertical line, a fundamental shift in how software gets made.

- **Deep Integration:** A combined 43.66% of professionals are now using AI coding assistants frequently or constantly, deeply integrating AI into their daily work [**Q7**].
- **Pervasive Use of Open Source AI Models:** The use of open source AI models from communities like Hugging Face is even more widespread. Nearly 97% of organizations are using these models in the software they build [**Q14**].
- **The Shadow AI Problem:** AI was adopted so quickly that governance hasn't had a chance to catch up. Our data shows a significant shadow AI problem. A notable 10.69% of respondents admit to using AI coding assistants without official permission, in an unverified or unmonitored way [**Q7**]. Developers are grabbing whatever tools they can to be more productive, and by doing so, they are exposing their companies to a whole new world of unmanaged security and compliance risk.

AI IS SEEN AS BOTH A POWERFUL NEW WAY TO IMPROVE SECURITY AND A **SIGNIFICANT NEW SOURCE OF COMPLEX, SCALABLE RISK**

Risk vs. Security

Most organizations are holding two completely contradictory ideas about AI simultaneously.

- **A New Threat Vector:** There's a clear consensus that AI is a new and serious threat. A majority of respondents (56.55%) agree that "the use of AI coding assistants has introduced new security risks" [**Q8**]. They're worried about real, practical things: the scale and speed of new vulnerabilities, compliance nightmares, and the risk of their proprietary code ending up in a training model [**Q10**].
- **A Force Multiplier for Security:** At the same time, there's an even stronger belief that AI can be a powerful ally. A significant majority—63.33%—of DevSecOps professionals agree that AI has "tangibly improved our ability to write more-secure code" [**Q11**]. They see the benefits in faster vulnerability identification, better coding consistency, and quicker fixes.

A Dangerous Disconnect?

We asked respondents how confident they were in their organization's ability to manage risks introduced by AI coding assistants like Copilot, Claude Code, and ChatGPT. Despite many respondents admitting that their current toolchains are manual, have low coverage, and are incredibly noisy, an overwhelming 88.81% of respondents feel confident that they can handle the new and complex security issues introduced by AI coding assistants [**Q9**]. An even higher number—93.90%—are confident they can manage the open source license risks from AI-generated code, a notoriously difficult problem that many security tools can't even see [**Q13**].

Organizations that use AI frequently or constantly have very high confidence in their ability to secure it. And many organizations feel that AI has aided them with secure coding. This implies two possibilities (as shown in Figure 4).

- Higher confidence in one's ability to secure AI results in more frequent use of AI assistants
- Frequently benefiting from AI's new security capabilities increases confidence in AppSec as a whole

The security benefits of AI are already being experienced early in the pipeline, with 19.78% of respondents noting that AI is producing faster identification/suggestion of potential security vulnerabilities in code as it's written [**Q12**].

QUESTIONS

Q9. How confident are you that you have the processes and tools in place to address security issues introduced by AI code generators/coding assistants?

Q12. From the list below, what is the PRIMARY security-related benefit your team has experienced from using AI code generators/coding assistants?

Q13. How confident are you in your ability to ensure AI-powered code assistants don't introduce any open source code that's subject to license obligations that put your intellectual property at risk?

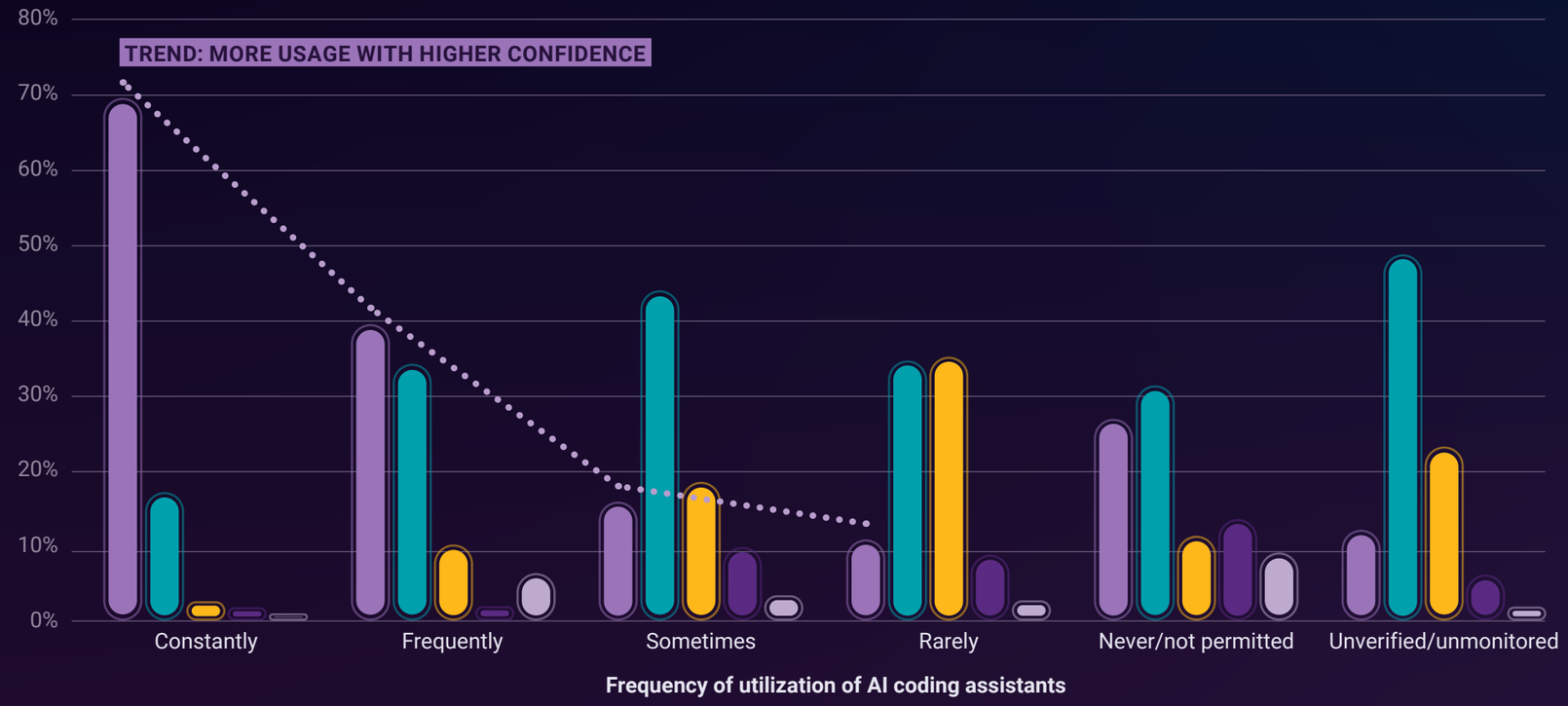


Figure 4. Respondents' level of confidence in the processes and tools in place to address security risks of AI usage in development.

Confidence in the processes and tools in place to address security issues introduced by AI code generators/coding assistants

- Very confident
- Moderately confident
- Slightly confident
- Not at all confident
- Not a priority

RECOMMENDATIONS AND OUTLOOK

Our survey data shows a strong consensus on where DevSecOps professionals need to focus, and it signals a fundamental shift in how to think about application security.

The Mandate for Workflow Integration

We asked our survey participants to name their single most important priority for improving application security in the next 12 months. The answer was loud and clear. The top priority, chosen by 27.27% of all respondents, is "better development workflow integration" [**Q15**].

This response beat out every other answer, including "increasing speed of tests" and "expanding coverage." This means our survey respondents have correctly diagnosed the root cause of their pain. The problem isn't a lack of tools, it's the inefficient and painful way developers are forced to interact with those tools. This is a mandate for a new, developer-centric approach to make security a seamless, invisible, and valuable part of the developer's native workflow.

Actionable Recommendations

For Technical Leaders (CISOs, Directors)

- **Establish a Robust AI Governance Framework:** The AI confidence paradox and the shadow AI problem are clear and present dangers to your organization. You need a formal AI governance framework, and you need it now. You must have clear policies on what's allowed, address the top concerns of data privacy and IP protection, and establish who is accountable for the code AI produces.

QUESTION

Q15. What is your organization's SINGLE MOST important priority for improving its application security testing capabilities in the next 12 months?

- **Rationalize and Optimize the AST Toolchain:** Do a ruthless audit of your current security portfolio to find the redundancies and the main sources of noise. The goal is to consolidate around solutions that integrate into existing, AI-enabled build pipelines and give you (and developers) high-fidelity results faster. This is how you improve the ROI of your security spend and reduce the friction that's killing your development velocity.
- **Invest in the Developer Experience of Security:** Shift from buying standalone security boxes to investing in platforms and tools that are built for deep, native integration into developer workflows. Start measuring success not just with security metrics, but with developer-centric metrics like mean time to remediate.

For Hands-on Practitioners (Developers, DevOps Engineers, Security Architects)

- **Champion Integrated Tooling:** *You're* the one who will need to sprint to the finish line or risk missing a release deadline because of a failed late-stage security test. Be the person who demands security tools that live where you live—in the IDE and the CI/CD pipeline. And ideally, those tools should be developer-focused and *support* security rather than having to be *force-fitted* into dev workflows. Use the data in this report to prove to management that this is what the entire DevOps community is demanding.

27% OF RESPONDENTS SAY THEIR TOP PRIORITY IS "BETTER DEVELOPMENT WORKFLOW INTEGRATION"

THIS IS A MANDATE FOR A NEW, DEVELOPER-CENTRIC APPROACH TO **MAKE SECURITY A SEAMLESS, INVISIBLE, AND VALUABLE PART OF THE DEVELOPER'S NATIVE WORKFLOW**

- **Quantify the Cost of Noise:** *You're* the one suffering from alert fatigue. Start tracking the time your team wastes by triaging useless alerts. Use our data—which found that over 71% of your peers report significant noise [**Q5**]—to show that this is a massive productivity drain on your own organization. A data-backed business case is the best way to get your toolchain rationalized.
- **Lead the Charge on Secure AI Enablement:** *You're* the one on the front lines of AI disruption. Don't let it be defined *only* as a risk. Champion AI's benefits, like faster vulnerability identification and quicker remediation. Show how it can be a security force multiplier, eliminating risks before they get tossed to AppSec teams, lessening their backlog with each commit. Work with your leadership to build the guardrails that will let you and your team innovate safely.

QUESTION

Q5. Approximately what percentage of security test results are noise? For example: duplicative results, false positives, conflicting with other tests/tools.

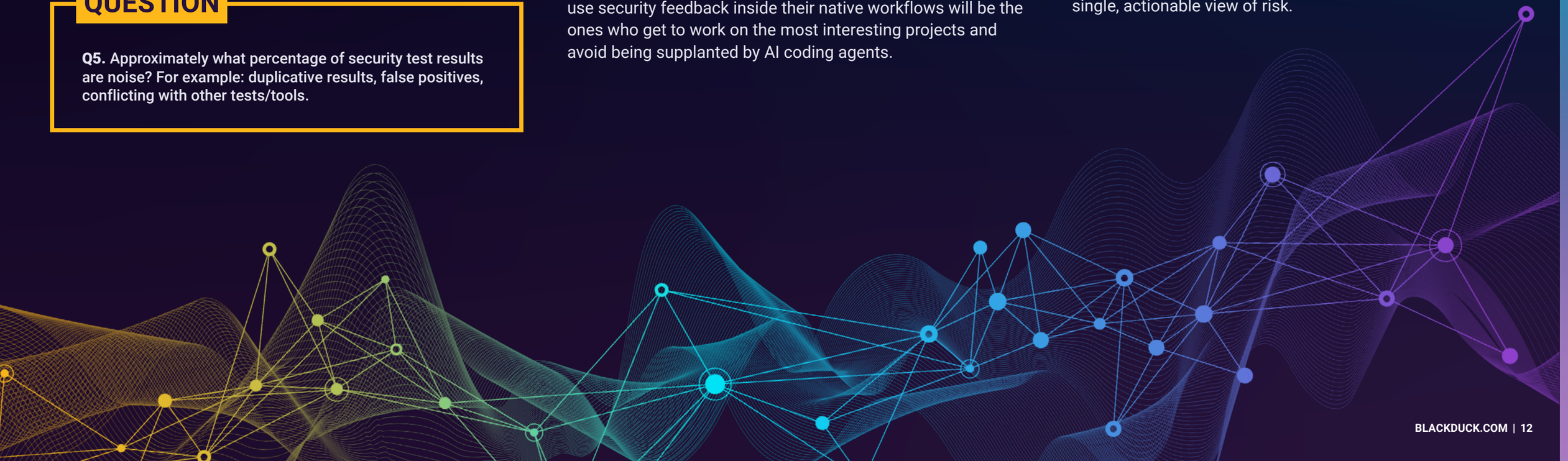
Future Outlook

The trends we've identified are only going to accelerate. Based on this data, here are three predictions for the next 18 to 24 months.

- **The Acceleration of AI Governance Tools:** The market abhors a vacuum. The combination of massive amounts of unsanctioned AI adoption and the risks of IP leakage and security flaws has created a critical need. Expect a flood of new tools and features designed specifically to give you visibility, governance, and security for AI-generated code and the use of open source AI models.
- **A Widening Developer-Centric Skills Gap:** The industry's demand for better workflow integration is going to separate the winners from the losers. Security pros who can think and act like developers—who understand APIs, IDEs, and CI/CD—will be invaluable. And developers who can effectively use security feedback inside their native workflows will be the ones who get to work on the most interesting projects and avoid being supplanted by AI coding agents.

EXPECT A FLOOD OF NEW TOOLS AND FEATURES DESIGNED SPECIFICALLY TO GIVE YOU **VISIBILITY, GOVERNANCE, AND SECURITY FOR AI-GENERATED CODE**

- **The Shift from Tool Acquisition to Toolchain Optimization:** The pain from the toolchain inefficiency cycle is becoming noticeable to the C-suite. We predict a major shift in security budgets. The smart money will move away from buying more standalone point solutions and toward investing in application security posture management platforms that can ingest, deduplicate, and correlate findings from all your tools into a single, actionable view of risk.



HOW BLACK DUCK CAN HELP

The story this report tells is clear and urgent: The old way of doing application security hasn't kept up with the current reality of software development. The security debt from lopsided DevSecOps maturity, the wasted money from toolchain friction, and the unmanaged risks from the AI gold rush are interconnected, systemic business risks that threaten your IP, create massive legal exposure, and slow down your ability to innovate.

To navigate this new world, you must make a strategic shift. The old approach of buying another disconnected point solution every time a new threat emerges has created a cycle of inefficiency and noise. Moreover, each new tool represents an additional point of failure, potentially breaking pipelines and causing inconsistent enforcement of internal standards or external regulations. The best way forward is to move from a reactive, tool-centric model to a proactive, platform-based strategy.

The goal is to turn security from a bottleneck into a strategic enabler of speed and trust. You won't do that by adding more tools. You do it by investing in enterprise-level visibility, comprehensive governance, and intelligent automation that is baked directly into the developer workflow. The companies that can accurately measure their security postures, enforce consistent policies across their entire toolchain, and safely govern the use of game-changing technologies like AI will be the companies that will win.

Black Duck provides the versatile, platform-based approach you need to manage risk at the speed and scale that modern, AI-enabled development demands. Here's how our integrated solutions directly solve the core problems identified in this report, helping you turn insight into action, and systemic risk into a durable competitive advantage.

From Security Debt to Security Assurance

The central theme of this report is that DevSecOps maturity is dangerously uneven in many organizations. They may have mastered "Dev" and "Ops" to get the speed they need, but their "Sec" is stuck in the past, full of manual processes and huge coverage gaps. What remains is a fragile system where development speed outpaces security, leading to a massive, often invisible "security debt."

Security debt—unaudited components and untested code—gets bigger with every release. The data shows the scale of the problem: a staggering 45.56% of organizations still rely on manual processes just to get new code into the security queue [**Q2**]. The result is a critical failure in test coverage, with over 61% of organizations testing 60% or less of their applications [**Q3**].

The solution isn't to test more. In their current state, most AppSec initiatives will just accrue longer, noisier backlogs by implementing more tests. Integrating risk detection, prioritization, and remediation mechanisms throughout the pipeline is the only way to fix the underlying visibility and governance deficit.

Black Duck solutions are engineered to solve this exact problem. We provide the industry's most comprehensive visibility into your software—whether written by your developers, generated by AI, or supplied by third-party vendors—and we deliver the informed, automated governance you need to manage security at the scale and speed AI-enabled pipelines require.

Unifying the AST Landscape with a Single Platform

The first step to breaking the inefficiency cycle is to attack the root cause: tool sprawl. Black Duck Polaris™ Platform is the integrated, cloud-based solution designed for exactly this. Polaris brings together Black Duck's market-leading SAST, SCA, and DAST engines into a single, easy-to-use SaaS solution. This unified platform gives you a central point of control for your security policies and a single, correlated view of risk across your entire application portfolio. Polaris lets you prioritize true risks and accelerate the feedback loop for faster remediation so you can make the shift from tool acquisition to toolchain optimization. Polaris is architected for versatility, speed, and scale—purposefully tuned for AI code generators.

QUESTIONS

Q2. What is the PRIMARY mechanism used to ensure new code sources (e.g., projects, repositories, significant feature branches) are included in your organization's application security testing program?

Q3. Approximately what percentage of your projects, branches, and repositories are being included in your application security testing queue?

Embedding Security into the Developer's Native Workflow

The ultimate remedy for friction is to make security an invisible, nondisruptive part of the developer's natural workflow. Black Duck's developer-centric strategy is underpinned by two powerful mechanisms.

- **Code Sight™ IDE Plug-in:** Code Sight makes security the natural end point of developers' daily tasks. It puts the power of our SAST and SCA engines directly in the developer's favorite IDE, alongside clear, prioritized risk insight and fix guidance. As developers write code, Code Sight gives them real-time, in-line feedback, flagging vulnerable components, security weaknesses, and quality defects on the spot. Code Sight helps prevent vulnerabilities from ever being checked in, which is the most cost-efficient and fastest way to address them. It eliminates the productivity-killing need to switch between separate tools and dashboards, and it creates a continuous learning loop, making your developers better at secure coding every day.
- **DevOps Integrations and SCM Automation Templates:** Black Duck provides deep integrations across the entire CI/CD toolchain, including popular platforms like GitHub, GitLab, Azure DevOps, Jenkins, and Bitbucket. Development teams can easily configure, test, and fix workflows optimized for the project and repository. Your AppSec teams can rest assured that risk tolerance policies will be automatically enforced, establishing security gates only when necessary and keeping pipelines flowing.

Leveraging AI as a Security Force Multiplier

Managing AI risk is critical, and 63.33% of professionals believe that AI is already helping them write more-secure code [**Q11**]. But because not every developer is a security expert, organizations must establish ways to cultivate security-capable developers without adding burdensome exercises or extraneous tools to their workload.

Black Duck is leading this charge with **Black Duck Assist™**, an AI-powered security assistant that works directly inside the developer's IDE via Code Sight. It uses a large language model supercharged with decades of our own security insights to give developers

- **Easy-to-Understand Issue Summaries:** Clear, simple explanations of complex vulnerabilities and rationales for their priority and remediation.
- **Actionable Code Fix Recommendations:** Concrete, context-aware, AI-generated suggestions on how to fix the problem, including lines of code ready to be cut and pasted into the project, so developers can move quickly and avoid late-stage refactoring.

By providing intelligent help right where the code is being written, Black Duck Assist not only helps developers fix issues faster but also acts as a continuous, on-the-job training tool. It turns AI from a source of risk into a powerful partner in building secure software.

QUESTION

Q11. To what extent do you agree or disagree with the following statement: "The use of AI coding assistants has tangibly improved our ability to write more-secure code and more effectively address security risks within our DevSecOps life cycle (shift left)."

Protecting Valuable Intellectual Property from Shadow AI

The shadow AI problem is especially tricky because many AI assistants inject small snippets of code that traditional SCA tools, which just look at declared dependencies, can completely miss.

Black Duck® SCA **snippet analysis** is uniquely built to solve this challenge. It can identify these small code fragments, match them to their original open source projects, and expose any associated license obligations that may put valuable IP at risk. Our massive KnowledgeBase™, sourced and curated by our in-house Cybersecurity Research Center (CyRC), catalogues more than 8.7 million open source components from over 57,700 forges and repositories.

Black Duck SCA snippet analysis can be activated via API and triggered with each commit. This allows snippet analysis to scale alongside AI coding assistants and eliminates the delays of batched analysis.

Ultimately, snippet analysis can help you ensure that revenue-generating projects or critical release branches don't include licensed open source components that could eventually result in asset write-downs or costly legal issues.

Tracking Open Source AI Models in Critical Projects

The pressure to stay competitive is likely driving your development teams to integrate AI models into your business applications. However, building and training these models in-house requires significant resources and expertise. As a result, many organizations are turning to open source AI models as a practical solution.

Black Duck SCA detects and manages risks in open source AI models within your projects, enabling you to govern the use of AI models within your organization as well as include information about those models in your Software Bills of Materials (SBOMs).

By cataloguing each associated AI model card, Black Duck SCA provides developers with the necessary information and insight to make informed choices about the AI models they use in their applications, enabling them to

- Assess the suitability of a particular AI model for their specific use case
- Compare different models and choose the most appropriate one
- Identify potential issues or biases in the models and take corrective action
- Ensure compliance with organizational policies and regulatory requirements

A Tailored Approach for Your Role

The challenges and solutions in this report mean different things to different people. For the solutions we've outlined to work, their value must be clear to every stakeholder. Black Duck solutions for AI-enabled pipelines are designed to deliver specific, measurable value, both to the executives setting the strategy and to the practitioners in the trenches.

For Executive Leaders: Transforming Systemic Risk into Competitive Advantage

Black Duck gives you the enterprise-level visibility and governance you need to turn the systemic risks identified in this report into a durable competitive advantage. We help you protect your valuable IP from AI-introduced license risks, ensure that you can meet the market's demand for accurate SBOMs, and provide a single source of truth for your management and engineering teams. This allows you to innovate faster, with a clear and manageable risk posture.

For Hands-on Practitioners: Building Secure Software Without Sacrificing Speed

For developers, DevOps engineers, and security architects, Black Duck is about eliminating friction. Our tools are designed to live where you live—in the IDE and the CI/CD pipeline—providing fast, accurate, and actionable feedback without forcing you to switch contexts. We help you cut through the noise of false positives, automate the enforcement of complex compliance rules, and give you the guardrails you need to safely use the power of AI. This lets you focus on what you do best: build and ship great software.

INNOVATE FASTER, WITH A **CLEAR AND MANAGEABLE RISK POSTURE**

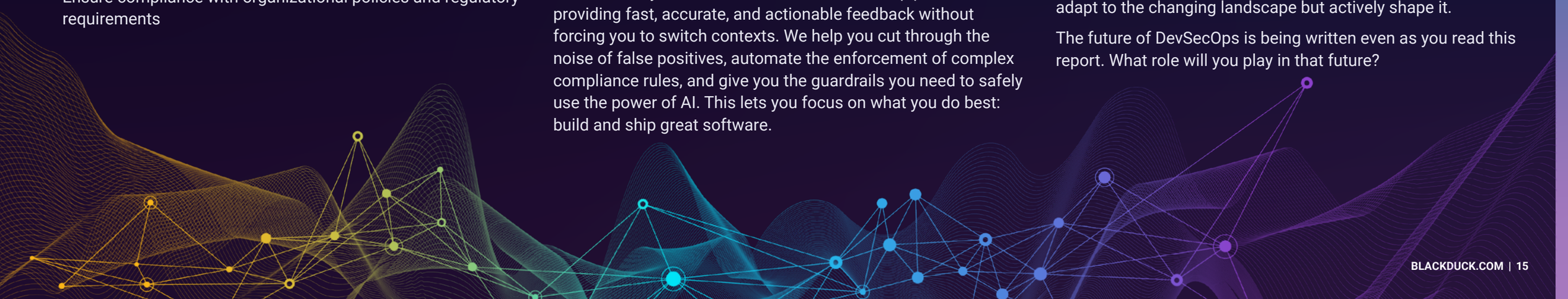
Conclusion: Building Trust in Your Software

The data is clear: DevSecOps is at a crossroads. The industry has achieved speed, but at the cost of mounting security debt. Companies bought more tools, but they created more friction. And now, AI is forcing a re-evaluation of everything.

The path forward isn't about more tools or more processes. It's about a fundamental shift to a developer-centric model of secure software development. It's about integration, automation, and insightful feedback that makes security the natural result of existing development efforts. It's about turning security from a roadblock into a strategic enabler for your business.

The companies that thrive in this new era will be the ones that can effectively balance rigorous security with the relentless demand for innovation. They will be the ones who don't just adapt to the changing landscape but actively shape it.

The future of DevSecOps is being written even as you read this report. What role will you play in that future?



APPENDIX A: FULL SURVEY QUESTIONS AND RESPONSES

1. On average, how often does your primary team/organization deploy code changes to production for your primary or business-critical application(s)?

Multiple times per day	27.77%
Daily	31.97%
Weekly	27.57%
Biweekly/monthly	11.39%
Quarterly or less frequently	1.30%

2. What is the PRIMARY mechanism used to ensure new code sources (e.g., projects, repositories, significant feature branches) are included in your organization's application security testing program?

Fully manual	20.88%
Primarily manual	24.68%
Balanced mix	21.88%
Primarily automated	18.18%
Fully automated	13.99%
I am not familiar with this process	0.40%

3. Approximately what percentage of your projects, branches, and repositories are being included in your application security testing queue?

Up to 20%	0.80%
21% - 40%	20.38%
41% - 60%	40.46%
61% - 80%	27.07%
81% - 100%	10.49%
I do not have enough visibility to approximate test coverage	0.80%

4. Which types of application security testing (AST) tools does your organization currently utilize? (Select all that apply)

Software composition analysis (SCA)/open source security	34.17%
Dynamic application security testing (DAST)	32.77%
API security testing	31.87%
Infrastructure-as-code (IaC) security scanning	30.47%
Static application security testing (SAST)	30.27%
Manual penetration testing/security reviews	30.17%
Interactive application security testing (IAST)	29.47%
Application security posture management (ASPM)	29.17%
Runtime application self-protection (RASP)	27.27%
Container security scanning	23.58%

5. Approximately what percentage of security test results are noise? For example: duplicative results, false positives, conflicting with other tests/tools.

Up to 20%	4.60%
21% - 40%	37.16%
41% - 60%	34.47%
61% - 80%	16.78%
81% - 100%	6.19%
I do not have enough visibility into all tests and results to identify noise	0.80%

6. Which statement best describes the relationship between application security testing and software development/delivery?

Application security testing severely slows down development/delivery	19.48%
Application security testing moderately slows down development/delivery	32.77%
Application security testing slightly slows down development/delivery	28.97%
Application security testing does not slow down development/delivery	17.58%
I do not have enough visibility to assess the relationship accurately	1.20%

7. On average, how frequently do you or your team utilize AI coding assistants (e.g., GitHub Copilot, Amazon Q, Gemini Code Assist, Tabnine) during software development?

Never/not permitted	4.40%
Unverified/unmonitored usage without permission	10.69%
Rarely (e.g., less than once a month, experimental use only)	18.08%
Sometimes (e.g., few times a month, for specific tasks)	23.18%
Frequently (e.g., multiple times a week, integrated into some workflows)	31.07%
Constantly (e.g., daily, widely adopted and integrated into standard workflows)	12.59%

8. To what extent do you agree or disagree with the following statement: "The use of AI coding assistants has introduced new security risks into, or made it harder to detect issues within, our codebase."

Strongly disagree	6.69%
Disagree	16.88%
Neutral	19.38%
Agree	34.97%
Strongly agree	21.58%
Not have enough visibility/too early to know	0.50%

9. How confident are you that you have the processes and tools in place to address security issues introduced by AI code generators/coding assistants?

Very confident	30.77%
Moderately confident	39.36%
Slightly confident	18.68%
Not at all confident	6.19%
This is not a priority at this time, as using AI-generated code is against company policies	4.10%
I do not have enough visibility into our processes to manage and secure AI-generated code	0.90%

10. What is your organization's PRIMARY security-related concern regarding the implications of using AI code generators/coding assistants?

Potential for AI to introduce security vulnerabilities at a scale and speeds that exceed AppSec	16.28%
Challenges ensuring AI-generated code meets specific compliance/regulatory requirements	14.99%
Risk of sensitive data or intellectual property leakage through prompts or model training	12.99%
We have no significant security concerns regarding AI coding assistants	12.49%
Over-reliance on AI leading to erosion of developers' secure coding skills	11.59%
Inaccuracy or unreliability of AI-generated security fixes or suggestions	11.09%
Difficulty integrating AI security findings with existing security toolchain (SAST, SCA, etc.)	10.79%
Lack of transparency/explainability in how AI arrives at security recommendations	9.79%

11. To what extent do you agree or disagree with the following statement: "The use of AI coding assistants has tangibly improved our ability to write more-secure code and more effectively address security risks within our DevSecOps life cycle (shift left)."

Strongly disagree	5.99%
Disagree	11.19%
Neutral	18.88%
Agree	38.26%
Strongly agree	25.07%
Not have enough visibility/too early to know	0.60%

12. From the list below, what is the PRIMARY security-related benefit your team has experienced from using AI code generators/coding assistants?

Faster identification/suggestion of potential security vulnerabilities in code as it's written	19.78%
Improved consistency in applying secure coding standards across the team	17.88%
Quicker remediation of identified vulnerabilities through AI-suggested fixes	16.48%
Automation of repetitive security documentation or compliance tasks	16.38%
Help understanding and refactoring complex code to reduce security risks	15.38%
Assistance in generating more comprehensive security test cases (e.g., unit, integration)	13.19%
We have not yet experienced significant security benefits	0.90%

13. How confident are you in your ability to ensure AI-powered code assistants don't introduce any open source code that's subject to license obligations that put your intellectual property at risk?

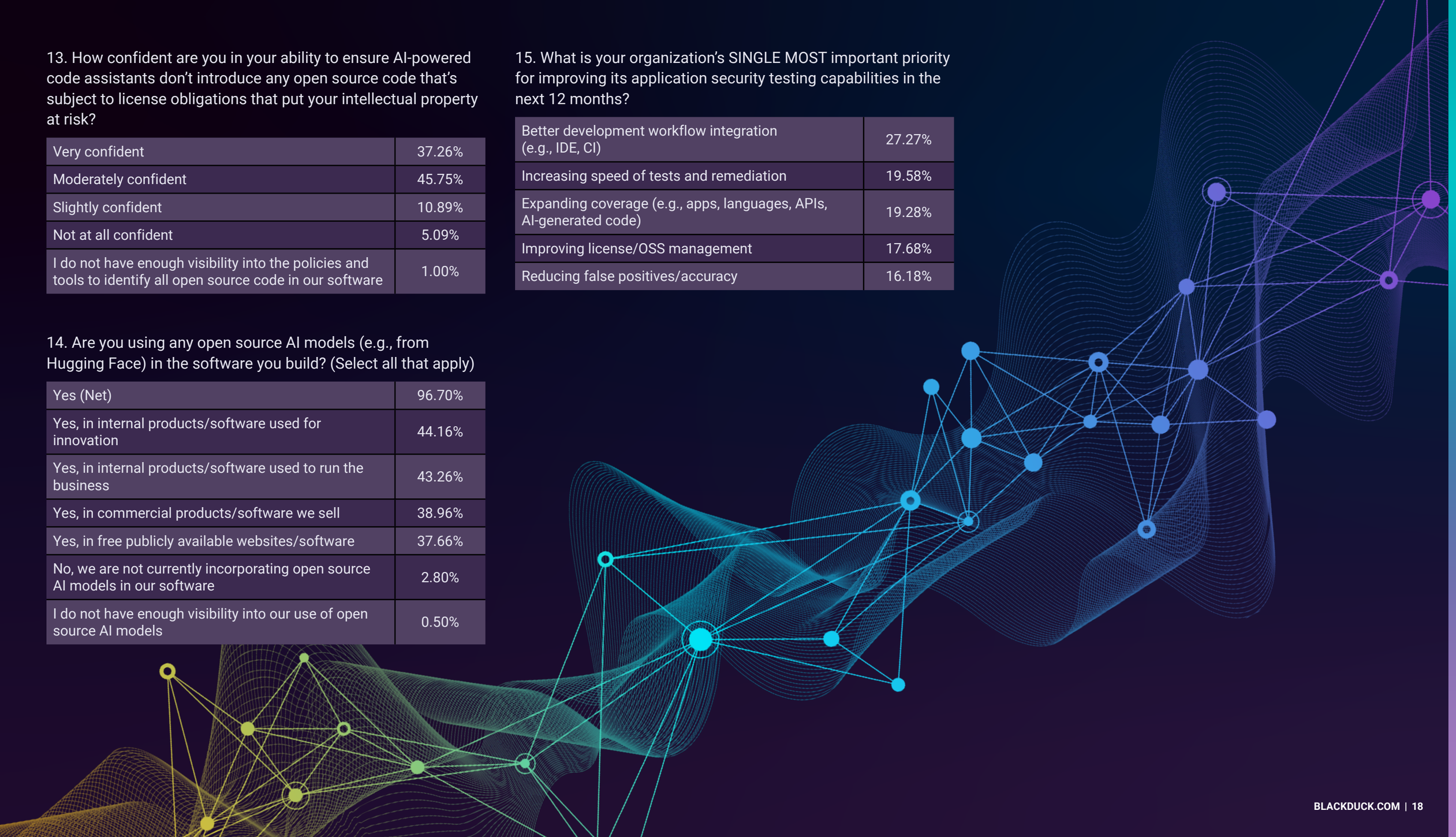
Very confident	37.26%
Moderately confident	45.75%
Slightly confident	10.89%
Not at all confident	5.09%
I do not have enough visibility into the policies and tools to identify all open source code in our software	1.00%

14. Are you using any open source AI models (e.g., from Hugging Face) in the software you build? (Select all that apply)

Yes (Net)	96.70%
Yes, in internal products/software used for innovation	44.16%
Yes, in internal products/software used to run the business	43.26%
Yes, in commercial products/software we sell	38.96%
Yes, in free publicly available websites/software	37.66%
No, we are not currently incorporating open source AI models in our software	2.80%
I do not have enough visibility into our use of open source AI models	0.50%

15. What is your organization's SINGLE MOST important priority for improving its application security testing capabilities in the next 12 months?

Better development workflow integration (e.g., IDE, CI)	27.27%
Increasing speed of tests and remediation	19.58%
Expanding coverage (e.g., apps, languages, APIs, AI-generated code)	19.28%
Improving license/OSS management	17.68%
Reducing false positives/accuracy	16.18%



APPENDIX B: DETAILED RESPONDENT DEMOGRAPHICS

The analysis in this report is based on a comprehensive survey of 1,001 global software and security professionals. The respondent base was designed to provide a holistic and statistically relevant view of the DevSecOps ecosystem, capturing perspectives from a wide range of geographies, organizational roles, company sizes, and industry verticals.

Respondents by Geography

United Kingdom	125	France	125
United States	126	Germany	125
Singapore	125	Japan	125
Finland	125	China	125

Respondents by Company Size (Number of Employees)

Fewer than 100	14	5,001-10,000	180
100-500	75	10,001-15,000	91
501-1,000	121	15,001-50,000	54
1,001-2,000	190	50,001-100,000	56
2,001-5,000	219	More than 100,000	1

Respondents by Job Title

Application Security Architect	132	Operations Engineer	9
Application Security Manager	48	Product Security	2
CISO	237	Product Security, AppSec	1
Developer	56	Programmer	23
DevOps Engineer	149	QA/Tester/Test Manager	4
Director, Application Security	23	Release Engineer/Manager	4
Director, Cybersecurity	47	Security Architect	5
Director, IT Risk Management	66	Security Director	6
Director, IT Shared Services	50	Security Engineering Manager	7
Director, Product Security	15	Senior Director, Product Security	11
Director, Security Assurance	5	SVP, Product Security	13
Executive Director, Product	9	Technical Lead	12
Incident and Security Manager	3	VP, Product and Application	9
Information Assurance Director	8	VP, Security Architecture	16
Manager, Software Security	23	VP, Security Compliance	5

Respondents by Industry Sector

Automotive	34	MedTech	15
Technology	401	Healthcare	26
Cybersecurity	110	Retail	35
Application/Software	96	Media	22
Manufacturing	50	Government	27
FinTech	14	Transportation	29
Banking, Financial, Insurance	56	Utilities	22
Telecommunication/ISP	42		

BLACK DUCK®

Black Duck® meets the board-level risks of modern software with True Scale Application Security, ensuring uncompromised trust in software for the regulated, AI-powered world. Only Black Duck solutions free organizations from tradeoffs between speed, accuracy, and compliance at scale while eliminating security, regulatory, and licensing risks. Whether in the cloud or on premises, Black Duck is the only choice for securing mission-critical software everywhere code happens. With Black Duck, security leaders can make smarter decisions and unleash business innovation with confidence. Learn more at www.blackduck.com.