ANALYST BRIEF
Sponsored by: GrammaTech

DevOps processes and third-party components are increasingly being used to create safety-critical products, which means that organizations need modern DevSecOps security solutions to protect the code they write and the components they embed.

Application Code Security for Safety-Critical Products and Applications

October 2022

Written by: Jim Mercer, Research Vice President, DevOps and DevSecOps, and Melinda-Carol Ballou, Research Director, Agile ALM, Quality and Portfolio Strategies

Introduction

Software is embedded in the devices we rely on every day. Businesses and individuals worldwide use software in a range of products (cars, thermostats, medical devices) and for multiple purposes (credit card transactions). As the software content in products has increased, it has created an insatiable demand for new features, upgrades, and connectivity, making the security of these products paramount.

Composite Software Development

Though most people don't realize it, much of the technology we rely on every day is built using a supply chain of third-party commercial components and open source software (OSS). Using existing software components enables modern software development teams to create new software capabilities in a composite manner using a combination of natively developed code and already built software modules. A recent IDC survey revealed that the use of open source has gone mainstream, with 89% of respondents indicating that they are currently using or planning to use open source.

AT A GLANCE

KEY TAKEAWAYS

- Modern critical safety solutions include increasing numbers of embedded third-party components.
- The number of software vulnerabilities and bad actors continues to climb year over year.
- Safety-critical systems are particularly attractive to bad actors due to their essential role.
- BSCA and SAST security solutions are essential for protecting safety-critical products from vulnerabilities and licensing issues.

This composite approach is beneficial because it enables development teams to deliver more features and functionality faster. Still, it also leaves the software, the devices the software runs on, and the users of those devices susceptible to bad actors or a catastrophic breach due to unknown provenance.

Product development organizations must be diligent about securing their software supply chains to avoid susceptibility to a software breach. Given the risk to the business or their customers' business, product development organizations must prioritize software security and product safety while delivering software with increasing velocity.

The recent Apache Log4j crisis provided a stark reminder that virtually every organization is susceptible to the next big vulnerability. The Log4Shell vulnerability (CVE-2021-44228) also brought to light that even commercial software components that are expected to be secure can introduce unexpected risk. Even as companies rely on third-party commercial software, including OSS, they are ultimately responsible for the quality and security of the software they produce.

So, product development organizations must architect good up-front practices that incorporate software assurance methodologies early in the software development life cycle (SDLC) to ensure the hygiene of the code they write as well as the code they integrate into their software solutions.

Executive Order 14028

Also, while your organization may not be a U.S. government agency or do business with the government, it is only a matter of time before you are affected by Executive Order 14028. This unprecedented government action is a mandate to protect against software supply chain and infrastructure attacks. It emphasizes that the prevention, detection, assessment, and remediation of cyberincidents is a top priority and essential to national and economic security.

As more organizations adopt these standards, they will expect the same due diligence from their business partners or vendors. Software composition is increasingly becoming a part of compliance regulation language in standards such as PCI as well as products for healthcare (FDA), automotive (NHTSA), energy (NERC), and consumers (FTC).

Civil Penalties

On January 4, 2022, the FTC warned companies and their vendors to take reasonable steps to remediate OSS vulnerabilities, such as Log4Shell. In its warning, the FTC stated that it intends to use its full legal authority to pursue companies that fail to take reasonable steps to protect consumer data from exposure due to future Log4j exploits or similar known vulnerabilities. There are already precedents for this: Equifax paid $575 million for a data breach caused by an unpatched vulnerability in the OSS Apache Struts, and the board of SolarWinds is currently being sued for breach of fiduciary duty following the high-profile Orion cyberattack.

DevSecOps Automation

Because the number of developers is significantly higher than the number of security team members, it is challenging for organizations to find the resources to review the security integrity of even a fraction of their applications. Furthermore, developers already have a lot being added to their plates, and if additional demands are made on their time, important security improvements simply will not get done.

The need for security automation has driven the adoption of DevSecOps, which enables DevOps teams to act as key stakeholders in defining and implementing security policies. IDC defines DevSecOps as a methodology that asserts that security needs to be prioritized at the beginning of the DevOps delivery pipeline and continuously throughout the SDLC.

DevSecOps encompasses application code checking that looks at the code you write natively and the code from others you integrate into your application (i.e., OSS and third-party commercial software). So, to effectively secure your application code, you need to evaluate source files in a code repository (e.g., Git) and the fully built binaries from third parties. Further, third-party software comes with licenses that need to be evaluated to avoid possible conflicts of license usage that could result in litigation.

Safety-Critical Products

While all the benefits of DevSecOps and automated code reviews apply to every enterprise, they are essential for organizations developing safety-critical and life-critical products. Failure is not an option when it comes to many of these applications, and the safety and quality of the software are more important than the testing speed. Examples of safety-critical products include medical devices and automobiles. For these systems, the stakes are high, and insecure software code leading to the exploitation of a vulnerability could result in loss of life, permanent disability or major injury, loss of a system, and/or significant equipment damage. Logistically, it is not always easy to patch these systems once they have been deployed into the field.

Therefore, the standards and capabilities for DevSecOps solutions used with safety-critical products need to be extremely strong to ensure the utmost protection of these products and those who depend upon them. These products require modern DevSecOps tools that can examine the code you write and the external third-party components embedded in your product with the highest degree of accuracy. However, DevSecOps includes more than just application security tools.

While the right security solutions lay the foundation, process and cultural changes are also needed to bring together software development, quality, and security teams to ensure they collectively prioritize the right solutions for the business.

Benefits

To reap the benefits of DevSecOps for safety-critical products, you need to leverage both static application security testing (SAST) and binary software composition analysis (BSCA) security technologies across the SDLC to ensure the sanctity of your security-critical product code. And given that many of these safety-critical systems include software embedded in other products, the costs to update, patch, or replace any at-risk software obligate producers to emphasize comprehensive testing of all components during development to identify and remediate vulnerabilities prior to each production release. Both solutions are vital because the impact of a vulnerability in safety-critical products can have an extensive blast radius that extends to your customers and often your customers' customers (and regulatory agencies as well).

SAST

SAST solutions provide a white-box method of testing while inspecting your source code to identify areas of potential threats or security exposures within the code. Because SAST is usually used before the software is built, it can prevent vulnerabilities from being introduced during the software development process. Another benefit of SAST is it can provide developers with real-time feedback as they code (e.g., introduce security warnings into the engineer's integrated development environment [IDE], helping them fix issues before they pass the code to the next phase of the SDLC). Because SAST provides developers with fast feedback on their code, it implicitly provides training on secure coding practices. So, SAST can help ensure you catch insecure code even if your developers do not have secure coding skills.

An essential benefit of using SAST is its ability to analyze 100% of the source code base exponentially faster and more accurately than manual secure code reviews performed by humans. You can even use it if your developers use a no-code, low-code development platform.

BSCA

The increased adoption of OSS and third-party commercial binaries in safety-critical products necessitates the use of a BSCA solution to identify any vulnerabilities that may be present in the inherited components.

A fair number of different vendors offer SCA solutions, but not as many provide BSCA capabilities. Conventional SCA solutions identify potential risks and vulnerabilities of commercial OSS dependencies integrated into an application. However, most of these solutions are programming language and framework dependent and are not capable of scanning component binaries.

The distinction between BSCA and SCA became apparent during the December 2021 Log4j vulnerability event (*The Log4j Vulnerability: Widespread Impact and How DevOps Teams Can Respond*, IDC # lcUS48600422, December 2021) when organizations scrambled to try and find the Log4j vulnerability in third-party commercial software. Organizations using a BSCA solution could scan the commercial binaries and quickly identify their exposure to the vulnerability.

BSCA examines your built application binaries across many different computing platforms, such as desktop, server, mobile, and embedded, including binary files from third-party commercial providers, to identify the first-party and transitive dependencies creating a software bill of materials (SBOM). An SBOM is a dependency graph showing the transitive artifacts in your application. This information is essential because the OSS and commercial software you consume also use third-party code to build their components, so you inherit vulnerabilities several layers deep into the SBOM dependency graph.

The standards and capabilities for DevSecOps solutions used with safety-critical products need to be extremely strong to ensure the utmost protection of these products and those who depend upon them.

A BSCA solution enables organizations to locate and remove the manual effort around software licenses included as part of their safety-critical products. Understanding the software licenses will help determine your risk and whether the embedded licenses are permissive, public domain, proprietary, and so forth. Legal and compliance teams can use this information to mitigate the risk of litigation resulting from improper software license use.

Better Together

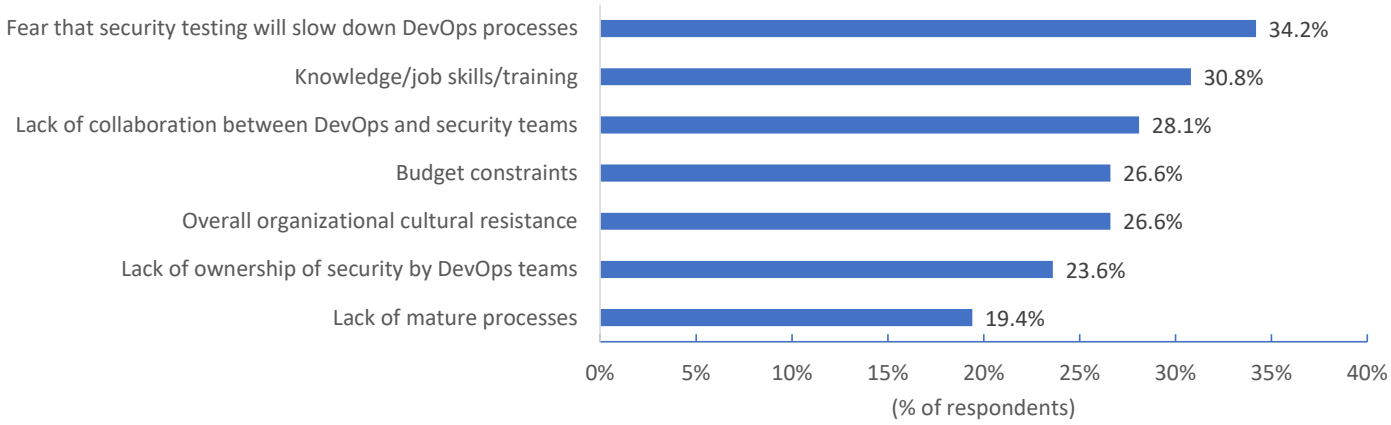
Individually, SAST and BSCA each provide important protections for safety-critical software solutions. However, combining these two technologies protects both the code you write and the code you embed into your product, helping create a holistic defense of all the code within your application.

Considerations

While there are several capabilities for developers to consider when researching SAST and BSCA solutions, an often overlooked feature is whether the solutions are developer friendly. Figure 1 shows that developers are most concerned that security tools will slow them down by adding friction to the process.

FIGURE 1: ***Developers Need DevSecOps Tools That Move at the Pace of DevOps***

Q What are your top 2 organizational challenges concerning DevSecOps adoption?



n = 263

Source: IDC's DevSecOps Adoption, Techniques, and Tools Survey, April 2022

SAST Considerations

- The SAST solution not only should be developer friendly but also should provide:
  - Code security where the developers are doing their work continuously (agile "shift left")
  - Integrations with IDEs and DevOps orchestration tools
  - Findings that include in-context code information for faster resolution
  - The ability to move at the pace of DevOps without adding unnecessary friction to developer workflows
- How do you need to consume the solution?
  - Is your source code scanned on premises to ensure maximum protection, or will it be sent to the security vendor's cloud/SaaS locations for processing?
- Can the solution manage large code bases? Many safety-critical solutions have large code bases, so the SAST solution needs to be able to scale appropriately.
- Does the solution support multiple languages, and does it support the languages you use?
- Does the solution provide access to developer training, especially in-context training? As security vulnerabilities are found in developers' code, do developers have a resource to understand why the code is vulnerable and how they could fix it?

BSCA Considerations

- To be a true BSCA solution, it must be able to independently scan binaries without the need to access the source code or build files.
- Vulnerability analysis should run deep. The solution reference should go beyond the National Vulnerability Database (NVD) and include enriched vulnerability data from multiple sources of independent research.
- Support for emerging SBOM machine-readable standards such as CycloneDX or SPDX should be available.
- The solution should support a framework such as OpenChain, an international standard (ISO/IEC 5230:2020) for open source license compliance, which helps ensure that your program yields a trusted and consistent outcome.

Other Code Security Considerations

- Consider the criticality of your application — prioritize imperatives and efficiency.
  - Don't compromise on accuracy for speed.
  - Consider all the embedded systems and devices that make up your safety-critical product.
  - Design and architect your applications based on historical SAST, BSCA, and other security vulnerability data for greater resilience.
- Ensure the security of air-gapped systems.
  - Air-gapped systems can be hacked in several ways, including thumb drives (i.e., Stuxnet), rogue employees/contractors, or even an adjacent system.
- Comply with industry standards and frameworks.
  - Motor Industry Software Reliability Association (MISRA), Automotive Open System Architecture (AUTOSAR), electronic protected health information (ePHI), and Payment Card Industry (PCI) standards should be supported.

Key Trends

Several trends are reinforcing the need for the comprehensive security of safety-critical products, including:

- The number of vulnerabilities continues to rise.
  - The explosion of new applications across multimodal platforms has increased the overall attack surface, making applications a soft target for malicious actors.
  - IDC estimates the number of applications worldwide will grow from 195 million in 2021 to 750 million in 2025 (*750 Million New Logical Applications: More Background*, IDC #US48441921, December 2021). This explosion of new applications will continue to increase the overall attack surface, making software applications a soft target for malicious actors.
  - According to a recent vulnerability report by Risk Based Security (now Flashpoint), the company that produces the popular VulnDB vulnerability database, 28,695 vulnerabilities were disclosed in 2021, the highest number ever recorded.

- The cost of a security breach continues to climb.
  - IBM Security's *Cost of a Data Breach Report* projected that the average price of a data breach has reached an all-time high of $4.35 million (up 13% over two years).
- Emerging regulations, including Executive Order 14028, will continue to pressure the enterprise. The follow-on Cyber Incident Reporting for Critical Infrastructure Act of 2022 (CIRCIA) should motivate organizations to secure safety-critical products.
  - Civil penalties will become a reality. On January 4, 2022, the FTC warned companies and their vendors to take reasonable steps to remediate OSS vulnerabilities, such as the Log4Shell vulnerability (CVE-2021-44228).
- Geopolitical tensions, most notably the Russia-Ukraine War, pose increasing risks to enterprises. IDC anticipates increased activity from rogue nation-states or activist hacker groups that attack wherever they can find a weakness.

Conclusion

The number of new software products continues to rise such that now software is embedded in virtually all the devices and systems that we rely on every day. Bad actors have noticed, and software applications have become a preferred attack surface. The increased risk of an application breach drives the adoption of DevSecOps methodologies to protect application code. Two critical DevSecOps technologies for securing application code are SAST to protect the code you write and BSCA solutions to protect the third-party code and binaries included with your application.

Code security technologies should provide high levels of accuracy and be integrated into DevOps workflows for developer acceptance. Further, keep in mind that enterprise IT teams can benefit from security solutions that meet the rigorous requirements for safety-critical products to address the imperative to create resilience for life-critical systems and products.

About the Analysts

Jim Mercer, *Research Vice President, DevOps and DevSecOps*

Jim Mercer is a Research Vice President within IDC's DevOps and DevSecOps Solutions research practices. In this role, he is responsible for researching, writing, and advising clients on the fast-evolving DevOps and DevSecOps markets. Mr. Mercer's core research includes topics such as rapid enterprise application development, modern microservice-based packaging, GitOps, application security, software supply chain security, and automated deployment and life-cycle/management strategies as applied to a DevOps practice.

Melinda-Carol Ballou, *Research Director, Agile ALM, Quality and Portfolio Strategies*

Melinda-Carol Ballou serves as Research Director for IDC's Application Life-Cycle Management (ALM) program. In this role, Ms. Ballou provides thought leadership, expert opinion, and analysis through comprehensive research on end-to-end application life-cycle management — from requirements to quality, testing, change, continuous release, process, project, and portfolio management (PPM) with a focus on agile DevOps software life-cycle strategies.

MESSAGE FROM THE SPONSOR

About GrammaTech

GrammaTech is a leading global provider of application security testing (AST) solutions used by the world's most security conscious organizations to detect, measure, analyze and resolve vulnerabilities for software they develop or use. The company is also a trusted cybersecurity and artificial intelligence research partner for the nation's civil, defense, and intelligence agencies. GrammaTech has corporate headquarters in Bethesda MD, a Research and Development Center in Ithaca NY, and publishes Shift Left Academy, an educational resource for software developers.

Visit us at https://www.grammatech.com/, and follow us on LinkedIn and Twitter.

IDC Custom Solutions

The content in this paper was adapted from existing IDC research published on www.idc.com.

IDC Research, Inc.
140 Kendrick Street
Building B
Needham, MA 02494, USA
T 508.872.8200
F 508.935.4015
Twitter @IDC
idc-insights-community.com
www.idc.com