EBOOK

Contrast
SECURITY

Bringing an End to Security Roadblocks

How Development Teams Can Push Code Continuously While Prioritizing Security as a Quality Metric

> The average number of serious vulnerabilities per application is 26.7—
> The same as in 2000.[2]

Executive Overview

Thanks to approaches like Agile and DevOps, software developers have become exponentially more productive in recent years. But application security (AppSec) still requires many manual processes and has not kept up, in either efficiency or effectiveness. When the speed of security lags the speed of development, it is no wonder that the average number of serious vulnerabilities per application has not decreased for two decades.[1] From a developer's perspective, legacy tools create delays at every turn, creating coding bottlenecks during scans and forcing developers to do extensive manual work to answer questionnaires, triage false positives, and identify and remediate vulnerabilities.

Fortunately, there is a better approach that solves the delays to coding caused by traditional AppSec tools and processes. Security instrumentation builds security monitoring and response into an application itself, continuously providing insight that developers can actually use to quickly address problems. Tools like interactive application security testing (IAST), next-generation open-source security (OSS), and runtime application self-protection (RASP) use instrumentation to provide continuous protection throughout the software development life cycle (SDLC).

Using instrumentation as the basis for an AppSec strategy eliminates the inefficiencies that create roadblocks for the development teams—repeated security scans, high false positives, and dealing with non-risky open-source vulnerabilities. It also virtually eliminates false negatives, which can result in huge delays to future projects while remediation is performed on applications after they are released in production. As a result, development and security teams can become true partners in ensuring the delivery of safe, innovative applications with aggressive timelines.

[1] "2019 Data Breach Investigations Report," Verizon, April 2019.

[2] "2019 Data Breach Investigations Report," Verizon, April 2019.

Table of contents

01

Eliminating Security Scans: Saying Goodbye To A Constant Interruption

Legacy AppSec tools like static application security testing (SAST) and software composition analysis (SCA) rely on periodic scans, and any development work that takes place during the scan period requires a new scan. These scans can be very time-consuming: One test found that vulnerability scans can sometimes take more than two and a half hours to complete.[3] Scans must be conducted every time changes are made to the software, resulting in frequent interruptions to the development process—and often significant delays in the delivery of the application.

Instrumentation eliminates the need to stop development for vulnerability and security scans across the SDLC. Agents inside the application itself continuously monitor code and provide code-level feedback that empowers developers to fix problems on the fly. The IAST functionality within an instrumentation platform provides more complete and timely identification of vulnerabilities than legacy SAST and dynamic application security testing (DAST) tools combined, while OSS keeps a detailed database of all open-source dependencies—without interrupting development work with scans.

To ensure that an instrumentation platform is as effective as possible in eliminating scans and other coding delays, developers should look for an integrated platform that provides built-in, automated AppSec across the SDLC. It should support all applications, application programming interfaces (APIs), libraries, and frameworks and should provide integrated protection across development servers, test servers, and production servers.

> “
>
> One test showed that vulnerability scans can take code offline for as long as 164 minutes.[4]

> “
>
> [B]ecause security is integrated into the application, security no longer needs to disrupt coding and release cycles.[5]

[3] Michael D. Ernst, et al., “Boolean Formulas for the Static Identification of Injection Attacks in Java,” University of Washington, accessed April 14, 2020.

[4] Michael D. Ernst, et al., “Boolean Formulas for the Static Identification of Injection Attacks in Java,” University of Washington, accessed April 14, 2020.

[5] Tim Freestone, “AppSec Instrumentation Addresses AppSec Skills Shortage,” Security Boulevard, March 9, 2020.

02

Doing Away With False Positives: Returning Lost Time To Developers

Because they scan lines of code without any consideration of how users interact with the software, legacy SAST and SCA tools are notorious for false-positive. In fact, the Open Web Application Security Project (OWASP) Benchmark Project finds that the average SAST tool has a nearly 23% false-positive rate.[6] The result is significant alert fatigue for developers. Every security scan results in wasted time—and delays in pushing code—as developers sift through irrelevant and unprioritized alerts. For applications in production, web application firewall (WAF) tools show a similar propensity to false positives that can potentially pull developers off their current projects—in addition to impacting security operations (SecOps) productivity—to investigate an extraneous alert for an earlier project.

> “At the end of the day, your security tools need to give you less, but significant, Alerts that contain the correct intelligence to best inform your security and development teams.[7]

[6] “Accurately Assessing AppSec With the OWASP Benchmark Project,” Contrast Security, December 2016.

[7] Patrick Spencer, “Accuracy in AppSec is Critical to Reducing False Positives,” Contrast Security, April 8, 2020.

Instrumentation virtually eliminates false positives because it takes a totally different approach from legacy AppSec tools. Instead of security testing and development operating in separate, asynchronous silos, the two processes run in parallel. Instrumentation weaves sensors into the application that watch what happens. Unlike SAST, which simulates a control flow and data flow graph, IAST and RASP leverage the code flow graph that was created by the runtime. This provides deep visibility into both the application code and its runtime environment.

IAST provides direct, real-time vulnerability analysis and threat telemetry—with unparalleled accuracy. Once an application is in production and a zero-day attack occurs, RASP provides true self-protection from within the application, providing the same highly accurate telemetry and combining it with policy-based threat response. Instead of relying on pattern matching or behavioral learning, RASP simply watches from inside the running code to understand how it is vulnerable.

To achieve the greatest reduction in false positives, developers should look for a security instrumentation solution that uses multiple datasets in its continuous analysis. Ideally, an IAST tool will combine the best features of SAST, DAST, configuration analysis, and open-source analysis with real-time, code-level feedback.

The best instrumentation solutions also include route intelligence—the analysis of the data movement that takes place when a user interacts with an application. Rather than analyzing lines of code, route intelligence maps URLs to code paths that inform developers on how an application is accessed. Because it analyzes how real users will interact with the software, route intelligence provides the most complete visibility of the entire application attack surface.

> “
>
> [Rasp] can distinguish between actual attacks and legitimate requests for Information, which reduces false positives and allows network defenders to spend more of their time combating real problems and less time chasing digital Security dead ends.[8]

[8] John P. Mello Jr., “What is Runtime Application Self-Protection (RASP)?” TechBeacon, accessed April 15, 2020.

03

Automating Remediation Verification: Eliminating A Time-Consuming Manual Process

Another time-consuming security process that developers must perform is verifying that their fixes to identified vulnerabilities have actually corrected the problem. With legacy approaches to AppSec, this is a totally manual—and often frustrating—process that results in further coding delays. Developers and SecOps teams must spend valuable time tracing different iterations of code to verify vulnerability remediation.

Instrumentation can address this problem through automation, using both IAST and RASP solutions. After receiving actionable insight from the continuous scans that take place in the background, a developer can adjust code and receive immediate feedback as to whether the fix was successful.

Instrumentation platforms that include route intelligence provide even more robust verification feedback for fixes that are identified. This functionality can compare successive security assessment results for each application route to ensure that the vulnerability originally discovered on an entry point is no longer present. And because route intelligence is employed, remediation verification is automated, even if application source code changes.

> “
>
> When combined with analysis techniques, interactive application security testing can identify a broad range of potential vulnerabilities and confirm control Effectiveness.[9]

> “
>
> An appsec platform powered by instrumentation... Automates vulnerability Identification as well as the verification of vulnerability remediation.[10]

[9] “Security and Privacy Controls for Information Systems and Organizations,” National Institute of Standards and Technology (NIST), Draft Special Publication 800-53, March 2020.

[10] Tim Freestone, “AppSec Instrumentation Addresses AppSec Skills Shortage,” Security Boulevard, March 9, 2020.

04

Prioritizing Open-Source Vulnerabilities: Eliminating Needless Manual Work

The development community is driving further efficiencies through an increased use of open-source code. In fact, Forrester recently found a 40% jump in the use of open-source code in one year.[11] At the same time, the number of vulnerabilities identified in open-source code is skyrocketing at an unprecedented clip.[12] Having to track down huge numbers of Common Vulnerabilities and Exposures (CVEs) is a major time sink for developers, and the vast majority of them are not risky. In fact, only 0.6% of CVEs are ever exploited in the wild.[13]

Instrumentation solves this problem by providing deep insights into open-source dependencies and the level of risk actually posed by specific open-source vulnerabilities. The OSS solution in an instrumentation platform continuously maintains a detailed database of open-source dependencies and tracks newly discovered CVEs that might cause problems in an application.

The best OSS solutions also analyze which vulnerabilities found in a code scan are actually used by the application, eliminating a set of CVEs that pose zero risk to an organization. Developers should also seek a solution that enables custom policies across the SDLC, and has the ability to block attacks at runtime.

This risk management-based approach to open-source vulnerabilities, combined with real-time intelligence from the IAST platform, virtually eliminate coding delays for developers resulting from open-source vulnerabilities. The vulnerabilities that truly pose a risk are identified early and rise to the top of the list, where they can be addressed in near real time.

> “
>
> Use of open-source code by developers grew by 40% in a single year.[14]

> “
>
> Only 0.6% Of all CVES are ever exploited in the wild.[15]

[11] Amy DeMartine and Jennifer Adams, “Application Security Market Will Exceed $7 Billion By 2023,” Forrester, updated March 29, 2019.

[12] Liam Tung, “Open-source Security: This is Why Bugs in Open-source Software Have Hit a Record High,” ZDNet, March 13, 2020.

[13] Roger A. Grimes, “Are Zero-day Exploits the New Norm?” CSO, February 21, 2019.

[14] Amy DeMartine and Jennifer Adams, “Application Security Market Will Exceed $7 Billion by 2023,” Forrester, updated March 29, 2019.

[15] Roger A. Grimes, “Are Zero-day Exploits the New Norm?” CSO, February 21, 2019.

05

Avoiding False Negatives: Escaping Huge Delays Later On

False negatives are ticking time bombs that are destined to blow up at a later date, and legacy AppSec tools are notorious for missing vulnerabilities. The OWASP Benchmark Project finds that the overall accuracy score is just 20% for the average SAST solution and only 18% for the average DAST tool.[16] When these vulnerabilities are discovered—during final testing or in production—they are costly and time-consuming to remediate. For applications in production, developers can be pulled off new projects for time-consuming emergency remediation of old ones, resulting in huge delays to both. And remediation of vulnerabilities is significantly more time-consuming and expensive at this stage.[17]

Instrumentation results in a dramatic reduction in false negatives, again because of the completely different approach it takes compared with legacy tools. Instrumentation platforms do continuous scanning and evaluate applications from a variety of angles. Again, instrumentation platforms that include route intelligence provide further protection against false negatives, as they analyze an application the way users interact with it.

> “The cost of remediating a vulnerability in an application in production is 100X MORE than with vulnerabilities addressed during the design phase.[18]

[16] Amy DeMartine and Jennifer Adams, “Application Security Market Will Exceed $7 Billion by 2023,” Forrester, updated March 29, 2019.

[17] Mukesh Soni, “Defect Prevention: Reducing Costs and Enhancing Quality,” iSixSigma, accessed April 9, 2020.

[18] Mukesh Soni, “Defect Prevention: Reducing Costs and Enhancing Quality,” iSixSigma, accessed April 9, 2020.

06

Achieving True Devsecops

Instrumentation is a game changer for developers when it comes to AppSec. Embedding continuous security analysis into an application eliminates virtually all the frustrating coding delays that developers have come to expect from security processes.

This removes the friction that often exists between these two teams, which have historically been measured by different metrics that sometimes had them working at cross purposes. Developers can take care of the vast majority of vulnerabilities without the involvement of the security team, removing another source of coding delay. The result: more of a partnership between security and development, and more of an integrated approach that could be called DevSecOps.

With security instrumentation, developers are freed up to focus on what they are good at—innovating and pushing code—with the knowledge that the application they deliver will be secure.

> A new approach that combines sast, dast, software composition analysis (sca), and interactive application security testing (iast) breaks down the silos separating different security tools and processes.[19]

> “Instrumentation-based application testing improves security without skilled security staff or the need to change code.[20]

[19] Tim Freestone, “AppSec Instrumentation Addresses AppSec Skills Shortage,” Security Boulevard, March 9, 2020.

[20] Erik Costlow, “Changing the AppSec Game with Security Instrumentation,” Security Boulevard, April 2, 2020.

Contrast Security provides the industry's most modern and comprehensive Application Security Platform, removing security roadblocks inefficiencies and empowering enterprises to write and release secure application code faster. Embedding code analysis and attack prevention directly into software with instrumentation, the Contrast platform automatically detects vulnerabilities while developers write code, eliminates false positives, and provides context-specific how-to-fix guidance for easy and fast vulnerability remediation. Doing so enables application and development teams to collaborate more effectively and to innovate faster while accelerating digital transformation initiatives. This is why a growing number of the world's largest private and public sector organizations rely on Contrast to secure their applications in development and extend protection in production.

240 3rd Street
2nd Floor
Los Altos, CA 94022
Phone: 888.371.1333
Fax: 650.397.4133

Contrast
SECURITY

contrastsecurity.com