Contrast SECURITY

Route Coverage Through Instrumentation and Automated Vulnerability Management

Achieving More Effective Risk Management and Efficient Operations

Executive Overview

The identification of vulnerabilities and verification that they were remediated can consume substantial time and energy for security teams as well as developers. At the same time, these manual processes slow development cycles—thereby inhibiting digital transformation. These legacy application security (AppSec) approaches simply cannot support the velocity and agility demanded by modern software development. Code scanning—whether static or dynamic—is a broken model, and AppSec requires a fundamental change.

This involves security instrumentation that uses route intelligence to determine application route coverage—which ones have and have not been exercised. This combination enables security teams and developers to automate vulnerability identification and the verification that vulnerabilities have been fixed. This new approach to AppSec embeds security within the software, improving detection accuracy and pinpointing only those vulnerabilities that pose risk. This, in turn, eliminates false positives that consume valuable time and resources to investigate while pinpointing false negatives.

The success of any AppSec program centers on the ability to continuously identify vulnerabilities and to verify their remediation. If vulnerable code is exploited in production runtime, the results can be dramatic, including breaches of critical data, brand damage, noncompliance with various industry regulations and standards, legal action, and operational outages and disruptions.[1]

Failings of Legacy Application Security and Vulnerability Management

Legacy AppSec, such as static application security testing (SAST) and dynamic application security testing (DAST), is plagued with manual processes used for vulnerability management. In addition to manual, frustrating workflows and processes used to identify vulnerabilities and then verify their remediation, legacy AppSec testing models generate large volumes of false positives due to incorrect assumptions about how data flows through the application.

Sorting through the vulnerabilities that do not matter—namely, false alerts—is time-consuming. Plus, because legacy AppSec is point in time rather than in real time and solutions rely on signature-based scanning to identify vulnerabilities, these traditional security testing solutions incur false negatives that increase application risk and consume valuable time fixing once they are discovered in production runtime.[2]

> “43% of data breaches are caused by exploitation of web application vulnerabilities.[3]

Contrast SECURITY

Automated Continuous Vulnerability Identification and Remediation Verification

A paradigm shift is required in order to address the deficiencies of legacy application security testing approaches. Rather than scanning code line by line using point-in-time signatures from outside of the software (outside-in), AppSec needs to be embedded within software using instrumentation. This integration with existing development tools and workflows enables both applications and associated application programming interfaces (APIs) to be analyzed for vulnerabilities. Further, an instrumented approach unlocks automation and is part of continuous integration/continuous deployment (CI/CD) processes. This eliminates time spent scanning each code commit—which, in turn, delays development cycles and wastes time.

AUTOMATE VULNERABILITY IDENTIFICATION

A reliance on manual processes for vulnerability identification increases workloads on security and development teams. In order to achieve the velocity demanded by Agile and DevOps, organizations must ensure AppSec testing does not slow coding and release cycles.

There are multiple positive outcomes. First, automation of vulnerability identification within the CI/CD pipeline removes these obstructions—freeing up valuable time for developers to spend on writing more code and releasing it faster. Second, automated vulnerability identification enables developers to learn from their mistakes, allowing them to more efficiently generate secure code in the future.

To provide fully automated vulnerability identification, an AppSec solution must be capable of integrating with DevOps and Agile workflows. Application testing must be continuous and performed automatically as part of the CI/CD processes. Vulnerabilities that are identified need to be automatically added to the development team's issue tracking and project management systems for remediation.

> Web application attacks doubled from 2019 to 2020—ratcheting up the risk of application vulnerabilities even further. [4]

AUTOMATE THE VERIFICATION OF VULNERABILITY REMEDIATION

Triaging the root cause of a vulnerability and then verifying its remediation is a time-consuming and manual process. With legacy application security testing, the quantity of alerts—some of which are false positives—and inability to prioritize vulnerability fixes can be overwhelming. Correlation of test results and remediation of discovered vulnerabilities are manual processes, and it is not always easy to know how to fix them. For example, modifications to the code may fail to fix the identified vulnerability, and it may even introduce new ones. This requires repeated testing to verify that the vulnerability was fixed—which wastes even more time.

Automated vulnerability remediation verification using instrumentation and route intelligence is an essential component of an AppSec solution for DevOps and Agile—dramatically speeding the vulnerability verification process. To be effective, automated vulnerability remediation testing performs vulnerability identification and correlates discovered vulnerabilities with previous tests to determine if a vulnerability was successfully remediated—namely, if a vulnerability is not present when the application route is exercised again, then it is deemed fixed. Successful remediation is then automatically logged in the development team's issue tracking system.

VULNERABILITY MANAGEMENT NOW REQUIRED IN NIST CYBERSECURITY FRAMEWORK

Most AppSec testing solutions produce reports that provide a snapshot of the vulnerabilities present in an application at one point in time. This certainly may be useful for regulatory compliance, but the Agile and DevOps CI/CD pipeline necessitates vulnerability management that is continuous, as a single change in a line of code can introduce a new vulnerability.

Acknowledgement of the need for continuous vulnerability identification and remediation is evident in the new interactive application security testing (IAST) standard in the National Institute of Standards and Technology (NIST) Cybersecurity Framework: SA-11 (9), Developer Security Testing and Evaluation: "Require the developer of the system, system component, or system service to employ interactive application security testing [IAST] to identify flaws and document results."[5] Compliance, in short, requires application security testing that is instrumented and continuous.

> The average web application has 1,000 different dependencies.[6]

Comprehensive, Accurate Vulnerability Identification

Accurate vulnerability identification is essential for scalable AppSec. Overreporting of potential vulnerabilities that do not pose a threat wastes valuable time and resources, while missed detections leave an organization vulnerable to attack. Comprehensive, accurate vulnerability detection requires full visibility into custom and open-source code, complete coverage of an application's execution paths, and the ability to differentiate legitimate threats from extraneous detections.

VISIBILITY INTO OPEN SOURCE

With the growth in open-source libraries and frameworks, the number of third-party dependencies in software has grown exponentially. Indeed, recent research indicates that the average web application has over 1,000 different dependencies.[7] Many applications, as a result, depend on millions of lines of third-party code, which can introduce new vulnerabilities into code that depends on it.

Minimizing false positives when analyzing third-party code requires a solution with visibility into the execution state of an application. By monitoring the routes an application follows, APIs exercised, and the data that passes through the application, an application security solution eliminates false positives by identifying vulnerabilities that are only at risk of exploitation.

> One-quarter of security alerts are false positives.[8]

INTERNAL VISIBILITY INTO CODE EXECUTION PATHS

A modern software development life cycle (SDLC) requires AppSec solutions that provide internal visibility into an application and reveal comprehensive vulnerability identification. This includes the ability to follow runtime execution paths within an application during testing rather than performing line-by-line code or API-based analysis.

In particular, an API may expose several potential entry points to an application. However, complete test coverage of the entry points of an API does not guarantee coverage of all possible code execution paths within an application. This mistaken assumption contributes to high false-negative rates in DAST solutions.[9]

To address these issues, an AppSec approach must secure the application and APIs regardless of the execution environment. Analysis areas include HTTP requests and responses, libraries and frameworks and how they are used, data flow, back-end connections, and configuration parameters. This inside-out application security approach eliminates the need for specialized security toolsets to protect containers and microservices. In addition, AppSec must be extensible and follow an application regardless of changes in connections and configurations as well as the addition of containers or microservices.

> Hundreds, if not thousands, of potential vulnerabilities exist in the places where applications and functions come together—application programming interfaces.[10]

FOCUS ON RISK VERSUS SIGNATURES

Legacy AppSec approaches employ signatures for scanning software for vulnerabilities. These pinpoint known vulnerabilities but incur false negatives—missing vulnerabilities that are unknown or zero day. With upwards of one-quarter of identified vulnerabilities not posing any risk,[11] this translates into a significant waste of time for security and development teams. In contrast, analyzing route coverage identifies vulnerabilities that are previously known and unknown. Only vulnerabilities that pose a risk are identified, saving developers significant time triaging through false alerts.

For applications in production runtime, perimeter defenses—web application firewalls (WAFs)—employ the same outside-in approach as SAST and DAST scanning engines. This results in high numbers of false positives that can overwhelm security teams. Indeed, 99% of attacks against web applications do not reach a targeted vulnerability.[12] When security instrumentation is extended into production runtime, security testing is performed at the point of exploitation—runtime application self-protection (RASP). Only those vulnerabilities that can be exploited are identified.

> The importance of identifying and fixing vulnerabilities in development is immensely cheaper than remediating them in production runtime—100x per one report.[13]

Vulnerability Management at the Speed of DevOps

Digital transformation demands velocity and accuracy. Legacy application security testing that slows code commits and release cycles ratchets up pressure on development teams, with more than half of developers admitting that they have scaled back security measures to meet a business deadline. This should not be a surprise: More than two-thirds of organizations have a mandate from the CEO that nothing should be allowed to slow down the development process.[15]

By automating vulnerability management—from identification to verification of remediation— organizations can remove gates that create operational inefficiencies and slow development cycles, enabling them to release with confidence.

[1] Kelly Bissell, et al., "Innovate for Cyber Resilience: Lessons from Leaders to Master Cybersecurity Execution," 3rd Annual State of Cyber Resilience Report, Accenture Security, March 2020.

[2] "Integrated Security Instrumentation Is the Future of AppSec: Interview with Surag Patel," Inside AppSec, Episode #3, April 2020.

[3] Patrick Spencer, "43% of Data Breaches Are Connected to Application Vulnerabilities: Assessing the AppSec Implications," Contrast Security Blog, May 20, 2020.

[4] "2020 Data Breach Investigations Report," Verizon, May 2020.

[5] "Security and Privacy Controls for Information Systems and Organizations," National Institute of Standards and Technology, March 2020.

[6] Rui Ribeiro, "Enterprise Web Security: Risky Business," Dark Reading, November 5, 2019.

[7] Ibid.

[8] Ericka Chickowski, "Every Hour SOCs Run, 15 Minutes Are Wasted on False Positives," Security Boulevard, September 2, 2019.

[9] "How to Secure APIs at DevOps Speed," eBook, Contrast Security, May 2020.

[10] Curtis Franklin Jr., "How to Manage API Security," Dark Reading, December 17, 2019.

[11] Kelly Sheridan, "Security, Networking Collaboration Cuts Breach Cost," Dark Reading, February 24, 2020.

[12] "Contrast Labs Application Security Intelligence Bimonthly Report, January–February 2020," Contrast Security, April 2020.

[13] Mukesh Soni, "Defect Prevention: Reducing Costs and Enhancing Quality," iSixSigma, accessed May 8, 2020.

[14] "52% of Companies Sacrifice Cybersecurity for Speed—Webinar Recap," Threat Stack, March 13, 2018.

[15] Ibid.

Contrast SECURITY

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