SYNOPSYS®

How to Leverage ASOC to Manage Software Risk

An Application Security Orchestration and Correlation Handbook

Table of contents

Development is accelerating and changing

In everything from web applications to medical devices, consumer electronics, and automobiles, software is changing how value is delivered. This is turning software engineering teams into an increasingly critical piece of the value chain and driving the adoption of rapid development processes such as agile, DevOps, and continuous integration / continuous delivery (CI/CD).

For most software projects, the days of quarterly releases are long gone. Today, some companies are releasing new features and software every few seconds.

- Facebook delivers between 50,000 and 60,000 Android builds each day.
- Amazon reportedly deploys new software to production every second.
- Netflix's DevOps teams deploy new releases 100 times each day.
- Etsy uses its continuous delivery practice to deploy a single monolithic application more than 60 times a day.

It's clear that this level of activity requires automation. This includes running the right security tools at the right time and managing the results of those scans. The growing adoption of security automation led Gartner to define a new category of solutions that merge the application vulnerability correlation and application security testing orchestration market segments into one, termed application security orchestration and correlation (ASOC). In addition to selecting, configuring, and scheduling security testing tools, ASOC tools help security, operations, and development teams understand and prioritize risk in applications. These solutions also save time and effort by aggregating, normalizing, correlating, and prioritizing scan results from multiple and disparate tools, and then recommending remediation steps based on the criticality of the project and its development pipeline.

Rapid development and deployment challenges security

Adopting DevOps and CI/CD methodologies can help organizations deliver better code faster, but they also introduce new challenges to development, security, and operations. These practices require teams to integrate security earlier in the development life cycle.

Security-centric vs. developer-centric approaches

Development teams are measured primarily on their ability to deliver a specific set of features by a specific date. When security testing occurs late in the development life cycle, vulnerabilities are more difficult to remediate and cause delays. This can result in development teams viewing security teams as an impediment to timely releases.

The solution, of course, is to "shift left"—identify vulnerabilities earlier in the development life cycle, when remediation is simpler. This can be accomplished by integrating security teams into the development process or by providing development with the responsibility and tools needed to test for vulnerabilities.

- **A security-centric approach** involves defining security requirements and secure coding policies in the design phase of the development life cycle, and then testing builds incrementally to catch coding errors that can result in security issues early. This approach requires organizations to scale security resources in a tight labor market.
- **A developer-centric approach** can mean less friction and lower demands on security for testing, but it also requires controls that security can monitor to ensure that policies are followed, risk mitigation is taking place, and appropriate reporting is available for management and audit teams. It also can put the burden on development teams to normalize and interpret the results from a variety of security testing tools.

Rather than taking a security-centric or developer-centric approach, it makes more sense to identify common needs of both teams. Either way, it's vital to define clear requirements and assignment of duties. ASOC tools provide a balanced approach that enables developers to build secure software at DevOps speed, and validates testing and controls so security can ensure internal or external compliance.

An expanding security toolbox

The maturity of an organization's application security programs can vary between business units and project teams. Likewise, security tools can vary widely and report issues in unique ways.

Different tools return different results. Static analysis can identify many common security bugs in code, including SQL injection, cross-site scripting, and buffer overflows. Dynamic analysis can find many of the same vulnerabilities as static testing, but it's particularly useful for injection errors and privilege escalation. Neither static nor dynamic analysis will find open source components with known vulnerabilities—software composition analysis (SCA) tools can do that. Because of this, multiple tools are recommended to cover an application adequately.

Penetration testing: Many organizations use third-party penetration tests (also called ethical hacking) to identify vulnerabilities in their web applications. Penetration tests require a completed application in a staging environment, and therefore can only be performed late in the development process.

Static analysis: Static application security testing (SAST) tools generate a model of an application's source or binary code by mapping the control flow and data flow of the application. The model can then be queried by rules to identify coding errors that might result in security vulnerabilities. SAST tools can be used on early builds of an application, but due to their complexity, they are typically used when an application is nearly complete. Results are reported where the error occurs, by file name and line number.

Dynamic analysis: Dynamic application security testing (DAST) tools, like penetration testing, run on complete or nearly complete applications and use techniques such as malformed inputs to attempt to bypass security controls, identify vulnerabilities, or crash the application. Results are reported as a URL/action/result.

Interactive analysis: Interactive application security testing (IAST) tools instrument the application and then monitor activity during normal functional testing to identify vulnerabilities.

Software composition analysis: The need for rapid time to market has led to the broad adoption of open source libraries. While this greatly reduces development time, thousands of new vulnerabilities are disclosed in open source each year. SCA tools scan codebases and package managers to identify open source components and then map those to a database of known vulnerabilities such as the National Vulnerability Database, or more complete private databases provided by SCA vendors.

	Static Analysis	Dynamic Analysis	Interactive Analysis	Source Composition Analysis	Penetration Testing
Identifies vulnerabilities from coding errors	✓	✓	✓		✓
Identifies vulnerable open source components				✓	

Scarce security resources

No organization has all the security expertise it wants or needs. Security resources are in high demand; research indicates that there are between 3.5 million and 4 million unfilled cyber security jobs worldwide, and help from universities is not expected soon. Finding and retaining security personnel to build policies, run tools, interpret results, and offer remediation guidance continues to challenge organizations.

Overlapping regulatory requirements

Organizations are subject to an increasing number of regulatory standards. The goal of these standards is to ensure that software and systems have adequate controls to protect sensitive data. Financial services organizations in particular operate under a host of regulatory standards. This makes sense, as the assets and information managed by these firms is valuable, sensitive, and targeted by attackers daily. Most are subject to the Gramm-Leach-Bliley Act (GLBA) and the Payment Card Industry Data Security Standards (PCI DSS). If an organization is a public company, the Sarbanes-Oxley Act applies. Personally identifiable information about European consumers is covered by the General Data Protection Regulation (GDPR), and U.S. customers are covered by Section 5 of the Federal Trade Commission Act. Individual states also have regulations, including the California Consumer Privacy Act.

In addition to general security controls, standards like the PCI DSS obligate organizations to test for specific types of vulnerabilities. Knowing which tools and scans to run for each application complicates security policies.

Introducing ASOC

Clearly, automation is critical to a successful DevOps or CI/CD program. With the need for rapid delivery and testing, manual processes ensure failure. ASOC tools provide that automation while allowing granular control over each step of the process. These tools automate the deployment of the right tools at the right time; allow granular policy enforcement; aggregate, deduplicate, normalize, and correlate findings; and provide audit control and reporting to support organizational and regulatory standards.

How it works

Application onboarding

When a new project is started, development and security teams fill out a brief questionnaire. The ASOC solution can use the information provided to rank the level of risk of the application under test based on the application's criticality to business goals and the consequences of a successful attack on the application. For most organizations, the application falls into one of five categories: critical, high, medium, low, or unranked. This ranking ensures consistent and objective characterization of all projects and the application of all appropriate policies.

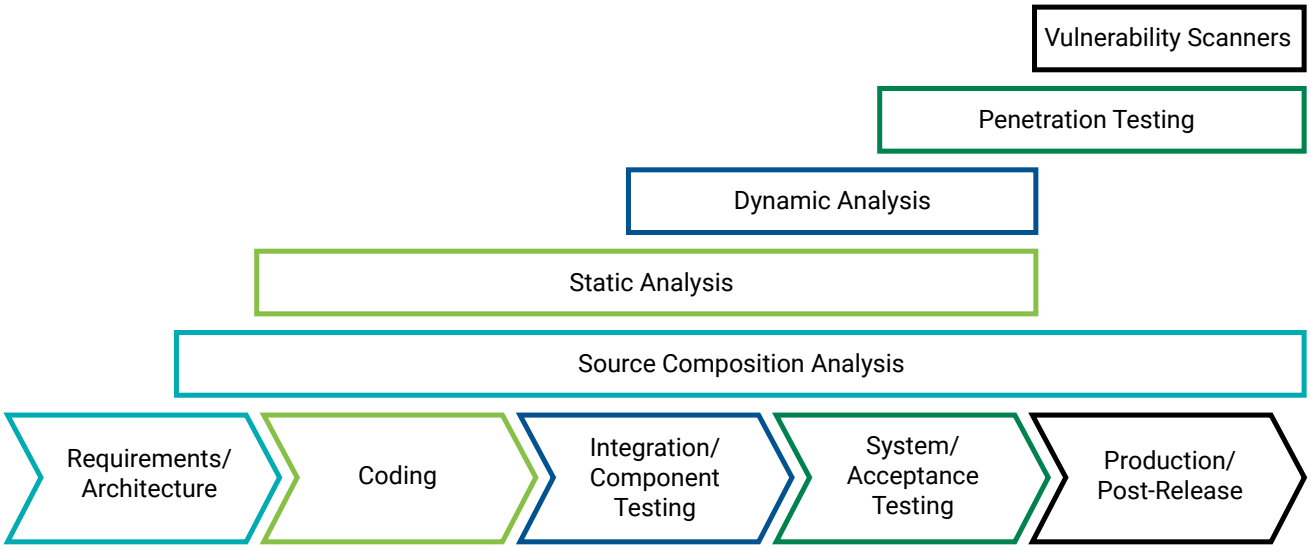
Policy management

Smart organizations understand that different applications require different levels of scrutiny, and they set security policies appropriately. Policy management includes varying the policy by build pipeline, so development isn't hindered by overly strict policies for projects early in development. It should also include any specific policies required by regulatory standards.

Tool selection

Based on the ranking of the application, internal policies, and regulatory requirements, the ASOC solution determines security scan requirements. This can include which tools to run based on the application's programming language or corporate policies.

Tool Usage in the Development Life Cycle



Broad tool support is necessary

Development environments can differ between teams, and they are characterized by frequent changes. To be usable in an enterprise setting, ASOC solutions must support a wide variety of commercial and open source tools including:

- **Developer tools** including a variety of integrated development environments (IDE), source code repositories, and binary repositories. Integrations allow developers to launch a security scan, view results and remediation guidance, and designate false positives, all without leaving the normal working environment.
- **Build tools** compile human-readable source code into machine-readable binary code. ASOC solutions schedule the tasks required for each build based on appropriate policies. These policies can include breaking a build—stopping the code from advancing in the development process—if certain criteria are unmet. Popular build tools include Jenkins, Maven, Gradle, Travis CI, and Bamboo.
- **Security testing tools** look for coding errors and third-party components with known vulnerabilities. It's important to include manual testing techniques in addition to automated scanning tools. Since each vendor takes a different approach to analyze code, many teams will use multiple scanners.
- **Developer and security reporting solutions** provide information on security bugs and flaws, including prioritization and remediation guidance. ASOC solutions should support integration with leading bug tracking tools. In addition, a full audit trail should be available to provide evidence that teams adhered to accepted policies.

Tool configuration

Based on the criticality of the application, organizational policies, and regulatory requirements, the ASOC tool will configure the rule sets used by the scanning tools. Many teams prefer to run a condensed set of highly accurate rules that produce minimal false positives in early runs. This minimizes the time required to analyze the results and eliminate false positives and informational issues while maximizing developer productivity.

Which tools to use in which pipelines

Not all software projects require the same level of security scrutiny. Public-facing applications require more testing than internal projects with a limited attack surface. Applications that manage sensitive data warrant more testing. Further, some projects require specific rule sets. For example, PCI DSS requires testing for industry benchmarks such as the OWASP Top 10, SANS CWE

Top 25, or CERT Secure Coding Standards. Determining which tools will be used for each application in each pipeline—and which rules are critical for each—can be difficult to manage manually.

Enforcing security gates

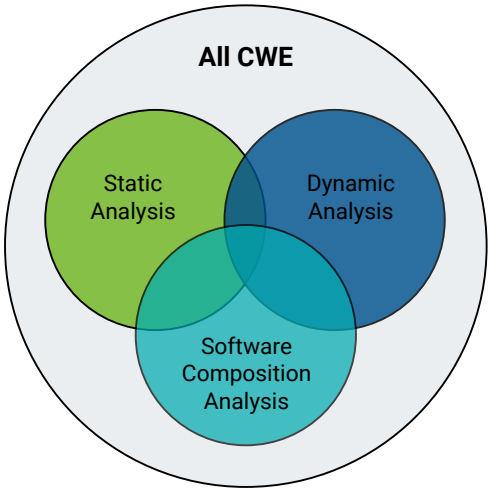
When choosing and configuring security tools, it's important to determine the optimal scanners for each stage of the development life cycle. Rapid development and deployment methodologies often include rules or "gates" that applications must pass prior to being promoted in a pipeline. However, it isn't always necessary to require the same security gates across each pipeline. For example, teams may elect to break builds if a component with a critical vulnerability is detected in a production pipeline, but let the build complete in a test pipeline. Determining whether an application may be promoted by manually analyzing the results of each security scan can slow down development.

Results aggregation and normalization

Different types of tools detect and report vulnerabilities in different ways. Instead of viewing the results from each scanner in that tool's dashboard, ASOC solutions aggregate the results from each tool and provide a single place to view and analyze vulnerability reports. Since each vendor reports severity in its own way, ASOC solutions also normalize the results, allowing teams to use a single scale to communicate the criticality of any vulnerability. The table below is an example of how four different tools might rank the severity of the same vulnerability.

	Tool 1	Tool 2	Tool 3	Tool 4
Scoring System	High	Severe	8 (of 10)	5 (of 5)

Results correlation



As shown in the adjacent figure, different testing methodologies report results in different ways. SAST results are reported as a file name, line number, and issue type. Dynamic analysis and penetration testing results are reported as a URL/action/result. IAST results are reported as a file name, line number, and issue type. And SCA results are reported as the vulnerable component, CVE identifier, and CVSS score. It can be difficult to know if a vulnerability is exploitable by an attacker.

Root cause analysis

ASOC solutions use advanced algorithms that match results from different testing tools to provide a complete and succinct set of results. In addition, leading ASOC solutions provide root cause analysis. For example, input validation issues like cross-site scripting occur when untrusted data enters an application without checks to control the type of data intended for that field. Some testing tools report a vulnerability each time that data is used. While these are true vulnerabilities, the most efficient method to correct all the errors is to add input validation when the data enters the application. This saves hours of remediation effort and retesting.

	Static Analysis	Dynamic Analysis	Interactive Analysis	Source Composition Analysis	Penetration Testing
Results Reported As	File name, line number, vulnerability type	URL, Input, Result	URL, Input, Result	Component name, component version, CVE identifier	URL, Input, Result

Triage

Unfortunately, security testing tools are notorious for high numbers of false positives and "informational" findings—those related to style rules or low-priority issues. Adding more tools compounds the issue and introduces duplicate findings.

Triage is the process of reviewing the aggregated, normalized, and correlated results to determine which are most critical to an organization. A triage exercise determines which issues need to be prioritized, which can be safely ignored, and which will represent residual risk.

Deduplicating findings and sorting critical issues from hundreds or thousands of findings has real costs. Assessing a single finding takes an analyst 10 minutes. With even a single tool generating hundreds or thousands of findings, triaging results to determine which are critical can require weeks of labor effort. This leaves teams with a poor choice: either slow down development to scrub results, or release software without fully understanding its security profile.

During triage, some findings will be deemed as presenting acceptable risk, and the issue will be suppressed in the report. The ASOC solution should be able to retain that suppression in results from future scans and save organizations the effort of reviewing those items again.

Auditability

In the event of an audit, ASOC solutions should provide evidence that an organization's policies and activities have been followed. This includes verifying that tests were performed as required, confirming that vulnerabilities were remediated according to policy, and documenting the reasons and approvals for any exceptions. This is difficult to achieve with multiple tools and reports, but ASOC tools can automatically apply and track policies and testing, and then aggregate, normalize, and correlate results, simplifying the process.

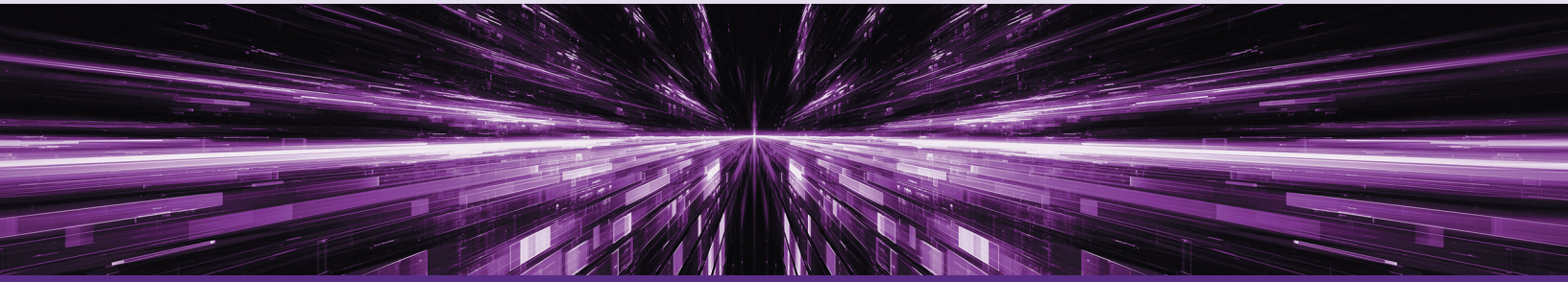
The future of ASOC

ASOC solutions are evolving as DevSecOps teams continue to adopt rapid development and deployment strategies. A natural next step is the application of machine learning to analyze security testing results. This technology will accelerate the triage process by learning how teams prioritized vulnerabilities in the past and applying that to new findings, or by looking at past false positive/true positive rates for each rule and each tool.

More than scheduling scans

Security testing tools have advanced greatly over the past 10 years, helping organizations build more secure software. However, rapid development and deployment strategies pose several challenges to tool vendors, as well as to security and development teams. In earlier times, security would test builds using one or more tools and then present reports to development itemizing the vulnerabilities. This approach is antithetical to the rapid development methodologies and quick feedback of DevOps teams.

ASOC solutions do more than simply schedule scans and host the results from testing tools. As teams run more tools more frequently, there will be more overlap in results and more confusion regarding which issues are critical to an organization's business goals. ASOC solutions can automate the aggregation, deduplication, and normalization of those results. Incorporating machine learning allows ASOC solutions to accurately prioritize issues based on the previous actions of an organization's security analysts for each project and in each pipeline. By doing so, ASOC tools enable security testing to scale and keep pace with DevOps delivery requirements.



SYNOPSYS®

The Synopsys difference

Synopsys helps development teams build secure, high-quality software, minimizing risks while maximizing speed and productivity. Synopsys, a recognized leader in application security, provides static analysis, software composition analysis, and dynamic analysis solutions that enable teams to quickly find and fix vulnerabilities and defects in proprietary code, open source components, and application behavior. With a combination of industry-leading tools, services, and expertise, only Synopsys helps organizations optimize security and quality in DevSecOps and throughout the software development life cycle.

For more information, go to www.synopsys.com/software.

Synopsys, Inc.
690 E Middlefield Road
Mountain View, CA 94043 USA

Contact us:
U.S. Sales: 800.873.8193
International Sales: +1 415.321.5237
Email: sig-info@synopsys.com