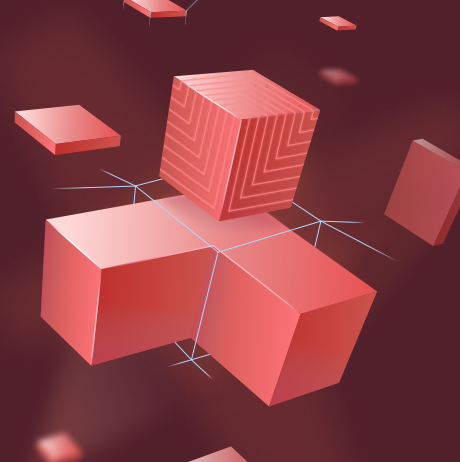


Next-Gen SCA: Pipeline Composition Analysis

 SOLUTION BRIEF



Application code dependencies like open source libraries comprise approximately 80-90% of modern application codebases. Organizations that aren't scanning these dependencies for vulnerabilities aren't securing their applications. However, scanning only application code—which is what traditional Software Composition Analysis (SCA) solutions focus on—is no longer enough to secure applications.

Vulnerable dependencies exist beyond application code in a wide range of tooling across the SDLC. Next-Gen SCA: Pipeline Composition Analysis closes the gaps created by traditional SCA solutions by identifying these dependencies. Pipeline Composition Analysis (PCA) also traces the path of a vulnerability from code through to production environments and gives more detailed prioritization advice based on runtime exploitability.

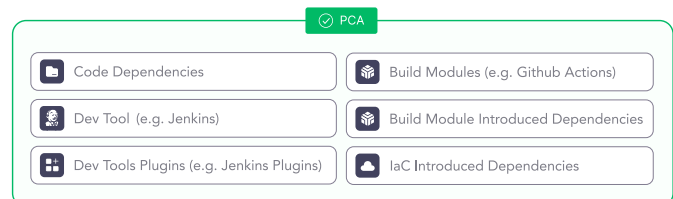
By scanning beyond application code and expanding the scope of where dependencies are located, PCA is better able to identify vulnerabilities that could lead to software supply chain attacks. Broadening the definition of dependencies to include those that exist in all phases of software delivery pipelines allows organizations to more accurately identify, prioritize, and remediate risk to prevent data breaches.

Find and Fix All Vulnerable Pipeline Dependencies

Dependencies that contain vulnerabilities or incompatible licenses expose organizations and put intellectual property at risk. These dependencies

exist throughout development pipelines, not just in application code, in development tools like Jenkins, plugins like Jenkins Plugins, build modules like GitHub Actions, build module introduced dependencies, IaC dependencies, and more.

With the interconnectedness of software development pipelines, a weakness in one component can expose the entire application. Organizations must scan all dependencies to find vulnerabilities. PCA takes the functionality of SCA and expands it to include a wide range of vulnerable dependencies that could compromise applications and lead to software supply chain attacks.

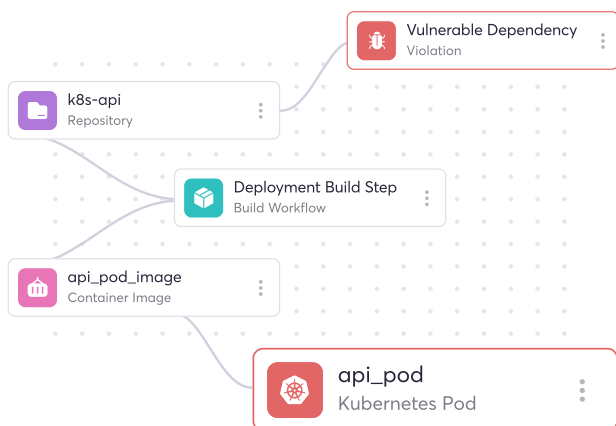


Bridge the Gap Between Code and Deployment

While traditional SCA is able to identify vulnerabilities in application code, it is unable to provide visibility into where that code has been deployed in production. When large-scale, high-risk exploits like Log4Shell occur, organizations must find every instance of a vulnerability fast. Without visibility into production environments, it is impossible to determine whether all deployed vulnerabilities have been remediated.

PCA provides complete visibility across the SDLC, including production environments. This enhanced

visibility allows the path of vulnerable components, from source code through to production environments, to be easily identified. When high-impact zero-day exploits are discovered, PCA enables organizations to respond quickly to threats and effectively remediate defects by identifying every production location in which a vulnerable dependency has been deployed. The result is faster response times and peace of mind that all deployed vulnerabilities have been remediated.



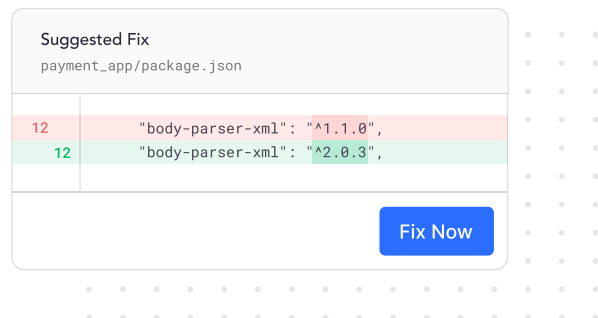
Prioritize Using Runtime Exploitability

Reducing security debt through effective prioritization is a top priority for SCA. Traditional SCA solutions, however, prioritize based only on whether a vulnerability is reachable in code. While this is one way to reduce the noise, a more effective solution is to prioritize vulnerabilities based on whether they are exploitable in runtime environments. Not all vulnerabilities are exploitable in all environments. Certain vulnerabilities that are reachable in code may require a specific environment in order to be exploited at runtime. If that runtime environment doesn't exist, the vulnerability can be deprioritized.

PCA prioritizes and remediates vulnerable dependencies by identifying whether a vulnerability is exploitable at runtime based on what components are in use in production environments. By determining whether a cloud environment has the right conditions for a vulnerability to be exploited, organizations are able to focus on high-risk vulnerabilities first to optimize precious development resources.

Shift Security Left with Developer Friendly Workflows

By giving developers tools they love to use, organizations can shift security left and help prevent vulnerable dependencies from entering codebases from the start. PCA gives developers scan results during pull requests that automatically recommend a fix with a single click. By seamlessly integrating with developer workflows and issue trackers, PCA helps remediate policy violations without hindering developer productivity.



Complete Software Supply Chain Security

Eliminating vulnerable dependencies is necessary, but it's only one part of application security. PCA helps secure application code, and the Cocode platform secures the entire software supply chain from code to cloud so organizations can manage their vast and interconnected attack surface.

Cocode hardens the SDLC's security posture by implementing consistent governance and reduces the risk of breaches with a series of scanning engines that look for issues like hardcoded secrets, code leaks, SCA, misconfigurations, SAST, and more. Cocode's Knowledge Graph tracks code integrity, user activity, and events across the SDLC to prioritize risk, find anomalies, and prevent code tampering.

 [Learn more at Cocode.com](https://cocode.com)