VERACODE Infosheet

SAST vs. DAST: What Are the Differences and Why Are They Both Important?

When it comes to application security (AppSec), no two scan types are the same. They are all designed with a different area of focus, much like health exams. Just as you would never assume your health is fully in order after one eye exam, you can't assume that your applications are fully secure if you only use one scan type.

CAPABILITIES	1 STATIC ANALYSIS	2 DYNAMIC ANALYSIS	3 INTERACTIVE APPLICATION SECURITY TESTING	4 SOFTWARE COMPOSITION ANALYSIS	5 MANUAL PENETRATION TESTING
Flaws in Custom Web Apps (CWEs)	✓	✓	✓		✓
Flaws in Custom Non-Web Apps (CWEs)	✓		✓		✓
Flaws in Mobile Apps (CWEs)	✓		✓		✓
Known Vulnerabilities in Open Source Components (CVEs)				✓	✓
Behavioral Issues (CWEs)	✓		✓		✓
Configuration Errors (CWEs)		✓			✓
DOM-Based Cross-Site Scripting	✓	✓			✓
Business Logic Flaws (CWEs)					✓
Coverage of Full Apllication	✓	✓		✓	✓
Repeatable for Process for Automation	✓	✓	✓	✓	
Scalable to All Corporate Applications	✓	✓	✓	✓	
Scan Speed	Seconds to Hours	Hours	Seconds to Minutes	Seconds to Minutes	Days to Weeks
Cost	$$	$	$$$	$	$$$$

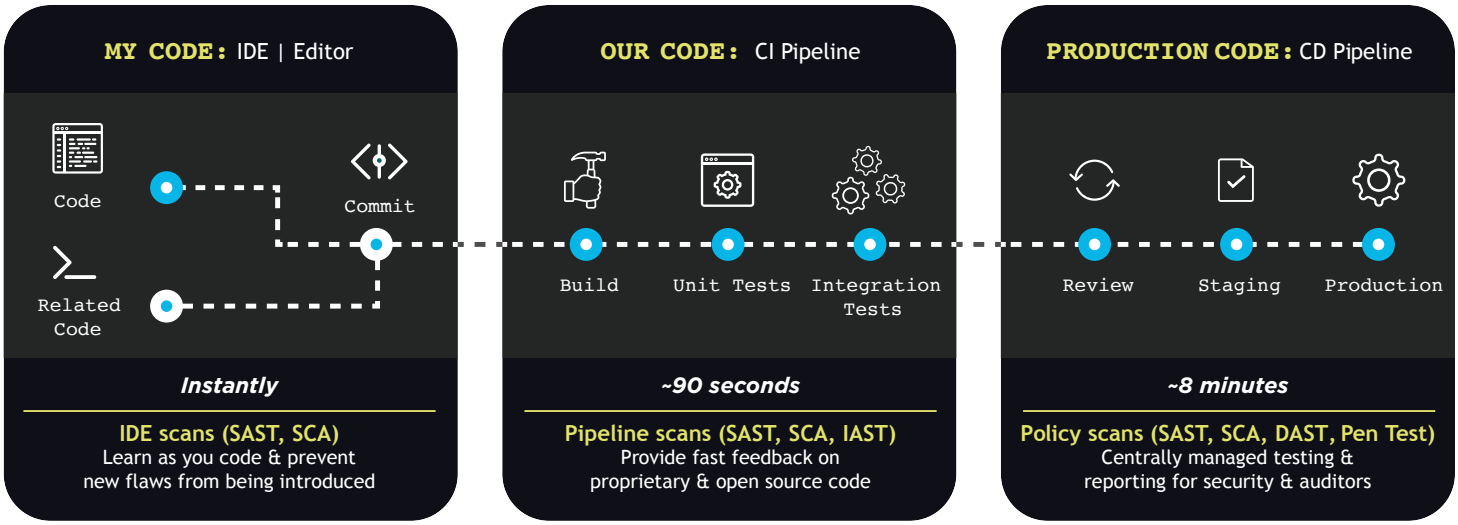
Take a look at the chart above. If you only use static application security testing (SAST), you won't find open source vulnerabilities, configuration errors, or business logic flaws. But if you use dynamic application security testing (DAST) with SAST, you still won't uncover all flaws, but you'll uncover more than SAST alone – like configuration errors. So the more AppSec scan types you employ, the more flaws you uncover.

Below, we will dive deeper into the differences between SAST and DAST and establish the benefits of using both scan types in unison.

What is SAST?

SAST scrutinizes the codebase for flaws and excels at finding many issues, such as directory traversals, Cross-Site Scripting, and various injection flaws. It can be conducted as soon as the first line of code is written through to production. For example, Veracode's SAST solution, ***static analysis***, delivers three methods of testing – the IDE Scan, the Pipeline Scan, and the Policy Scan – to meet the unique needs of development and security professionals across the SDLC.

The IDE scan is conducted in the pre-commit stage of development. It scans the code that a developer is currently working on and provides real-time feedback before they commit the code to the main repository. The Pipeline Scan happens in the build phase. It directly embeds into teams' CI tooling and provides fast feedback on flaws being introduced on new commits. Lastly, the Policy Scan evaluates the entire application against a given policy to ensure that it's compliant with industry standards and regulations.

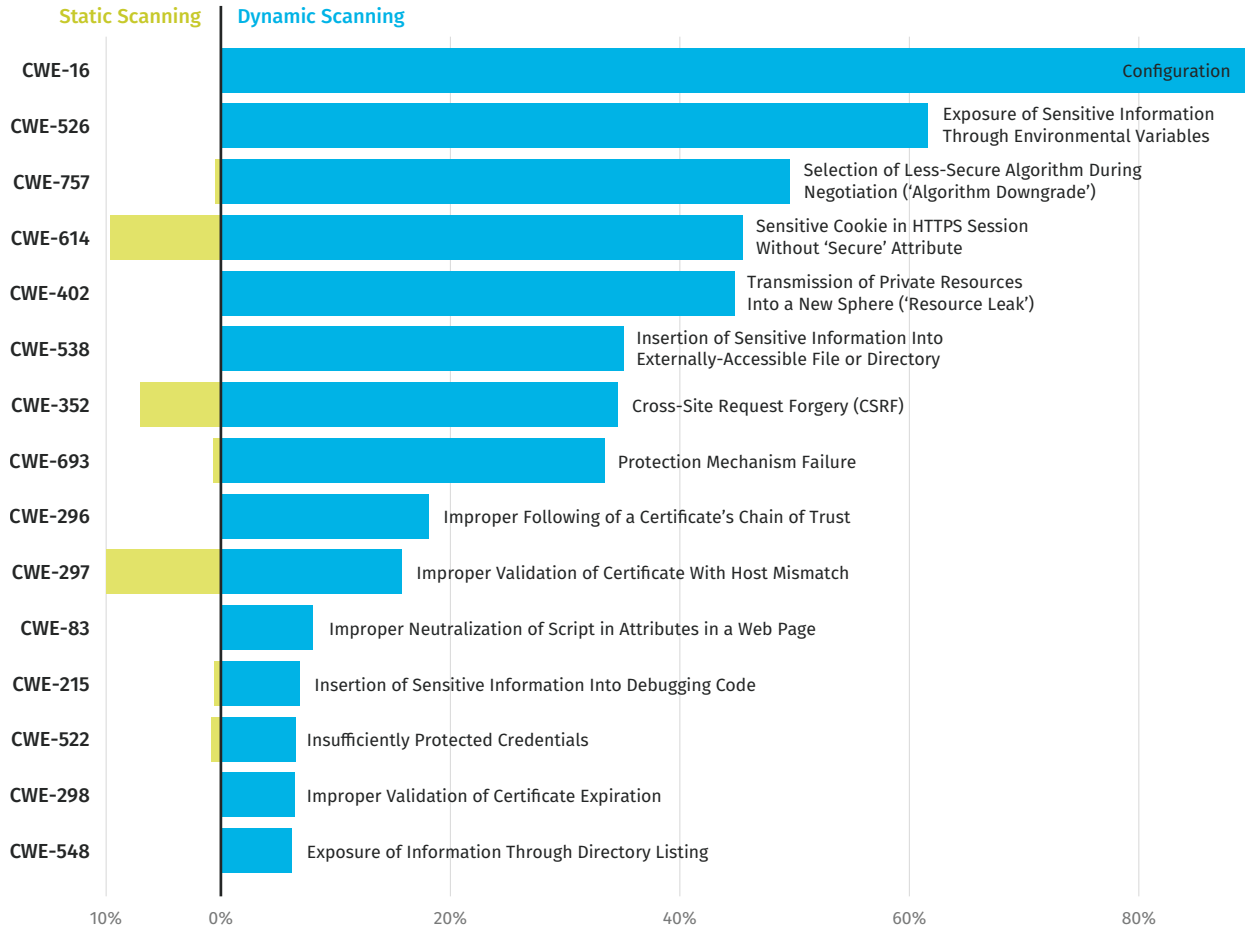


What is DAST?

DAST is an AppSec testing type used in the build phase of the SDLC. ***DAST*** interacts with the application like an attacker. It starts by performing a crawl to understand the application's architecture, including links, text, form fills, and other page elements that a user could potentially interact with. It also picks up on attack points that are less visible to the user, such as header values, cookies, and URL parameters. The scanner then audits the objects and attributes discovered by the crawl and sends attacks – like Cross-Site Scripting and SQL Injection – to the objects/attributes to see if they have any exploitable vulnerabilities.

Why is it important to use SAST and DAST together?

SAST and DAST find different flaws. SAST finds code issues and DAST finds exploitable flaws in runtime. When used together, you are able to find significantly more flaws than if you only use SAST or only use DAST.



As you can see in the figure from our recent ***State of Software Security (SOSS) report***, some flaws become more prevalent when dynamic scanning is used in conjunction with static analysis. For example, static analysis finds around 10 percent of CWE-297 flaws, but when DAST is added, the number of CWE-297 flaws discovered more than doubles.

Using SAST and DAST together also results in faster time to remediation. Our SOSS findings revealed that organizations using both DAST and SAST address 50 percent of their open security findings almost 25 days faster than organizations that only use SAST.

There are a couple of possible reasons why using SAST with DAST could result in faster time to remediation. One reason could be that since the flaws discovered by DAST are proven exploitable, developers may be more inclined to quickly remediate the flaws.

It's also possible that organizations using both SAST and DAST are further long on their AppSec journey. One of the steps in achieving AppSec maturity is having a comprehensive mix of AppSec testing types. So, if an organization adds DAST to their testing suite, chances are they are also taking other steps to mature their program like automating scans, conducting scans frequently, and remediating flaws in a timely manner.

For more detailed information on the benefits of using both SAST and DAST, download our recent ***State of Software Security (SOSS) report.***

VERACODE

Learn More

Veracode is the leading AppSec partner for creating secure software, reducing the risk of security breach and increasing security and development teams' productivity. As a result, companies using Veracode can move their business, and the world, forward. With its combination of automation, integrations, process, and speed, Veracode helps companies get accurate and reliable results to focus their efforts on fixing, not just finding, potential vulnerabilities.

Learn more at www.veracode.com, on the Veracode blog and on Twitter.