WHITE PAPER

Invicti

Security at the Speed of Software: DAST in the SDLC

Executive summary

You hear about it in every article and every conference keynote: the huge problem facing the cybersecurity industry. Threats are growing exponentially and security teams aren't.

The current climate, with huge breaches making international news on what feels like a weekly basis makes this reality more stressful and omnipresent. The truth is you cannot keep doing what "worked" 5 years ago. In fact, you can't keep doing what "worked" even last year. The world has changed and you have to adapt.

Large enterprises are managing 946 custom apps on average, and developing 193 more: a 20% increase from 2020. This scale and pace of development is a key reason for the ongoing security crisis. Estimates show that enterprise IT security teams have failed to secure 62%+ of their custom applications. [1]

So what are you going to do about it? What is the best solution to continuously improve software security across development, test, and QA in a way that scales at the same staggering pace of your application innovation?

Invicti Security™, through our products Netsparker® and Acunetix®, delivers a modern, orchestrated DAST+IAST (dynamic+interactive application security testing) solution that works for organizations of all sizes. This unique approach gives your team complete visibility into all of your applications. By feeding accurate vulnerability scanning results directly into existing automation toolchains and processes, you can catch and fix security issues earlier in the development process, shifting security testing left to avoid the cost and disruption of addressing vulnerabilities at later stages.

This guide outlines several key truths:

Traditional pre-release security testing is no longer enough for modern web application development

Modern DAST automates application security testing and integrates it into agile software development workflows

Shifting left with accurate dynamic testing is the only way to build scalable web application security and move towards DevSecOps

Organizations of all sizes can use Invicti products: smaller organizations can get started quickly with Acunetix, and Netsparker brings the scale and integration required in large enterprises

[1] https://downloads.cloudsecurityalliance.org/assets/survey/custom-applications-and-iaas-trends-2017.pdf

Building security into the SDLC

DevOps has become an everyday necessity not only to ensure that operations can keep up with frequent application changes but also to align development to increasingly dynamic cloud deployment models. But as DevOps accelerated, security testing was somewhat left behind. Now, organizations are looking to integrate security more closely with development in a new approach: DevSecOps.

A modern web application is a complex combination of changing parts where you have no control over the vast majority of the codebase.

Rapid development, cloud deployment

Part of the web security challenge lies in the nature of modern web development. Application development is no longer a process with a clear start and end point but rather a continuous cycle of design, implementation, testing, and deployment. Feedback and new or changed business requirements can be incorporated into production in weeks or even days, without going through the lengthy process of waterfall development. Applications are now overwhelmingly built using premade frameworks, components, and libraries, enabling companies to easily adopt new technologies and design trends without slowing down continuous delivery.

But this means that what you write is no longer what you run. The code that is actually processed by the browser has little or nothing to do with the framework-based development code. Under the hood, frameworks and site generators automatically pull in dozens of external dependencies and combine them with templates, external function calls, and custom code. This poses unique challenges for security testing, as developer-controlled code is just a tiny part of the code base and much of it is not directly rendered by the browser but merely used to guide every site generation.

On top of that, you might have dynamic dependencies that are only loaded at runtime, usually from a content delivery network (CDN). A modern web application is a complex combination of changing parts where you have no control over the vast majority of the codebase. Testing just your own code for vulnerabilities can't hope to provide complete coverage – you need to think much bigger.

Cloud deployment brings additional complexity and risk. Modern web applications are dynamic jigsaw puzzles glued together with a thin layer of custom code. Imagine this patchwork being split into dozens of microservices, each running in one of thousands of containers in a public cloud environment. It's a similar story with data storage and access – no longer restricted to a single physical server, your business data can now be spread all over the world (and certainly all over your cloud region). This creates a massive attack surface, exposing the application with its myriad components and sensitive data to global cyberattackers.

Dynamic testing is a vital part of the security testing process as the method that most closely approximates the actions of real-life attackers.

SAST is not enough

In software testing, developers have traditionally focused on static code tests, using linting to catch simple errors and unit testing to check code logic and sanity. If an application or module passes all the code checks, compiles, and runs, it is no longer the developers' concern as QA testers take over to run dynamic testing and report any bugs they find.

The same tried and tested approach was then applied to application security testing, with code-level tools used during development, runtime testing performed during QA, and penetration testing run separately. This is how we arrived at the two major categories of security testing products: static application security testing (SAST) and dynamic application security testing (DAST).

Applying the traditional software testing mindset to web application security brings many problems. First, each stage of software testing checks for correctness at some level: syntax, value sanity, business logic, business requirements, performance, data integrity, user acceptance, etc. It is designed to check if the software works as expected. But security testing is not about correct or incorrect – you are checking if an attacker can compromise the software.

Checking the source code for suspicious constructs and data flows can provide a starting point but comes with the inevitable problem of false positives. Static analysis tools don't know the developer's intent, so many of the warnings they generate will be irrelevant. In any case, source code checking by itself is insufficient: real-life attackers will try to compromise the entire resulting application as built and executed. This makes dynamic testing a vital part of the security testing process as the method that most closely approximates the actions of malicious actors. Organizations may attempt to use manual vulnerability assessment and penetration testing, only to discover how slow and costly this approach is, and how unsuitable it is for a weekly release schedule.

Only extensive automation across all stages of DevOps opened the door to rapid development and deployment in a continuous software development lifecycle (SDLC). The same is true of security testing: it must be highly automated to keep up with the breakneck pace of development. But there are two challenges with automation in security testing: finding tools that are accurate and advanced enough to deliver useful results and can then automate processing and management in a way that fits into a modern SDLC.

Traditional pre-release security testing is not enough

Organizations have scrambled to add security to their agile development workflows. But simply running occasional security testing or even adding manual vulnerability testing at the pre-release stage does little to measurably improve security. You might be able to find vulnerabilities and fix individual issues, but there is no realistic way of getting a large application environment to an acceptable level of security and consistently keeping it there.

Sweeping security issues under the carpet is no longer an option given the massive business risk. If a critical vulnerability is only discovered during pre-release testing, the whole release has to be put on hold while the issue is verified, triaged, fixed, and retested – consuming time and money. Shifting security left to earlier stages of the development pipeline has become a practical necessity to avoid the costs and delays associated with late-stage security testing.

The shortcomings of isolated security testing in agile development

Not automated

Manual issue processing by small security teams becomes a major bottleneck to development and releases. A large organization can have thousands of developers but only a handful of web security staff.

Not integrated

Security teams and developers often use separate tools and workflows. Security engineers need to manually process test results, create tickets, provide feedback, and follow up on fixes.

Not scalable

Agile development and operations environments can grow rapidly to respond to business requirements. Manual security testing and issue management can't scale and adapt to the same level.

Not agile

When vulnerabilities are detected late in the development cycle, it takes far longer to isolate, assign, and fix issues. This makes fixing more laborious and less effective, with a high risk of vulnerabilities resurfacing in the future.

The solution: integrating DAST into development workflows

To be truly effective and agile, modern web application security testing must be fully integrated into the software development lifecycle and cover the entire attack surface of the application as deployed. This means that any application security program must include dynamic testing as that is the only feasible approach to testing everything that an attacker will target.

Complete coverage for all applications

In traditional waterfall-based development, there was a clear-cut division of labor in application security: static testing (SAST) and other code-level checks were done during development, dynamic testing (DAST) was done in QA and staging. When it came to building security into the software development lifecycle, SAST was initially the natural choice as it was relatively easy to add into existing build scripts and test suites and could (at least in theory) provide full source code coverage. However, actually implementing static testing in the context of modern web development can be challenging and the results frequently need a lot of tweaking to minimize false positives.

But checking your own source code is not enough to provide full coverage in modern web applications. Even if you add software composition analysis (SCA) to ensure that known vulnerable dependencies are not brought in, some form of dynamic testing is still essential to test the final application as built and deployed, including all modifications, component interactions, and dynamic dependencies. At the same time, leaving dynamic testing to be done by security specialists in the late stages of development is inefficient and doesn't scale.

The obvious solution to both problems is to introduce dynamic testing early on in the development cycle. Modern web development workflows including CI/CD (continuous integration/continuous deployment) pipelines already make it possible to introduce DAST from the very start.

The importance of dynamic testing

Apart from its core purpose of catching runtime issues, dynamic security testing is essential for web applications because it reveals and probes the actual attack surface of the application as attackers see it. This includes not only your own application code but also all external dependencies and components brought into the application, whether directly in your code or in the underlying framework. You can test security without worrying about source code access or even knowing all your code repositories. In fact, you can even test dynamic dependencies that are only brought in at runtime and legacy or third-party components that you simply don't have the source code for.

All this gives you maximum visibility and also makes DAST extremely fast and easy to set up, run, and maintain – you just point the tool at a URL and start scanning, regardless of the application architecture or language. Even if the underlying language changes, the same dynamic scanner will still work without having to buy and configure new tools. Apart from uncovering exploitable vulnerabilities, modern dynamic scanners can also discover web assets to scan, return information about security misconfigurations, detect outdated software versions, and recommend security best practices to build a defense in depth.

Leaving dynamic testing to be done by security specialists in the late stages of development is inefficient and doesn't scale.

Debunking DAST myths

Early dynamic scanners were not standalone solutions but rather simple utilities to support the work of penetration testers. The first DAST products based on these simple tools had many limitations, partly due to the simplicity of the mostly static websites they were designed to scan. As web technologies advanced and web applications became ever more complex, the limitations of legacy DAST became obvious, leading to the lingering misconception that DAST can only handle simple static sites and find trivial issues. Modern DAST has come a long way from these rudimentary tools.

A quality DAST solution can accurately scan any modern web application, including JavaScript-heavy single-page applications (SPAs). It can handle automated authentication to ensure that high-value pages hidden behind login forms are also tested. It provides advanced crawling and discovery to detect and scan as many web assets as possible. And perhaps most importantly, especially in enterprise environments, it can be integrated into the SDLC to automate and streamline application security testing.

A quality DAST solution can accurately scan any modern web application, including JavaScript-heavy SPAs.

Prerequisites for integrating DAST into the SDLC

Trustworthy

To automate security testing, DAST needs to give you confidence that you are not automating false positive results.

Accurate

To get measurable security improvements, DAST must be advanced enough to find actionable issues and provide fix guidance.

Comprehensive

To ensure maximum test coverage and quick detection, DAST needs to scan every corner of a modern web application (including authenticated scanning).

Workflow-ready

To minimize security testing overhead and maximize adoption, DAST needs to integrate with existing development and collaboration tools.

Flexible

To be effective and adaptable in modern web environments, DAST needs to work with multiple web technologies and languages out-of-the-box.

The benefits of using a modern DAST solution in your SDLC

Invicti Security, the maker of industry-leading DAST solutions Netsparker and Acunetix specializes in solutions that **embrace automation** to achieve the scale needed for today's world.

The key to scaling a web security program is having confidence in the scan results. Bad experiences have made security professionals hypersensitive to false positives, and as a result, teams have come to believe (falsely) that all identified issues must first be manually checked and confirmed before they can be fixed.

Invicti has solved this problem and saved security teams untold hours with our proprietary approach to proving vulnerabilities at the time of the scan. Netsparker and Acunetix both automatically detect a wide array of web vulnerabilities as well as prove or confirm many common issues. This is combined with deep runtime insights from an additional IAST module to isolate issues with pinpoint accuracy and extend testing to assets that are inaccessible to crawlers. To ensure that all these capabilities bring real value, Netsparker and Acunetix also provide out-of-the-box integration with popular issue trackers and automation tools to minimize the time and effort required to deploy a working solution.

Application security testing at the speed of your SDLC

Improved security in the long run

Real value and tangible savings

Application security testing at the speed of your SDLC

To be an effective part of an agile DevOps pipeline, security testing must be automated as far as possible. In order to feed automated vulnerability scanning results into development workflows, you have to be confident that you are not raising false alarms.

This is the biggest benefit of Invicti's ability to confirm vulnerabilities: results are real issues that can go straight into the developers' issue tracker without the burden of manual confirmation and triage. In many cases, it is even possible to assign fix tasks directly to the developer who made the vulnerable commit and go from detection to bug ticket in a matter of minutes. This is a massive time-saver that also eliminates the inefficient (and annoying) practice of having to fix other people's code. This opens the way to true scale, even across hundreds or thousands of websites, applications, and services. Modern web application security requires automation of everything that can possibly be automated, simply to avoid an eternal backlog of issues that just keeps growing.

Invicti's products combine efficient and accurate vulnerability scanning with confident automation and deep SDLC integration, so your security testing can keep pace with development.

With Invicti's ability to confirm vulnerabilities, results are real issues that can go straight into the developers' bug tracker.

Improved security in the long run

Web applications keep growing in size and complexity and multiplying at an astounding rate. If you are serious about security, you face both the daunting task of finding and eliminating vulnerabilities in your existing application environment and the challenge of maintaining a good security posture. The only way to do this is to build application security directly into your SDLC and shift as much work as possible away from small security teams and towards large developer teams.

Invicti's unique combination of integration, automation, and fully trusted results with vulnerability confirmation allows you to build DAST into your development pipeline to eliminate vulnerabilities before they can reach production. Because developers get accurate feedback in real-time about the security bugs they introduce, they can not only fix them quickly but also avoid making the same mistakes in the future. This fosters a security-focused mindset among developers to improve application security in the long run.

Automated application security testing also alleviates internal friction and inefficiencies between security and development around reporting and resolving security issues. When developers get proven security bug reports from an accurate tool delivered directly into their favorite ticketing system, the often adversarial attitude to working with security teams changes to one of efficient collaboration. In more mature organizations, it may even be possible to handle all application security issues at the development team level, so the core security team can focus on high-level research, vulnerability management, and policy.

You need to build application security directly into your SDLC and shift as much work as possible away from small security teams and towards large developer teams.

Real value and tangible savings

The time-to-value calculation for an application security product starts from the moment of purchase to the first measurable security improvements. Invicti's products combine the advantages of DAST technology, such as ease of deployment and broad scope of testing, with their own unique benefits and IAST capabilities to deliver value within days of deployment.

By automating manual processes and eliminating inefficiencies in team collaboration, you are likely to find that deploying a modern DAST solution will reduce the cost of your application security program while increasing its effectiveness. Each automatically confirmed vulnerability means one less manual verification task for a security engineer, allowing your security staff to focus on vulnerability management, security education, and analyzing complex vulnerabilities that really need human expertise. All this translates to fewer man-hours spent on tasks that could be automated, improved security, and more satisfied employees.

Deploying Netsparker or Acunetix will reduce the cost of your application security program while increasing its effectiveness.

Summary

As web applications grow ever more complex and opaque, it is becoming clear that comprehensive dynamic testing is an indispensable part of any application security program. At the same time, effective security testing must be fully integrated with agile development workflows that are now the norm in web application development – and legacy DAST products intended for manual scanning just won't cut it. To combine these two pressing needs, you need a quality DAST solution that can integrate deeply into the SDLC. Invicti Security can help organizations of all sizes and maturity levels achieve continuous application security automation with our products, Netsparker and Acunetix. Fueled by over a decade of relentless security research and development dedicated to dynamic testing, our products combine accurate and actionable vulnerability scanning results with embedded interactive testing modules and extensive integration and automation capabilities.

The result is a powerful and flexible DAST+IAST solution that can streamline and automate application security testing across the development and testing workflow: security at the speed of your SDLC.

About Invicti Security

Invicti Security is changing the way that web applications are secured. A global leader in web application security for more than 15 years, Invicti's dynamic and interactive application security products help organizations in every industry scale their overall security operations, make the best use of their security resources, and engage developers in helping to improve their overall security posture. Invicti's product Netsparker delivers industry-leading enterprise web application security, while Acunetix is designed for small and medium-sized companies. Headquartered in Austin, Texas, Invicti serves organizations all around the world.

invicti.com

linkedin.com/company/invicti-security

facebook.com/invicti-security

twitter.com/invicti-security