WHITE PAPER

False Positives in Web Application Security

FACING THE CHALLENGE

Netsparker

Invicti

Contents

Executive summary

To keep up with the fast pace of modern web application development, vulnerability testing requires automated tools to assist in finding vulnerabilities. Unfortunately, apart from legitimate vulnerabilities, automated scanners can also report false alarms, or false positives, which must be further investigated manually just like real vulnerabilities. As systems and applications grow, the number of false positives can rapidly increase and place a serious burden on developers and security teams, with negative consequences for the development process, application security, and business results.

This whitepaper examines the impact of false positives across the software development lifecycle, suggests ways of eliminating false alarms in vulnerability scanning, and shows the business benefits that can be achieved.

Introduction: The everyday reality of web application security testing

In today's fast-paced development environments, web applications are updated on a daily basis, and agile, integrated methodologies such as DevOps are fast becoming the norm. Development teams use highly automated processes to create, test, and modify multiple applications and services, often making extensive use of ready application frameworks and open source libraries.

Development, operations, and security teams are under constant pressure to deliver more with less in a constantly changing threat environment.

Rapid development presents a serious challenge for application security testing. Manual testing is too slow, expensive, and often impractical across multiple applications. Automated scanners integrated into the software development lifecycle (SDLC) are a practical necessity, but bring their own challenges. In particular, the false alarms (or false positives) generated by many such tools can have a serious impact on the development process, application security, and business outcomes.

Your development, operations, and security teams are under constant pressure to deliver more with less in a constantly changing threat environment. Automated tools integrated into the SDLC are critical to their success, so they must be reliable, efficient, and trustworthy. This is no place for false positives.

The challenge: False positives in vulnerability scanning

There are two main types of vulnerability scan errors: false negatives, where the results don't include an existing vulnerability, and false positives, where the scanner indicates non-existent security issues. False negatives have a direct impact on security, because undetected vulnerabilities can't be fixed. False positives, on the other hand, can have serious consequences not just for security, but all across the organization.

In web application vulnerability scanning, false positives can be a real deal-breaker.

Modern web application development relies heavily on integration and automation, especially for approaches such as CI/CD and DevOps. This means that security testing must also be integrated into the development pipeline and automated as much as possible to ensure that issues are detected quickly and test results are efficiently communicated back to the developers. False positives in scan results introduce unnecessary additional work into the highly automated development pipeline and undermine the entire development process.

The purpose of automated vulnerability scanning is to ensure more effective security testing than with manual methods, especially as the number of applications and updates grows. However, if the number of false positives reported by an automated solution becomes unmanageable at scale, organizations might, for example, limit vulnerability scanning only to their top-priority applications – in effect negating the benefits of using an automated solution in the first place. In web application vulnerability scanning, false positives can be a real deal-breaker.

Static application security testing
(SAST, or white-box testing)

Uses source code analysis tools to find vulnerable constructs and analyze data flows at the development stage. Static tools can prevent vulnerabilities from ever leaving development and can precisely indicate flaws in the source code. However, they can be hard to configure and integrate, require access to source code, and cannot detect runtime issues such as misconfigurations. Crucially, they tend to report many false positives, which may limit their usefulness.

Dynamic application security testing
(DAST, or black-box testing)

Checks for flaws and vulnerabilities in a running application, reflecting real-life use, misuse, and attack attempts. Dynamic tools can detect all sorts of runtime issues, including system misconfigurations and flaws in external dependencies, but can't directly indicate errors in the code. While modern DAST solutions generally report fewer false positives than static tools, false alarms can still be a problem, and eliminating them is only possible with advanced confirmation technologies.

Consequences for the development process

Scalability is a major concern for any growing organization, and scaling up development processes can bring many challenges. Small-scale development often relies on manual processes and ad hoc toolkits, which can initially work well and remain manageable, even if the tools used for testing report too many false positives. However, as the number of updates and products grows and workloads increase, the number of false positives can grow exponentially, and manually dealing with each false alarm becomes impractical. When you start adding automation at scale, even occasional false positives can force the security team to manually screen all results, negating the performance benefits of automation.

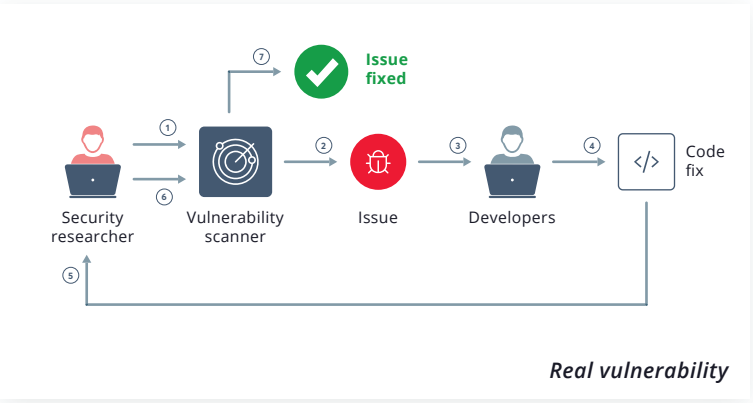
Dealing with real vulnerabilities	Dealing with false positives
■ The vulnerability scanner reports an issue	■ The vulnerability scanner reports an issue
■ Developers fix the issue	■ Developers try to locate the issue and eventually give up
■ The application is re-scanned to ensure the issue is gone	■ Developers dismiss the issue as a false alarm
■ RESULT: The issue has been provably resolved	■ The vulnerability scanner still reports the issue
	■ The security team sends the issue back to developers
	■ Developers have to convince the security team that it was a false alarm
	■ The vulnerability scanner continues to report similar issues in the future
	■ RESULT: The issue is not provably resolved, developers and security staff have wasted time, and now everyone distrusts the scanner

These scalability problems are compounded by the fact that dealing with a false positive can actually take longer than resolving a real vulnerability. This is because real issues are testable: you can test for the suspected vulnerability, fix it, test the fix, and have documented proof that the issue has been resolved. However, when dealing with a false positive, a lot more testing can be necessary until the developer decides that it's a false alarm. Crucially, someone has to take personal responsibility for ruling against the scanner and signing off code where potentially serious issues have been flagged as false alarms.

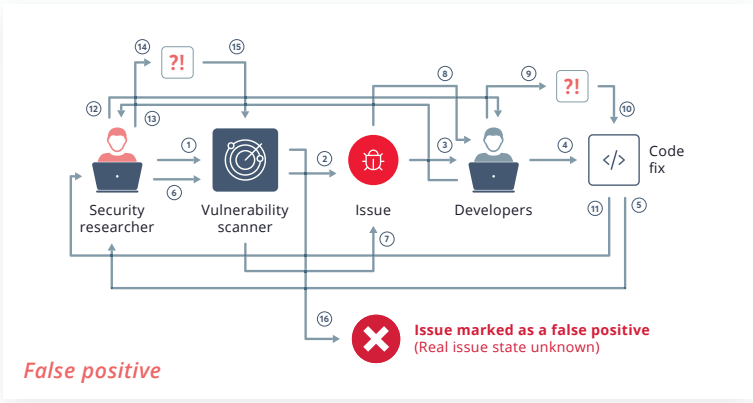
In an agile development environment, automation is king – and manual security processes are not a feasible option at scale. DevOps and CI/CD teams rely on their automated tools to do the legwork so they can focus on tasks that require the creativity and problem-solving skills of highly qualified specialists. False positives in vulnerability testing can force testers and developers to put their streamlined automated processes on hold and laboriously review each false alarm just like a real vulnerability.

> "Automation without accuracy cannot scale"
>
> *Ferruh Mavituna, CEO, Netsparker*

False positives can also be detrimental to team dynamics. Every time the security team reports a vulnerability, the developers have extra work investigating and fixing the issue, so reliability and mutual trust are crucial to maintaining good relations. This makes false alarms particularly aggravating, and if the vulnerability scan results burden the developers with unnecessary workloads, the working relationship may quickly turn sour. The dev team may start treating the security people as irritating timewasters, leading to an "us vs. them" mentality – with disastrous consequences for collaboration and the entire software development lifecycle.



Real vulnerability



False positive

Impact on application security

Apart from the burden they place on the development process, false positives can also directly affect application security. As developers and testers lose confidence in a vulnerability scanner that generates mostly false alarms, they might start routinely ignoring whole classes of issues from this tool. After all, each vulnerability report means extra work, so if, say, 2 out of 3 issues reported by a certain tool are false positives, human nature dictates that sooner or later someone will start ticking boxes just to make the errors go away – especially considering that every single false alarm is a huge problem at scale. Worse still, that one remaining issue might be a critical vulnerability that goes unnoticed in the flood of false positives and makes it into production, or is caught and fixed at far greater cost during later manual testing.

If this goes on long enough, developers and testers may become desensitized to vulnerability reports, causing the overall security culture to deteriorate. This goes back to the issue of trust: if security reports mostly cause unnecessary work due to false positives, developers may become wary not just of specific tools, but also of any security issues in general. Just as SecDevOps and similar approaches are being introduced to foster a security-first mindset, security solutions that flood developers with false alarms might undo all these efforts, relegating security issues to the backbench in development pipelines.

If 2 out of 3 reported issues are false positives, human nature dictates that sooner or later someone will start ticking boxes just to make the errors go away.

Time to resolution is another vital aspect of security that is impacted by false positives. Modern web applications can be updated several times a day, and each modification could potentially introduce new vulnerabilities. To maintain system and data security, vulnerabilities in production applications must be detected, confirmed, triaged, and addressed with maximum efficiency. Again, false positives in vulnerability reports can be a serious headache for security teams and developers, who must go through false alarms before addressing real, exploitable vulnerabilities. Apart from the cost and frustration of extra work, this increases the time to resolve actual vulnerabilities and leaves production applications vulnerable for longer than absolutely necessary.

Financial consequences

We have already seen that false positives in vulnerability scanning can have serious consequences for application security and the development process, but the financial side is equally important. In business, delays due to unexpected problems have a measurable financial impact, so false positives can really hurt your bottom line.

For many if not most organizations, employee salaries and contractor fees are a major cost component, and extra work means more expenses. Dealing with false positives can be especially costly, as investigating a false positive can take longer than fixing a real issue. This waste of productivity is compounded by the fact that if developers are chasing false alarms, they are not generating business value – or fixing real vulnerabilities.

With the risk and impact of cyberattacks growing from year to year, leaving avoidable vulnerabilities in your software is a recipe for disaster.

Timely delivery of products and features is crucial for the business success of any development operation. However, when security testers and developers are spending too much time investigating false positives, delays can easily creep in, with very real financial consequences. Features that are still awaiting implementation can't bring in new revenues, and project schedule overruns may mean lost business opportunities as staff are unable to take on new work. Just as importantly, missed client deadlines are bad for repeat business.

If staff gets used to waving away vulnerability reports because they are most likely false alarms, real vulnerabilities may sometimes slip through and make it into the production application. With the risk and impact of cyberattacks growing from year to year, leaving avoidable vulnerabilities in your software is a recipe for disaster, whether developing software for internal use or for paying clients. Data breaches, system outages, data loss, malware infections – all can be very costly in terms of time, money, and reputation.

Global cyberattacks in 2018

>$45bn	**2,000,000**	**95%**
financial losses due to cyberincidents	cyberattacks worldwide (est.)	of breaches could have been prevented

Organizations that see security as an investment rather than just another expense will be interested in the return on investment (ROI) from their vulnerability scanning solution. Integrating an enterprise-class web vulnerability scanner into the development and operations processes can bring measurable savings due to increased efficiencies all across the pipeline. However, tools that return too many false positives can eat away at these benefits by adding unnecessary man-hours to the company payroll. From a wider financial perspective, false positives can quite simply reduce the return on investment in security tools.

The solution: Proof-Based Scanning™

So far, we've seen that false positives can be a lot more than an inconvenience, and if they get out of hand, they can seriously affect the entire development pipeline. But how do you get rid of them? Let's step back and think about this. A false positive is reported when a tool mistakenly suspects a certain kind of vulnerability. To go from suspicion to certainty, you need proof that the vulnerability really exists – and can be exploited. So why not create a solution that can deliver this proof by automatically exploiting suspected vulnerabilities? Well, this is exactly what Netsparker by Invicti's Proof-Based Scanning™ technology does.

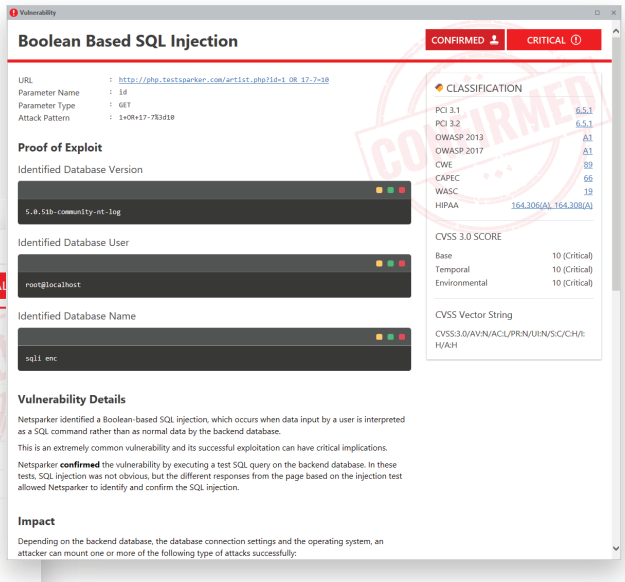
If a vulnerability can be exploited, it is not a false positive.

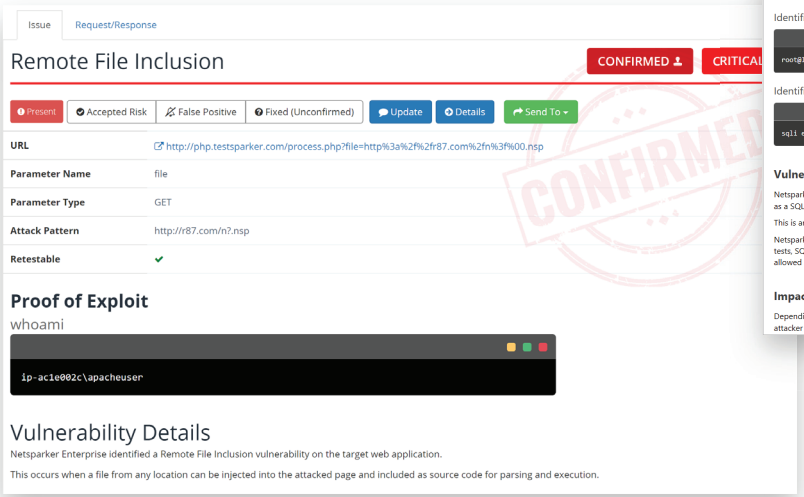
Proof-Based Scanning™ is based on a fundamental insight: if a vulnerability can be exploited, it is not a false positive. Combined with a meticulously developed and continuously maintained auto-exploitation engine, this simple idea has allowed Invicti's engineers to create an industry-leading vulnerability scanning solution that provides positive proof of exploitable vulnerabilities to deliver accurate and actionable reports.

When Netsparker finds a vulnerability, it attempts to automatically and safely exploit the flaw to make sure it is not a false positive. If the vulnerability is exploitable, the scanner generates a proof of exploit (extracted sample data) or a proof of concept (exploit code used in the test attack). Both types provide hard-and-fast evidence that the reported vulnerability is not a false alarm and can aid developers in locating the underlying issue. Especially with proofs of concept, developers can use the actual exploit code to quickly and effectively pinpoint the vulnerability.

Netsparker can generate proofs of exploit for:

- SQL Injection
- Boolean SQL Injection
- Blind SQL Injection
- XML External Entity (XXE) Injection
- Remote Code Execution via Local File Inclusion
- Remote File Inclusion (RFI)
- Command Injection
- Blind Command Injection
- Remote Code Evaluation
- Local File Inclusion (LFI)
- Server-Side Template Injection





Proof of exploit

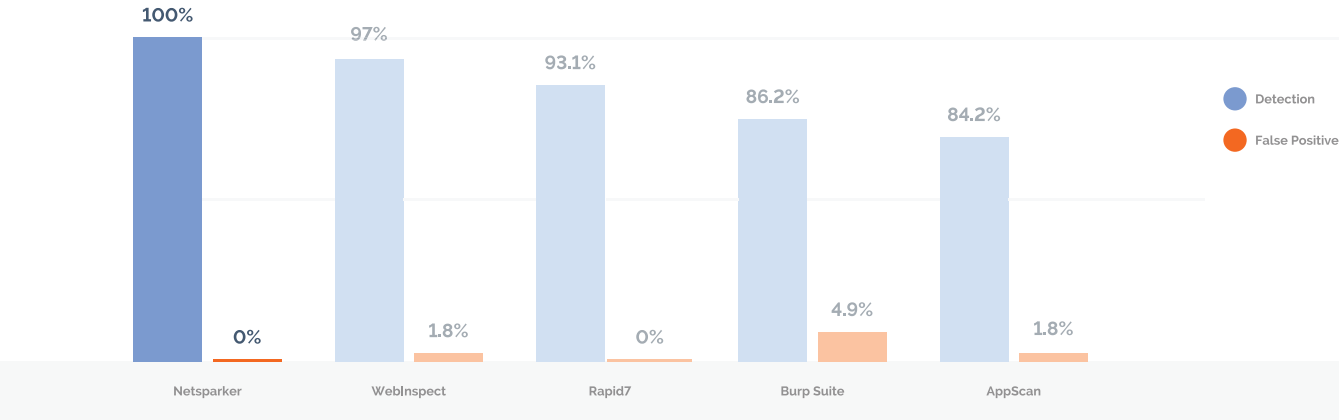
Sample data extracted from the target application after exploitation. This is collected in a **safe**, **read-only way**, without affecting the vulnerable system. For example, after discovering and exploiting an SQL injection vulnerability, Netsparker may extract and report information about the database server version and configuration. This provides **direct proof** that a system is open to attack.

Proof of concept

Actual **exploit code** used to confirm the vulnerability. For example, after exploiting a cross-site scripting (XSS) vulnerability, Netsparker will report the payload that was used to inject code. Apart from providing **evidence of the vulnerability**, a proof of concept can also help developers isolate the exact issue that made exploitation possible.

2018 Vulnerability scanner comparison

Independent benchmark results have shown Netsparker to be the only tested solution that detected all vulnerabilities in the test set with zero false positives.



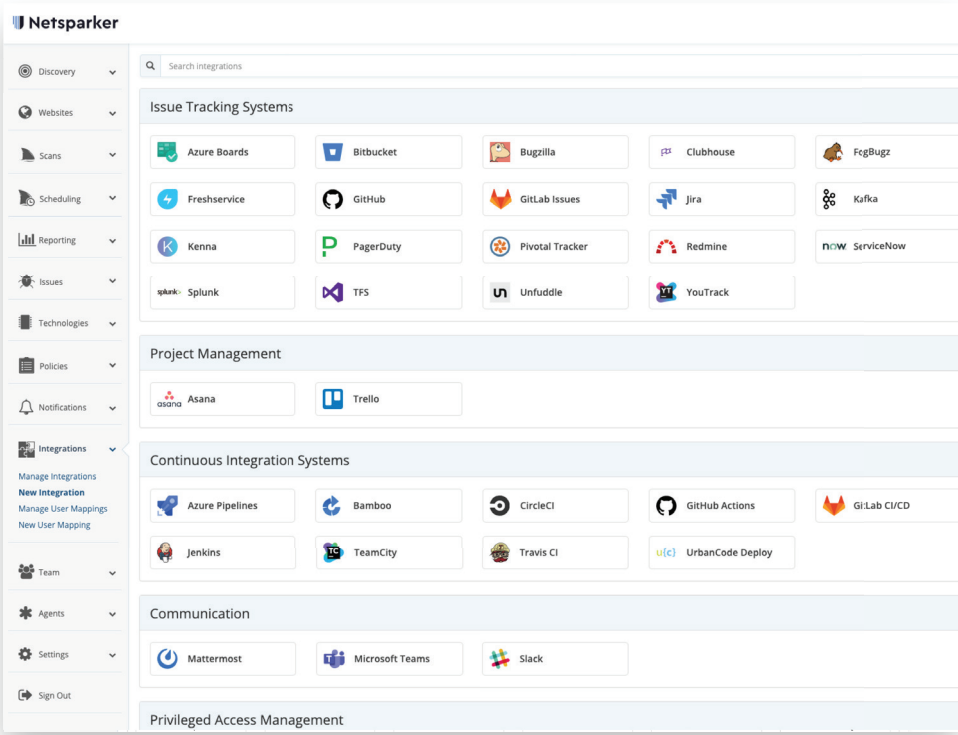
Benefits for the development process

We've already seen that accuracy is crucial to reap the benefits of automation and effectively scale tightly integrated development processes such as DevOps. By clearly indicating verified and actionable issues, vulnerability scanners can finally live up to these expectations for web application security. Thanks to enterprise-class solutions such as Netsparker, teams can seamlessly integrate vulnerability scanning into their automated workflows without the drag of excessive false positives.

Accurate and trustworthy vulnerability scan results help organizations truly integrate security into the software development process.

With vulnerability management that is truly accurate, automated, and reliable, developers can focus on fixing verified vulnerabilities instead of poring over lists of suspected issues and running (and re-running) manual checks. Actionable evidence of vulnerabilities obtained using Proof-Based Scanning™ helps developers to quickly locate and resolve issues, resulting in improved productivity and less frustration. For maximum effectiveness, vulnerability notifications can even be integrated into many popular issue tracking systems.

Largely due to problems with false positives, many organizations require the security team to check vulnerability scan results before assigning issues to developers. When a small security team has to deal with hundreds of web applications, this additional step can create a bottleneck that delays the resolution of critical issues. However, when scan results are trusted to be free of false alarms, proven vulnerabilities can be automatically assigned directly to developers in their issue tracker. That way critical vulnerabilities are addressed more quickly while also decreasing the workload of security teams.



In terms of the overall security culture, accurate and trustworthy scan results help organizations truly integrate security into their automated software development process. Evidence provided by auto-exploitation technology can reduce the back-and-forth between developers and security professionals trying to convince them that a vulnerability is real, increasing efficiency and improving team relations. The same evidence can be invaluable for effective, fact-based time and resource allocation within teams when additional man-hours need to be signed off.

Security benefits

Accurate scan results from fully integrated tools bring clear and immediate benefits for application security. When developers receive an automatic vulnerability notification, they can quickly get to work fixing it without wondering if it's just another false positive. With trustworthy automatic assessments of severity and exploitation potential, critical risks can be addressed immediately and accurately to patch applications as soon as possible.

When testers and developers don't have to sift through multiple false positives to confirm and triage actual issues, they have more time to resolve real vulnerabilities, resulting in better quality fixes and improved application security. Just as importantly, there is less risk of vulnerabilities going unnoticed in a flood of false alarms and slipping into production. Trustworthy scan results are also treated more seriously, so all suspected issues are given the attention they deserve.

For vulnerabilities that can't be patched immediately, web application firewall (WAF) rules can be configured to temporarily block attack attempts. Netsparker makes this easy through integration with leading web application firewalls. Confirmed vulnerability scan results can be automatically applied as real-time WAF patches or exported as WAF rules ready to apply manually. This allows organizations to effectively manage vulnerabilities even when fixing them is not immediately possible.

Netsparker can export scan results as rules for popular web application firewalls (WAFs):

- ☑ ModSecurity
- ☑ F5 Big-IP
- ☑ Imperva
- ☑ FortiWeb
- ☑ AWS
- ☑ Cloudflare

Business benefits

Increased confidence in scan results boosts efficiency across the development process and the entire organization, bringing tangible financial benefits. Savings start with reducing labor costs on multiple levels. With more accurate automatic scan results, staff can spend less time sifting through false alarms and manually confirming issues. Developers can focus on working with the code to quickly resolve verified vulnerabilities and do what they do best: build functionality that adds business value.

With Proof-Based Scanning™, exploitable vulnerabilities are automatically confirmed by the scanner, so security professionals don't have to manually reproduce the results to confirm them. This reduces the number of issues that require the attention of specialist technical staff, bringing further cost savings. Increased efficiency across the software development lifecycle can also mean faster and more predictable product delivery, helping projects stay within time and budget constraints.

Choosing a vulnerability scanner that consistently delivers accurate results and clearly indicates verified and actionable vulnerabilities can make the difference between buying just another tool and investing in the future success of your organization.

Of course, improved application security also brings its own business benefits, starting with the reduced overall cost of vulnerability management. Better security means a lower risk of attacks, with all their attendant dangers: data breaches, system outages, data loss, regulatory liability, and so forth. Minimizing high-profile incidents by maintaining solid application security and rapidly resolving critical issues also helps organizations maintain a good reputation and ensure client satisfaction.

Looking at the big picture, all this contributes to the return on investment in your vulnerability scanning solution. By eliminating uncertainty from scan results, you can confidently apply automation to streamline the entire software development process, reduce risk, and achieve cost reductions across the organization. More efficient development and testing also mean a shorter time to market – and to profit. Choosing a product that consistently delivers accurate results and clearly indicates verified and actionable vulnerabilities can make the difference between buying just another tool and investing in the future success of your organization.

Invicti

About Invicti Security

Invicti Security is changing the way 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.

Netsparker

Netsparker's technology has been the best performing in vulnerability detection and accuracy from third party benchmarks. World leading and renowned businesses such as Samsung, NASA, Microsoft and Siemens trust Netsparker to ensure the security of their web applications, web services, and web APIs.