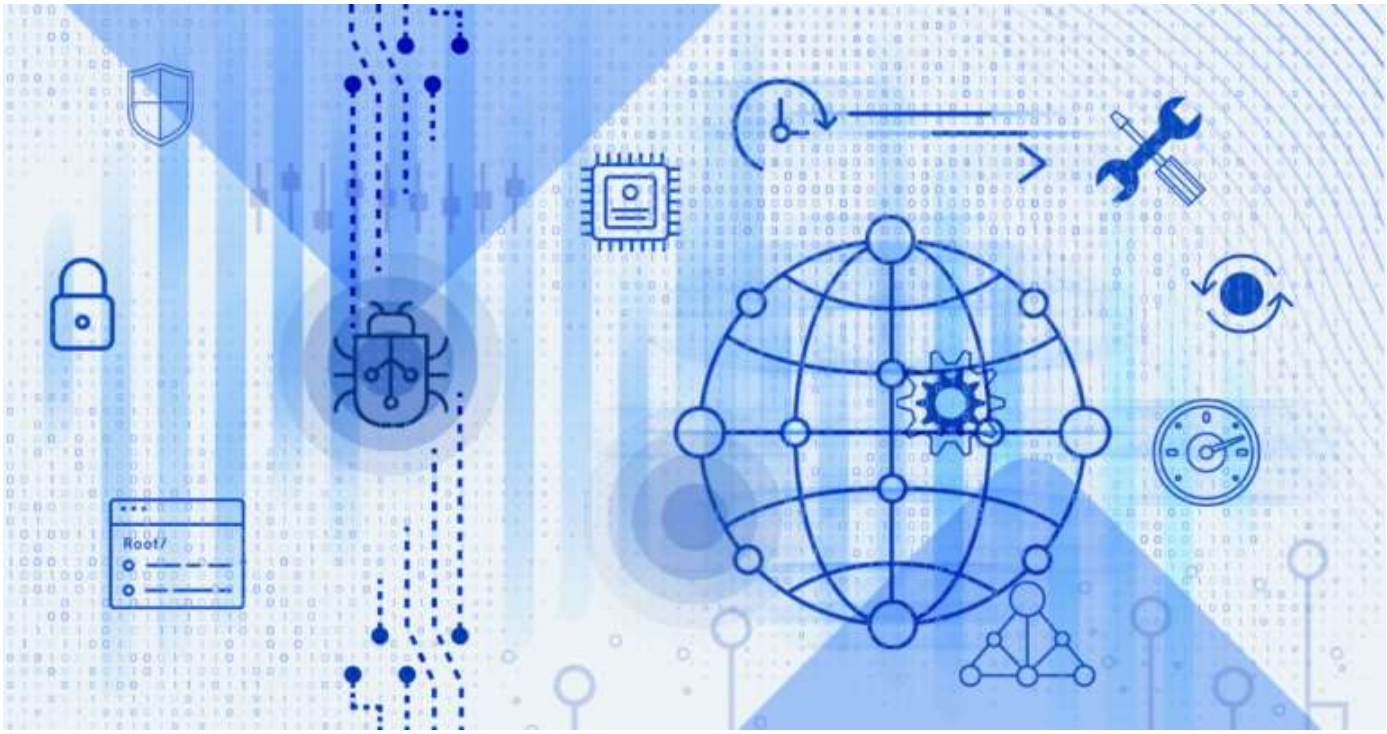

7 reasons why development teams skip security steps

Tomasz Andrzej Nidecki - Thu, 26 Jan 2023 -

Studies confirm that bypassing security during application development is the rule rather than the exception – but why? Learn to recognize common signs that your organization isn't doing everything it should to support secure software development.

Back in 2021, the Invicti Fall AppSec Indicator revealed that a full **70% of development teams skip security steps**. To double-check this incredibly high number and also to see if things are improving, we asked a similar question last year for our 2022 AppSec Indicator. This time, we found that **74% of companies frequently or routinely release software with unaddressed vulnerabilities**, confirming that application security can't keep up. But why is that? Here are seven potential reasons that you should explore as a business leader, along with ideas on how to eliminate these causes as quickly and efficiently as possible.

Reason #1: Management not leading by example

The primary reason why development teams may skip security steps is if they do not think these steps are important enough. And to learn what is important for your company, you look up. If business leaders don't drive for strong web application security, developers and their direct managers are less likely to take care of it all by themselves. More often than not, they already have too much work and too few resources to go out of their way to worry about keeping their web applications properly secured.

There are different ways to demonstrate your focus on web application security, but the first step is to recognize that it exists and is fundamentally different from other areas of cybersecurity. Starting from the budgeting level, if your business invests heavily in endpoint security solutions like antiviruses and network security solutions such as firewalls but makes zero investment in web application security, that's exactly the priority you are communicating to your development teams.

To rectify this:

- If you are a business leader, emphasize your commitment to web application security.
- Make sure both you and your teams are aware that efforts around endpoint and

- Ensure you are using good quality automated security tools to help secure your websites and web applications.

Reason #2: Assuming that web application security is not your problem

While many business leaders are already aware of web application security being important and distinct from endpoint/network security, they may also feel that their web applications don't really need any internal security efforts. Here are a few common misconceptions:

- **Our application is built externally, so the application provider takes care of web application security.** If you don't develop web applications internally but outsource development or use an open-source product like WordPress, you may assume that the creators ensure sufficient web application security. In reality, you can never be sure.
- **Our application is only for internal or restricted use.** If you allow only internal network access to an application or restrict the IPs that are allowed to access it, you may think that your application is safe. It is not – attackers can still get at the application through compromised user accounts, misconfigured servers, or vulnerabilities in other systems.
- **Our website/application is not at risk because it is simple.** This is a common assumption especially for marketing sites and other websites that seemingly hold no sensitive data. Yet even such sites, if vulnerable, may provide database access or allow privilege escalation to help attackers access other, more critical systems.

Once again, without a top-down mandate to take application security seriously, development teams don't have the time or the tools to systematically take care of web application security, and the best you can hope for is a few security-conscious developers who will painstakingly try to address any issues on their own. And while you can, of course, prioritize securing some applications over others, a systematic security program must not skip any of them.

- Take responsibility for all your web application security, even if you believe you can trust the security practices of a third party. You can invest in your own tools or hire an MSSP to handle security for you.
- Treat every single web application and website as part of your organization's attack surface – even internal applications and even the simplest websites.

Reason #3: Lack of time and resources

Even business leaders who are aware of the importance of web application security and ready to invest in suitable tools may still not realize that ensuring web application security requires some extra time and resources. For example, writing a simple web form might take a few seconds. Writing the same form with effective cross-site scripting protection is likely to take longer – at least a few minutes, especially if secure coding practices are not a routine part of development. Whenever developers are under pressure, they simply might not have the time to do something the slower but safer way.

It's the same with tooling. To take the example of **dynamic application security testing (DAST)**, modern DAST tools don't hog the pipelines or cause bottlenecks in the SDLC, but the scanning and resolution process can still add some overhead on top of existing QA testing time, especially at the beginning. Business leaders must be aware that to maintain web application security, development teams need extra time and effort – and they must make sure that developers feel comfortable taking this time and not rushing. If time constraints force your developers to rush and copy-paste snippets from StackOverflow without checking, you can only dream of web application security.

To rectify this:

- Allow for slight development delays, especially while your teams are still getting used to focusing on web application security.
- Make it clear to your development teams that they are allowed and expected to take extra time to write secure applications rather than rush a release by quickly

Reason #4: Security and development silos

Many businesses automatically assume that anything with “security” in the name is handled by a dedicated security team – development teams do development, so security teams should do security, right? While this may be true for endpoint, network, or general IT security, it will not work for web application security, where developers play a crucial role.

When application security is treated as another responsibility for the cybersecurity team, businesses often end up with a security team that works completely separately from development. Assuming the security team deals with web application security at all, they do it only at late stages – staging, pre-production, or even live production only. This leads to web application security issues being found and eventually fixed weeks or months after they were introduced, with nobody sure what exactly happened and how to stop it from happening again.

To rectify this:

- Make sure that your cybersecurity and development teams don’t work in isolation. Web application security experts should work directly with or even within development teams to investigate and remediate issues as early as possible.
- Put development teams in charge of fixing security problems discovered early on in the software development lifecycle by automated and integrated tools such as DAST. Security teams should be involved only in the setup and maintenance of suitable tools, not in their everyday use.

Reason #5: Insufficient education in application security

Many developers have formal education covering many aspects of information technology. However, many development courses, even at the university level, don’t sufficiently cover web application security, which is partly due to the cybersecurity skills gap. Even businesses find it hard to attract and retain cybersecurity experts. At

With few security experts among the faculty, many schools turn out developers who have been taught everything *except* application security.

Business leaders should definitely emphasize the importance of web application security but also be aware that inexperienced junior developers (and not only they) might feel anxious about their inadequate security skills. Beyond hearing that they are expected to pay attention to security, they also need to get the necessary tools and educational resources – and the reassurance that it's all a learning process. Again, this involves senior developers as well as cybersecurity teams working together with junior developers to provide practical instruction on avoiding security vulnerabilities in their code.

To rectify this:

- Make sure that junior developers are not penalized for any gaps in their security knowledge or skills and that they are not afraid to ask for help.
- Foster security champions in your development teams, giving them the time and tools they need to teach others how to avoid and fix web application security vulnerabilities.
- Use existing web resources to educate your teams about typical vulnerabilities and ways of fixing and avoiding them. Invicti Learn is a good starting point.

Reason #6: Having to fix someone else's mistakes

Even if you are doing your best to foster security education for developers, improvements can be slow if you don't have the tools and processes to provide rapid feedback. In a typical environment where web application security testing is only performed in late stages, such as staging, pre-production, or even live production, developers have no way of knowing whether they've committed vulnerable code. Security issues can surface weeks after they are introduced and will often be assigned to someone else to fix, which is inefficient. If fact, even if the original developer gets the ticket, they are unlikely to remember the details of code from weeks ago.

In an ideal environment, code should be tested for security vulnerabilities at the same time as it is being tested for functional correctness, or even earlier if possible. In practice, this means that dynamic security testing should be performed along with QA testing, so your security tool should run right before or right after your Selenium tests (or whatever other test automation you use).

To rectify this:

- Integrate your application security testing tools, especially your vulnerability scanner, with your CI/CD environment so they are a routine part of the development pipeline.
- Run web application security scans at multiple stages – not just for releases but for every new commit that goes into your master branch.

Reason #7: Frustrations due to inadequate tooling

Let's say you've already addressed all the issues listed above and integrated security testing tools into your development pipeline but find your teams are still skipping security steps. If so, the problem may lie with the tools themselves, whether it's the type of tool, level of workflow integration, or simply low-quality results. Unless the tools are efficient and accurate, developers may still sidestep security simply to avoid a tool that makes their work harder, not easier.

For a developer, false positives are the main source of frustration and time-wasting. Whatever the tool is, if the majority of alerts are false positives, developers will first waste time chasing a non-existent problem and then start ignoring similar alerts. Investigating a false positive may mean ten times more work than fixing a real vulnerability, so your developers are likely to start cutting corners as soon as they lose trust in the tool.

To rectify this:

- Don't rely only on SAST tools alone, as they are prone to raising lots of false

- Invest in tools designed to help you keep false positives under control. For example, Invicti uses Proof-Based Scanning technology to automatically confirm the majority of common vulnerabilities, letting your developers know that reported issues are real and need to be addressed – without the risk of wasting time.

Effective application security starts at the top

Whenever you find your development teams skipping security steps, you are likely to find some of the above symptoms in your organization. Any one of them can make delivering secure applications an unrealistic requirement for your teams. To remedy the situation and make secure software development your new normal, you need to make the right business and management decisions – simply demanding improved security will not work. Choosing the right application security platform and integrating it into your software development lifecycle can mitigate many of the problems that get in the way of your developers delivering innovative yet secure applications on schedule.

To learn more about building a realistic and effective web application security program, see the Invicti white paper **Enterprise Web Application Security Best Practices: How to Build a Successful AppSec Program**.

About the Author

Tomasz Andrzej Nidecki (also known as tonid) is a Primary Cybersecurity Writer at Invicti, and the author of Invicti Learn. A journalist, translator, and technical writer with 25 years of IT experience, Tomasz has been the Managing Editor of the hakin9 IT Security magazine in its early years and has been behind the Acunetix by Invicti blog since early 2019.

Most Recent Articles

A quick guide to telling apart SSDLC, SDLC, SDL, and the security life cycle

Top 4 resources for building a security champions program

Former security chief's prosecution is a warning to prioritize ethics in AppSec

Choosing an MSSP? Ask about DAST for your web application security

invicti

Invicti Security Corp 1000 N Lamar Blvd Suite 300 Austin, TX 78703, US

RESOURCES

Features

Integrations

Plans

Case Studies

Advisories

Invicti Learn

USE CASES

Penetration Testing Software

Website Security Scanner

Ethical Hacking Software

Web Vulnerability Scanner

Comparisons

Online Application Scanner

WEB SECURITY

The Problem with False Positives

Why Pay for Web Scanners

SQL Injection Cheat Sheet

Getting Started with Web Security

Vulnerability Index

Using Content Security Policy to Secure Web Applications

COMPANY

About Us

Contact Us

Resources

Partners

Legal

Privacy Policy

California Privacy Rights

Terms of Use

Accessibility

Sitemap