MEND

Developers Are Taking Over AppSec

Report

Application security is a top priority today for companies that are developing software. However, it is also becoming more challenging and complex as release frequency continues to rise and the requirements for data security are getting stricter. Thanks to new DevOps practices and tools, development cycles are getting shorter, allowing organizations to meet market demands and deliver a superior customer experience, but is application security keeping up? Application security has undergone a transition in recent years, as information security teams testing products before release became irrelevant, developers started playing a leading role in the day-to-day operational responsibility for application security.

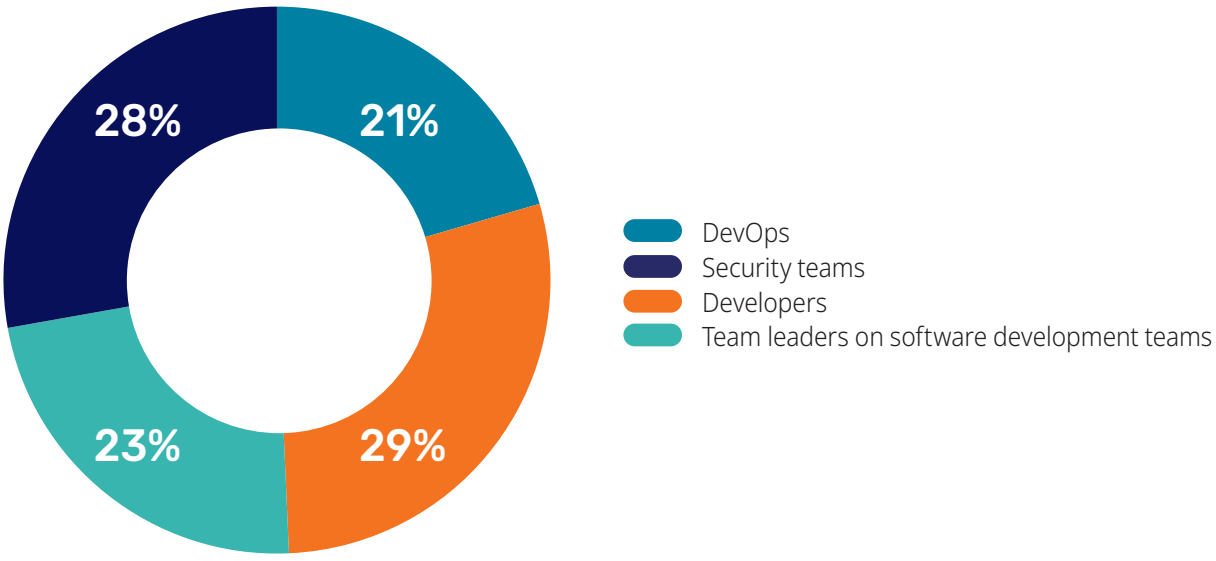
This change has led to application security testing shifting left into early stages of software development in order to alert developers on security vulnerabilities when it is easier and quicker to remediate. Although these application security tools have helped to bring serious issues to developers' attention earlier in the software development lifecycle (SDLC), there are still glaring inefficiencies that must be overcome. In hopes of a better understanding of how developers are dealing with their security responsibilities, we polled over 600 software developers. Their responses provide insights into how organizations are transferring application security responsibilities to support faster deployment cycles, the steps taken to support this transition and the challenges facing security and development teams.

WHO OWNS APPLICATION SECURITY?

Responsibility for the security of our applications has traditionally been in the hands of the company's security professionals. Developers would design and build the product, and security experts would perform their reviews and flag issues for remediation. Everyone pretty much knew what their role entailed.

However, these days, the lines over who owns security are being blurred. What we see is a clear movement of ownership for the day-to-day operational responsibility for application security with 71% of the respondents stating the ownership lies in the software development side, whether it is by the DevOps teams, the development team leaders or the developers themselves.

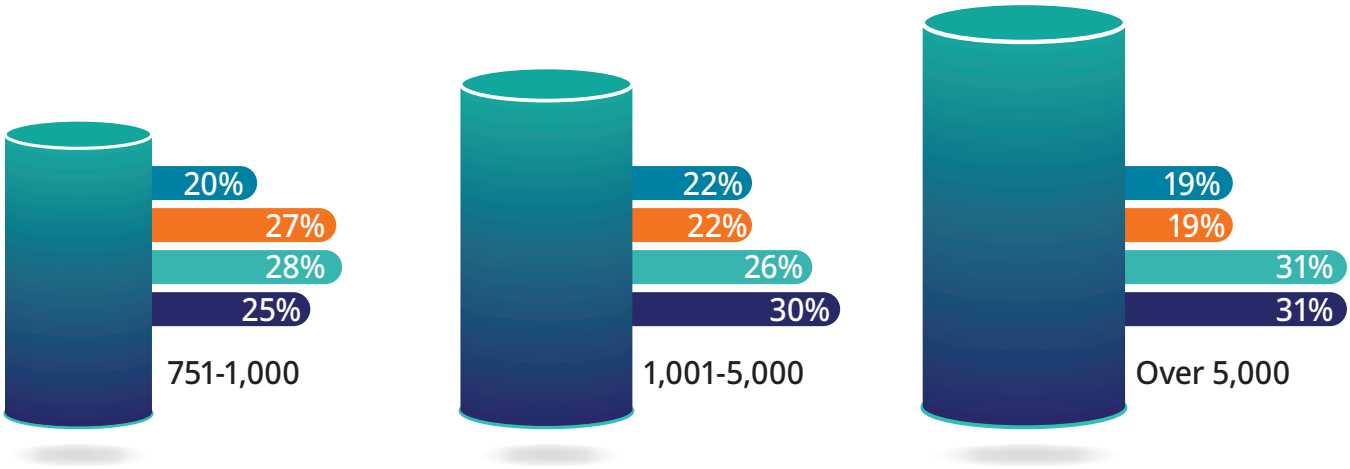
The reasoning is clear, as fixing a security vulnerability earlier in the software development process (during coding) significantly reduces the cost and effort needed, which not only increases agility but also produces better-secured applications from the get-go.



WHO OWNS THE DAY-TO-DAY OPERATIONAL RESPONSIBILITY FOR APPLICATION SECURITY IN YOUR ORGANIZATION? (BY COMPANY SIZE)

As AppSec continues to shift left into the design and development phases and responsibility over security is shared with developers, secure coding practices and tools become an important part of the DevSecOps pipeline.

In order to gain insights on secure coding with open source components, we decided to dive deep into the data on the most common CWEs in vulnerable open source components detected in 2020.

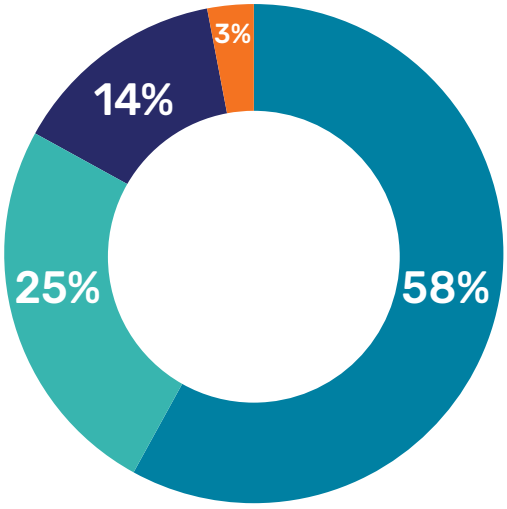


The smaller organizations, which have more freedom to define new processes, are seeing more progress in turning more ownership over towards developers. That said, even the SMEs and large enterprises are showing leftward movement away from their security teams maintaining control, putting developers in the driver's seat.

- DevOps
- Developers
- Team leaders on software development teams
- Security teams

INSIGHTS INTO HOW DEVELOPERS TAKE CHARGE OF SECURITY

When responsibility for security was more squarely in the hands of an organization's security team, developers mostly focused on functionality and meeting timelines. Now that developers have been tasked with secure coding, we see that their mindset has changed accordingly and most view security as a top priority while coding.



- Yes its a top priority - we have processes in place to detect and remediate vulnerabilities
- I only make sure the code is secure before deployment
- I think about security but only when an issue arises
- I don't think about security - it slows me down

INSIGHTS INTO HOW DEVELOPERS TAKE CHARGE OF SECURITY

In contrast to the waterfall methodology of the previous era where security was only reviewed before a release and sent back a mountain of security issues for developers to investigate, the modern DevOps approach relies on testing for vulnerabilities early and often in an attempt to avoid time wasting bottlenecks before releases.

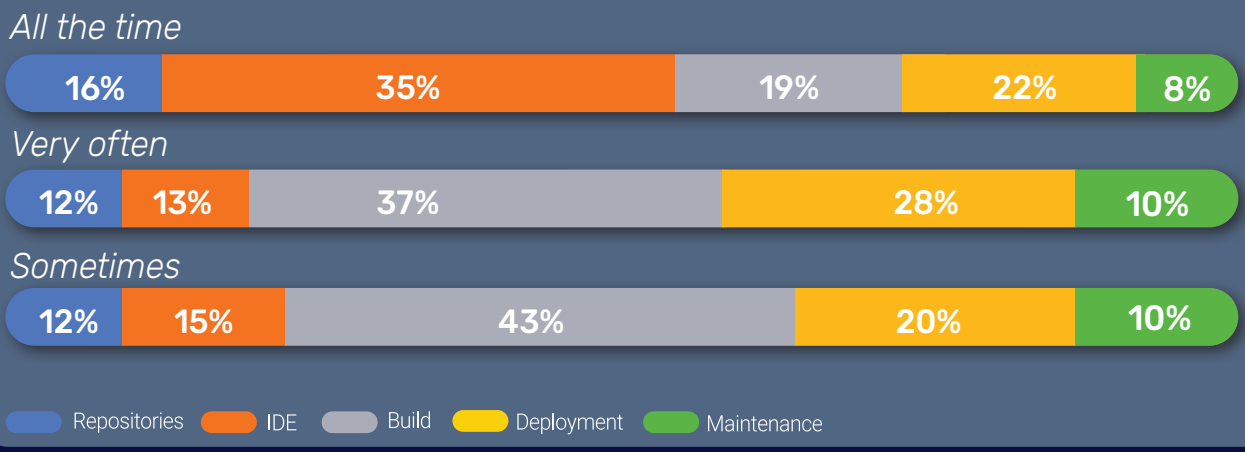
It follows that we see the build stage ranked highly as a testing point by many developers who are integrating security into their DevOps pipeline, since it is an easy jumping off point. What is more interesting is that the 36% of organizations have moved past the initial implementation at testing at the build stage and are starting to integrate security testing tools at earlier points in the SDLC like the IDE and their repositories.

In what stage of the SDLC do developers start testing the security of their application?

The decision to implement security testing tools before the build is part of a conscious decision by organizations to shift left their security efforts just like their quality testing to find issues when it is easier and cheaper to fix. Interestingly, we didn't notice a major difference when comparing SMBs, SME and large enterprises.

We did find a strong correlation between companies where developers testify to rely heavily on open source components, and companies who test their applications' security before the build. It is not much of a surprise as both are characteristics of a mature DevOps organization.

Testing application security in different stages of the SDLC, by open source usage



ORGANIZATIONS ARE INVESTING HEAVILY IN SECURE CODE

Companies are investing significantly in terms of testing tools, training, and time spent on handling security vulnerabilities, all in an effort to improve the security of their applications.

The good news is that these efforts appear to be paying off. 36% of developers say that their company provides them with security training that helps them code better.

How much emphasis does your organization put on security training for developers?

Along with training, developers are tooling up with a range of application security testing (AST) technologies with 68% of developers reporting using at least one of the following technologies: SAST, DAST, SCA, IAST or RASP. For organizations that are working with DevOps, the question is not if they should integrate automated tools into their pipeline, but which ones should they adopt first.

These technologies inform security teams and developers of potential security issues in their products, before and after production, adding visibility to the application's security and enabling teams to be proactive.

However, the integration of these automated application security testing tools is bombarding developers with security alerts, which developers are now required to research and remediate. It is unreasonable to ask developers to handle all security alerts, especially as most application security tools are developed for security teams focused on coverage (detecting all potential issues), rather than accuracy and prioritization.

While it is clear that organizations are investing in a number of measures to ensure that security is being integrated into development, it appears that developers are still struggling with this transition. Developers claim that they are spending a considerable amount of their time on dealing with remediations, with 42% reporting that they spend between 2 to 12 hours a month on these tasks, while another 33% say that they spend 12 to 36 hours on them.

What tools and processes can help developers continue to prioritize security without slowing down the development lifecycle and costing them so many valuable work hours?

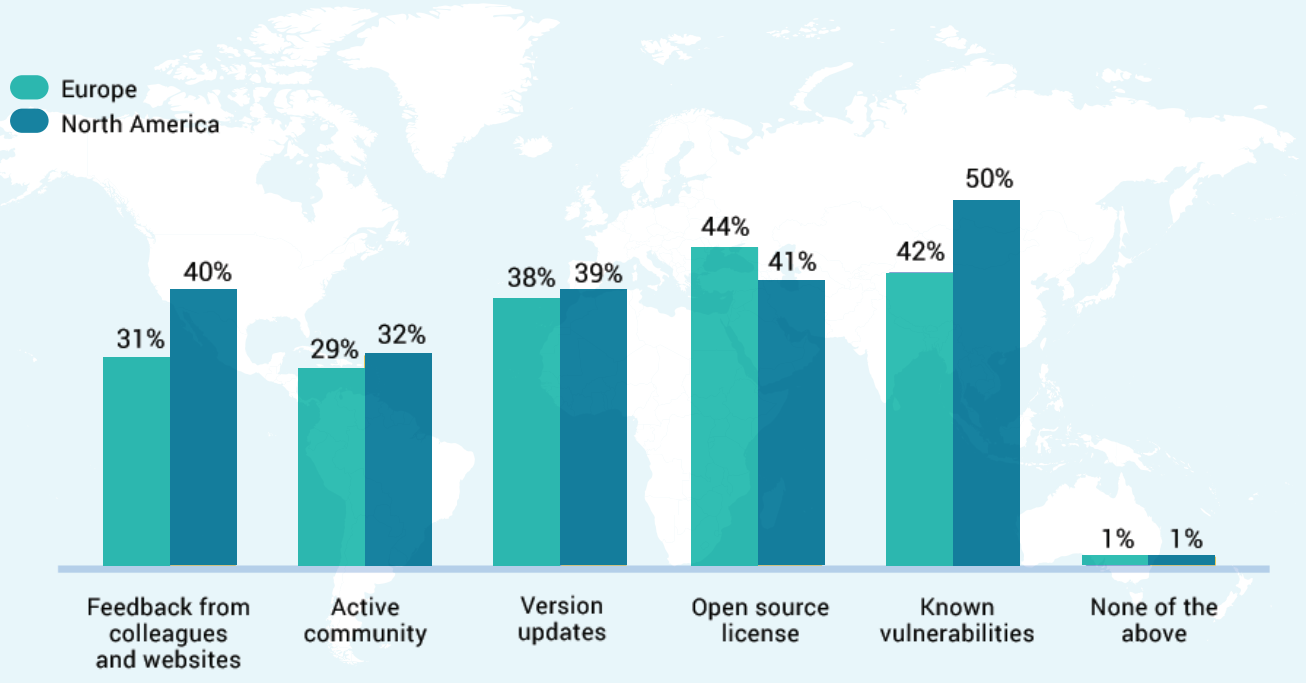
OPEN SOURCE SECURITY AS A USE CASE FOR THE ADOPTION OF DEVELOPER-FOCUSED SECURITY TOOLS

We decided to look at open source security vulnerabilities management as a use case to learn how developers are handling security throughout the development lifecycle: which processes are currently in place, how are automated tools helping them, and what challenges do they present.

Checking for known security vulnerabilities was top on the list of parameters developers said that they check when they choose an open source component, proving that developers are highly aware of the importance of ensuring their open source components are secure from the earliest stages of development.

Interestingly, we saw that respondents from North America (U.S. and Canada) showed a higher level of awareness to check the vulnerability status of the open source components that they were choosing. For the Europeans though, open source compliance rated higher on their priorities.

Which of the following parameters do you check before choosing an open source component, by region?

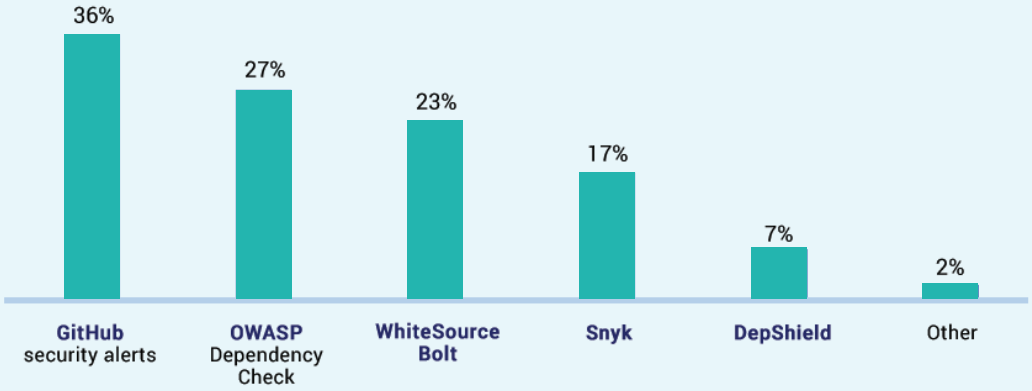


HOW DOES YOUR ORGANIZATION DETECT VULNERABLE OPEN SOURCE COMPONENTS IN YOUR APPLICATIONS?

The increased awareness is translating to 34% of developers implementing automated Software Composition Analysis (SCA) tools to detect open source components with known vulnerabilities in real-time.

It is important to state that most of these tools are free tools with limited capabilities offered by different application security and DevOps players. They have proved themselves to not only be effective at identifying known vulnerabilities but perhaps even more importantly, have been well-received by developers since they fit into their existing workflows.

If you're using a free tool, which one is it?



As probably one of the most beloved players in software, it is no surprise that most developers turn first to GitHub's Security Alerts tool for help. However, tools like Mend Bolt, which is available in GitHub and Azure DevOps, are also leading the pack as developers seek out reinforcements for handling the challenge of open source vulnerabilities.

However, knowing is only half the battle. Once the developers discover that they have a known vulnerability in their product they need to find a quick and effective path to remediating it.

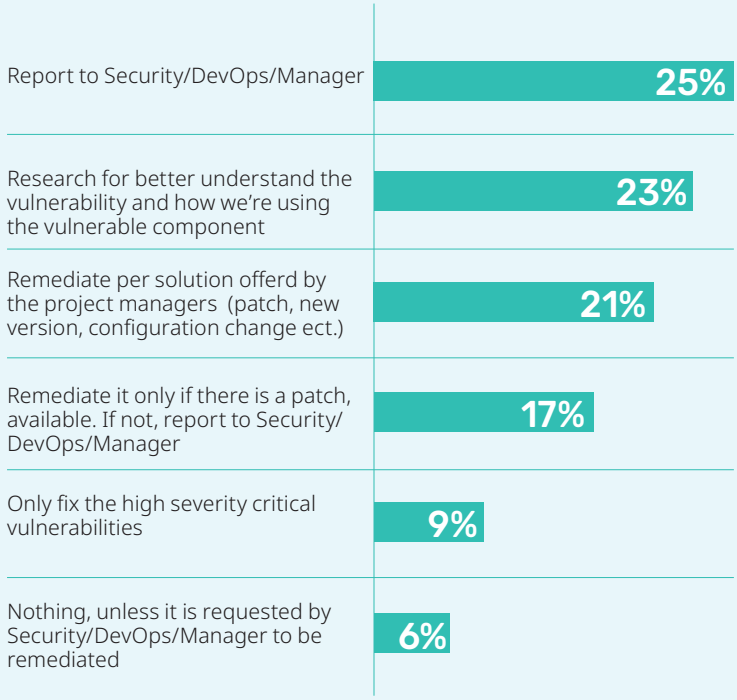
CLOSING THE LOOP FROM DETECTION TO REMEDIATION

Although we see developers taking a more proactive approach to detecting vulnerabilities we are not seeing the same shift when it comes to remediation. 25% of developers only report on detected vulnerabilities and 53% are taking actions only in specific cases.

As seen before, developers are investing many hours is research and remediation so why aren't we seeing more developers taking action? The reason probably lies in the fact that most application security tools' main goal is to detect, alert and report.

We believe that the time has come for a new generation of security tools built for developers, focused on not only on detecting but also on prioritizing issues, as well as automating remediation processes, to help developers handle the workload, and enable them to fix reported issues.

If a vulnerable open source component is detected in your application, what would you do?



Based on this research, we conclude that developers are faced with rising expectations for them to take a more substantially leading role when it comes to application security, but are lacking the necessary tools to do so.

Advances in the adoption of detection tools are a step in the right direction, likely due to the recent proliferation of free options, but we need to close the loop between alerts and remediation in a way that makes sense for developers.

From the moment that a vulnerability is detected, developers are required to investigate the impact and validity of the vulnerability, research the possible remediation options and carry on the remediation path chosen. The problem is that these are manual processes that, in addition to being time-consuming, they also demand a certain amount of skill. In order to have a significant impact on the security of applications, security tools must address these challenges.

The next generation of application security tools will be those that are developer-focused, closing the loop from detecting of an issue, all the way through validation, research, and remediation of the issue.

It is not enough to simply slap the word "developer" on tools that were designed for the information security team. Developer-focused security tools need to offer seamless integration to the developers' development environment and provide security alerts when needed while automating and simplifying the tasks needed for vulnerabilities remediation.