VOICE OF THE MODERN DEVELOPER

What Security Should Know About Developers

TROMZO | www.tromzo.com

A WORD FROM

Harshit Chitalia

CTO, Tromzo

Development sprints pressure software developers to produce more and more code quickly. Everyone wants secure code, but organizations do not incentivize developers to spend precious time addressing the endless stream of issues from the security team. Experience tells them that much of what security is concerned about will turn out to be noise, and they haven't the time to ferret out the critical signals.

Without the background or perspective to easily discern which security concerns are crucial and which are the result of ineffective or poorly tuned security tools, it is little wonder that developers ignore security issues.

Instead of propagating the blame game between Dev and AppSec teams, we believe it is more productive to better understand the challenges developers face, how they feel about security, and what organizations can do to bake security into the development process. To that end, we commissioned a survey of over 400 AppSec professionals for our first annual Voice of the Modern Developer Report.

The key findings here shed an informative light on how developers work with security teams and why.

42% of developers push vulnerable code once per month.

When a developer knowingly publishes code they believe to be vulnerable, it is clear that they think it is not their responsibility to fix the code before it is pushed.

Developers fix only 32% of vulnerabilities.

Given the volume of false-positive alerts that teams deal with today, fixing 32% of vulnerabilities could very well produce an acceptable result if developers could determine which 32% to fix. Unfortunately, without security training and experience, developers can not be expected to make that determination accurately.

A third of vulnerabilities are noise.

To reduce false-positive vulnerabilities, scans must have access to all of the required asset information so that security tools can accurately determine whether a vulnerability exists. Reducing security noise will allow developers to address security issues confidently.

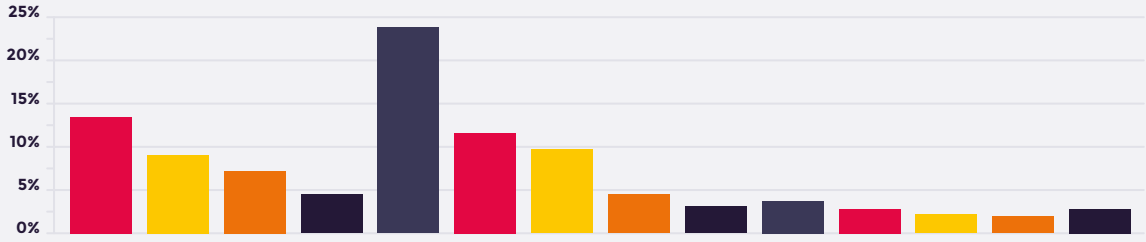
33% believe that developers and security are siloed.

When developers and security teams operate in insulated silos, it leads to inefficiencies and gaps in security across the software development lifecycle. These silos ultimately lead to security vulnerabilities and bad user experiences.

Starting on September 24, 2021, we surveyed 402 US-based developers who work at organizations where they are using CI/CD systems. The survey was conducted online via PollFish using organic sampling. Learn more about the Pollfish methodology here.

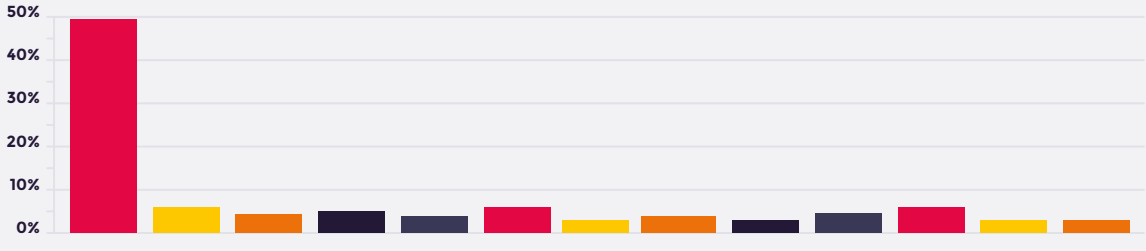
Profile of Who We Surveyed

What best describes your job title on the development team?



Junior Software Developer	13.1%	Software Architect	4.2%
Web Developer	9.4%	UX/UI Designer	2.9%
Programmer	7.2%	Backend developer	3.7%
Front End Developer	4.2%	Full-stack Developer	2.7%
Software Developer	23.8%	QA Engineer	2.2%
Software Engineer	11.6%	Database Administrator	1.9%
Senior Software Developer	9.7%	Other	2.7%

What industry does your company primarily operate in?



Technology	49.7%	State/Local Gov	3.9%
Finance	6.7%	Education	2.4%
Insurance	4.2%	Manufacturing	4.7%
Healthcare	5.7%	Services	3.9%
Hospitality	3.7%	Retail	2.9%
Utilities/Energy	6.7%	Other	2.4%
Federal	2.4%		

Current State of Application Security

Today, many developers find themselves in the unenviable position of serving two masters. They are responsible for creating elegant apps that facilitate critical business needs while simultaneously protecting sensitive data, networks, and endpoints. With most organizations having suffered at least one security breach from an application vulnerability, it's obvious how aggressive and pervasive application-based cyber attacks have become.

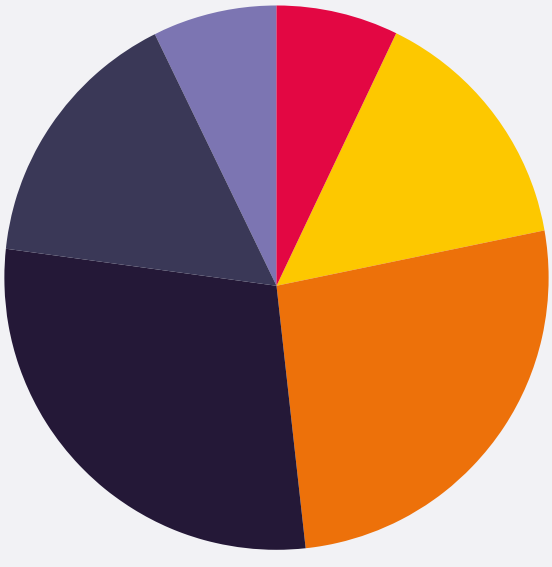
PART #1

62% of developers are using 11 or more application security tools.

There are scores of code scanning and security testing tools on the market today. Many have a unique purpose or better identify one type of security vulnerability than others. Some are more generalized but tout ease of use or scalability. However, using too many tools comes with its own set of challenges. Most (62%) respondents indicated that they use eleven or more application security tools, and 17% use twenty or more tools.

A few tools analyzing a large amount of data is the preferred method for attaining high-value security signals. Unfortunately, too many companies have resorted to piling on more tools to outwit their cyber adversaries. There are security testing tools and application shielding products - tools for static, dynamic, interactive, and mobile testing. The result for developers can be a tsunami of security signals that can hide critical vulnerabilities.

Overall, how confident are you in your organization's application security posture?



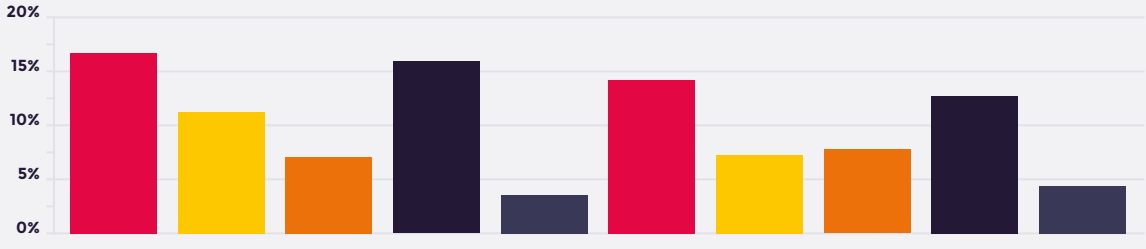
1 - 5	8.2%	16 - 20	28.3%
6 - 10	5.1%	20+	15.9%
11 - 15	25.1%	I don't use any	7.2%

Developers not following through on security requests is cited as the #1 challenge by a plurality of respondents.

The fact that developers fail to do what security teams ask of them is a strong indication that security testing needs to be shifted left and integrated into the development process. When organizations employ more than 11 different security tools (see key finding 1) that generate a flood of security alerts, it should come as no surprise that developers struggle to fix all the suspected vulnerabilities, especially when security remediation requests are given to them after the development life cycle has ended or is nearly complete. In this light, the case for shifting security left becomes apparent.

16.5% of our respondents feel that developers' lack of follow-through was the primary challenge for the AppSec program, but 15.5% maintain that lack of visibility across scanners and tools was an even more significant challenge. Both findings substantiate the notion that developers are too often left without context or visibility into security concerns.

Which of the following do you consider to be the #1 challenge of your application security program?



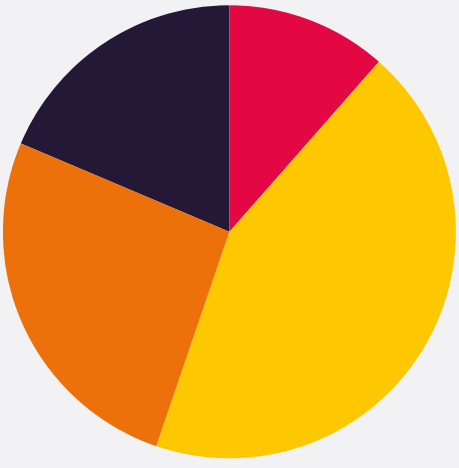
Developers not doing what security asks	16.5%
Too much noise and false positives	11.1%
Too many tools	6.9%
Lack of visibility across scanners and tools	15.5%
Too much time spend on manual work	3.9%
Changing culture of organization to cultivate a security awareness	14.0%
Training development staff in secure coding practices	7.1%
Educating management on the threats and mitigation strategies	7.6%
Reviewing legacy and third party applications for security risks	12.6%
Shortage of capable security staff	4.7%

42% of Developers Push Vulnerable code once per month.

To be sure, there is a significant difference between knowingly pushing vulnerable code and inadvertently doing so. The former could indicate a callous indifference toward the quality of their work product–which is highly uncharacteristic for most developers–or a belief that someone else in the organization is primarily responsible for catching and fixing vulnerabilities. Confusion can creep into security processes without a clearly defined and continuously reinforced AppSec program.

Inadvertent publication of vulnerable code, on the other hand, signifies a need for more robust quality control methodologies and additional testing. For organizations overwhelmed by security alerts, it is often a matter of no one having the time to investigate a suspected vulnerability thoroughly. While done without malicious intent or indifference, the outcome can be just as devastating.

How often do you push vulnerable code?



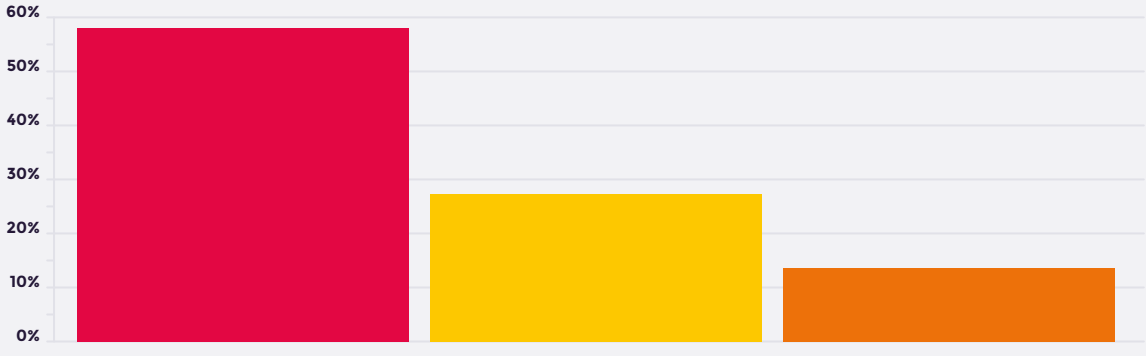
Never	12.6%
Roughly once per month	42.7%
Roughly once per quarter	25.3%
Roughly once per year	19.1%

80% would be surprised if they saw their company in the news for a security breach.

A developer that is surprised when their application is compromised is at once a little naive and appropriately confident in the quality of their work. Slightly naive to believe that the isolated efforts of even the best developer are enough to produce an impenetrable application and yet confident that they have done everything as securely as possible. As a development team leader, you would hope that number would be at least 80%, and even higher would be better. As a risk manager, you would know that when you become the target of an advanced cyber adversary, there is little chance that your code will hold up.

Dev teams must incorporate secure coding practices into all life cycle stages of the application development process. Ensuring the development of a secure application is a responsibility that exceeds the capability of any single member of the team. The entire team must focus on security risks and integrate secure coding practices into their day-to-day operations.

How surprised would you be if you saw your company in the news for a breach related to a vulnerability in your applications?



Very surprised	59.2%
Somewhat surprised	28.6%
Not many surprised	12.1%

Developers & Vulnerabilities

In Part 2 of this report, we examine some of the survey's key findings that reveal important information about how developers interact with vulnerabilities and remediation requests. We look at how signals from security tools affect how developers work and how they react to the pressures exerted by security teams.

PART #2

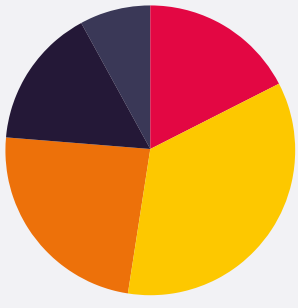
69% of Developers Have At Least 1000 vulnerabilities/issues to address, which has increased 3x for 22% of them.

It's not difficult to imagine what it would be like to work on a development team that faces an insurmountable avalanche of security vulnerabilities. You would feel like you are drowning. Add to that picture knowledge that even as you struggle to keep your head above water, there is a continual increase in the rate at which the depth of the problem is rising.

The only plausible solution is to prioritize the vulnerabilities the best you can and hope for the best. No one wants to believe that the security of their application hinges on a hope that the most critical vulnerabilities were accurately plucked from the sea of security signals the Dev team swims in, but that is the reality of many developers.

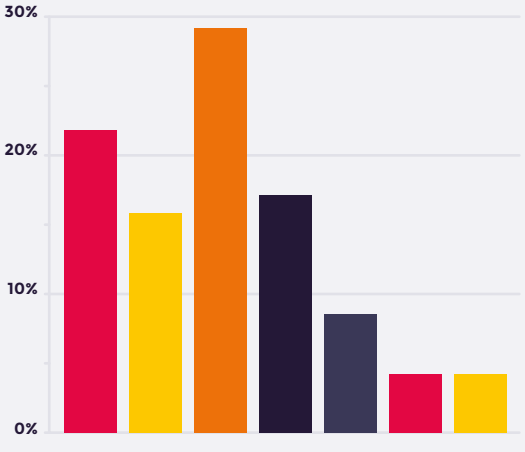
The size of an organization and the complexity and scale of its applications can affect how many unaddressed vulnerabilities they can reasonably live with, but 20% of our respondents said they have 10,000 or more still to contend with. That is unreasonable on any scale.

How many security vulnerabilities/issues do you currently have that need to be addressed?



Less than 1,000	18.9%
1,000 - 5,000	32.0%
>5,000 - 10,000	24.3%
10,000+	17.4%
I don't Know	7.2%

Which of the following do you consider to be the #1 challenge of your application security program?



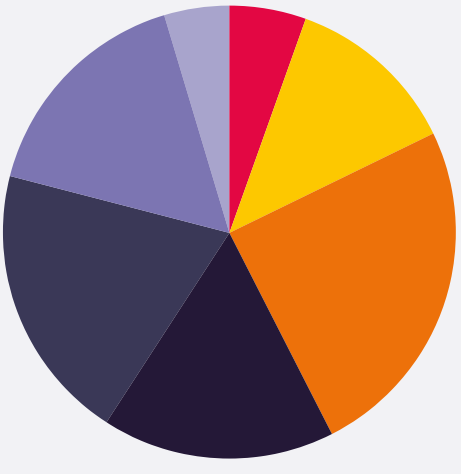
Increased at a rate of 5x or more	20.9%
Increases at a rate of 4x	15.6%
Increases at a rate of 3x	28.3%
Increases at a rate of 2x	16.6%
No increase	8.4%
Decreased	4.9%
I don't Know	4.9%

Only 17% of Developers feel 51% or more of these vulnerabilities/issues are truly important to fix.

IGiven the number of unaddressed security issues that companies have to deal with, it is understandable that a plurality of developers feel that most of the vulnerabilities and security issues that land on their plate do not warrant action on their part. Of course, the danger inherent in continually contending with this noise level is the natural tendency to discount the urgency of all, except the most extraordinary, security alerts. This signal desensitization undermines the value of the organization's security tools.

Ironically, as a team generates more security alerts, it can mean a degeneration of their development security posture. Providing developers context around security alerts, scan results, and testing outcomes enables them to contribute to the process of filtering out false-positive results.

What percentage of vulnerabilities/issues are truly important to fix?



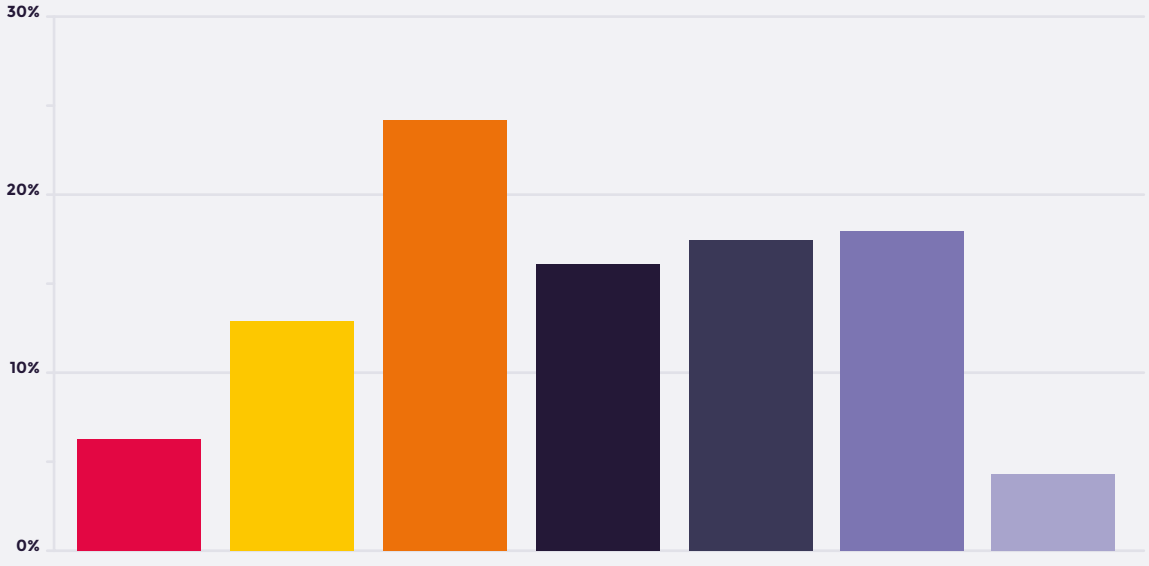
Less than 10%	5.9%
11 - 20%	13.1%
21 - 30%	24.1%
31 - 40%	16.9%
41 - 50%	17.4%
51% or more	17.6%
I don't know	4.7%

Developers fix only 32% of vulnerabilities.

The implication of this question in the context of this survey is that the tools used by AppSec programs are ineffective. Not because they miss too many critical signals; instead, along with alerting developers to significant security issues, they can easily overwhelm them with too many unactionable alerts.

The optimal condition is that developers respond to and, where applicable, mitigate each alert generated by the organization's security tools. To accomplish this, security testing should be an integral part of development rather than an external function that feeds untimely information back to developers.

What percentage of these vulnerabilities/issues do you fix?



Less than 10%	5.9%
11 - 20%	13.1%
21 - 30%	24.1%
31 - 40%	16.9%
41 - 50%	17.4%
51% or more	17.6%
I don't know	4.7%

False Positives

False-positive security alerts consume valuable resources in the form of the time it takes developers to validate or dismiss in the indication of a vulnerability. No systems are perfect, so a zero false-positive rate suggests that the tool is missing critical vulnerabilities.

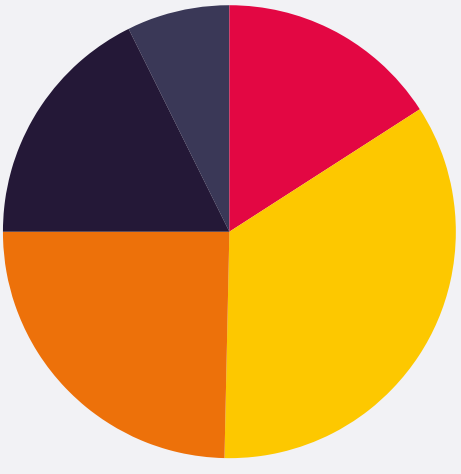
PART #3

A third of vulnerabilities are noise.

This survey indicates that many teams believe that at least a third of the vulnerabilities indicated by their security tools are only unactionable noise. Nearly 20% believe that more than 75% of alerts are spurious.

These results are a severe condemnation of the security tools commonly used in business today. Modern security tools provide the ability to suppress or reclassify alerts to reduce the false-positive rate, but taking these actions can mask critical vulnerabilities and create a false sense of security.

What percentage of these vulnerabilities/issues are false positives or noise?



Less than 25%	17.4%
25 - 50%	32.3%
50 - 75%	25.3%
75% or more	18.1%
Don't Know	6.7%

Three Changes Teams Have Made to Reduce the Number of False Positives.

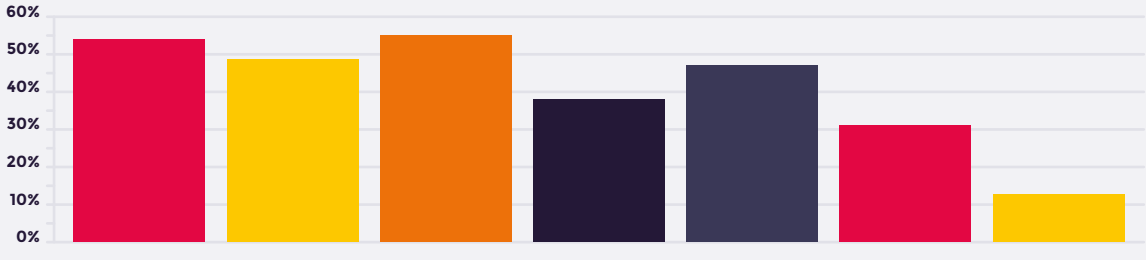
Respondents that told us they were reducing the number of false-positive alerts shared how they achieved this.

The single most effective way teams reduce false positives is by implementing modern AppSec tools like Software Composition Analysis, Runtime Application Self-Protection, and IAST (Interactive Application Security Testing). These tools analyze code for vulnerabilities while the app is running and report vulnerabilities in real-time, which does not add extra time to the CI/CD pipeline.

Next, teams found defining custom detection rules in their scanning engine helpful. Creating custom analytics rules to help discover threats and anomalous behaviors in your environment is a method of fine-tuning scan alerts. Begin with existing rules and then define more stringent conditions to reduce false positives.

Additionally, developer teams find it helpful if security analysts manually triage results before sending a mass of alerts to the development team for remediation.

If less than usual, what changes have you made to reduce the number of false positives and noise?



Created custom scanning rules to tune the scanner for the application	54.7%
Integrated security testing earlier in the development process	49.2%
Update coding standards to return consistent error messages used by scanning tools	54.9%
Integrate validation testing with automated scanning	38.3%
Implement Web Application Firewall to prevent questionable requests before received by application	48.5%
Improve collaboration between security and development teams	31.8%
Other	11.6%

The relationship between security & developer teams.

Overall, development teams still see security as a barrier. It is a common perception that security stifles innovation and slows down the pace of development, which in some cases has an adverse effect on developer financial incentives. The farther security processes are from developers, the more prevalent these perceptions are.

As organizations shift security left, the relationship between security teams and developers for most organizations improves. The respondents for this survey gave us an informative glimpse into how these two crucial teams interact and how those relationships are changing.

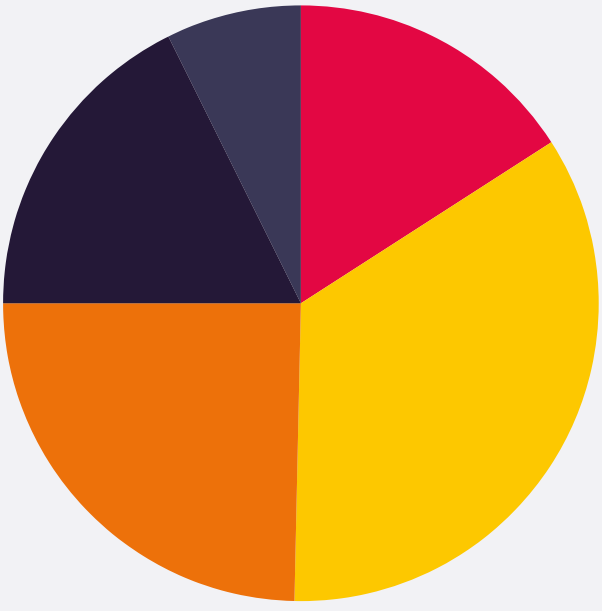
PART #4

23% believe that developers and security are siloed.

Operating in silos with little communication between development and security teams is an unfortunate reality for nearly a quarter of organizations. This condition is a breeding ground for animosity and misunderstanding. Developers inundated with a barrage of security remediation requests can understandably harbor resentment toward the security team unless they have visibility into the background and limitations of the AppSec program.

Fortunately, most organizations report at least essential communication between these teams, with developers acting on security service requests regularly. Over half of our respondents say that these relationships are improving. Security teams affirm that developers understand the risks of pushing insecure code and take security seriously.

Which of the following best describes the relationship your security team has with developers



Teams are siloed with little communication between them	22.9%
Security and development teams have basic communications established. Security services are done on request	21.4%
Security and development teams work together. Each team has a security champion and there are regular security trainings for developers	22.9%
Teams collaborate together. Security reviews performed with developers and system admins. Security processes are automated and integrated into SDLC	17.7%
None of the above	15.0%

What organizations can do to improve the relationship between security and developer teams.

When asked what organizations can do even further to improve the relationship between developers and security teams, a plurality of respondents indicated that integrating automated security checks throughout the software development life cycle to keep pace with release cycles would do the trick.

To shift security left, many organizations agreed that having at least some Dev team members qualified to focus on security would improve communication and help bridge the gap between that team and security. Having these security specialists embedded in the development organization can help developers be more agile and raise the security awareness of the entire team.

Even those respondents that couldn't envision a world where security checks were integrated throughout the SDLC acknowledged that the integration of security checks earlier would reduce work on developers. In every case, respondents stated or implied that the more security could be integrated into development (the sooner, the better), and the more control developers have over security outcomes, the more secure the code will be.

The Future is Developer-First

The AppSec team will always be the experts in making risk-based decisions. Their training and experience qualify security practitioners for making critical determinations and risk evaluations. While they can drive security accountability across development teams and be invaluable in solving complex security challenges and training developer specialists, the entire organization will benefit from shifting the security team's day-to-day security testing and scanning tasks to the developers responsible for creating applications.

This survey clearly demonstrates that the flood of false-positive alerts and security noise generated by many AppSec programs contributes to tension between security and developer teams. Security teams become frustrated that developers are pushing vulnerable code. Developers without adequate context and latitude to make security decisions can only, at best, prioritize security remediation requests in the dark and hope for the best.

At Tromzo, we are committed to helping to create a world where developers effectively determine appropriate security measures as they go about developing elegant applications to meet critical business needs. A world where secure code is paramount in developer workflows and security practitioners can focus on making the organization more secure.

As organizations shift security left in the software development life cycle, application security professionals have a valuable role in gathering intelligence, addressing complex issues, training developers in security best practices, and refining the AppSec program.

The evidence is clear that developers can address security vulnerabilities most effectively when allowed to do so during the development life cycle and not as an untimely remediation.


TROMZO
Make Application Security
Easy for Developers.