Checkmarx

White Paper

The Road to
DevSecOps

The Road to DevSecOps

Holding on to the way things were has never worked for developers, security practitioners, or anyone in the technology ecosystem and software development is no exception.

Unfortunately, it also hasn't worked for cybersecurity.

The shift to cloud-native applications, has allowed for a continuous development and deployment cycle with new tighter deadlines.

In response, DevSecOps offers a new approach that nips vulnerabilitlies in the bud, before they become unmanageable later.

However, there are some clear problems with the dialogue:

1. DevSecOps is often misunderstood as being distinct from DevOps. But in reality, they're the same thing. More correctly: DevSecOps is the necessary evolution of DevOps.

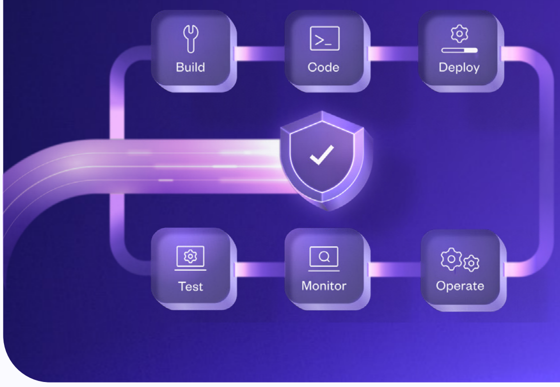
2. Almost no one is actually doing DevSecOps.

Only **20%** of companies surveyed have begun automating security controls as part of a DevSecOps effort.

The point of DevSecOps is to realize that AppSec tools must become developer tools. They must seamlessly integrate into the developer mindset and process. The goal of DevOps is to deploy and continuously improve high-performing code.

This is the intersection between DevSecOps and DevOps - code cannot be high-performing if it is not secure. So, if security is necessary to create high-performance code, AppSec must provide useful tools for developers to write secure code!

If we haven't truly implemented a DevSecOps approach yet, how can we get there from where we are now?



In 2024, Checkmarx surveyed 500 developers and 200 CISOs about their workflows and compiled some of the answers to help shed some light on the road to DevSecOps. In addition, we also developed our own quick and dirty DevSecOps maturity model, including five key requirements to get your team moving towards a DevSecOps future.

↘ **Keep reading to get our survey results, maturity model, and five key requirements for DevSecOps!**

DevSecOps

What DevOps Needs To Be When It Grows Up

↘ Where Are We Now?

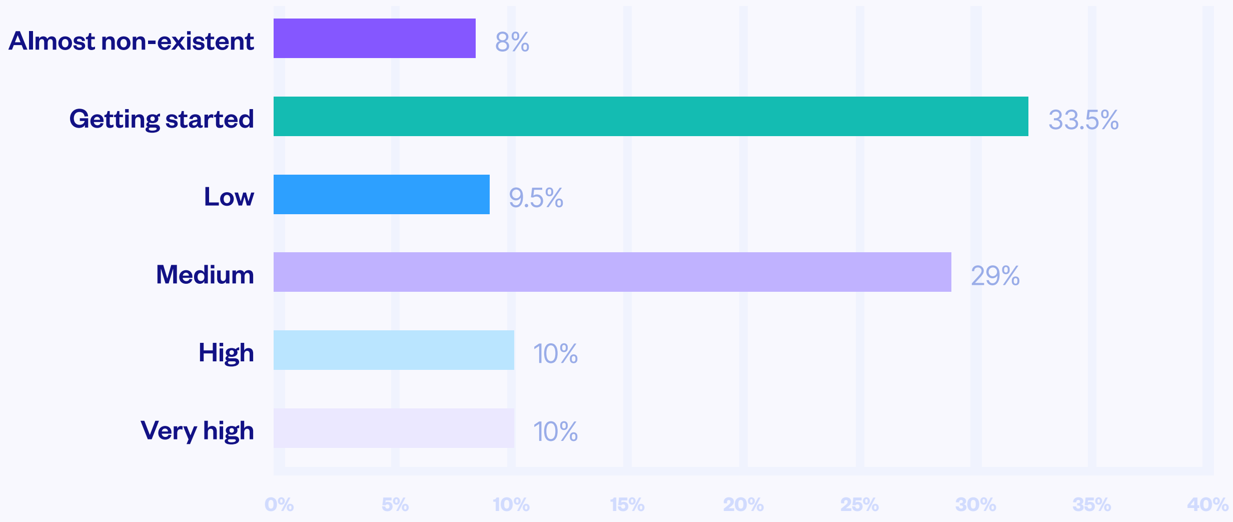
DevOps represents the fundamental cultural shift in software engineering that prioritizes performance – high performing teams with high-performance code.

However, DevOps' emphasis on performance often neglects security considerations.

DevSecOps reflects the reality that DevOps must expand to encompass security as a core component. It is the continued merging of organizational cultures that began with DevOps, with the ultimate goal being that high-performance code is also secure by default.

We're not there yet, so what's the next step?

In mid-2024, Checkmarx surveyed CISOs about the state of their current AppSec programs. We asked, "Where are you on your DevSecOps journey?" Here's how they answered:



- ▌ **Almost non-existent**
- ▌ **Getting started** - Our DevOps and security teams remain separate, with minimal collaboration
- ▌ **Low -** Our DevOps and security teams are collaborating on developing joint policies and strategies
- ▌ **Medium** - Our DevOps and security teams have already defined joint policies and strategies
- ▌ **High** - We've begun integrating security with DevOps and automating security controls
- ▌ **Very high** - Security controls are mostly/fully integrated and automated within our DevOps pipeline

FIGURE 1

On the surface, this doesn't look too bad... until you consider the details. Based on our scaling, "medium" only represents the definition and strategy phase. The process of integration and automation - when companies begin implementing DevSecOps - is only active in 1 of 5 companies surveyed.

Let's be clear – the actualization of DevSecOps is only possible through integration and automation. DevSecOps takes the needs and outcomes of application security – specifically risk mitigation and management – and integrates them with the processes and culture of DevOps. We shouldn't ever be differentiating between DevOps and DevSecOps; DevSecOps is simply what DevOps needs to be when it grows up.

DevSecOps

Why Go Through the Effort?

Getting your security and development teams marching to the same beat seems like a lot of hard work. Why even bother? Here's one reason: Our CISO survey reveals that application security and secure coding practices are becoming essential factors in the purchasing process across various industries (not just in SaaS).

How often do your prospects consider your level of application security maturity when they make purchasing decisions regarding your company's products?

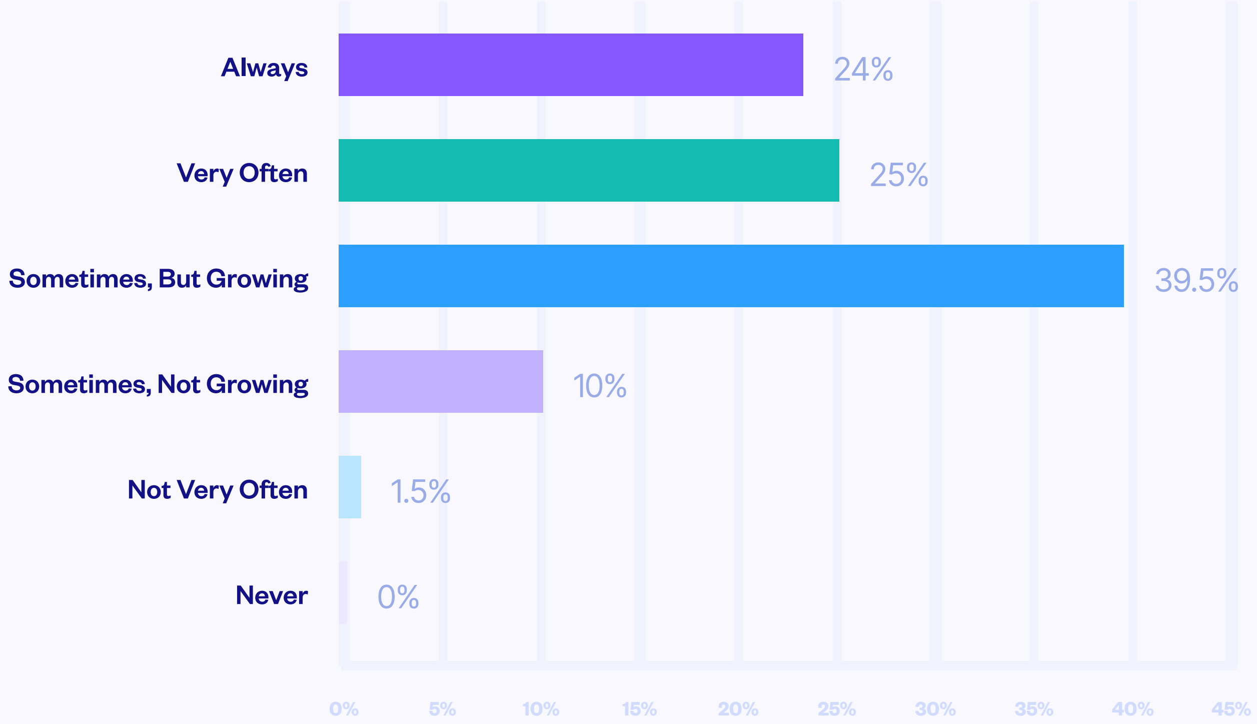


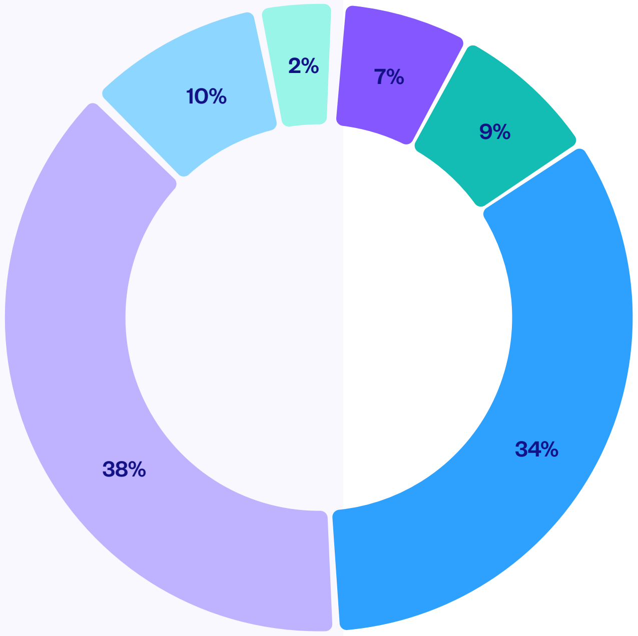
FIGURE 2

In fact, **88.5% of respondents** replied that prospective customers increasingly or very often/always consider application security in the purchase process. This is aligned with other trends observed by Checkmarx' analysts that buyers are increasingly prioritizing security in their decisions and holding vendors accountable when falling behind.

↘

We also recently surveyed 500 developers and found that 93% of companies reported at least one security breach in the past twelve months due to an internally developed, vulnerable application.

In the past 12 months, how many times, if any, has your organization experienced a security breach as a direct result of a vulnerable application that your organization developed?



No data breach in the past 12 months	7%
Once	9%
Twice	34%
3 times	38%
4 times	10%
5 times	2%
Total	**93%**

FIGURE 3

Organizations experienced an average of

2.4 breaches in the past 12 months.

The Path to Maturity

Companies that are mature in their DevSecOps journey behan with a security-first focus and incorporated the developer experience along the way.

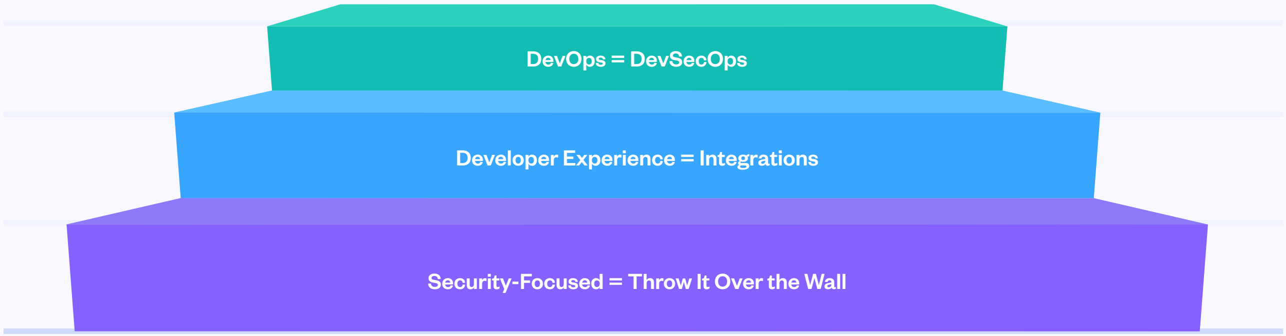

DevOps = DevSecOps
Developer Experience = Integrations
Security-Focused = Throw It Over the Wall

FIGURE 4

Let's start at the bottom of the model, where AppSec traditionally locates vulnerabilities and tosses them to developers to fix. And this starting point, where AppSec managers blast developers with hundreds of vulnerabilities to fix, is what's commonly known as "shift left." Maybe that's a bit brash and contemptuous to shift left efforts, but when you look at the big picture, this truly is the base level of maturity. That said, even though shift left is the very bottom of the model, it plays an important role in getting organizations moving towards DevSecOps, and you have to start somewhere!

The next level focuses on developer experience and the point of maturity where both AppSec teams and developers realize that shifting left doesn't work. And it doesn't stem from laziness or lack of focus, but because shifting left is only intended to be the first step. In this stage, the AppSec teams already have the tools needed to find and triage vulnerabilities, but the developers don't necessarily have the correct tools for them to fix these vulnerabilities without disrupting their workflow. The developer experience stage of maturity focuses on tools that facilitate IDE integrations, remediation guidance, and minimizing workflow disruptions that help keep developers focused.

However, like shift left, focusing exclusively on developer experience eventually leads to diminishing returns. Once organizations hit this point, they need to continue moving towards the third step of maturity: establishing a DevSecOps culture. To get there, security and development teams need to define and agree upon joint policies, governance, and methods of collaboration. Then, they can begin to build security automations based on the successful merging of DevOps and security cultures. This will ideally encourage AppSec teams to align with development goals and developers with AppSec goals, uniting both teams around shared direction and purpose!

GOOD NEWS!

Only **13%** of developers described their communication with security teams as bad or worse.

AppSec Is From Mars – Developers Are From… Mercury?

We know that building a DevSecOps culture is difficult. The proposed merger between security and DevOps is particularly thorny because of the environments that these groups of people generally come from. We emphasize groups of people to drive home the idea that moving to DevSecOps is fundamentally a human issue.

And here's the difference between these two groups of people. Developers and DevOps teams live in the IDE. They (should) have big chunks of uninterrupted "maker time." According to Paul Graham's 2009 essay, *Maker's Schedule, Manager's Schedule*, this means that developers need time dedicated to coding, with zero outside interruptions such as meetings and alerts. They really need this maker time to "move fast and break things." Get to minimum viable product. Get feedback. Iterate. Produce!

Security teams, on the other hand, operate differently. While DevOps teams move fast and break things, security teams don't ever want to let anything break and are constantly faced with a flood of alerts that disrupt (but also direct!) their workflow.

When Gartner says "only 29% of those surveyed say the two groups strongly agree with each other" – it probably comes down to these different cultures with different goals.

> “According to Gartner® survey data, there is a 27% improvement to security outcomes when there is a high-level of collaboration between developers and security. However, only 29% of those surveyed say the two groups strongly agree with each other.”
>
> Gartner.
>
> DevSecOps Maturity Model for Secure Software Development | 29 August 2024

The key to changing these two distinct cultures is understanding the fundamental mindset of these two teams and getting them aligned. To this end, we've come up with **five key requirements** to keep in mind when pushing this internal, cultural change.

If you're familiar with DevOps, you've likely heard of Jez Humble's (co-author of The DevOps Handbook) CALMS framework. CALMS refers to the five proposed pillars of DevOps:

- Collaboration/Culture
- Automation
- Lean
- Measurement
- Sharing

We have come up with five requirements, aligned to CALMS, that will help move the needle on the culture shift from DevOps to DevSecOps:

01 **Integrations**

02 **Shared measurements**

03 **Security education**

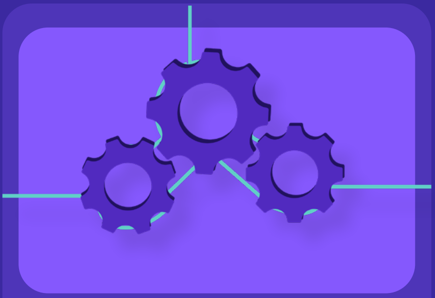
04 **Security velocity that matches DevOps**

05 **Automating security processes**

Five DevSecOps Requirements

The remainder of this report will review these five requirements in detail, supported by relevant survey data that highlights their significance.

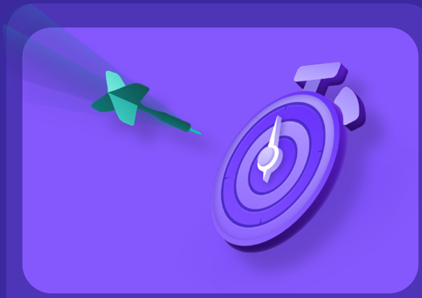
Before we dive in, note that each requirement aims to align DevOps and security teams with each other's needs:



#01

Integrations

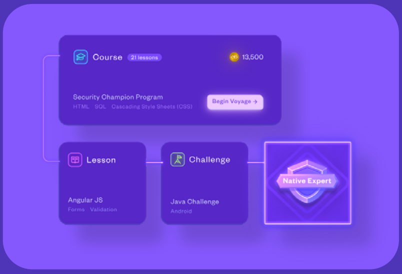
encourage security teams to consider how developers work and how to keep them productive.



#02

Building Shared Measurements

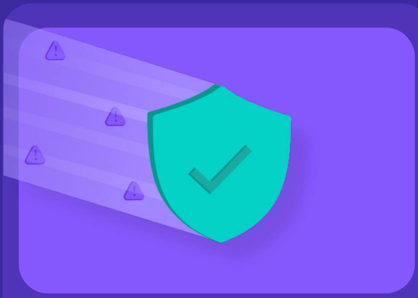
aligns teams on their goals and fosters a shared language and outcomes they can agree on.



#03

Security Education

helps developers understand security, while also enhancing their own skillset advancing their careers, and streamlining security tasks.



#04

Matching Security Velocity to DevOps

helps developers feel like security is part of the process rather than an obstacle.

#05

Automations

are the core of DevOps, fostering efficiency and boosting team satisfaction!

DevSecOps Key Requirement #01

Integrations

Looking back at our maturity model, the first step towards DevSecOps is thinking about the developer experience. A requirement for DevSecOps is keeping developers in their flow state, which means security tools need to be delivered directly to the IDE to keep things moving effectively.

This is one of the rare cases where a change in tooling is actually a cultural step – and integrations are the first manifestation.

Properly integrating security tools into the developer workflow fosters communication and trust, helping security teams provide developers with the resources they need to achieve their goals.

Our survey shows the importance of this step, as 67% of developers rated IDE integrations as very important or critical:

How Important is Seeing Vulnerability Scan Results in Your IDE?

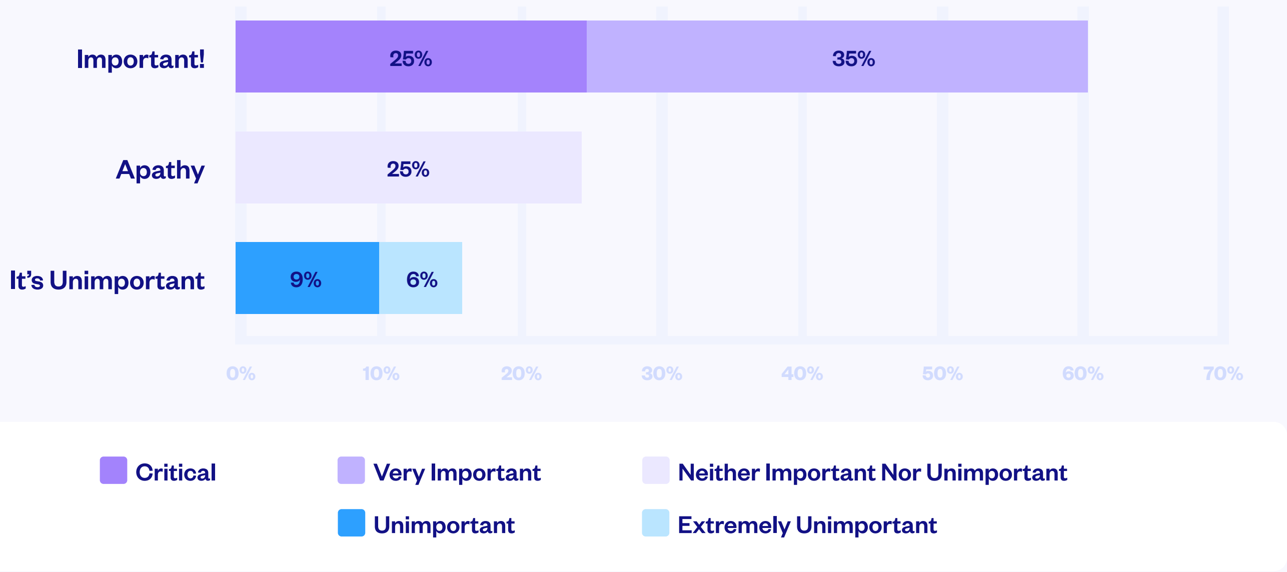


FIGURE 5

Apathy Among Developers!

Note that the center column of Figure 5 is labeled "Apathy." We asked this question in a similar format to the same group of developers across seven different domains (IDE integrations, automations, training, prioritization, GenAI, SSO, and low false positives), with approximately 20% answering "neither important nor unimportant" for each question. This is troubling and likely represents a block of developers who are either completely checked out or do not care one way or the other. They are neither believers, nor non-believers, instead they seem to be non-participants. It's worth seeing who these developers are on your team so you can figure out how to get them more engaged.

This same survey highlighted another data point supporting the need to focus on developer experience. We asked developers to choose their top three reasons for choosing or recommending their most recent AppSec solutions.

Out of twelve options, the top five characteristics all focused on developer experience:

1. Fast onboarding
2. Improved the developer experience
3. Better integration or built in with CI/CD processes
4. Better integration into the IDE and tooling
5. Able to remediate across the SDLC with a single tool

RECOMMENDATIONS

At this point, security teams must choose which integrations will make developers' lives easier. Don't forget to advocate for yourself!

Integrated Development Environment (IDE)

Devs do their work here. If you're not in an IDE, you're not in a dev's workflow.

Source Control Management Tools (SCM)

SCM tools track changes to the repo.

Feedback Tools

Feedback tools are usually integrated into a developer's IDE, but integrating with a feedback tool helps developers stay in their workflow by delivering vulnerabilities and remediation advice directly to them.

CI/CD Pipeline Tools

Tools that automate building, testing, and deploying software changes are key integration points for security. If you want to insert security processes, they'll need to cover the whole SDLC and these pipeline tools are important points for control.

Starting with integrations should help your teams quickly ramp up on the maturity curve and go from "throwing vulnerabilities over the wall" to purposefully nurturing the developer experience.

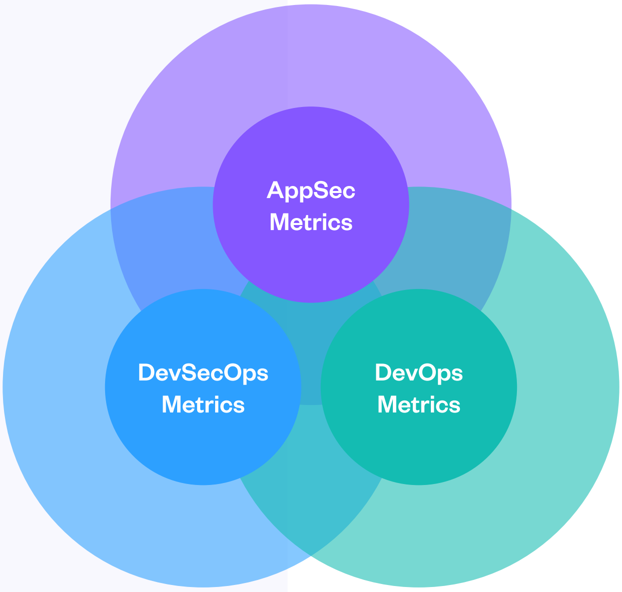
Next, we're going to explore metrics that both security and developers relate to, because shared measurements are key to developer experience – and DevSecOps. When you're finally measuring the same things, you're moving in the same direction!

DevSecOps Key Requirement #02

Shared Metrics

Simply put, shared metrics are metrics everyone – both developers and AppSec managers – can relate to. Earlier, we explained that security and DevOps teams come from different mindsets, which also means that they have different goals and KPIs. Security and development teams operate and track very different metrics.

Want to move to DevSecOps? Begin thinking beyond a security-only mindset. While vulnerabilities and remediation stats are one metric, they fall short of the big picture DevSecOps promotes. Good metrics for a DevSecOps approach are those that measure how quickly your integrated team is delivering organizational value. They are metrics that focus efforts and identify pipeline problems to help your teams work more efficiently.



AppSec Metrics

- Number of vulnerabilities (by severity)
- Fixed vulnerabilities (by severity)
- Scans
- Scanned LOC

DevOps Metrics

- Lead time for changes
- Change failure rate
- Deployment frequency
- Cycle Time

DevSecOps Metrics

- Mean time to remediate (the other MTTR)
- Mean time to detect (MTTD)
- Issue volume
- Top vulnerable applications
- Security coverage (depth of scanning)

FIGURE 6

AppSec and DevOps metrics should not be disregarded in the move to DevSecOps. Traditional metrics still have their place within parts of the organization, but a DevSecOps unit should begin tracking and aligning with these key, shared metrics.

There are other relevant metrics as well, but these are the foundational building blocks of a successful DevSecOps approach.

What security metrics do you collect to show you have a successful AppSec program? (Select all that apply)

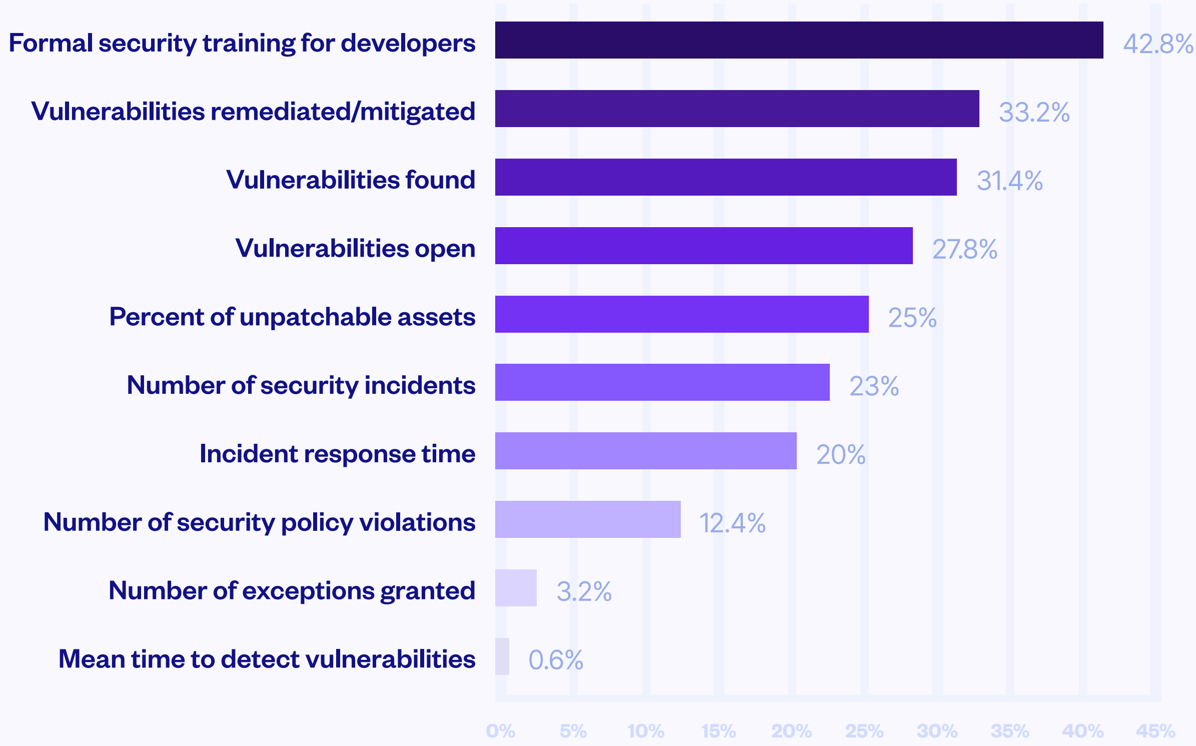


FIGURE 7

↘ As you can see, even MTTR for critical vulnerabilities is only being tracked by 43% of developers. It's clear that we still have a long way to go in terms of getting teams aligned.

RECOMMENDATIONS

Identify what DevDecOp metrics trackable in your system- and what isn't .

How much work would it take to start tracking these metrics?

Do you have the right technology to do so?

How can you get your security and DevOps teams together to figure it out?

↘ This is the second stage of working together within our maturity model.

Once developers can address security concerns within their own workflows (integrations), it will be time to focus on better alignment. Metrics are a great next step in that conversation.

DevSecOps Key Requirement #03

Security Education

Performance is firmly engrained in DevOps culture. However, this wasn't always the case in development – it's taken almost fifteen years for DevOps to become the norm. Culture changes take time.

The good news is that it can and will change.

We can bring about this change through security education. Unfortunately, as we're all aware, universities and coding bootcamps don't train developers to code security – something the industry has been complaining about for years. Apart from the few developers who are formally trained in secure coding (according to our survey, only 23% are providing secure coding training to developers), most have only learned security through experience. Or worse, they haven't learned any security at all. You probably know that one dev on the third floor who got really good at figuring out XSS vulnerabilities after a particular incident, or maybe there was the small group of devs that helped during log4j, but otherwise the knowledge is inaccessible and tribal.

RECOMMENDATIONS

It's not really the developers' fault though – they haven't received the proper training. So, what do we do to fix this?

Give them options!

01 **Formal security training**

02 **Just-in-time training**

03 **In-line support**

Formal training should be structured, personalized, and modular. This helps developers understand vulnerabilities at the highest level, gives them broad knowledge of the types of vulnerabilities, and helps developers learn how vulnerabilities manifest in code. Just-in-time training is generally offered through AppSec tools in one form or another.

It could be as simple as providing remediation guidance alongside vulnerability information or giving a link to a formal training module. In-line support is relatively new and requires the support of cutting edge AppSec vendors deploying GenAI tools to support developers. Here tools can scan code in line and generate suggested code snippets as fixes while a developer codes. This helps developers maintain their workflow and reduces toil on remediation.

For the most part, developers agree that security education is something they need.

How Important is Secure Code Training?

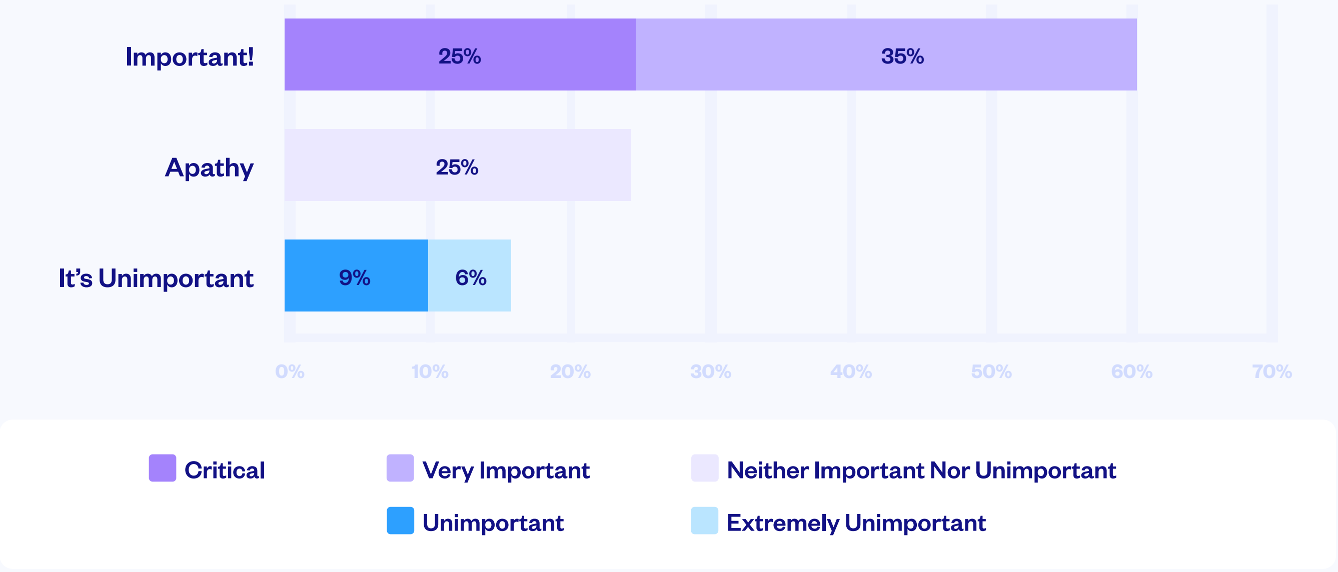


FIGURE 8

According to our survey, 46% of developers are not using secure coding training, but plan to in the next twelve months. This is encouraging news, especially given that in Figure 9 you'll see that 0% said that no program was in place.

Does your organization offer any programs or tools to train you or other developers in secure coding? (Select all that apply)

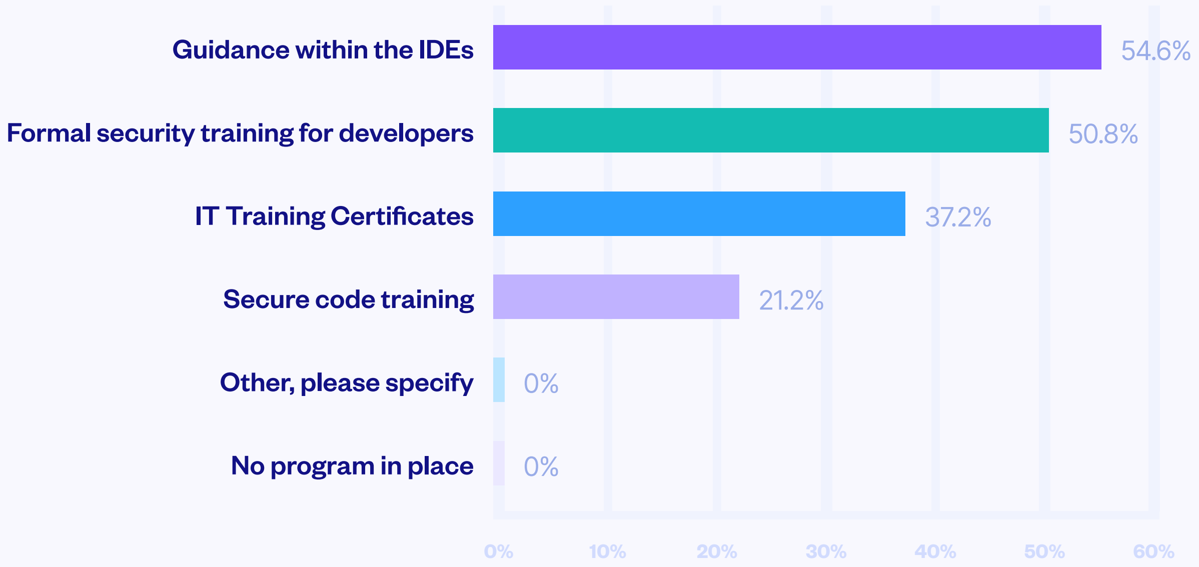


FIGURE 9

DevSecOps Key Requirement #04

Security Velocity That Matches DevOps

How often your developers release must drive the rhythm for everything else, including security.

That poses the question: how does security fit into that release schedule? Checkmarx is an AppSec vendor and if you asked one of our salespeople about speed, we'd review questions like:

- How fast does your SAST/SCA/DAST/etc. tool scan?
- How fast do the results get into developer queues?
- What does your tool do to reduce noise so developers can move more quickly?

These are good questions to ask when comparing tools during a purchase cycle and these are also great for internal requirements building. Can this tool do these things for me? You should ask these questions of your vendors because these can all help reduce developer toil.

But...do positive answers to these questions get you to do DevSecOps?

No. Because DevSecOps is people, processes, and *then* tools. When you think about speed, you need to think about it from from a DevOps perspective: what is the business goal?

The business goal of DevSecOps is to quickly deliver secure features.

This requires a mindset reorientation from both DevOps and AppSec teams.

In Figure 3, 93% of developers said that their organizations had been breached due to a vulnerable application. We asked those same developers why the vulnerable code was deployed and almost 1 in 3 responded "to meet a business, feature, or security-related deadline." We also asked the same developers what their biggest concerns are around developing secure code and 61% answered "security getting in the way of the development process."

If your team rushes to release an app or feature quickly, but it later gets breached, resulting in fines, brand damage, and an emergency patch to take it offline – did DevOps really succeed? Was it truly fast?

No. Not really.

The original agile manifesto focused on responding to business needs: Did the business need that breach? No. It needed a secure application.

Speed is how quickly you as a team solve problems. It's mean time to remediation and training developers to understand risk and security teams to understand how they contribute to developer velocity. It's about redefining high-performing code as secure code, where the biggest risks have been mitigated. If your organization does not encourage this kind of teamwork, it cannot do DevSecOps. Everyone needs to be on board with everyone else's needs for DevSecOps to work.

RECOMMENDATIONS

We recommend forming your first DevSecOps team with a group of people already willing to build common goals.

> **Remember that DevSecOps is about taking the needs and outcomes of security – specifically risk management and mitigation – and integrating them seamlessly into the process and culture of DevOps.** This must start with a small, cross-functional group of people who are dedicated to making this positive change within the organization.

DevSecOps Key Requirement #05

Automating Security Processes

Automation is the last requirement, because even though it's a core tenant of DevOps, from a cultural perspective you absolutely cannot successfully implement security automation without everything else coming together. Reducing friction between DevOps and security teams requires everyone to be on the same page. Without this high level of trust, security automation simply won't work.

↘

In our developer survey, we asked about the importance of security automation. While the results weren't quite as strong as the excitement around integrations, they still showed a strong preference for automation.

How Important is having your AppSec solution be integrated/automated into the DevOps pipeline?

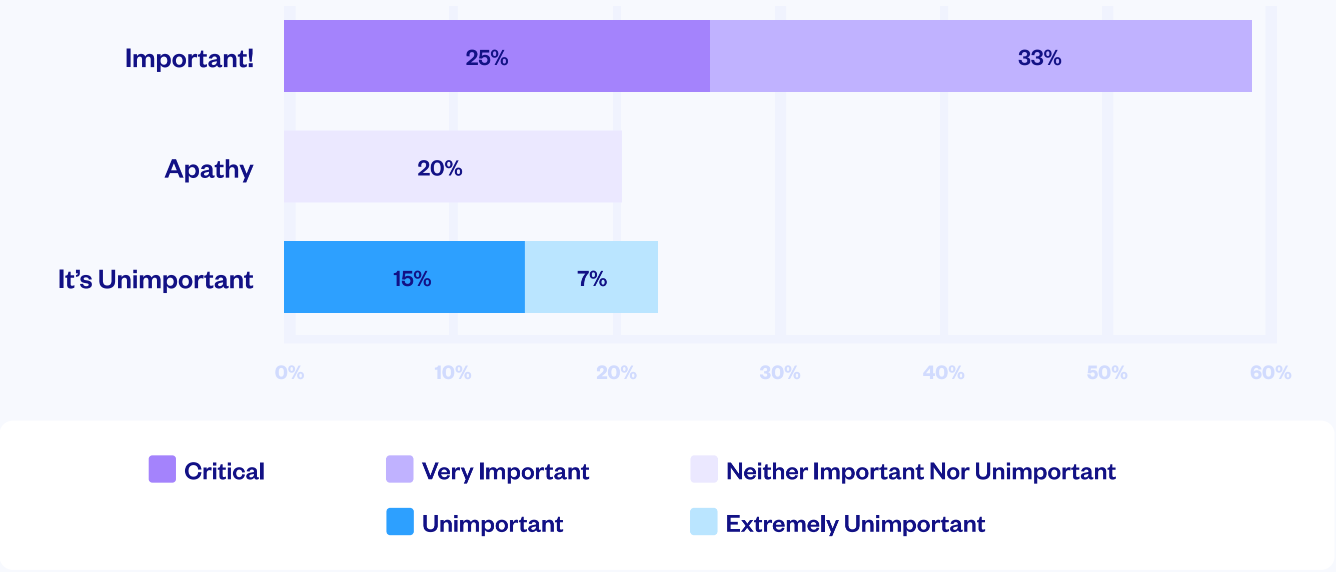
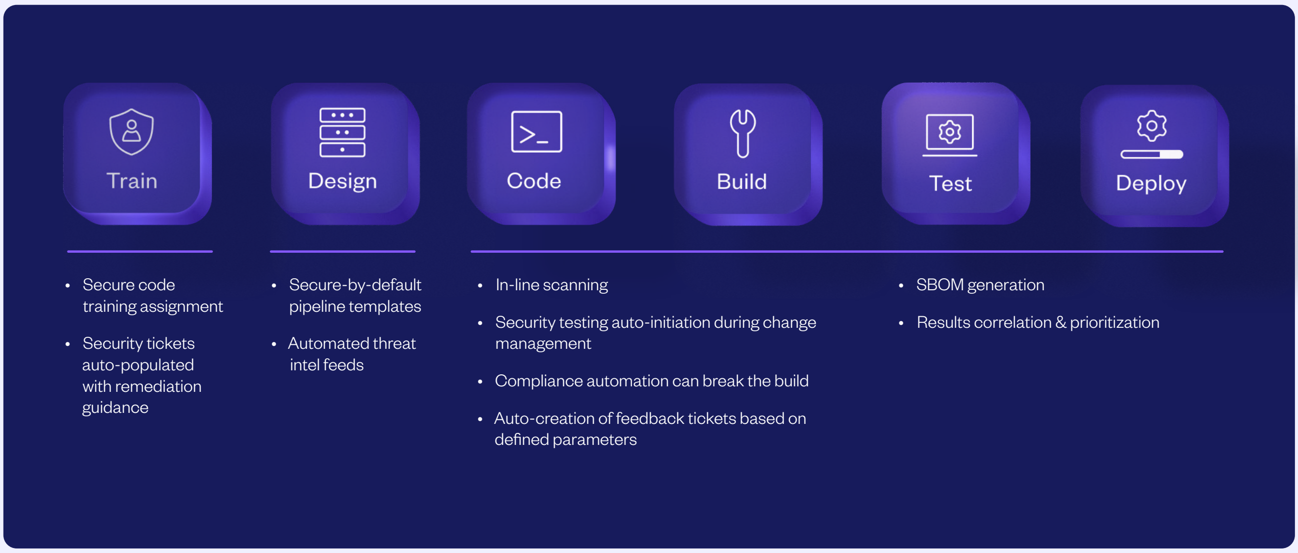


FIGURE 10

We'll explore this question further in a future survey, but for now, we believe more developers rated automation as "unimportant" because they may not fully realize how beneficial it can be.

Luckily, at Checkmarx we have great imaginations and a lot of hands-on experience helping customers build out their DevSecOps operations. Let's look at what's possible.

RECOMMENDATIONS



- Secure code training assignment
- Security tickets auto-populated with remediation guidance

- Secure-by-default pipeline templates
- Automated threat intel feeds

- In-line scanning
- Security testing auto-initiation during change management
- Compliance automation can break the build
- Auto-creation of feedback tickets based on defined parameters

- SBOM generation
- Results correlation & prioritization

FIGURE 11

In Figure 11 we lined automation options up against the software development lifecycle (SDLC). There are plenty of places to start, depending on what you and your team believe is the easiest to implement and will deliver the biggest results for your organization.

While you're looking at what can be automated, it's also important to address what cannot be automated. That's remediation. While you can do a lot to automate the scanning and triage processes to present developers with the highest impact vulnerabilities to fix, we still can't automate remediation safely.

But can we help make remediation as fast as possible for developers?

Of course. Just review requirements 1-4 of this report!

Provide developers with the integrations they need. Align security and DevOps on KPIs to ensure security is focused on supporting developers in making high-impact fixes. Educate developers so they're prepared when security issues arise, and embed security into your DevOps culture, where risk management and mitigation goals are valued by the entire team. Then, automate everything possible to free up people to tackle the biggest challenges.

If this sounds a lot like the original DevOps philosophy of uniting development and operations – that's because it is!

You did it before, and you can do it again.

Conclusions

The Business Requires Secure Applications to Succeed

ChatGPT has it right when it keeps saying that **“the development of applications is rapidly evolving.”** What is in the code (custom, open source, AI-generated code, APIs, etc.), how it’s developed, packaged, and deployed – it’s all changing every year. And development teams are living those changes.

You have the opportunity now to think more about security, and to align DevOps and AppSec teams to build out DevSecOps pipelines. If your business doesn’t already demand it, it will soon.

In this report we presented:

- A DevSecOps maturity model
- Five key requirements to start building DevSecOps in your organization
- A lot of survey data to back it all up!

I urge you to take some of these key concepts back to your organization and think about how you would do your job differently if:

01 **Everyone believed that high performing code was secure code**

02 **AppSec tools were developer tools that must be part of their workflow**

03 **DevSecOps was a culture problem that your organization needed to solve**

↘ **Like it or not, our modern world demands more attention to security and DevSecOps is one way to make sure your organization doesn’t fall behind.**

Checkmarx

Checkmarx is the leader in application security and ensures that enterprises worldwide can secure their application development from code to cloud. Our consolidated platform and services address the needs of enterprises by improving security and reducing TCO, while simultaneously building trust between AppSec, developers, and CISOs. At Checkmarx, we believe it’s not just about finding risk, but remediating it across the entire application footprint and software supply chain with one seamless process for all relevant stakeholders.

We are honored to serve more than 1,800 customers, which includes 40 percent of all Fortune 100 companies including Siemens, Airbus, Salesforce, Stellantis, Adidas, Walmart and Sanofi.