WHITE PAPER

Dev to DevSec:
The Five-Point Tactical Guide for Secure Developers

Waterfall, Agile, DevOps... it seems that every few years, a new methodology is born for optimum software creation within an organization. While these processes all have their strengths and weaknesses, the streamlining (and, er, previously absent red tape) they bring can feel like somewhat of a hindrance to the main goal of the developer:

BUILDING KICK-ASS FEATURES.

Many moons ago, when the world moved at a seemingly slower pace, organizations would task their developers with creating new applications without input from security or operations teams. They were free to write their code, putting their digital stamp on the world with a laser focus on functionality, innovation and releasing new features. Once it was complete, they would normally hand it off to an operations team to install in their company's production environment. Security was an afterthought most of the time, if directly considered at all. The software looked great, functioned well, and the developer's job was done. Until one day it wasn't, and security debugging started to interrupt free-flow coding in a business environment, with increasing risk to the company and end-users.

SECURE CODE WARRIOR

aws

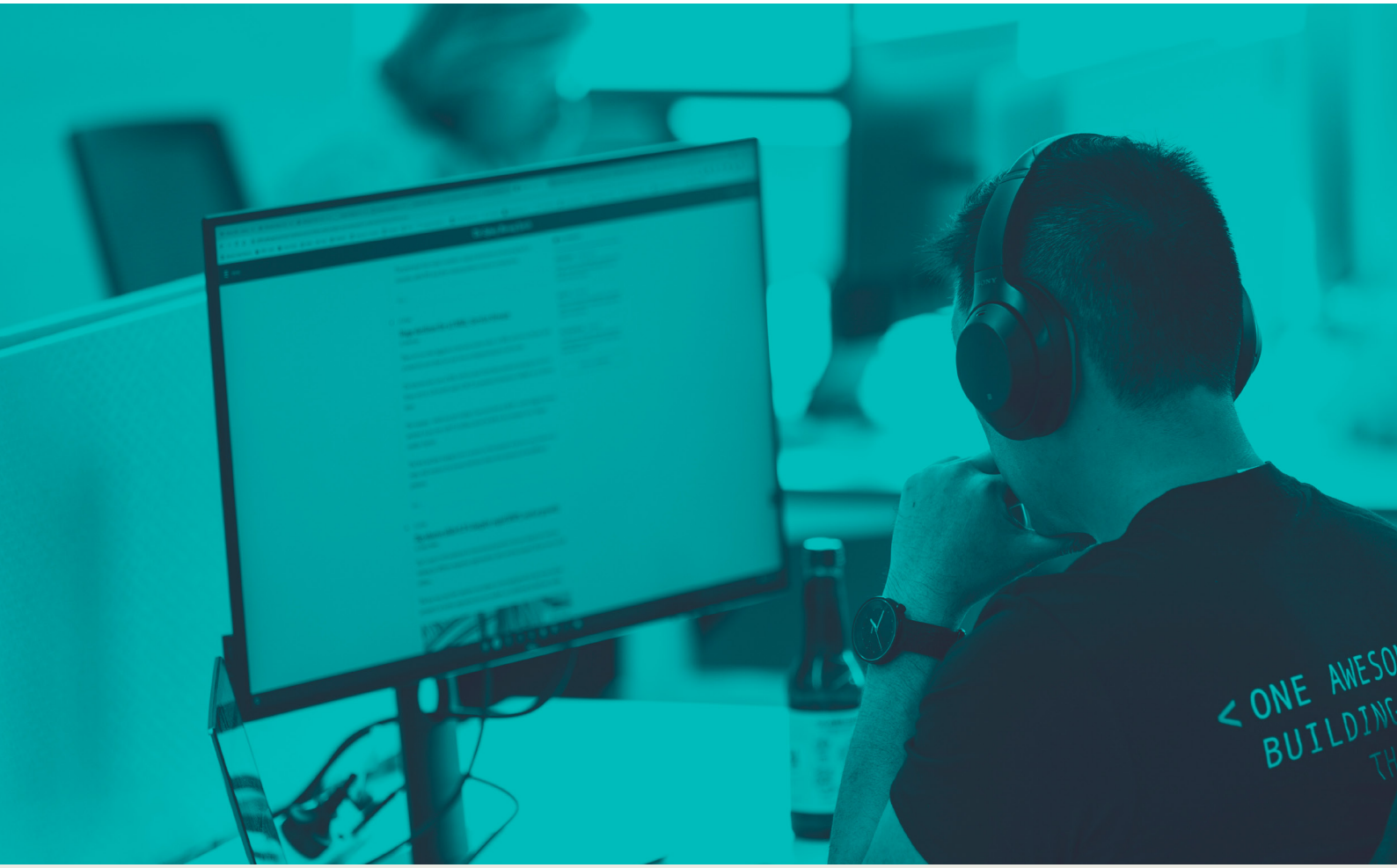
A new era of security headaches
(and no particular ownership)

When security flaws and program vulnerabilities were discovered, which sometimes only happened after a user exploited an application, the operations team would pull the app and send it back to the developers to patch. By that stage, developers were already working on new projects, effectively having to pause those efforts in order to fix old problems. They did their best, but going back and shoehorning security features into an application that a development team completed a year ago is hardly efficient. And sometimes fixing one problem would create many others, ensuring the whole frustrating cycle would continue.

Does this seem familiar?

This bogged down new application development twenty years ago, and this pain continues today. Organizations are still exposed to an unacceptable risk by deploying potentially unsafe code, and developers remain hamstrung with the responsibility to continue feature-building to a deadline, while debugging old projects when required. Mitigation plans had to be implemented with haste, and updated often to deal with an ever-increasing demand for software.

Enter, DevOps.



DevOps: A methodology with more room to shift left

The DevOps movement does address this onslaught of code-building and feature delivery, namely by ensuring the development team is involved with key operations teams from the beginning. Taken as a whole, DevOps can be a successful methodology. It certainly streamlines the development process, a necessity these days when organizations are sometimes creating hundreds or thousands of new applications every year.

However, this rapid pivot to keep up with demand isn't the silver bullet to a growing security problem; DevOps isn't perfect[1]. Having developers and operations teams work together helps to make application deployment more efficient, but doesn't fully lock down every security issue, so when they crop up it still puts a lot of strain on developers who already have many plates to spin. Dedicated security professionals are required, who are trained in how to spot and stop the latest vulnerabilities, security issues, and exploits (and they usually have a tech stack to support their efforts). That is why many organizations have added application security groups, or AppSec, to their development process.

With the security pressure diverted to a specialist team, that's all fixed, right? Over two decades of inefficient, disruptive debugging for developers can finally come to an end, and they can get back to their main priority: building awesome features.

Bzzt, wrong. But there is hope.

With the security pressure diverted to a specialist team, that's all fixed, right?

1. Danhieux, P. (2019) Why DevOps Implementation is Often Unsuccessful (and How You Can Fix It)

DevSecOps: Security as a shared responsibility from the start

Right now, we have some organizations that are working within a DevOps environment, but with a separate AppSec team. And some groups have not yet made the plunge to DevOps at all. Perhaps they are still coding using something like Agile, which is efficient, but doesn't take security into account as a primary focus.

The main oversight in many of these previous approaches is the lack of security teams' involvement from the start. With zero-day exploits alone tipped to reach one per day by 2021[2], there is an urgent need to marry security best practice with the enormous amount of code being produced, and to do it as early as possible. More code means more vulnerabilities, and so an updated methodology was born: DevSecOps.

In all cases, the end goal should be evolving into a full DevSecOps program.

DevSecOps has become both a software engineering tactic, and a culture that advocates security automation and monitoring throughout the software development lifecycle. The primary goal of DevSecOps is to break down barriers and open collaboration between development, operations, and security teams so that all of them can contribute to the creation of new software, and organizations can deploy new apps with secure, working and efficient code. But make no mistake, security is the primary goal.

These days, deploying a secure application is just as important to most organizations as whatever core function the app will be conducting. Coding an application that works fine, but which exposes a business to a potential exploit, is just as much a failure as making an app that doesn't function properly.

However, finger-pointing and blame games are a waste of time. Security is everyone's responsibility, and every department needs the resources to take ownership. What's more, the development team can play a critical role in reducing the attack vectors in software from the very beginning, eliminating common vulnerabilities and saving a whole lot of security debugging later.

Think about the processes you currently work with to create software; how well would your team handle this approach? How aware are you of how you, the developer, would fit into a DevSecOps scenario?

2. Cisco and Cybersecurity Ventures (2019). 2019 Cybersecurity Almanac: 100 Facts, Figures, Predictions And Statistics (Accessed 2020).

If DevSecOps is so good, why isn't everyone doing it?

Moving an organization to an efficient DevSecOps program isn't always easy; it's equal parts an operational and cultural change for most companies, but the end result is worth it. In today's cyber environment, hackers are continuously scanning for vulnerable code. Insecure apps can be attacked within moments of their deployment, destroying a team's hard work in an instant (not to mention the company's reputation).

With humanity's insatiable demand for amazing digital experiences – to the tune of approximately 111 billion lines of code being written each year[3] – a sustainable method of project management is necessary to keep everyone, especially developers, sane and productive. Put simply, anything less than a formidable DevSecOps operation at this point in time is a business risk.

In this white paper, we will provide a FIVE-POINT TACTICAL GUIDE that developers just like you can follow to move from 'Dev' to 'DevSec', becoming a higher standard of engineer with the security awareness to make DevSecOps a roaring success.

3. Kerravala, Z. Network World (2017) Cisco to network engineers: Get comfortable with software. It's here to stay. (Accessed 2020).

Making the move from Dev to DevSec

If you're a developer wondering where you might fit in within the DevSecOps process, you've come to the right place.

It's true that developers, in general, are extremely clever people, many of whom have a love for problem-solving and learning with hands-on experience4. It's also true that many developers haven't had the most positive experience when it comes to dealing with security within their working life. Hearing from the AppSec team is rarely an uplifting experience; they tend to only come knocking when code is broken at a security level, effectively telling the creative development geniuses behind the software that their baby is ugly.

Security training and awareness is hit-and-miss, if it's implemented at all. If you're a developer who feels left behind and a little short-changed in the security upskill department, you're not alone. According to Veracode's DevSecOps Global Skills Survey[5], just 24% of respondents were required to participate in security training as part of their education, and a staggering 86% of respondents felt their organizations were not investing enough in application security training.

The simple fact is, without secure developers, the whole DevSecOps process falls apart. AppSec specialists are way outnumbered by software engineers and the amount of code being produced, and developers who are security-aware are crucial in bridging the security knowledge gap.

Secure developers are valuable, sought-after and stand out amongst their less-trained peers. While the ultimate goal of building awesome features will remain, with security front-of-mind, these features can be secure by default (and lead to fewer disruptions down the track; after all, who wants to hotfix something that was forgotten about months ago?). For software and digital innovation to be truly great, it must be secure.

And moving from "Dev" to "DevSec" isn't that big of a jump. With these five steps, you will have the foundation to start your security journey and upskill efficiently.

4. Singh, J. (2019) Contextual, Hands-On Learning: The Supercharged Way to Train Your Brain for Security.
5. Zorabedian, J. Veracode (2017) Veracode Survey Research Identifies Cybersecurity Skills Gap Causes and Cures (Accessed 2020).

What successful DevSecOps teams share

Defining a single strategy to create a DevSecOps team that works in every organization and every situation is virtually impossible. It's a bit like examining a winning professional sports team and then trying to mirror their success by building a second one that is exactly the same. There are too many variables for an exact copy to work every time.

DevSecOps is as much a culture as it is a program, therefore not every blueprint will work perfectly within every company (this should serve as a timely reminder to check the pulse of your overall security culture for signs of life, and opportunity to improve it). But the majority of successful DevSecOps groups, like winning sports teams, will share many of the same elements.

Fostering those key elements – modified as appropriate for your role as a developer within the organization – will help you forge your path in a winning DevSecOps team that can both improve application development efficiency and security.

These are some of the most important elements that highly successful DevSecOps teams share:

1. Successful DevSecOps teams recognize that security is a shared responsibility.
2. DevSecOps teams thrive best in a highly supportive environment, where everyone – from individual team members to the supervisors and management – embrace the culture and support collaboration efforts.
3. The top DevSecOps teams train consistently and improve their skills as they work.
4. Successful DevSecOps teams employ a suite of tools that are tailored specifically for the job at hand, and can be understood and used by developers, operations teams and security personnel equally.
5. The most successful DevSecOps teams are transparent with one another. They understand each team's core functions, strengths and limitations, and they play to those strengths. The effort is made to ensure each piece of the DevSecOps puzzle has the breathing space to do their best work.

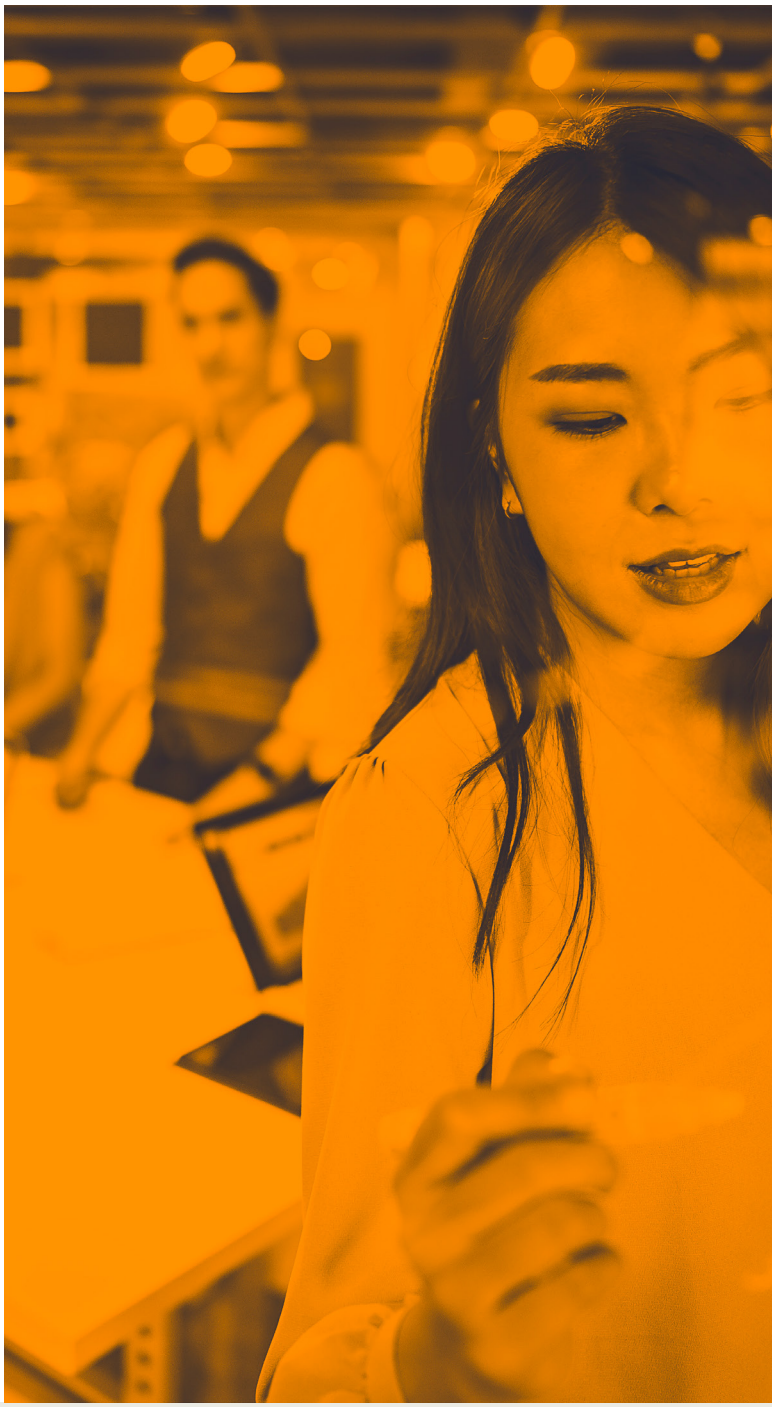
These don't represent a complete list of everything that successful DevSecOps teams share, but they are some of the most important elements. If your company takes meaningful steps to help you thrive, it will go a long way to making sure your DevSecOps program starts strong and maintains efficient, secure coding and application development for years to come.

So, how are we going to get you, the next security superhero, DevSec-ready and making a big, positive mark on your organization's software development process? READ ON

Implementing successful DevSecOps elements for developers

Although creating a DevSecOps operation and culture within your organization is the overall goal (and not one that should be shouldered by one team or one individual alone), the process by which you as a developer, and a counterpart in the AppSec team will get there won't be identical. Developers will initially have a completely different outlook and skillset, with priorities mostly at odds with the security team. Once aligned and given the tools to succeed, however, their end-goals with implementing DevSecOps will be similar.

The following is some practical advice that you can follow to become an integral part of the DevSecOps program in your organization. By studying tested and proven elements of engineers that have effectively operated in a DevSecOps environment in the past, you can make the transition to an invaluable DevSec-powered member of the team.



Five-point tactical guide

1. The DevSec mindset recognizes that security is a shared responsibility.
2. Seek to be part of a highly supportive environment.
3. The top DevSecOps teams train constantly and improve their skills as they work.
4. Successful DevSecOps teams employ a suite of tools that are tailored specifically for the job at hand.
5. The most successful DevSecOps teams are transparent with one another.

1

The DevSec mindset recognizes that security is a shared responsibility

While paradoxical, this step is both the easiest and hardest hurdle for developers to get over. It's easy in that the statement makes complete sense – it's a huge job, and it truly does "take a village". But it's also difficult to accept because developers have traditionally been graded more on the speed of their coding, as opposed to the secure qualities of that code. If an application worked, looked good and was delivered in a timely manner, then that was considered a victory. If the application was later kicked back to them because of security problems in the operational environment, it wasn't their fault. It was just those annoying AppSec personnel giving them more work and slowing down their efforts with any new applications they were already in the midst of coding.

Think about whether this resonates: The fear shared by most developers is that implementing DevSecOps will add yet another layer of responsibility to their already full plate. How will they be able to maintain their high production volume and speedy coding record, if they now must develop security and "bake it into" every application they create? It is, understandably, a stress-factor.

The shift you will require to open your mind to the new reality that security is at least as important as an application working correctly, will not be overnight and will cause a little stress. If the software works well, but exposes a company to attackers through a security vulnerability, then deploying that program is far worse than missing some arbitrary deadline.

In this environment, the best coders will always be those who are security-aware and can create good, secure code. In fact, the demand for[6] talented DevSecOps engineers is skyrocketing with no end in sight. It is beneficial for you to embrace DevSecOps as both a way to protect the organization, and a pathway to improve your skills.

Becoming a DevSecOps engineer will make you far more valuable than an average coder. You absolutely will become a highly sought-after employee in an increasingly vital field, and you'll be able to negotiate a better salary as a result.

2

Seek to be part of a highly supportive environment

Due to the fact that development teams are the magicians creating new applications, while the operations and security teams are supporting those efforts, much of the success of a DevSecOps program will come down to the attitude of the developers.

In a functioning DevSecOps environment, developers are ranked by their team supervisors more on their ability to create secure software as opposed to their ability to simply code quickly. They also allow adequate time for the whole development team to use the security tools available to them, including frequent bursts of training as part of the job.

If you and the team embrace the new security training and improve your work, there should be recognition, reward and public acknowledgment of your commitment to a higher standard of code.

In addition, senior developers within teams that constantly show an aptitude towards creating secure code should be encouraged to become security champions. Those champions should then be empowered to work with AppSec teams making security decisions about the applications being created. They should also become mentors and trainers for other developers, and receive rewards and praise for successfully taking on their new roles.

You as a standalone developer can't foster a highly supportive environment across an entire company, but the reality is that the development team is the first and arguably most important group when it comes to creating secure software. What you can do is commit yourself to higher security standards, supporting others and working positively with your new, equal partners from the operations and security teams.

If your organization has implemented a DevSecOps program, and they are not helping you work towards the "DevSec" part of the process, this is worth flagging with senior managers. Success depends on support, and being given enough time to operate efficiently in this relatively new process.

6. Rowe, S. DevOps.com (2018) How to Become a DevSecOps Engineer (Accessed 2020).

If you and the team embrace the new security training and improve your work, there should be recognition, reward and public acknowledgment of your commitment to a higher standard of code.

3

The top DevSecOps teams train constantly and improve their skills as they work

Unless individual developers were highly focused on cybersecurity as a personal interest, most coders will have a lot to learn when becoming part of a DevSecOps program. The key to that success will be employing a training program with a proven track record of fostering better secure coding skills.

Training should teach you to look at the software you're designing through the eyes of someone who is trying to exploit it

A good training program should reinforce the concept of thinking like a hacker. It should teach you to look at the software you're designing through the eyes of someone who is trying to exploit it. It should empower you with the skills to anticipate what a hacker wants to do, and what methods they will use to accomplish their insidious goals. It should then give you the knowledge to write secure code that thwarts those efforts and plugs any vulnerabilities.

Ideally, secure code training should be continuous, not a "one and done" compliance exercise that dates quickly and isn't contextually helpful. Some of the best developer training programs use real-world code, the kind you'd work with in your day job, in the language and framework you actually use.

4

Successful DevSecOps teams employ a suite of tools that are tailored specifically for the job at hand

Going from not worrying much about security, to building it into every piece of software can be quite a transition. Having good training will go a long way toward educating you about common vulnerabilities and teaching you think like attackers. However, to effectively put that knowledge into practice requires good tools. The best DevSecOps teams have access to a suite of them, because no singular tool is perfect in every situation, with the ability to scan for and squash every possible security bug in every conceivable environment.

For example, both static application security testing (SAST) tools and their close cousin, dynamic application security testing (DAST) tools, are designed to help find security flaws hidden inside code, often before it can get to a production environment. But they approach security testing in different ways and at different stages of code development, so having both available would be beneficial for developers on DevSecOps teams.

You should also seek to be aware of security policy and best practice rules that are specific to your company, like which secure code libraries can be used when designing any new software (within the OWASP Top 10, using components with known vulnerabilities still sits at number nine[7]).

The key is to have the knowledge and the right toolset across the team, using whichever one is most appropriate for the task at hand.

7. Rowe, S. DevOps.com (2018) How to Become a DevSecOps Engineer (Accessed 2020).

5

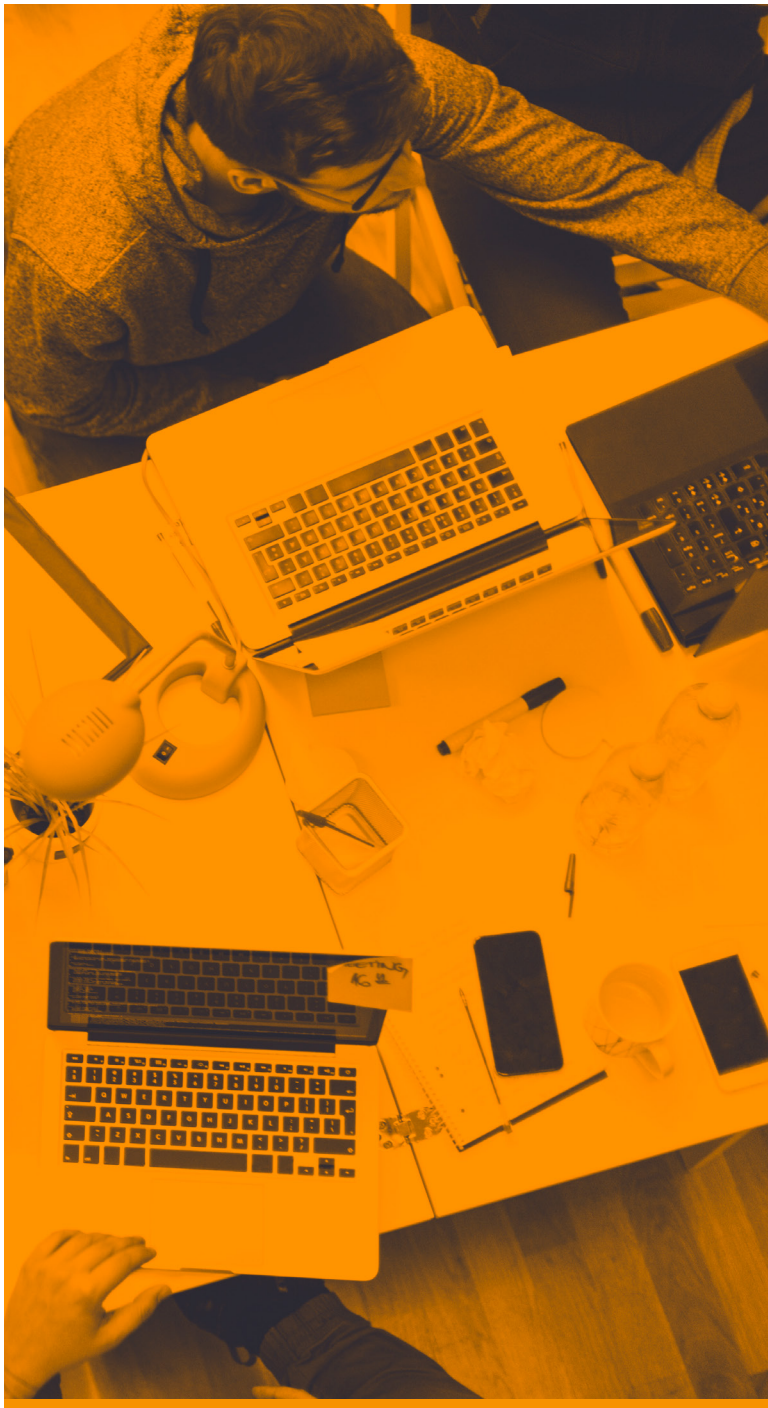
The most successful DevSecOps teams are transparent with one another

Transparency is the thread that pulls the other key elements of a successful DevSecOps program together. It means that every group should understand their new role, their strengths and limitations, and ideally be working with the same toolset and training programs as other teams in a collaborative environment. Structurally, you should be empowered to talk with operations and AppSec teams as equals, and in a way that promotes universal understanding. In a blame game, nobody wins, and that also means that others in the DevSecOps environment should seek to understand your challenges too.

One specific area of concern regarding transparency for developers joining a DevSecOps program is the need to ensure that their new roles as security practitioners do not detract from their primary responsibility of creating applications.

Transparency is important, but you need to be careful that this new focus on security does not overwhelm your role as a coder. The tools put to use should not only be transparent, with operations and AppSec teams working from the same playbook, but also designed to support rather than impede your coding efforts.

With the right tools, training, and transparency, developers will be free to apply their best security development practices to every application they create.

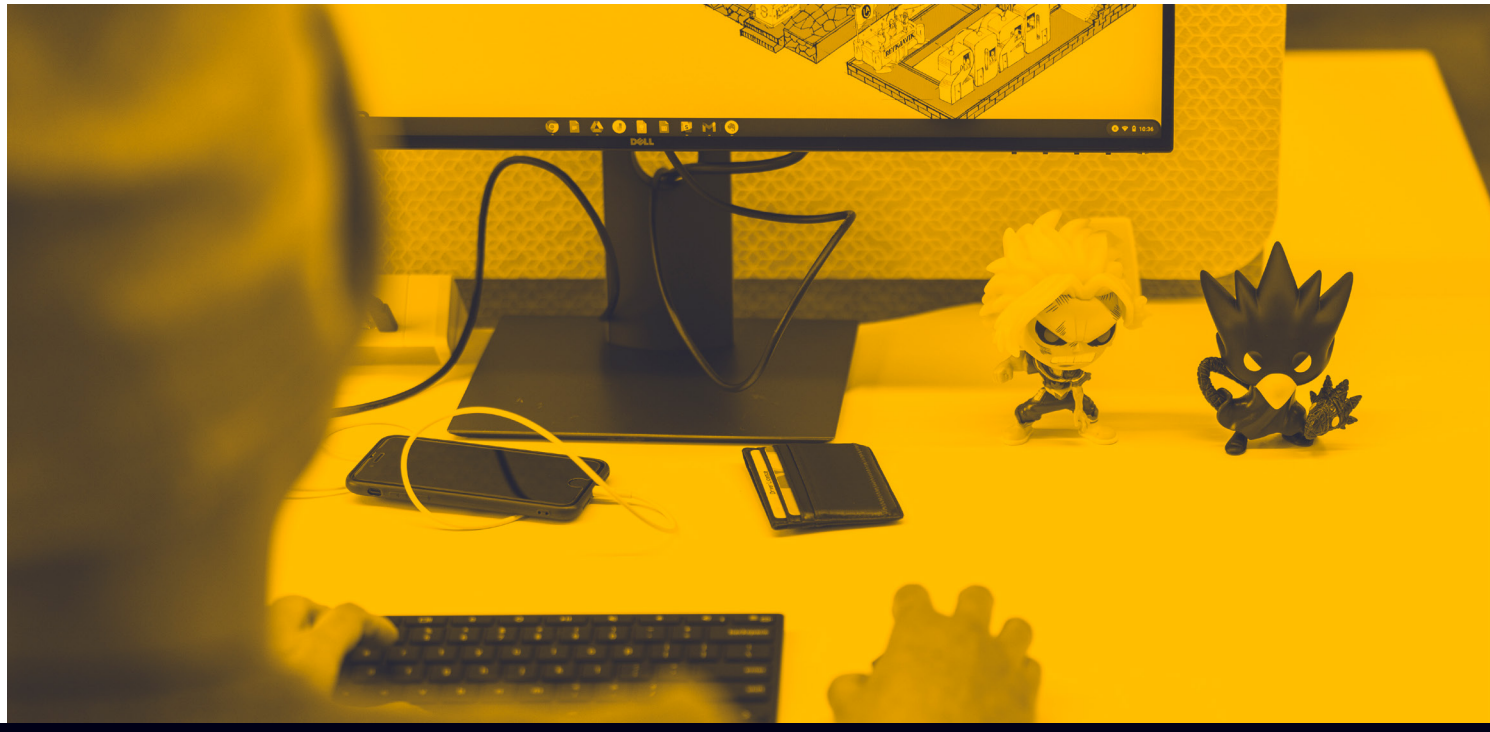


Transparency is important, but you need to be careful that this new focus on security does not overwhelm your role as a coder

Are you ready to begin with DevSecOps?

We hope that this white paper has given you a lot to consider as part of your role in an efficient DevSecOps program, one that supports the creation of secure applications. Lots of organizations have made their move to DevSecOps, but the journey getting there is not always easy.

It will take time to train up, it will take a mindset shift and an all-hands effort to build a positive security culture ... but you can do it and reap the personal and practical benefits.



ABOUT SECURE CODE WARRIOR

Secure Code Warrior is the developer-chosen solution for growing powerful secure coding skills. By making security a positive and engaging experience, our human-led approach uncovers the secure developer inside every coder, helping development teams ship quality code faster.

Through inspiring a global community of security-conscious developers to embrace a preventative secure coding approach, our mission is to pioneer a people-first solution to security upskilling, stamping out poor coding patterns for good.

info@securecodewarrior.com

securecodewarrior.com