eBook

Your handbook to developer-driven security and agile learning

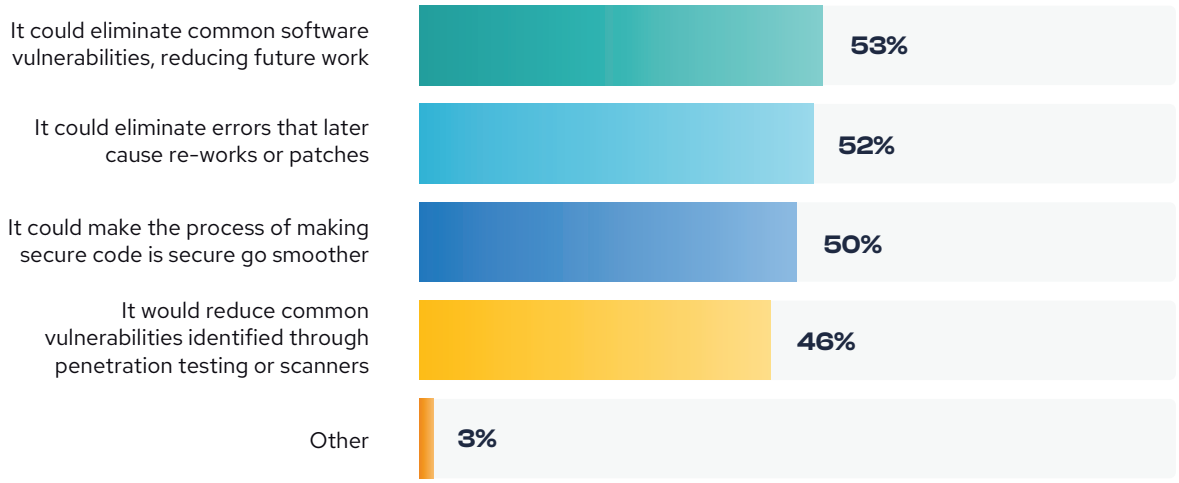
SECURE CODE WARRIOR

Agile learning in the age of DevSecOps

The migration to a DevSecOps approach has enabled organizations to move away from rigid waterfall releases that were often late and miss-scoped to flexible sprints that could meet fast-changing customer requirements. An agile approach breaks work into small pieces that can be executed quickly across teams to move the needle in an incremental way with successive layers building to produce high-quality products. Just as agile practices have overtaken the waterfall approach, agile is now revolutionizing developer security training. It's time for traditional training practices to evolve. To learn more about agile learning, read our blog.

Only 14% of developers in a survey conducted by Secure Code Warrior[1] said that security was a top priority for them, which means they are leaving a huge portion of the work of vulnerability remediation for AppSec teams to find. DevSecOps was meant to facilitate shipping code, securely and successfully, without significant delays. But why does it seem to be so difficult to do in practice?

In what way would secure coding most improve productivity?



Security breaches and vulnerabilities are extremely costly to businesses, time-consuming for developers to fix, and a distraction from more productive development. The traditional approach to risk management is reactive and wasteful because vulnerabilities are cleaned up at the end of the SDLC, when they are harder and more costly to remediate. Organizations need to adopt a proactive, developer-driven approach to security that snuffs out security breaches before they even begin. Applying agile learning principles to secure code is the best way for developers to internalize secure coding skills that they can apply right away.

It's predicted that there will be nearly 30 million software engineers by 2024[2] to drive digital transformation. Engineering resources are famously scarce and expensive, with developers commanding some of the highest salaries in the world. Finding developers with proven secure coding skills is even more of a daunting task. Most organizations work with the resources they have, with security as an afterthought. To conserve precious developer resources, organizations should consider investing in an agile learning program for secure code that helps developers produce secure code from the start and remove the wasted effort of fixing software later in the development cycle.

Secure Code Warrior has integrated agile principles into the design of our learning platform to break the mold of traditional security training with a developer-focused, agile learning experience. This in turn has delivered business value of 2x to 3x improvement across several dimensions for our customers, from reduced risk and cost to increased developer productivity.

In this ebook, you will learn how to integrate agile learning principles into the DevSecOps process, understand how developer security maturity evolves over time, and learn the best practices and processes to implement your own agile learning program for secure code.

The challenge facing developer teams today

60%[3] of developers report that they ship code faster than ever before. This may seem like a good thing, but it introduces a new risk because security is often deprioritized in order to meet tight deadlines and market demand. This pressure doesn't give developers enough time to code securely, and teams commonly emphasize functionality over security. This pressure to constantly deliver, however, actually sets teams back because of the lost productivity. Developers often do not have the time nor the knowledge to address common vulnerabilities.

> **60% of developers report that they ship code faster than ever before.**
>
> This may seem like a good thing, but it introduces a new risk because security is often deprioritized in order to meet tight deadlines and market demand.

The phases of security maturity

Every organization has to start somewhere and we recognize that there are many components to developer-driven security. It is important to think about where you are on the spectrum of security maturity as you seek to improve your overall posture and build a robust DevSecOps approach.

Building security code learning in developer teams can be approached in stages. Based on Secure Code Warrior's experience with 600+ organizations, we've identified the common practices and traits that align with specific phases of secure code learning maturity. Through our experience and research, we have identified three different phases organizations evolve to when implementing a secure coding culture:

Defining

Identified the need to define and build security maturity

Adopting

Beginning to adopt secure coding practices into all stages of the SDLC

Scaling

Have implemented a cohesive approach and created a security forward culture

Defining

The first phase usually begins with defining your goals and identifying your most common recurring vulnerabilities discovered in scanning and pen-testing audits. **This phase is all about starting to audit and identify where your organization stands today: what are the strengths and weaknesses?** What is the plan you are actually going to develop? Most organizations at this stage lack a secure code learning strategy, and think of security as a check box at the end of the software development process. In this phase, developers lack a general understanding of security knowledge at times when they need it the most.

Often organizations in this phase reuse existing code without properly scanning it, or need to consistently rework their code as vulnerabilities are exploited or discovered.

One best practice in the defining stage is to conduct a company-wide tournament with SCW that baselines the organization and gets developers curious about secure coding techniques. This gives you a starting point so you can plan for where you want to be in the future. A tournament can be very useful as a mechanism for identifying which developers need the most coaching, where their weaknesses lie, and whom evangelize security internally as a security champion.

Adopting

During the adoption phase, the work has to be done to gain your developers' trust and build their confidence in secure coding practices. This can be done through an easily accessible and secure code learning program. Developers have said they want more hands-on training, so it's crucial that you invest in an engaging program that covers the vulnerabilities your developers commonly face.

Though meeting compliance and mitigating risk is also a major goal during this phase, it's important to keep in mind that 75%[4] of bugs are missed by developers and are left for AppSec to find. Unfortunately, it's not an "if" but a "when" situation when it comes to a vulnerability being exploited, forcing people to improve their security posture. The time to start adopting a broader approach to security is now, when you can be proactive rather than reactive to address a potential security threat.

Building a program that is both fun, engaging, and highly relevant to your developer's challenges and tailored to your goals around compliance and security will help improve adoption over time.

Scaling

The last and most important phase is how to scale your secure code learning program. Broad adoption of a security-first culture can take time. **By shifting your organization's focus to developer enablement and your strategic approach to ongoing education–you will see broad adoption and scale over time**.

More and more security professionals are joining cross-functional teams and taking part in the daily tasks of software development. Yet despite increased collaboration and knowledge sharing, security teams find bugs that are easily preventable. With broader adoption and scale, you will see increased collaboration between engineering and AppSec and even the implementation of DevSecOps where security is implemented in all stages.

Rolling out a scalable secure code learning program requires support from all levels - from CISO to AppSec to Development Managers. Frequently holding tournaments or hackathons is a good way to keep security top of mind. Requiring secure code learning and certifications helps ensure widespread adoption from all developer teams.

Developers express that they are willing and able to learn about secure coding. With roles becoming more cross-functional, now is the time to invest in an agile learning program for secure code that scales to your organization's needs and helps developers upskill their secure coding abilities.

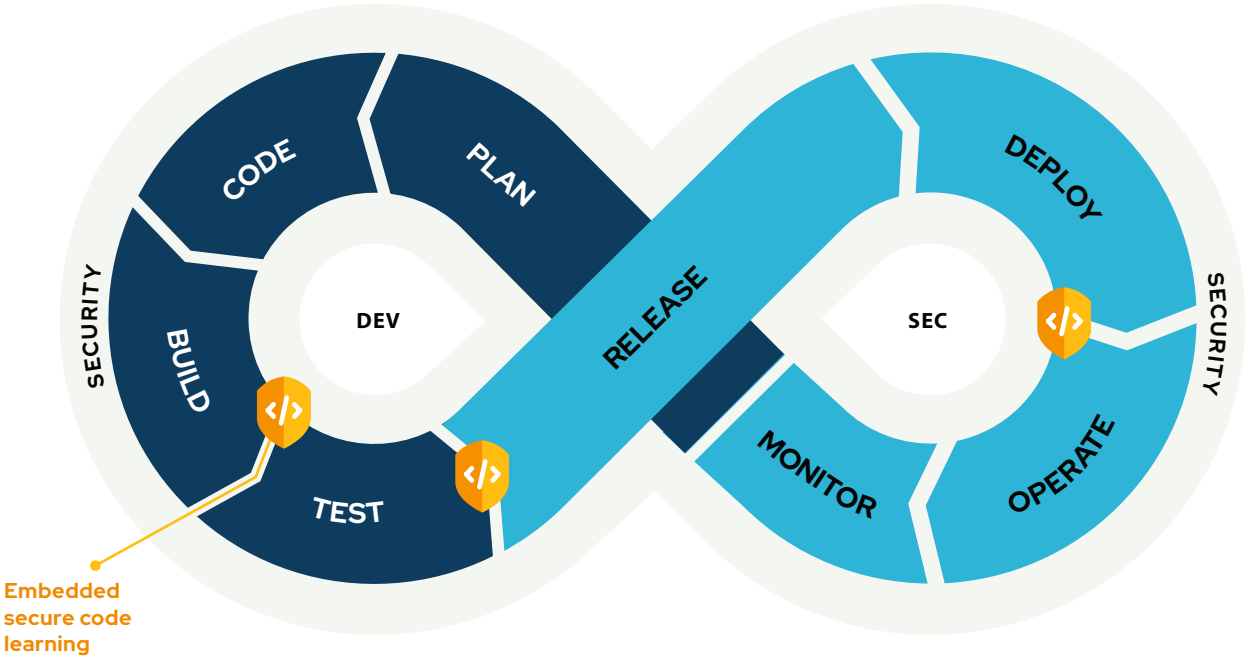
This ultimately helps them save time on exhaustive code reviews, rework, and last-minute fire drills that can be mitigated by enabling developer teams to be hands-on with their secure code learning agenda.

Implementing agile secure code learning with Secure Code Warrior

At Secure Code Warrior, we've built a platform that enables an agile approach to secure code learning.

> **Secure Code Warrior teaches developers how to identify, avoid, and fix vulnerabilities as they code, in the context of their everyday jobs.**

Our agile learning platform for secure coding gives developers the ability to learn how they want, with the most complete and reliable vulnerability content in the industry today. Enterprises choose Secure Code Warrior to add the most effective secure code learning solution as a strategic layer of defense in their security programs.



But as they say, Rome was not built in a day. Many challenges face organizations that need to first win over developers who are skeptical of training programs. They worry the content will be static, unengaging, and disruptive to their work. The steps outlined below illustrate how to win their buy-in, and eventually drive adoption in order to scale across any size organization.

1. Agile Learning is embedded

Increased developer engagement starts with micro-bursts of learning that are woven into the workflows and tools they already use. SCW eliminates context-switching and lands microburst learning at the exact point of need to ensure it's relevant, time-bound, and actionable.

This unique learning approach allows developers to build a strong foundation of skills with learning modules that keep leveling up in difficulty depending on their skillset. Developers can also transition easily from learning to doing with hands-on practice modules like Coding Labs, Challenges, and Missions that provide just-in–time coaching as tutorials on each security concept.

In addition, Secure Code Warrior's SCORM integration allows you to manage all developer learning, including secure coding training, in a single platform through your enterprise Learning Management Systems.

> **Secure Code Warrior also offers contextual tools and integrations with Jira, GitLab, and GitHub to enable just-in-time remediation and deep learning within the tools developers use every day.**

Key best practices:

- Embed learning into developer tools and environments – they will appreciate learning that feels tailored to daily challenges and that is easily accessible.
- Choose your content–with over 63 different languages and frameworks, you can create a program with content covering your organization's unique needs and common vulnerabilities in specific languages and frameworks.
- Challenge developers to self-train in a familiar environment – the training ground mimics an in-browser IDE and increases the stickiness of what developers are learning and practicing. Encouraging extra practice here will help knowledge carry over and impact production code.

2. Agile learning is data-driven

To design the right education program and goals, start with a baseline of your existing team. You can do this with assessments that help you understand areas of strength and improvement by testing your developers' knowledge around specific vulnerabilities, common threats, and in the language or framework they most commonly use.

Key metrics like the number of courses completed and time spent on courses, either at the team or individual level, help you to make strategic decisions as to how to build richer course content. However, it's important to not just measure the training but to measure the impact. Measure the number of weaknesses that get picked up in the development life-cycle through code analysis, bug bounties or classic vulnerability testing before you start the program in each team. One of the simplest ways to know your program's desired impact is by measuring the decrease in vulnerabilities developer introduce into your code base.

Recommended metrics to track:

- **Engagement** - how much time are your developers spending on training? Are they completing courses and assessments, and participating in tournaments?
- **Skills** - what are the areas of strength? What areas have you identified as needing improvement?
- **Vulnerability reduction** - have you noticed a measurable decrease in vulnerabilities during code review? Are you seeing less rework come back from AppSec?
- **Productivity** - how long does it take to remediate an issue? Have you noticed an increase in productivity or velocity of vulnerability reduction?

Engagement

- Completion rates
- Time spent
- Participation
- User activity, signups, logins
- User adoption
- Developer feedback

Skills

- Compliance rates
- Accuracy
- Language proficiency
- Belting program progress
- Areas of strength & weaknesses

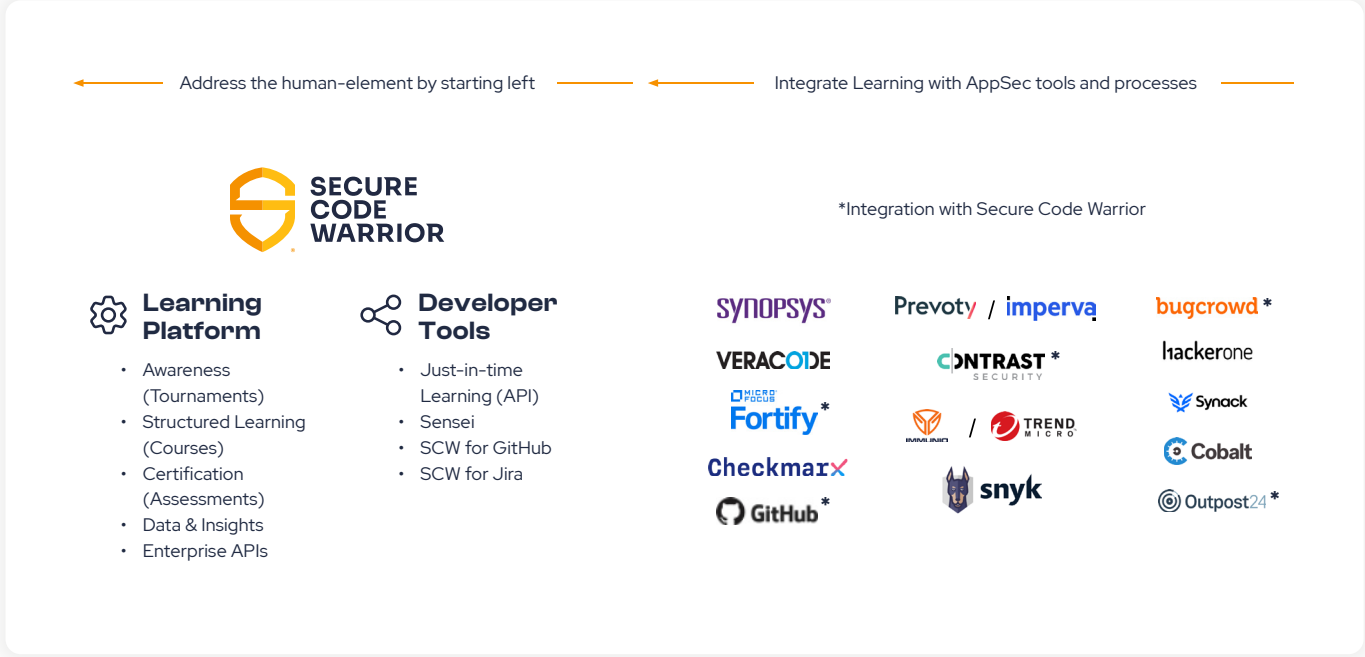
Vulnerability reduction

- Correlation between training and decrease in vulnerability during code review
- Amount of rework passed back from AppSec

Productivity

- Impact on Mean Time to Remediation
- Code productivity, where secure code = quality code

Be proactive. Start left.



The second thing to measure is the time it takes to fix a vulnerability. If it takes a developer a month to fix it, this clearly shows they need some additional training, but if they can fix it in an hour, you know they have mastered those skills.

Key best practices:

- Leverage insights and reporting to view individual, team, and departmental performance.
- Benchmarking and assessing your teams will help you verify skills and understand your areas of expertise and opportunities for improvement.
- Give incentives to become a security-skilled developer - the more you reward and recognize security champions, the more others will want to follow suit.
- Measure the impact of your training program by how many vulnerabilities are reduced and how long it takes to remediate an issue.

3. Agile learning should build a community of experts

If your organization employs dozens, hundreds, or maybe even thousands of developers - it can feel like finding a needle in a haystack to identify those who prioritize security and write the most secure code. This is a missed opportunity to nurture your security champions to mentor other developers to code more securely.

Many organizations are familiar with Hackathons, which help teams collaborate and compete to improve upon or build a new application. Why not do something similar for security? Holding regular tournaments creates a fun, competitive environment that emphasizes continuous learning about real-world vulnerabilities and aid in building a strong, security-minded culture.

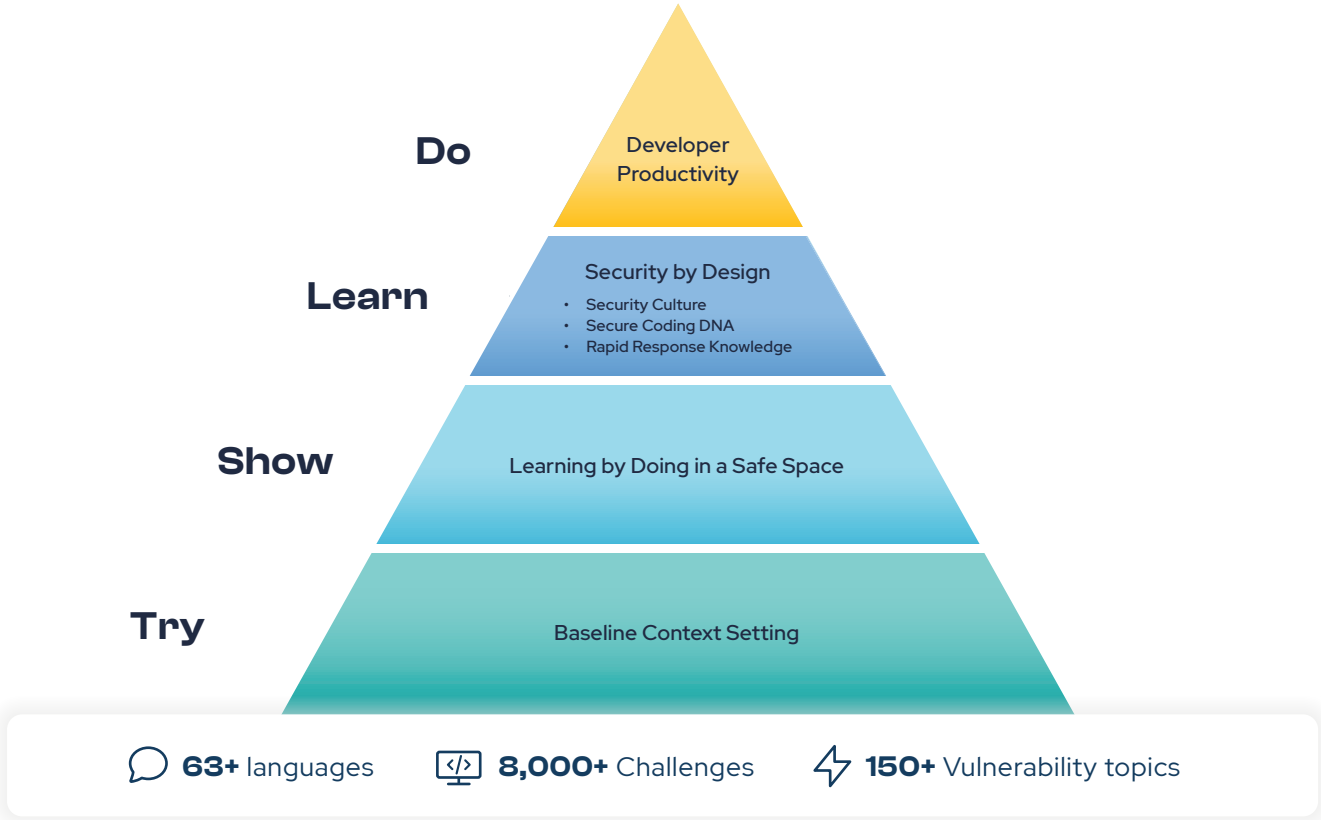
> **Tournaments are gamified, so developers can bring their competitive spirit and vie for the prize of becoming the top security champion.**

This helps you recognize those who are your best advocates for security and institute bite-sized,interactive learning to help your teams train throughout the year, not just during compliance cycles.

Key best practices:

- Test your developers' knowledge against their peers–either within your organization or across the globe with tournaments or hackathons.
- Find and recognize your security champions so they continue evangelizing and instituting a culture of secure code across your organization.
- Make it fun and consistent! Secure Code Warrior makes it easy to stay on top of training throughout the year, not just during annual compliance certification.

Realizing true business value with agile learning



How to implement security maturity with agile learning for secure code

Driving culture change isn't easy, but Secure Code Warrior helps you to identify your security champions and equip developers and organizations with the right skills to tackle today's ever-changing security environment.

> **Implementing an engaging and scalable agile secure learning program for secure code is a worthy investment because of the long-term preventative approach to security, instead of the reactive ways of the past.**

This ultimately helps mitigate costly risks of a breach, educates developers on how to find and fix vulnerabilities quickly, and facilitates better collaboration with AppSec. All of which helps improve focus on revenue-generating product development and faster time to market.

Business impact - large enterprise outcomes

Secure code learning reduces risk and our customers have seen business impact gains of 2x to 3x. Our solution positively impacts productivity, risk reduction, and prevention.

PRODUCTIVITY

2.48x

SCW educated developers fixed on average 2.48 times the number of vulnerabilities than non-educated developers, delivering a 148% improvement in vulnerabilities fixed

BREACH PREVENTION

47%

Fewer vulnerabilities introduced into production code over a two year period at a large payment processing company

RISK INTELLIGENCE & AWARENESS

2x

After SCW, developers at a major financial services company retired twice as many vulnerabilities – in half the time – freeing teams to focus on the next unknown vulnerability

RISK REDUCTION

236 → 115

Average days open for vulnerabilities in the tech debt queue fell by 121 days (a 51% improvement)

Compared to their peers, SCW-educated teams:

- Are more productive– they can fix almost 2-½ times more vulnerabilities than non-educated developers.
- They reduce risk – SCW-educated developers introduce about half as many vulnerabilities into production.
- They solve vulnerabilities faster – SCW-educated developers can lower the average days open in the tech debt queue by as much as 50%.
- Show risk intelligence and awareness. SCW-educated developers fix twice as many items in half the time so they can focus on the next unknown.

Empowering and enabling your developers helps them to catch security weaknesses earlier in the development process before they become expensive or worse, leaving vulnerabilities that can be exploited later. Secure Code Warrior helps shift your security culture left by starting at the source - your developers.

Sources:

[1] Secure Code Warrior: The Challenges and Opportunities of Secure Coding

[2] Statista: Global developer population in 2024 https://www.statista.com/statistics/627312/worldwide-developer-population/

[3] GitLab 2022 Global DevSecOps Survey: Thriving in an insecure world

[4] GitLab 2022 Global DevSecOps Survey: Thriving in an insecure world

About Secure Code Warrior

Secure code learning for today's developers

Secure Code Warrior gives your developers the skills to write secure code. Our learning platform is the most effective secure coding solution because it uses agile learning methods for developers to learn, apply, and retain software security principles. Over 600 enterprises trust Secure Code Warrior to implement agile learning security programs, deliver secure software rapidly, and create a culture of developer-driven security.

Request a demo

Try Secure Code Warrior for free

Find us on social: