Checkmarx

A Modern Approach to Developer AppSec Awareness and Training

The Checkmarx Complete Guide to Secure Coding Education

Enter

CONTENTS

INTRODUCTION

Security is increasingly becoming integrated and automated into software development initiatives, and application security (AppSec) specialists often work closely within cross-functional software development teams to improve the security that software organizations create. However, there is still confusion about the role developers play in an organization's secure software initiatives and how much of the security responsibility rests on the shoulders of developers today.

As a result, the best way to clear up the confusion and address secure coding practices head on is to first acknowledge there is a deficiency, then next, apply a modern approach to developer AppSec awareness and training. But before we reveal the Checkmarx proven approach, let's take a look at where many organizations are today.

In research conducted by ESG, **79%** of respondents shared their organizations regularly or occasionally push code to production with known organic vulnerabilities. Developers, AppSec teams, and operations are clearly feeling pressure to knowingly deploy vulnerable code, with **54%** citing the need to meet a critical deadline.

Read more

In a study conducted by DarkReading, **78%** of IT and security managers surveyed describe security as being important enough to delay deployment of applications, but only **26%** of organizations say their developers are very knowledgeable about secure coding practices.

Read more

Finally, in an investigation conducted by Checkmarx, AppSec ownership continues its gradual shift from IT to developers. When asked about skills improvement, **46%** of developers said they have prioritized learning or improving secure coding skills in the past 12 months and **36%** said more AppSec training is high on their wish list.

Read more

A Relationship Unfolds

From the data points just mentioned, there is an intrinsic relationship between:

- Vulnerable software making its way into production.
- Vulnerabilities causing delays resulting in slower releases.
- Developers desiring skills improvement to help alleviate this situation.

Clearly, there is a gap between an organization's need to produce secure software and the level of AppSec awareness, training, and developer secure coding skills.

This guide is designed help you to bridge that gap, enabling you to ensure your developers are effectively building their AppSec awareness, and are able to embrace AppSec learning as an immersive daily activity, resulting in more-secure and rapid software releases.

It covers everything you need to know to ensure your developers get the most effective AppSec awareness and training solution of all – one they will actually use.

SECTION 1

UNDERSTANDING THE DEVELOPER'S PERSPECTIVE

From all indicators, developers want to create more-secure code. Be it a competitive thing amongst their peers, a heightened sense of responsibility, or even a personal desire for perfection, developers acknowledge that security training is imperative to meeting their goals. However, and resonating loudly, security cannot impact their primary objective of software development and delivery.

A developer's most valuable resource is time

While today's fast-paced development requires rapid delivery – with minimum bugs – for continuous integration and continuous delivery (CI/CD) requirements, secure coding education solutions that slow developers down, are not relative to their daily duties, or are considered a nuisance will end up becoming a point of contention.

Therefore, organizations must consider solutions that raise AppSec awareness and secure coding skills in a way that integrates seamlessly with the development environment, while code is being written, and provides what developers need, when they need it.

Developers are primarily hired to write code

You probably won't find "secure code delivery" in most job requirement ads, developer contracts, or onboarding processes. Secure coding is often looked upon as "nice to have if you have time", and time is not an abundant resource in our modern age of digital transformation.

Today's developers are measured by speed and delivering workable code – but not by the amount of security vulnerabilities that it contains. This means that although developers are well aware of the need to deliver bug-free code – and most will therefore put in an effort to do just that – this is not usually the case in the context of coding errors leading to exploitable vulnerabilities.

For developers to deliver secure code, their development team leaders must begin treating security vulnerabilities in the same way as coding bugs. Once the importance of secure coding is established, you can then implement a programmatic approach to AppSec awareness and training.

SECTION 2

WHY APPSEC TRAINING OFTEN FAILS

Video tutorials, lectures, slide decks, periodic classroom training, and mandatory online courses are all standard approaches to AppSec training, yet they often fail to achieve developer AppSec awareness and skills improvement that helps them write more secure code. This is because these approaches are generally treated as boxes that needs to be checked on a to-do list, not as important tools for securing an application.

Because developers' priorities lie elsewhere, antiquated, and unproven educational approaches are understandably not working well. So, what is the best way to invest in security education for developers?

Why checklists don't work

General studies on training effectiveness (such as Arthur, et al., 2003) have shown that:

- Learning occurs best when the training is targeted to a specific set of behaviors or skills, is delivered in a context relevant to the learner, and is actionable.
- Training and development need to be easily accessible, relevant, and immediately applicable, rather than simply a means of delivering information to instill knowledge.
- Simulations may augment procedural, declarative, and experiential cybersecurity knowledge, enhancing learning effectiveness (although evidence is largely anecdotal, as few studies have systematically tested this proposition).

While it may be tempting to provide a simple list of technical topics in a checklist, it's clear that this is the wrong approach for building awareness of application security issues.

Presented in purely informational form, topics such as OWASP Top 10 Risks to avoid, SANS Top 25 software errors, secure coding guides, and best practices tutorials will fail to ignite the behavioral change that is essential to cybersecurity responsiveness.

Immersion in an environment via simulations and live activities are proven to be critical to applied learning performance: and for training that leads to improved AppSec awareness, we recommend gamification and contextual learning.

SECTION 3

EFFECTIVE APPSEC AWARENESS AND TRAINING

Effective AppSec awareness and training programs should harness all of the great things modern technology provides us today. Much in the same way that an engaging mobile app can influence the behavior of its user, the foundation for efficient secure coding practices can be rooted in gaming principles and technology-driven traits that keep users engaged long-term.

Lesson gamification

Gamification is the application of game-design elements and game principles in non-game contexts. The benefits of gamification-based lessons are recognized, yet most AppSec awareness and training solutions fall behind in taking advantage of it. Gamification can be thought of as simulations that include attacker vs. defender scenarios within the context of exploitable vulnerabilities and the steps to remediate them.

It's been proven that we learn better when we enjoy the learning process and take an active role, being less likely to quickly tire. Since developers spend most of their day in front of a screen viewing lines of code, our experience with thousands of developers has taught us that training that spices things up, rather than boring lesson that require deep concentration, is highly welcomed, and therefore more successful.

Since this is the case, there are four steps to achieving effective gamification for AppSec training.

1

Make it interactive

Chief Learning Officer® could not have phrased it better:

"Just because a learner clicks frequently does not necessarily mean learning content is engaging. The learner may be trying to speed through the course to reach completion."

Recognize the pattern? We see this often when workers are required to complete online training on a subject they don't fully understand the importance of, for example, mandatory IT security courses. They already have plenty on their desk, so they want to get through the training as quickly as possible and get back to tackling their workload. This leads to ineffective online courses and quizzes that people quickly click through without internalizing the importance of the content.

Chief Learning Officer goes on to point out that the stories and examples featured within interactivity (e.g., attacker and defender simulations) are an integral part of learning engagement, enabling the participant to feel directly and emotionally involved with the content, which can then improve retention.

Interactivity results in developers paying more attention to the content, which will then result in a higher chance of actually learning from it and retaining it. This is especially important when you consider that many people learn more effectively by doing and experiencing, rather than just by hearing or seeing.

Interactivity caters to the needs of this group, especially when short bursts of interactive training material can be consumed in-the-moment to remediate coding errors vs. taking a course and maybe a week later working on issue remediation. This immediacy closes any gap between learning and doing and is a valuable form of reinforcement leading to better retention.

And if all that wasn't enough, it's worth remembering that it is simply harder for someone to click their way through an interactive training session.

2

Tell a story

Characters, role-play, and a narrative are all effective ways to help embed information: stories stimulate our brains and spark a reaction.

AppSec training presented in the form of bullet points, questions and answers, or dull text is bound to bore developers. Stories, characters, and problems that need to be solved are the stuff that effective education is made of.

Offering your developers characters whose shoes they can step into and a storyline they can follow (in our case, a vulnerability that needs to be remediated) is extremely helpful in ensuring they remember what they have learned. Storytelling also forms the essence of many games and is often a fun-catalyst, hence its connection to gamification.

3

Keep it short

Although it's arguable whether the human attention span is getting shorter, it is still best practice to keep things concise when presenting information of any kind.

Short content is generally precise and to the point, eliminates irrelevant information, and increases the likelihood that people will engage with it. Given that time is a precious resource for developers, the briefer your training sessions are, the better.

4

Harness competition and the winning mentality

According to Dr. Ian Robertson's research "The Winner Effect," the most underestimated brain-enhancing agent is empowerment. He writes that winning can be applied to education, as it leads to increases in dopamine activity in the brain – giving us the feel-good factor.

While short and interactive storylines are important to establishing an AppSec awareness and training program, it's also important to consider the importance of "winning" the training session, that is, achieving success in remediating a vulnerability. Providing rewarding learning experiences for developers makes them feel good, and they will be more likely to return to the program or solution to discover new vulnerability challenges.

While the enjoyment gained from individual personal wins is an incentive for returning again and again to a gamified learning environment, it's also worth considering competition as a means of motivation on a larger scale. Which is where tournaments come in.

SECTION 4

THE POWER OF TOURNAMENTS

Organizations often use annual training sessions to make sure developers are continuously sharpening their secure coding education and skills. Turning these into tournaments, makes learning FUN!

A tournament is a live competition where developers compete with peers to resolve AppSec-related issues. It's intended to sharpen their secure coding and vulnerability remediation skills. Done right, developers experience an energetic competitive environment, where a dynamic leaderboard increases friendly rivalry, and challenges are established within set time limits.

Why tournaments make sense

They increase AppSec awareness through an **enjoyable and rewarding shared experience.**

They engage developers while **sharpening their AppSec skills.**

They measure the organization's overall **AppSec awareness,** while revealing any knowledge gaps.

How to leverage tournaments

When kicking off an entire training program

Using a tournament to launch a new learning solution or kick off a training program introduces your developers to AppSec training while allowing you to collect KPI data. This will provide you with a clear picture of your developers' AppSec skills, giving you and them a starting point from which to measure improvement and set new objectives. This will also help everyone to focus training efforts on the areas of most need.

When concluding a learning program

As a dynamic and exciting event, a tournament is a great finale for a learning program, and can motivate your developers to complete their training to increase their chances of winning.

As part of an AppSec Awareness week/month/event

If you occasionally host an AppSec Awareness event, a tournament can be one of the activities you include. You can run a tournament and intersperse rounds with other activities, such as speaker sessions, panel discussions, and recognition activities.

SECTION 5

BENEFITS OF CONTEXTUAL LEARNING

Preparing an engaging, gamified AppSec learning program is critical to ensuring that developers actually use it and learn from it. However, if you neglect to make it contextual, you may ruin all your hard work.

It isn't just the lack of gamification that makes periodic classroom training ineffectual, it's also the lack of context. Once you take developers away from their development environments, you remove them from their day-to-day coding routine and make it that much harder to recall a problematic line of code. AppSec knowledge and awareness should appear at the exact moment when it is needed: when coding.

The secret to successful AppSec training is integrating it into the developer's work routine, so that it is just a click away.

However awesome and gamified your AppSec content may be, there is no need for your developers to go through it all at once. We found that offering developers ongoing access to learning while they code – fully integrated into their development environment – is what encouraged them to seek it out when encountering a security vulnerability.

The other important aspect of contextual learning is to ensure it stays up to date with the latest coding languages. Secure coding best practices differ, so there isn't a one-size-fit-all type of solution. To learn more, check out the Checkmarx secure coding guides, such as:

The JavaScript Guide:
Web Application Secure Coding Practices

The Go Language Guide:
Web Application Secure Coding Practices

Kotlin Guide:
Mobile Application Secure Coding Practices

SECTION 6

PERIODIC ASSESSMENTS

Your AppSec awareness metrics should always be on the rise. After all, what's the point of investing in AppSec awareness and training solutions if all these efforts don't pay off by reducing your software security risk in general? To make sure this is the case, you need to keep a close eye on the progress of your development teams. Continuous improvement is the desired result, and to achieve that, you need to periodically assess the current state of your developers' security mindset.

A quick and easy way to measure and baseline the secure coding skills of your developers is to use assessments that take developers 10-15 minutes to complete and can be assigned to individuals or teams. We recommend you use assessments to:

- Establish a clear baseline that will allow you to see the impact of training over time.

- Reduce unnecessary training by validating developers' skill level.

- Identify knowledge gaps and nurture those that require more training.

- Measure the effectiveness of your security awareness program.

- Challenge your more senior developers while setting a bar for those less senior.

- Assess the secure coding familiarity of candidates and new hires.

The whole concept of assessments is to determine if developers need more training, identify the areas of weakness, measure and report on their improvements, and finally, reduction of repetitive coding errors.

SECTION 7

CHOOSING AN APPSEC TRAINING SOLUTION VENDOR

The Checkmarx AppSec Awareness and Training solution – CxCodebashing – has been created **by developers, for developers.**

Here are some important questions to ask when talking to solution vendors, complete with the answers we would provide.

Q Are training programs that deliver a different kind of approach available for my developers?

Yes.

CxCodebashing is a next-gen AppSec awareness and training solution. It accelerates learning, improves secure coding skills, and engages your developers with up-to-the-minute guidance you'd expect from a leader in the AppSec domain.

Q Will it be appropriate to what my developers do?

Yes.

CxCodebashing, when integrated with CxSAST, educates your developers on the specific day-to-day security issues they encounter. It not only describes the issues detected from a CxSAST scan, but also educates them on why it's happened and how to remediate it.

Q Does it meet enterprise needs?

Yes.

Straight out of the box, you get enterprise-grade management, communication, and analytics tools that enable you to seamless scale Codebashing from 10 developers to 10,000+.

You also get a focus on compliance, helping your developers build an understanding of compliance issues related to GDPR, PCI DSS, NIST 800-53, and HIPAA.

Q Can it keep my developers up to date without slowing them down?

Yes.

CxCodebashing is a companion for your developers. It's not distracting, nor does it remove them from their working environment. Instead, it sits alongside their daily workflow and can be accessed at any time without breaking the rhythm of whatever they're engaged in. This means that learning happens in a timely and relevant manner.

Q Can it be relied upon to keep my developer audiences engaged for the long term?

Yes.

Research shows that the most effective form of learning is achieved through contextual and simulated content. Your developers will see real progress in how securely they work.

And it's fun too. The CxCodebashing platform is built with enjoyment in mind – it's like no other learning solution.

SECTION 8

CALCULATING THE RETURN ON EFFECTIVE APPSEC TRAINING

Taking a programmatic approach to AppSec awareness and learning through a platform like Checkmarx Codebashing can deliver a positive impact on the bottom line.

This table shows an example of a mid-sized organization with 350 software developers, and applies industry-standard numbers to the productivity and cost parameters that need to be considered when making AppSec training investment decisions.

Estimated lines of code written per developer, per day	67.5
Working days a year	230
Estimated lines of code that a developer writes in a year	15,525
Estimated # of vulnerabilities per 1,000 lines of code	1
Estimated # of vulnerabilities per developer, per year	15.53
Estimated training effectiveness	10%
# of vulnerabilities reduced per developer, per year	1.55
Estimated # of hours to fix a vulnerability	22
Time saved due to reduction in vulnerabilities per developer, per year (in hours)	34.16
# of hours to complete Codebashing course	3
# of developers	350

Total time saved per developer per year (in hours)	31.16
Developer cost per hour	USD 35
Total cost saved per developer per year	**USD 1,090.43**

Total time saved per year (in hours)	10904.25
Codebashing cost per year	USD 80,000

Total savings per year with Codebashing	**USD 301,648.75**

SECTION 9

BEST PRACTICES AND PRINCIPLES

To help you successfully create AppSec awareness, upskill your developers in secure coding, and demonstrate the return on your investment, here are some useful pointers to help you launch and run your program successfully.

Implementation and launch

Pre-launch

Communication with developers should begin ahead of launch, and continued until the program is firmly established. Set clear expectations: don't just tell them what you're doing, but explain why, share goals and objectives, and let them know that this will be fun, productive, and beneficial to them as individuals, as well as to your organization.

Rollout

Taking a phased approach to your rollout makes the introduction of a new program easier to manage. You can stagger the launch by location, business unit, application, language, or any other criteria that help you to create distinct manageable groups for each phase.

Assess

Establish a baseline at the beginning of your program, so that you can measure improvements in developer skills, in addition to understanding where your strengths are and identifying areas for improvement. This approach will enable you to demonstrate value to key stakeholders after the initial training has been conducted.

This approach allows you to track developers' progress and your ROI using your organization's relevant KPIs for developers. As a result, accelerated improvement in the code being produced will have a significant impact on your bottom line.

Ongoing management principles

Train, train, train

Conduct focused training for dedicated periods of time on a regular schedule

Foster friendly competition among developers

Since everyone loves some level of competition, add it to the mix, and make sure no one feels left behind or left out.

Address challenges head-on

If problem areas are identified, address those areas first, and then share positive progress with the group, and with the larger audience.

Incentivize and recognize

Since most people are motivated by awards and kudos of some kind, think of incentives your teams would appreciate. It doesn't always have to be money based – sometimes public recognition is enough to motivate people.

Assess, track, report

Continuously assess your program and team performance. Share that performance regularly with relevant internal stakeholders in a report format that can be easily digested. Keep good records of all progress and problem areas, and remember to let teams and individuals know how they are doing.

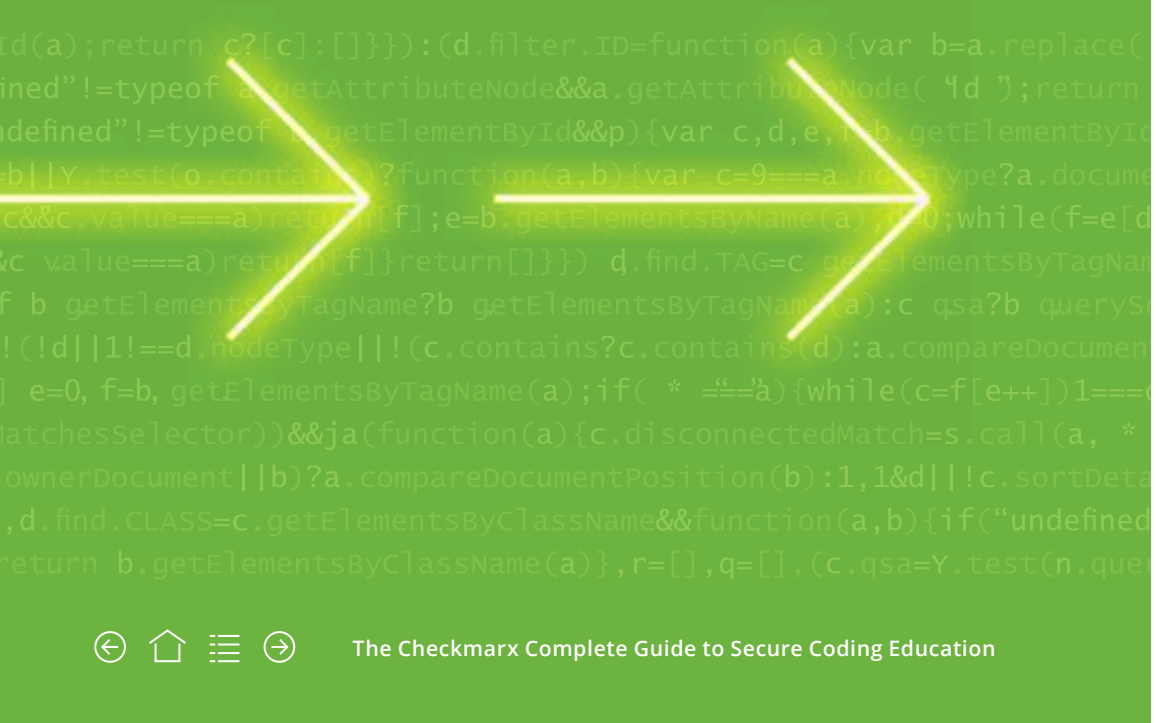
Repeat

If what you are doing is working, don't fix what isn't broken.

SECTION 10

NEXT STEPS

Taking a programmatic approach to AppSec awareness and learning through a platform like Checkmarx Codebashing can deliver a positive impact on the bottom line.



Ask your developers to try our **free Codebashing lessons**, which include all the necessary gamification and in-context tips presented in this guide.

Free Codebashing Lessons

If your developers are interested in playing a game to test their application hacking skills, **Game of Hacks** is a good place to start. In this game, the player will be presented with vulnerable pieces of code and their mission, if they choose to accept it, is to find which vulnerability exists in that code as quickly as possible.

Game of Hacks

About Checkmarx

Checkmarx is the global leader in software security solutions for modern enterprise software development. Checkmarx delivers the industry's most comprehensive Software Security Platform that unifies with DevOps and provides static and interactive application security testing, software composition analysis, and developer AppSec awareness and training programs to reduce and remediate risk from software vulnerabilities. Checkmarx is trusted by more than 40 of the Fortune 100 companies and half of the Fortune 50, including leading organizations such as SAP, Samsung, and Salesforce.com.

Learn more at Checkmarx.com