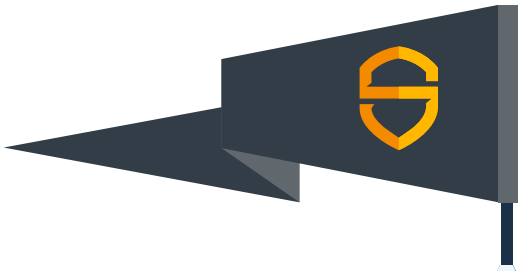
SECURE CODE WARRIOR



EMPOWERING DEVELOPERS TO WRITE SECURE CODE

An INNOVATIVE Approach to Improving Software Security

By Pieter Danhieux and Matias Madou

securecodewarrior.com

CONTENTS

INTRODUCTION

Economic growth today is largely based on digital technology, which means that every major business has become a software company in some form. With 23 million software developers around the world today[1], software developers are now the primary architects underpinning the success of many public and private organizations. Within this environment, many development teams have moved from a niche corner of their organization right into the 'hot seat'. They are challenged to rapidly translate business needs into competitive applications that are convenient, trustworthy, and secure.

'Convenient, trustworthy and secure' are easy words to write on a page, but infinitely more challenging to deliver, especially when there are 111 billion lines of code written every year.[2] In 2018, software security continues to be a major challenge, with the threat and reality of breaches growing every day. Agreeing with many concurring reports, Verizon's 2018 Data Breach Investigations Report found that 21 percent of data breaches today are caused by web application vulnerabilities, something that has been a consistent finding over the past decade.[3]

DEVELOPERS DON'T NEED TO BECOME SECURITY EXPERTS, BUT THEY MUST BE EMPOWERED TO BE THE FIRST LINE OF DEFENSE FOR THEIR ORGANIZATION.

Despite ever-increasing application security budgets, testing platforms, tools and penetration tests, the number of successful cyber attacks keep rising. According to an Akamai report, attacks on web applications increased by 69 percent from Q3 2016 to Q3 2017[4]. Further to this, the same security errors are routinely found in software day after day, year after year, and the threats are expanding as a result. According to Veracode's recent report based on 400,000 application scans in 2017, applications passed OWASP Top 10 policy only 30% of the time, and this is consistent for the past five years[5]. Astonishingly, SQL injections appeared in almost one in three of newly scanned applications over the past 5 years, including last year.

The time has come to evolve developer software security training to be a constant and positive part of their everyday working routine. Writing great software means it must be secure. Improving secure coding skills and outcomes will add a powerful layer of cyber protection for organizations, and will help them write better, faster code. Developers don't need to become security experts, but they must be empowered to be the first line of defense for their organization.

PART 1.

What's wrong with the current approach to software security?

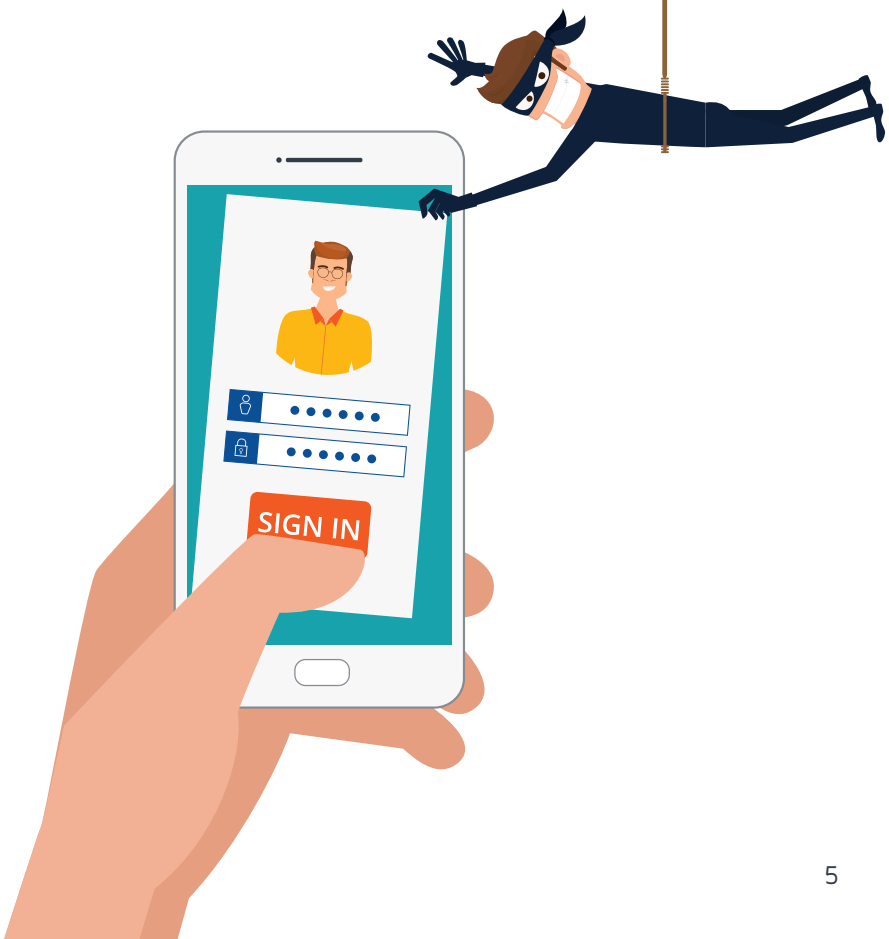
Developers are under pressure to deliver software with as little business risk as possible, but time and resources to help write secure software from the start are limited. With today's pace of development, even the best teams produce many security errors, causing delays, cost, and downstream risks that companies can't afford.

There is a widely held view among executives that cybersecurity impedes innovation, with numerous research reports indicating that uncertainty about cybersecurity is halting both digital innovation and “mission-critical” initiatives. According to a Cisco report, 71% of organizations believe that security issues are impeding innovation; 57% believe that security is likely to be compromised as organizations try to innovate quickly; and 39% have halted mission-critical innovation due to perceived cybersecurity risks.[6]

At the crux of the problem is the current negative mindset in which software security is often viewed. It is seen predominantly as a compliance and defensive strategy, rather than a positive enabler of growth. There are several reasons for this, namely the fact that insecure software is the cause for a lot of damaging cyber attacks, security teams and developers often work against each other and perhaps most importantly, the majority of emphasis is on finding flaws in the software, rather than fixing the problem at the source.

71% OF ORGANIZATIONS BELIEVE THAT SECURITY ISSUES ARE IMPEDING INNOVATION

57% BELIEVE THAT SECURITY IS LIKELY TO BE COMPROMISED AS ORGANIZATIONS TRY TO INNOVATE QUICKLY



Insecure software is responsible for a lot of successful cyberattacks

Successful attacks are commonplace occurrences, giving victim organizations a lot of bad press. There are regular statistics published about breaches due to flaws in the software, with one US government software assurance program quoting 90% of incidents are caused by defects in the design or code.[7] There's also a lot of blame going around after attacks and it is becoming common for breached organizations to point the finger at software bugs being responsible for the problem as they explain their circumstances.

Some organizations manage to identify serious flaws before they are breached, such as Twitter's warning in April 2018 that an internal defect in their password software had left its 336 million users vulnerable to hackers.[5] The narrative puts the responsibility on the user to change their password, and the company's rectification of the bug, but why is there such easy acceptance that the software being produced and used is vulnerable in the first place?

Security vs. Development is an unfair game

For a long time, security teams and developers have been pitted against each other in what feels like a very unfair game for both sides. Tension begins at the outset, because developers need to create functional code without any security issues, while security's job is to identify as many security problems as possible, and attackers only need to find a single security vulnerability, which feels like an even easier task. Few developers are formally taught security principles or are held to secure coding standards. They don't know how to code securely or how to fix the problems that security teams throw at them, as the recommendations provided are often too generic or not usable.

Security experts have a tough time too. They are responsible for security across, on average, 50-100 developers, meaning they are often stretched and stressed. According to BSIMM 8, the average percentage of security analysts to developers across their survey participants was 1.6%.[8] Also, although they are experts in security concepts and identifying security problems in code, security professionals are often not trained in specific languages and frameworks which would have allowed them to modify the developer's code and actually fix the issues they find.

Right-to-left is back to front

Current application security tools focus on moving from right to left in the Software

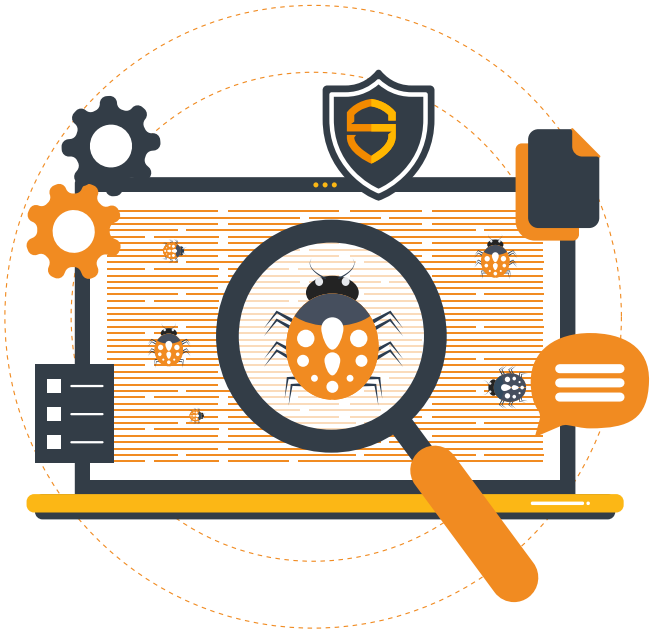
Development Life Cycle (SDLC). This process is good at finding problems and pointing them out, but doesn't teach anyone to prevent the problems or fix them. This approach supports detection and reaction—detect vulnerabilities in the written code and to fix them.

In addition to the developer-security antipathy, it is about 30 times more expensive to detect and fix vulnerabilities in committed code than it is to prevent them when writing code in the IDE (Integrated Development Environment).[9]

Current security tools and processes can't keep up with DevSecOps

In recent times there has been a real focus on short release cycles, with the emphasis on getting code out quickly and regularly, so a DevSecOps approach, which integrates development, operations and security teams under one umbrella with a common security goal, is becoming increasingly popular. In a DevSecOps world, there is a need to find security bugs within shorter release cycles, but many of the tools being employed are not able to do this. For example, executing a penetration test takes two weeks, and by the time the results come in, the version of the software and the bug are both already outdated.

MOST TOOLS ONLY POINT OUT PROBLEMS, BUT DON'T OFFER GUIDANCE ON HOW TO VALIDATE THEM, CORRECT THEM, OR HOW TO WRITE SECURE CODE.



Maturing AppSec programs is a slow process

Apart from large, highly regulated industries such as banking and finance, most formal software security programs begin only after an organization has been breached. At this point, organizations will generally start a program immediately to understand if there are more of the same issues and contain them. Penetration testing and code analysis would be common practice at this stage. Once they start testing, security will normally find dozens or even hundreds of issues in the code. Some companies will try and deal with the OWASP Top 10, but many will only try to deal with the OWASP Top 2 or 3.

As companies mature their AppSec programs, they generally "shift left," closer to the start of the Software Development Lifecycle (SDLC). They might invest in some white hat hackers, then shift further left and employ a static code analysis solution. Still, they are constantly finding hundreds of potential problems in their completed code, which are both time consuming and expensive to fix. There will also be many false positives and false negatives, and they will keep finding the same problems, which is frustrating for everyone. Even when companies move bug-finding all the way into the developer's IDE, most tools only point out problems, but don't offer guidance on how to validate them, correct them, or how to write secure code.

The take-out: We know lots about the problem; it's time to focus on the solution

It is important to have tools and processes to look for problems in the code and many companies are improving their ability to do this. However, for every bug found, developers are forced to go back into old code, re-familiarize themselves with it, learn about that problem, work out how to fix it, and then push it back into the system, which then triggers a new quality assurance process.

THE TIME HAS COME TO CHANGE OUR MINDSET AND OUR APPROACH

Apart from the obvious overhead, imagine how a developer feels knowing they have made that same mistake many times, but they aren't able to find and fix all the other instances until they receive them individually via a bug report. Each time they receive a bug report they must stop their code building and enter the same cycle again, without the ability to share the solution with other developers, who may be figuring out the very same solution by themselves somewhere else.

It is no wonder that software security is considered negatively by both development and security, and indeed by C-level executives generally. Innovation today has reached a place where although security is an essential foundation for innovation and growth, the current approach causes a lot of frustration and anxiety and frankly, makes security 'the bad guy' for everyone.

The time has come to change our mindset and our approach. Empowering developers to code securely requires a positive mindset around software security in general, as well as modern training and tools that enable developers to build important security skills, rather than just execution of the same old inadequate processes.

PART 2.

Empowering developer security requires a positive approach

At the risk of oversimplifying things, the first step is for an organization to decide that instead of perceiving software security as a burden or an impediment to innovation, it will be embraced as an opportunity for your development teams and the organization to write better, more secure applications.

Developers as security heroes: The opportunity

Although every developer is unique, it is fair to say that software development is a true craft, and most developers take immense pride in their work. Developers are creators, keen to solve complex, abstract problems and build clever features. They get excited about shipping an excellent product, and although there is enormous pressure to build applications quickly, no one wants to produce a shabby product.

BOTH SECURITY AND DEVELOPMENT TEAMS NEED A MINDSET SHIFT IN THE WAY THEY THINK AND OPERATE, ENABLED BY INNOVATIVE TOOLS AND TECHNIQUES THAT ENCOURAGE SECURE CODING FROM THE START.

What does empowering developer security mean?

Empowering developer security moves the focus from reaction to prevention. Rather than dull "compliance-based" training and a negative focus on pointing out every perceived and real flaw, the approach evolves to see security issues from a developer's perspective. This means making it relevant and engaging for them, while genuinely teaching developers how to identify and fix their own issues, as well as learning how to securely construct their own code.

Can you imagine if developers were taught to write secure code in real time, and to avoid creating many of the bugs in the first place? What if they could fix secure coding problems as they arise? And your highly skilled security managers and testers – what if they could focus on finding and fixing the really challenging, complex bugs rather than sending developers after nebulous minor issues with limited instructions?

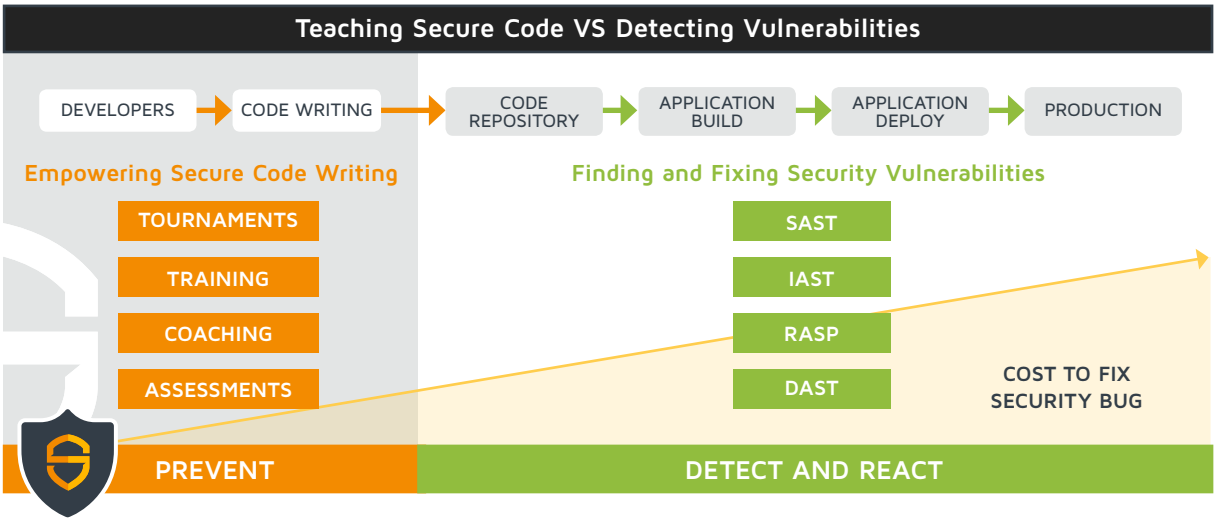
The developer persona is clever, problem-solving, and keen to build their skills. This must be applied to security training and tools, rather than the current model of boring videos and CBT training – tools that focus just on finding faults. This approach will not replace security experts and does not require developers to be security professionals themselves. However, it flips the strategy from tedious training followed by continually scanning and finding errors, to genuinely motivating and aiding developers to make their code secure at the source.

CAN YOU IMAGINE IF DEVELOPERS WERE TAUGHT TO WRITE SECURE CODE IN REAL TIME, AND TO AVOID CREATING MANY OF THE BUGS IN THE FIRST PLACE?

Teaching developers to write secure code vs. detecting vulnerabilities

As previously mentioned there are many solutions that find vulnerabilities in code. We need more emphasis on teaching developers to follow security guidelines that will prevent them from making mistakes. Teaching developers how to code securely is proving to have a significant and positive impact on both the number of vulnerabilities created as well as the general mindset around software security.

Developers should be assisted to write secure code and fix any errors they make as they are writing code, and the technology now exists to achieve this. Just as spelling and grammar correction tools assisted in the real-time writing of this whitepaper, specific to the American English it was written in, developers can be helped in real-time to write securely in line with the relevant language and security policy.



IT IS ABOUT 30 TIMES MORE EXPENSIVE TO DETECT ANY VULNERABILITIES IN COMPLETED CODE THAN IT IS TO PREVENT THEM.

OWASP has an important role to play in empowering developer security

The Open Web Application Security Project (OWASP) was founded by Mark Curphey in 2001 and is a non-profit online community that produces freely-available articles, methodologies, documentation, tools, and technologies in the field of web application security.

Most famous for its Top 10 list of the most critical application security risks, many organizations rely on this for direction in their AppSec programs. Indeed, SCW supports the OWASP Top 10[10] through our secure coding training platform, also allowing users to customize their security policy in alignment with this and other industry standards. The OWASP Top 10 is useful, but like scanning and testing tools, it focuses on pointing to the vulnerabilities rather than teaching developers how to identify and fix problems as they code, and how to follow secure coding guidelines so they can prevent bugs in the first place.

Alongside the Top 10, OWASP is playing a role to help companies empower developer security. For almost two decades OWASP has created a global community of like-minded people helping each other. It provides free assistance to coders worldwide, including cheat sheets, scanning tools, secure coding frameworks, and testing environments. Its numerous consensus projects are powerful too, because the community decides what is good and bad practice. OWASP's Application Security Verification Standard (ASVS) Project[11] and the OWASP Top Ten Proactive Controls[12] are both making a tangible and valuable contribution to evolving developer secure coding education and skill development.

OWASP ASVS

Use as a metric	Provide application developers and owners with a yardstick with which to assess the degree of trust that can be placed in their Web applications
Use as guidance	Provide guidance to security control developers as to what to build into security controls to satisfy application security requirements
Use during procurement	Provide a basis for specifying application security verification requirements in contracts

OWASP'S APPLICATION SECURITY VERIFICATION STANDARD (ASVS) provides a basis for testing web application technical security controls and provides developers with a list of requirements for secure development.

The Top 10 Proactive Controls

- **C1:** Define Security Requirements
- **C2:** Leverage Security Frameworks and Libraries
- **C3:** Secure Database Access
- **C4:** Encode and Escape Data
- **C5:** Validate All Inputs
- **C6:** Implement Digital Identity
- **C7:** Enforce Access Controls
- **C8:** Protect Data Everywhere
- **C9:** Implement Security Logging and Monitoring
- **C10:** Handle All Errors and Exceptions

The OWASP TOP TEN PROACTIVE CONTROLS 2018 is a list of security techniques that should be considered for every software development project. One of the main goals is to provide concrete practical guidance that helps developers build secure software. These techniques should be applied proactively at the early stages of software development to ensure maximum effectiveness.

PART 3.

How to make secure coding training and tools simple, relevant, and engaging for developers

TOURNAMENTS

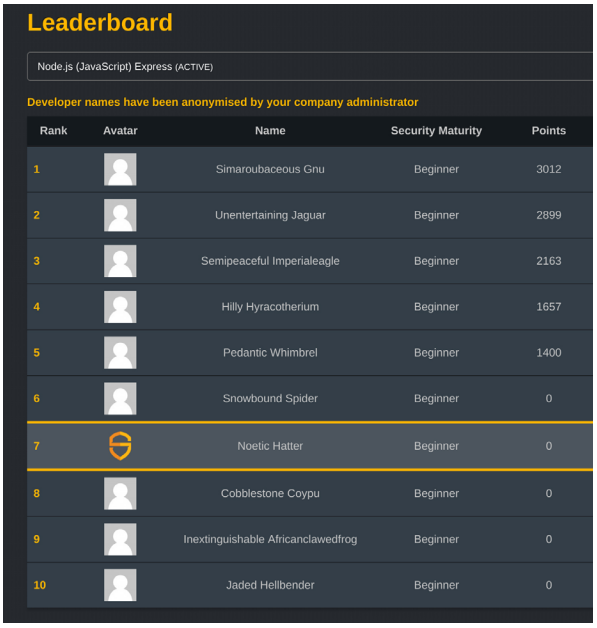
TRAINING

COACHING

ASSESSMENTS

Developers can be empowered to be the first line of defense in their organization by making security highly visible and providing them with the skills and tools to write secure code from the beginning. This section describes an innovative approach to improving secure coding skills and outcomes that is simple, scalable and positive, for both development and security teams.

DEVELOPERS ENTER A VIRTUAL ARENA, ESPORTS STYLE, EARNING POINTS IN THEIR QUEST TO THE TOP OF THE LEADERBOARD.



Tournaments: Building a positive learning experience that drives engagement

Secure coding tournaments are a fantastic way for companies to build awareness and enthusiasm for secure coding. They can be highly engaging face-to-face or virtual events that get the whole developer community involved and started on a secure coding journey. Tournaments can be short — only a few hours — or run for days, and include a series of vulnerable code challenges where developers identify problems, locate insecure code, and fix the vulnerabilities.

Challenges should generally be based on the OWASP Top 10 and enable developers to compete in a software language they are currently working in to ensure it is relevant. Throughout the tournament, developers earn points and watch each other (and themselves) climb to the top of the leaderboard. There are many exciting examples of companies holding innovative tournaments that attract developer attention and drive participation. It also helps companies to find security champions in each scrum team, which is important to ongoing skill building.

Secure Coding Tournaments

- Makes it fun!
- Keeps teams focused
- Builds awareness
- Identifies security champions

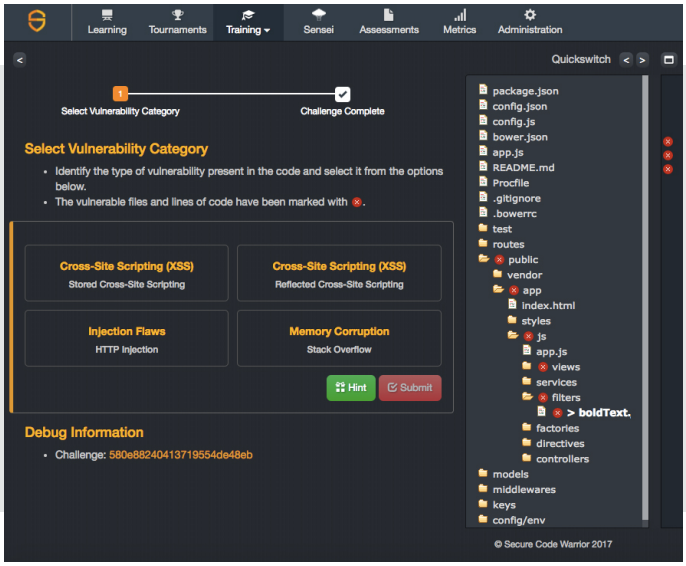
Training: Make it relevant, hands-on and fun, and developer security skills will grow fast

Given security is not the developer's priority, any security training must be fun, competitive, and engaging to motivate them to 'play.' The best learning platforms are gamified and allow developers to earn points and collect badges, with leaderboards for teams.

Training platforms must focus on building specific skills, actively engaging developers to learn, and build their secure coding skills. Developers should be able to work in real code and in their own languages/frameworks, to locate, identify and fix known vulnerabilities in code.

Challenges should be short and cover all the common vulnerabilities. They must be constantly expanded and updated so developers can continue to build their skills regularly over time, rather than completing a course once a year and forgetting about security afterwards. There must be engaging challenges for senior developers, as well as easier ones for those less experienced, as well as the opportunity to get assistance.

Enabling developers to view their progress throughout their journey is also important. Both developers and their managers should be able to see which challenges they have completed, their strengths and weaknesses, time spent on training and their accuracy.



TRAINING PLATFORMS MUST FOCUS ON BUILDING SPECIFIC SKILLS, ACTIVELY ENGAGING DEVELOPERS TO LEARN

EXAMPLE: 60% increase in secure development capability in 2 months

One of Secure Code Warrior's customers required their developers to play a single challenge (5 minutes) every day for two months. It tested their skills before and after the training period and observed a 60% increase in secure development capability over a group of hundreds of developers. This meant less resources spent on finding and fixing security bugs later in the lifecycle, and significant long-term savings.

Coaching: Make it helpful, scalable and available as developers are writing code

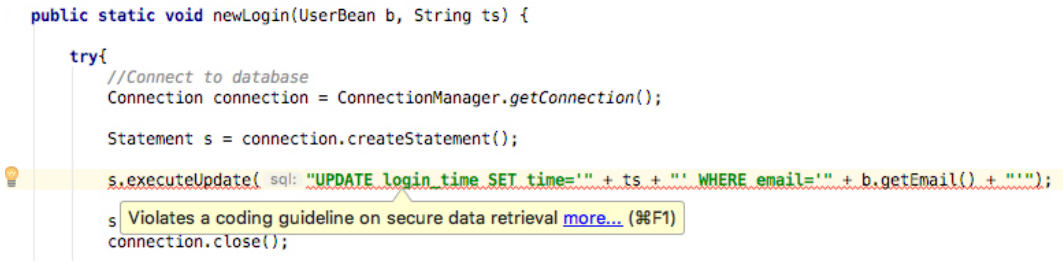
Just as writers use spell-checkers and grammar coaching tools, the code-writing profession has reached a point where a real-time security coaching plug-in must become an essential part of their code-writing toolkit. This should focus on positively teaching them to write secure code, rather than finding vulnerabilities.

Many organizations customize their security guidelines, so it is important for developers to validate code against both pre-set and customizable, authorized rule sets. As authors write code, developers need to know within milliseconds when they do not align with the company's secure coding guidelines. Just like a spell-checker, they should be offered instant quick-fixes for common vulnerabilities and link directly to specific training on that category of problems.

The more development teams use a quality real-time coaching and training tool, the better they will become at secure coding and the less time and money companies will need to spend on fixing bugs or managing security breaches.

Sensei

In 2018, Secure Code Warrior unveiled an exciting real-time enhancement to its secure coding platform that will help developers write secure code faster and more consistently. In the same way a spell-checker assists writers, SCW's Sensei coaching plug-in helps developers be more security aware, adhere to secure coding guidelines, and instantly transform 'bad code' into 'good code.' Sensei can be installed into a developer's IDE (Integrated Development Environment) in a few minutes to monitor, measure and correct mistakes.



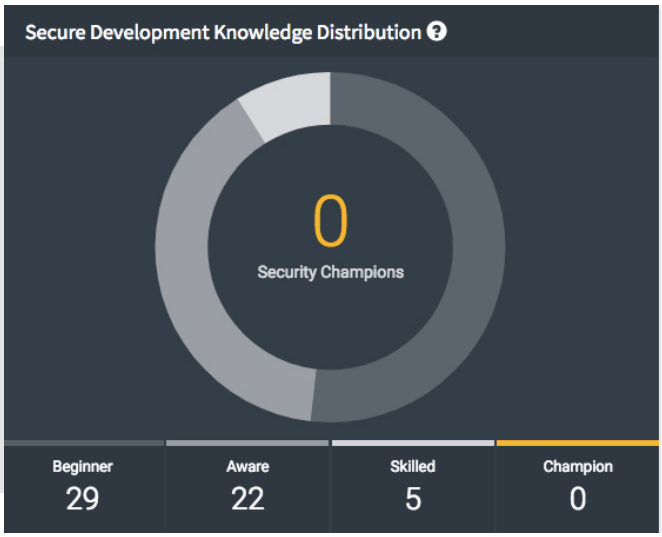
THE MORE DEVELOPMENT TEAMS USE A QUALITY REAL-TIME COACHING AND TRAINING TOOL, THE BETTER THEY WILL BECOME AT SECURE CODING

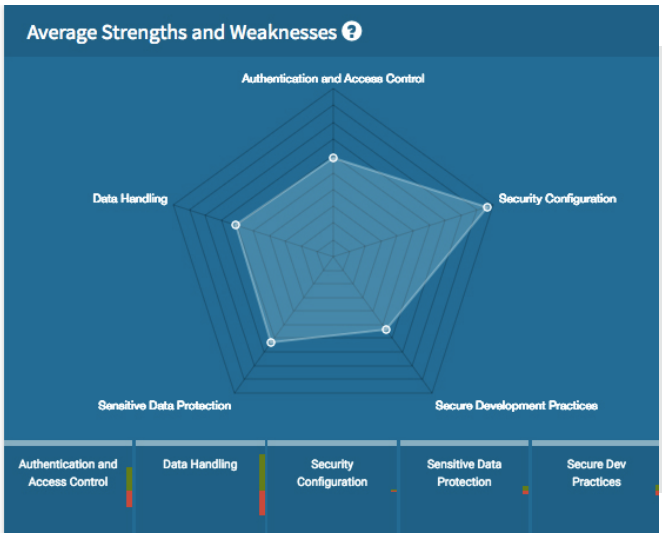
Assessment: Identify gaps and create strong evidence that secure coding skills are improving

Managers need to be confident that their developers have a base level of competency when it comes to securing their code. Assessment allows organizations to qualify and baseline the secure coding skills of their existing developers, offshore developers, new hires, and graduates. How easy would it be if you could identify developers who you were confident could write securely in your new project, before they have even started working on it?

Assessment also demonstrates to auditors that developers are learning the necessary secure coding skills that regulations like PCI-DSS (Payment Card Industry - Data Security Standard) outlines. Some companies are now starting to create an internal badge, belting or certification program that aligns with demonstrating certain security achievements.

SOME COMPANIES ARE NOW STARTING TO CREATE AN INTERNAL BADGE, BELTING OR CERTIFICATION SYSTEM





CONCLUSION

Taking Responsibility at the Source

As stated at the outset, the time has come to evolve developer software security training and tools, so they become a constant and positive part of their everyday working routine. Writing great software means it must be secure.

Developers need to take more responsibility for security, and a significant opportunity exists for companies to build a strategic business advantage by encouraging this kind of approach.

The solution not only involves building skills, but also having the right toolset to help every step of the process, from the first line of code until the last and beyond.

This paper outlines an innovative approach empowered by modern training and tools that we know work in alliance to improve developer security.

The powerful combination of Secure Code Warrior's training platform and the Sensei IDE plugin will assist security and development teams to collaborate constructively in building a positive security culture.

References

1 https://evansdata.com/reports/viewRelease.php?reportID=9

2 https://www.csoonline.com/article/3151003/application-development/world-will-need-to secure-111-billion-lines-of-new-software-code-in-2017.html

3 Verizon Data Breach Investigations Report 2018, p 35

4 Akamai Q3 State of the Internet Security Report. https://www.scmagazineuk.com/web-app-attacks-up-69-us-main-source-of-cyber-attacks/article/710175/

5 2018 Verizon DBIR report, https://www.verizonenterprise.com/resources/reports/rp_DBIR_2018_Report_execsummary_en_xg.pdf

6 Cisco, Cybersecurity as a Growth Advantage, 2016

7 https://www.us-cert.gov/sites/default/files/publications/infosheet_SoftwareAssurance.pdf

8 BSIMM 8

9 National Institute of Standards and Technology (NIST), www.peoplesec.org

10 OWASP Top 10, www.owasp.org

11 OWASP ASVS, www.owasp.org

12 OWASP Top 10 Proactive Controls, www.owasp.org

ABOUT SECURE CODE WARRIOR

Secure Code Warrior is a global security company that makes software development better and more secure. Our vision is to empower developers to be the first line of defense in their organization by making security highly visible and providing them with the skills and tools to write secure code from the beginning. Our powerful platform moves the focus from reaction to prevention, training and equipping developers to think and act with a security mindset as they build and verify their skills, gain real-time advice and monitor skill development. Our customers include financial institutions, telecommunications providers and global technology companies in Europe, North America and Asia Pacific.

securecodewarrior.com
info@securecodewarrior.com
insights.securecodewarrior.com
@seccodewarrior
/securecodewarrior
/company/secure-code-warrior