EBOOK

GUIDEPOINT APPLICATION SECURITY AS A SERVICE

# Secure Coding Culture Playbook

GUIDEPOINT SECURITY

# The Culture Gap Between Development & Security

There remains a cultural gap between software developers and application security practitioners. This gap challenges application security maturation within the Software Development Lifecycle (SDLC).

On one side of the gap, developers want to leverage the best-of-breed in familiar tools to ease their lives and work. *"In 2021, the tools used for application security that integrate into the toolchain must work much more rapidly, scale to cloud environments, and present actionable findings in a format that developers can understand and use to make quick fixes,"* says the 2021 Software Security Predictions: Our Experts Weigh In, Checkmarx. Developers must write code quickly to meet the needs of the business. These needs include shrinking Time-to-Market to earn a larger share of the marketplace ahead of competitors.

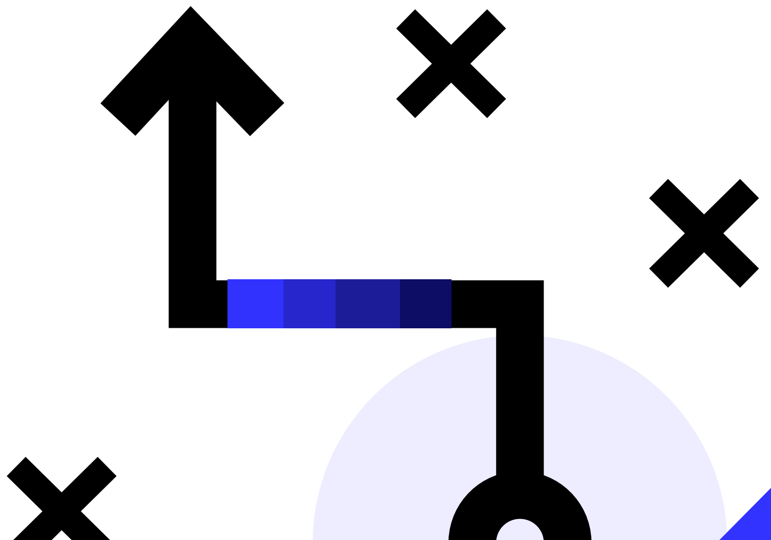
On the other side of the gap, Application Security strives to protect data as it moves through software in mobile devices, web applications, endpoints, and other aspects of software. Security must contend with existing and emerging privacy laws that require the business to geolocate data. Security needs to ensure that the code is free of significant vulnerabilities that could subject the user, the application, or the data to harm if an attacker leverages it. The goal is to eliminate security issues from the code proactively at the start of the development process rather than reactively after developers have written the code or at the end of the lifecycle.

But companies admit that they fall short of these goals. *"Forty-eight percent of organizations push vulnerable code into production. Forty-five percent say it's because the vulnerabilities were discovered too late in the cycle to resolve them in time,"* says a 2020 Modern Application Development Security report from Enterprise Strategy Group. It's not enough to test and fix code at the end of the development pipeline. Too many security flaws survive the process and make their way into production. Application security must dive deep into the heart of the development process when developers first stroke their keyboards.

But developers see application security as a roadblock to timely software delivery. They don't feel they have the skillset or the person-hours to touch application security. They want the security organization to address it without adversely affecting their work.

*"Developers focus on making things work and delivering releases on schedule, often under management pressure to get new features out of the door as soon as possible. The job of testers and security personnel, on the other hand, is to find bugs, vulnerabilities, and other issues–in other words, to prove that things don't work after all. To reconcile both approaches, organizations must incorporate security into the development culture and make it an integrated part of the application lifecycle rather than a separate phase,"* says the report, New Vulnerability Found: Executive Overconfidence, NetSparker.

A culture shift is underway to unite developers and security. The new culture maps a common destination for both groups to inject security into applications painlessly. Secure coding is that destination.



## Uniting Security and Development Through a Cultural Evolution

Thirty-percent of respondents said that ensuring that business-critical applications arrive without vulnerabilities was the top challenge, says The State of Application Security in the Enterprise, Micro Focus. Secure coding circumvents those vulnerabilities. But security needs to coax developers along in a way that makes writing secure code palatable for them. Security needs to expose developers to the philosophy of secure coding and its many benefits.

By taking baby steps, developers can soon write secure code from the start. Companies face some common challenges on the way to building a mature AppSec program. There are effective ways to meet these challenges in our AppSec playbook.

***Coding errors cause up to 90% of software security issues,* which is why secure coding standards are essential. – "What are security standards?"**

- PERFORCE

## The AppSec Culture

**THE CHALLENGE—**Establishing that application security is everyone's business.

**THE PLAY—**Reach out to all stakeholders

Security should initiate conversations with stakeholders, such as the QA specialists and architects, who touch the development lifecycle. Security needs to assess the development teams, their numbers, and their methodologies. With these conversations, security and development can work toward an AppSec recipe for siloed subgroups and the organization as a whole.

"More companies are concerned with the security of their applications than they were a year ago, survey results show. More than one-quarter (28-percent) say their level of concern has increased dramatically, while 20-percent say it has increased slightly," says The State of Application Security in the Enterprise, Micro Focus. The increasing buy-in from technology influencers and decision-makers eases the message that application security concerns the entire organization.

Once everyone is on board for cultural change, they can all do their part. For example, project managers can build more time into projects so that developers can code more securely.

*SAMM—the OWASP Software Assurance Maturity Model—is an industry-accepted framework with actionable ingredients for creating an optimal AppSec program recipe. SAMM evolved into version 2.0 towards the end of 2019 and into early 2020, offering more content about how developers deliver software so that they can code securely.*

**THE CHALLENGE—Developers are focused on differing software, old and new, and differing methodologies.**

While some developers support legacy software with more traditional architecture stacks, others work with new architectures including, but not limited to applications built on APIs and cloud-based applications where the clear lines of delineation between the application and infrastructure as code are becoming harder to determine. Each team may use DevOps, Agile, Waterfall, or hybrid development methodologies suited to the given project. The AppSec program must work for the various ways disparate groups within an organization perform the day-to-day business of software development.

And AppSec must protect all the software. That doesn't just mean securing the applications that the company considers to be mission- or business-critical. A holistic security strategy involves the entire application "portfolio." Leveraging solutions that address all applications–whether outsourced or built in-house or via open-source components–and the entire software development lifecycle (SDLC) is key to up-leveling your security posture," says 5 Irrefutable Reasons to Prioritize Software Security, Checkmarx.

**THE PLAY—Know the developers and their projects and approaches.**

Understand the developers' daily grind across projects and methodologies. Determine the first steps to introduce secure coding to each siloed team. AppSec enables security to apply the best SAMM ingredients in a recipe specific to their organization so that they can acclimate developers step-by-step. An AppSec recipe enables the enterprise to inject AppSec into the existing SDLC. Security can't drop policy or process requirements on SDLCs without knowing how developers function.

SAMM ingredients also enable the application security team to gauge developer software security practices and orchestrate an increasingly balanced AppSec program over time. SAMM ingredients lead to mature AppSec programs and activities that the organization can measure.

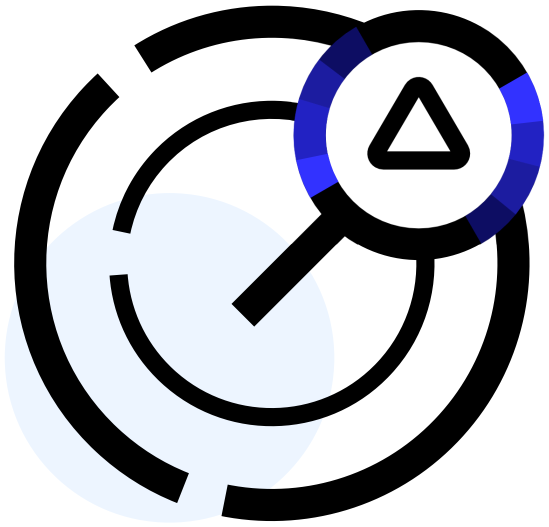
For example, SAMM has an ingredient for creating useful implementation review checklists for secure coding best practices based on the implementation language, platform, and typical technology stack. – Software Assurance Maturity Model, v1.5. in-house or via open-source components– and the entire software development lifecycle (SDLC) is key to up-leveling your security posture," says 5 Irrefutable Reasons to Prioritize Software Security, Checkmarx.

***OPEN-SOURCE SOFTWARE SECURITY VULNERABILITIES DOUBLED IN 2019, ACCORDING TO THE RISKSENSE SPOTLIGHT REPORT.***

**THE CHALLENGE—Developers chase increasing vulnerabilities through reactive testing, repeatedly fixing the same bugs in an endless cycle.**

Developers feel like they're running in a hamster wheel--coding, testing, and yet seeing the same vulnerabilities return[MOU1]  even as new vulnerabilities appear. The circular process is expensive and exhausting, and the technical debt grows as testing continues.

**THE PLAY—Introduce subtle growth in secure coding practices by seamlessly nurturing new programming habits close to how developers already write their code.**



***"It only takes a second to introduce a vulnerability but often many days to find and fix it. Without the right tools and streamlined workflows, many organizations are struggling with a backlog of issues that keeps growing."***

– REPORT, NEW VULNERABILITY FOUND: EXECUTIVE OVERCONFIDENCE – NETSPARKER

*"Complex procedures can lead to inconsistent results or worse, and developers may ignore them completely. You should avoid reinventing the wheel and stick to proven security and secure coding best practices. The OWASP Foundation offers many valuable resources, including the OWASP Top 10, which features the most common security risks and is thus a good starting point,"* says Secure coding practices every developer should know, Snyk.

Companies can monitor how well developers are assimilating secure coding best practices. SAMM has an ingredient for gauging developer progress in secure coding. Implementation Review 2 uses *"routine analysis results to compile historical data on per-team secure coding habits"* – Software Assurance Maturity Model, v1.5.

**THE CHALLENGE—Application security adds time and money to every project. But development needs to accelerate as users demand new feature releases with increasing regularity.**

**THE PLAY—Bring secure coding into developers' existing processes and workflow to keep projects moving along.**

*“We know that it is easier to find and fix issues in applications that have less coding baggage — small application size, using modern languages and frameworks. But even with the “baggage,” development teams that use secure coding practices, such as frequently scanning for flaws, integrating and automating security checks, and taking a broader look at the application’s health, are more likely to have better success with their secure software development efforts,”*

– VERACODE STATE OF SOFTWARE SECURITY, V11

According to the report, New Vulnerability Found: Executive Overconfidence, NetSparker, “the effectiveness of complex workflows related to development, testing, and operations now has a direct bearing on the business.” Fixing loads of bugs at the end of the development pipeline costs developers even more time, slows Time-to-Market, and costs the business money and competitive advantage. Secure coding is much less expensive and faster overall.

**THE CHALLENGE—Not all tools and processes that developers choose make their lives more comfortable in the long run.**

Development leadership is driving decisions on developer tools in a changing world where security is an increasing concern. Developers are excited at the prospect of increasingly automating their work.

**THE PLAY—Security can help developers choose the best tools for ease of development and secure coding.**

Optimal tools support secure coding while remaining intuitive and familiar. According to 5 Irrefutable Reasons to Prioritize Software Security, Checkmarx, organizations must adopt developer-centric tools that keep developers within their preferred environments to make it easier for them to embed security into their processes.

***“Shoutout to Python,* which still has a relatively low percentage of vulnerabilities, even though its popularity, especially in the open-source community, continues to rise. Hopefully, this is a result of secure coding practices and not lax security research for python projects.”**

- THE STATE OF OPEN-SOURCE VULNERABILITIES 2020, WHITESOURCE SOFTWARE

Though this is good news for developers, there are caveats. While new automated testing and development tools can help, security needs to apply governance to automation to ensure security milestones in the development pipeline. Security also needs to implement tools properly. For example, suppose development implements a static analysis development tool too quickly. In that case, it presents many false positives, and developers get much noise that they can’t use.

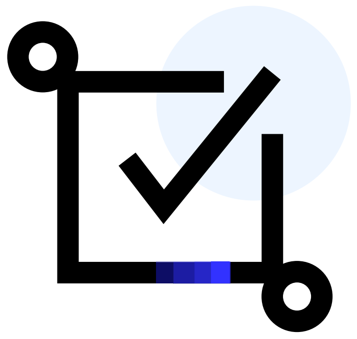
By working with developers on timelines for fixing vulnerabilities with certain severity-levels, security can ensure those deadlines are reasonable, so everyone is on board with them. SAMM has an exception process to ensure the governance of bug fix timelines even when there are factors beyond the organization's reach.

Say, for example, that an SLA requires development to address high severity findings within 30 days. Yet, there is a finding in a third-party component with no patch available. Governance can provide an exception process to analyze the risk and allow the developer to continue their work. The exception tracks the issue until there is a patch.

**THE CHALLENGE—Waiting for the culture change to produce secure code.**

**THE PLAY—SAMM has an ingredient for establishing baseline coding steps, using an Implementation Review. The Review assesses an organization's source code to aid vulnerability discovery and related mitigation activities and establish a baseline for secure coding expectations.** – The Software Assurance Maturity Model, v1.5.

The business can see results as soon as security injects baseline coding steps and requirements into the SDLC. Early skills such as configuring sensitive cookies and properly coding new input fields bear fruit. Then, security can layer on additional requirements over time to realize more progress.



## Three Tools to Ease Secure Coding

### 1. Create an application risk register

Use an application risk register to identify and prioritize applications the company should care about[MOU2], so everyone can apply their time optimally. An application risk register demands a risk score for every application so that security can determine the appropriate frequency and depth of testing for each application based on the level of risk. [MOU3] Prioritizing applications saves time for developers, too. They won't have to meet an excess of requirements for less critical applications.

### 2. Use a third-party dependency review process

"Seventy-percent of applications transfer at least one flaw from their open-source libraries. Thirty-percent of applications has more flaws in their open-source libraries than in-house code has" – Veracode State of Software Security, v11. A third-party dependency review process is essential. Suppose the code uses many third-party libraries with component dependencies that are not under the organization's control. In that case, security and development have to ensure they don't make secure source code vulnerable.

### 3. Apply governance rules so developers can keep working

Use an application risk register to identify and prioritize applications the company should care about[MOU2], so everyone can apply their time optimally. An application risk register demands a risk score for every application so that security can determine the appropriate frequency and depth of testing for each application based on the level of risk. [MOU3] Prioritizing applications saves time for developers, too. They won't have to meet an excess of requirements for less critical applications.

***COMPETITIVE DIGITAL COMPANIES UNDERSTAND THE CRITICALITY OF SECURE CODING. THEY MAINTAIN EFFORTS TO TRAIN EMPLOYEES IN CYBERSECURITY. THEY CHALLENGE THEM TO LOOK DEEPLY INTO CYBERSECURITY AND ELEVATE THEIR STANDARDS, ACCORDING TO "CYBERSECURITY IN A DIGITAL ERA,"***

MCKINSEY & COMPANY

**THE CHALLENGE—Thirty-five percent of organizations say that less than half of their development teams participate in formal security training, according to the 2020 Modern Application Development Security report from Enterprise Strategy Group. How does security get all this data out to developers when emails and new information constantly bombard them?**

**THE PLAY—Establish an AppSec one-stop-shop.**

A one-stop shop is an excellent self-service resource for busy developers. It offers consistency and ease of access to information to make secure coding simpler. Everything on security for development and anyone involved with the SDLC sits at one convenient link. Security can link to related policies and standards and keep links to outside information and announcements in reverse chronological order.

If security is finding common bugs across software, develop resources on coding against them and put those in the one-stop-shop. If the organization offers lunch-and-learns with guest speakers, publish notices in the one-stop-shop. Follow up by publishing the results of these presentations with any recordings from the event. Security can standardize secure coding checklists from across the various teams and put them in the one-stop-shop.

Publish a link to the one-stop-shop in every email that crosses the development group and stakeholders.

***CONSIDER ADDING FIVE PRACTICES TO SECURE CODING CHECKLISTS:*** minify and obfuscate code to make it harder to access and read; avoid shortcuts such as leaving hard-coded credentials and security tokens as comments; automate scanning and code reviews; avoid components with known vulnerabilities; audit and log to detect incidents when you deploy code to production. - Secure coding practices every developer should know

SNYK

***SIXTY-FIVE-PERCENT* OF APPLICATIONS TESTED HAVE AT LEAST ONE VULNERABILITY FROM THE OWASP TOP 10 VULNERABILITIES LIST. ANOTHER *59%* HAVE AT LEAST ONE VULNERABILITY FROM THE SANS 25 LIST.**

VERACODE STATE OF SOFTWARE SECURITY, V11

Security can create a one-stop-shop on a wiki, SharePoint site, or Confluence page, for example. It can include OWASP articles on how to fix common vulnerabilities.

“Secure code reviews use automated tools, checklists, threat modeling, software development experience, and security experience to identify security vulnerabilities. Integrate secure code reviews as part of the Software Development Life Cycle (SDLC),” says 9 Secure Code Review Best Practices For Your Web Application, Towards Data Science. Developers can use the secure coding checklists at the one-stop-shop in code reviews of junior developers’ work. But security must socialize and promote it in the one-stop-shop, so developers use it. Developers must get information from a one-stop-shop internally rather than by searching the web where they may get misinformation.

**THE CHALLENGE—Security personnel are outnumbered far and away by developers.**

The data demonstrates a valley between available security personnel and the glut of software developers. Seventy-percent of ISSA (Information Systems Security Association) members believe their organization has been impacted by the global cybersecurity skills shortage, according to The Life and Times of Cybersecurity Professionals 2020 report from the Enterprise Strategy Group. At the same time, the world can expect to see some 27.7 million software developers globally and climbing by 2023, according to Evans Data Corporation.

***SAAS VENDORS MUST ADVANCE THEIR SOFTWARE SECURITY CAPABILITIES.* THEY MUST BAKE SECURITY INTO THE DEVELOPMENT PIPELINE. THESE EFFORTS MUST INCLUDE TEACHING EACH DEVELOPER TO CODE SOFTWARE SECURELY, ACCORDING TO “CYBERSECURITY IN A DIGITAL ERA,”**

MCKINSEY & COMPANY

**THE PLAY—Extend AppSec resources as far out into the organization as possible.**

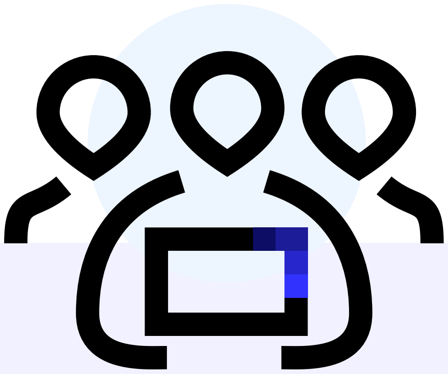
Create an AppSec Advisory Board or Security Champions Program. Get someone from development from each siloed team to join the board or program and appear for one hour a month. Developers should be amenable since the Advisory Board gives them a say in application security approaches that affect how they work. Keep the meetings actionable with SAMM initiatives that security and development can use. These people take what they learn back to their groups, so security staff doesn’t overextend themselves. It’s a practical and repeatable effort. “Only 15% of our survey respondents implement Security champions programs, widely recommended in OWASP Software Assurance Maturity Model (SAMM),” says The State of Open Source Security 2020 Report from Snyk. Organizations that fall into the remaining 85% are missing an opportunity to inject AppSec into their SDLs.

***SOFTWARE INEVITABLY HAS FLAWS, SO UNTIL NOW, WE'VE FOCUSED ON UNDERSTANDING WHAT IT MEANS WHEN WE SAY THAT APPLICATIONS HAVE FLAWS. HOWEVER, ACCEPTING THERE ARE FLAWS DOES NOT MEAN THERE IS NOTHING WE CAN DO. INDEED, MANY COMPANIES MAKE IT THEIR BUSINESS TO HELP DEVELOPERS WRITE INCREASINGLY SECURE CODE.***

VERACODE STATE OF SOFTWARE SECURITY, V11

**THE CHALLENGE—Building an AppSec program is challenging.**

**THE PLAY—Get involved with local OWASP (Open Web Application Security Project), ISSA, or ISACA chapters to meet and talk with peers from other AppSec programs. Through peer support and comparing notes, security teams can increase their confidence, knowledgebase, and resources in maturing their AppSec programs.**



## Keep Them Happy

To reach secure coding nirvana, strive to achieve the benefits in a happy coder checklist, based on data from Sonatype:

**1**

Happy developers are **3.6x** more likely to pay attention to security.

**2**

Happy developers spend more time thinking about security than grumpy developers in less mature organizations do.

**3**

Happy developers have the tools they need to complete their job.

**4**

Developers who receive training on how to code securely are 5x more likely to enjoy their work.

Stimulate cultural change to mature the software development group. Provide the best in breed development tools that address secure coding. Train developers so they know how to meet secure coding expectations.

Then, bask in the happiness of secure coding.

1. https://www.perforce.com/blog/qac/secure-coding-standards#:~:text=Secure%20coding%20standards%20are%20rules,that%20could%20compromise%20software%20security.
2. https://www.esg-global.com/hubfs/ESG-ISSA-Research-Report-Cybersecurity-Professionals-Jul-2020.pdf
3. https://evansdata.com/press/viewRelease.php?pressID=278
4. https://www.mckinsey.com/~/media/McKinsey/Business%20Functions/Risk/Our%20Insights/Cybersecurity%20in%20a%20digital%20era/Cybersecurity%20in%20a%20Digital%20Era.pdf
5. https://info.veracode.com/report-state-of-software-security-volume-11.html
6. https://www.businesswire.com/news/home/20200608005032/en/Open-Source-Software-Security-Vulnerabilities-Doubled-in-2019-According-to-RiskSense-Spotlight-Report
7. https://resources.whitesourcesoftware.com/research-reports/the-state-of-open-source-vulnerabilties-2020
8. https://towardsdatascience.com/9-secure-code-review-best-practices-for-your-web-application-28a519b079e9
9. Software Assurance Maturity Model, v1.5
10. 10.https://www.sonatype.com/hubfs/DevSecOps%20Survey/2020/DSO_Community_Survey_2020_Final_4.1.20.pdfhttps://www.sonatype.com/hubfs/DevSecOps%20Survey/2020/DSO_Community_Survey_2020_Final_4.1.20.pdf
11. 11.https://www.esg-global.com/research/esg-master-survey-results-modern-application-development-security
12. 12.https://owasp.org/www-pdf-archive/SAMM_Core_V1-5_FINAL.pdf
13. 13. https://www.checkmarx.com/blog/2021-software-security-predictions-our-experts-weigh-in/
14. 14. https://www.netsparker.com/executive-overconfidence-web-application-security-survey-report/
15. 15. https://snyk.io/open-source-security/
16. 16. https://www.checkmarx.com/blog/5-irrefutable-reasons-to-prioritize-software-security/
17. 17. https://snyk.io/learn/secure-coding-practices/
18. 18. https://www.microfocus.com/media/analyst-paper/the_state_of_application_security_in_the_enterprise_analyst_paper.pdf

GUIDEPOINT
SECURITY

2201 Cooperative Way, Suite 225, Herndon, VA 20171
guidepointsecurity.com • info@guidepointsecurity.com • (877) 889-0132