WHITE PAPER

Raising the AppSec Bar

GUIDEPOINT®
SECURITY

The need for a fully integrated application security process is critical

- Vulnerabilities in software and web applications account for a significant portion of the most common attack vectors.
- Application security is shifting to the left, but more still needs to be done to fully integrate AppSec earlier into the software development lifecycle.
- Organizations can prioritize and improve the AppSec process through the use of an AppSec Maturity Model.
- By incorporating several best practices into the software development lifecycle, organizations can achieve higher levels of AppSec maturity.
- Application Security as a Service (AppSecaaS) can extend an enterprise's security capabilities by offering additional security support.

How Organizations Can Benefit From Application Security-as-a-Service

To ensure improved application security (AppSec), enterprises need to plan for the continuing integration of AppSec into the software development lifecycle. This white paper provides the context in which to understand AppSec and incorporate it more fully into the development process to maximize security and minimize the impact of vulnerabilities, threats and breaches.

The statistics associated with the security of today's applications are sobering. Studies have found that almost half of all breaches are the result of a software vulnerability and that at least three-quarters of all applications tested have at least one security flaw, with many applications having multiple flaws.

Applications Are Under Attack

According to a 2020 report from Forrester Research, software vulnerabilities and web applications account for 42% and 35% of the most common attack vactors.

Looking Back to Look Forward

To appreciate and understand the importance of application security within the context of today's complex software development lifecycle (SDLC), it is important to take a brief look at how AppSec has changed over the last couple of decades.

The complexity of the applications themselves adds to the security concerns. Most rely on supplemental components, often from third-party vendors who may or may not be on top of security updates. Applications must also evolve regularly to support new platforms and frameworks, which in turn creates new attack surfaces. Many applications are built using open-source software, which makes the development process fast and nimble but also introduces vulnerabilities. And unsecured application programming interface (API) endpoints and malware injections into unprotected scripts enable threat actors to easily compromise software.

While not all applications contain flaws that will automatically result in a security risk, these issues do underscore the importance of application security (AppSec), particularly as the world becomes more dependent on the variety of cross-platform applications that support enterprise and personal needs.

What's clear from the notable increase in attacks on web applications and software is that companies need to think critically about securing the growing number of applications that support business activities.

In this paper, we'll explore the world of application security, including the current state of AppSec from the perspective of where it was twenty years ago compared to where it is today and where some anticipate it will be in the future. We'll also look at AppSec security models, including the most common models and how they integrate into today's DevSecOps process. We'll explore some best practices to help organizations integrate AppSec seamlessly into software development. And, finally, we'll discuss the benefits of using Application Security as a Service (AppSecaaS).

THE MILLENNIAL SHIFT

During the early 2000s, increasing cyber threats resulted in organizations moving away from security managed by a development team to a model that supported a separate group of security professionals. While the shift to dedicated security was necessary, an unintended consequence was siloed development and security departments occupied by groups engaged in their respective area of expertise. At this point, software development was driven by the heavy use of the Waterfall software development methodology, with the application security process more of an afterthought. Development teams engaged in very little pre-implementation security planning, primarily due to the time and expense. Few software developers were familiar with basic AppSec concepts such as cross-site scripting or some of the more common software vulnerabilities. Application security typically didn't happen until the very end, well after the application was completed.

TWO DECADES OF DIGITAL TRANSFORMATION

Twenty years later, as the world finds itself amid a massive digital transformation, the culture around disparate development and security teams is changing. Security is "shifting left" as organizations recognize the need for it to be an essential part of the software development lifecycle. Cybersecurity professionals credit the shift to the fact that more developers are now intimately aware of and understand security, thanks largely to the growth of mobile device use. With so many applications requiring the use of application program interface (API) endpoints, most organizations understand the need to institute the AppSec process early. Application developers also have mostly eschewed Waterfall methodology in favor of an Agile system to create a more responsive and integrated approach to software development and security. However, AppSec is still finding its place, often having to adapt to a blended DevOps development model.

With developers adopting a more fluid approach to development and also trying to maintain the speed of development, application security is still looking for ways to integrate security into that process, while maintaining overall governance.

APPSEC MATURITY WITHIN THE APPLICATION DEVELOPMENT PROCESS

Discussions abound as AppSec professionals anticipate a greater shift to the left when it comes to application security. However, testing earlier in the application development process means that developers need to have the skills to write more secure code, and this, in turn, means addressing the training and education skills gap that exists within the development community. To write secure code, developers need to understand what secure code is as well as have access to information about the latest tools and techniques threat actors are using to circumvent code. This presents an opportunity for AppSec practitioners to bridge the gap by mentoring, educating, and teaming with developers to help improve the overall level of security within code.

In addition, as frameworks and services are required to adhere to secured baseline containers, infrastructures, and code, security-as-a-default will become critical. This means AppSec professionals need to work with developers to baseline secure code requirements and standardized frameworks for key components such as input validation and the ability to assess components, images, and custom code to ensure that they meet a basic level of security compliance before use.

INTEGRATING APPSEC INTO INFRASTRUCTURE

The lines between application and infrastructure are blurring as they become more integrated, which makes AppSec increasingly important in the infrastructure-as-code development process. Not only must the infrastructure code itself be secure, but the application and infrastructure must also be strong enough to withstand vulnerabilities introduced by third-party applications and components. This makes the integration of software composition analysis (SCA) tools a critical supplement to existing AppSec tools used in the development pipeline.

AppSec professionals can also anticipate more automation in the area of compliance. This means integrating compliance frameworks and controls that can be managed as repeatable, testable, reusable, and with self-documenting audit capabilities. Two other automation areas may become prominent: the security content automation protocol (SCAP)—a blend of interoperable specifications derived from community ideas—and the DevSec hardening framework, which adds a layer to the automation framework to configure and secure operating systems and services.

THE FUTURE OF SECURITY IN THE WORLD OF APPLICATION DEVELOPMENT

As application development continues to evolve, organizations need to rethink how security can be more fully integrated into the code-writing process. Two key necessities stand out—supporting application security as a mature and fully assimilated component of the SDLC, including testing earlier in the application development process, and adapting secure code development to the increasingly blurred lines that exist between an application and the infrastructure where it is housed.

Measuring AppSec Maturity

Despite the shift to a more integrated SDLC, in which AppSec is incorporated actively into the development timeline, the reality is that security aspects are still sometimes overlooked as part of DevOps. This challenge is due in no small part to time and resource challenges and the inability to devote AppSec staff to software development since AppSec practitioners are usually vastly outnumbered by development staff.

The challenge then becomes—how do organizations prioritize application security given resource constraints? The use of AppSec maturity models can help.

Within the security industry, there are currently three commonly used AppSec maturity models:

- Software Assurance Maturity Model (SAMM)
- Building Security In Maturity Model (BSIMM)
- DevSecOps Maturity Model (DSOMM)

Software Assurance Maturity Model (SAMM)

The OWASP Software Assurance Maturity Model (SAMM) is an open framework designed to aid organizations in analyzing their software security practices, to help create a security process or program based on iterations, to demonstrate security practice improvements, and to measure and define security activities. SAMM is designed to enable organizations to determine the target maturity level for the security practices that are the best fit for their organization and then adapt process templates based on organizational needs.

SAMM was created in 2009 and based initially on the Waterfall development methodology. SAMM recently underwent a significant overhaul to a 2.0 version, which broadened the focus to adapt to the newer approach in application development. According to OWASP, the 2.0 version supports frequent updates through incremental changes to parts of the model and regular updates based on community contributions. What many AppSec professionals like about SAMM is that it supports the need to incorporate third-party components and dependency management, in addition to data protection, data loss, data retention, incident management, secure build process and other related issues. Like DSOMM, SAMM is an OWASP project, community-built, and has no proprietary components.

Building Security in Maturity Model (BSIMM)

The BSIMM framework is often called an observational maturity model. While sometimes compared to SAMM, BSIMM is somewhat different in that the data produced by BSIMM comes from studying actual organizational software security initiatives. BSIMM is not prescriptive—in other words, it doesn't tell organizations what they should do from a security perspective. Instead, it represents what organizations are actually doing. The BSIMM framework report includes data from more than 100 organizations and is

broken down by vertical and industry type if the data comes from a sufficient number of organizations to make it statistically significant.

BSIMM and SAMM share similarities in the areas of domains and activities. However, BSIMM observes a total of 121 activities, including governance, intelligence, software development lifecycle touchpoints, and deployment, with some organizations only engaging in a few activities. This means that an organization could have a relatively immature absolute AppSec program yet still get credit for having mature security activities. Activities may also disappear from certain BSIMM iterations depending on whether organizations are performing activities within those areas.

DevSecOps Maturity Model (DSOMM)

The DSOMM model focuses on specific DevOps approaches and how security can be applied to those processes. Created by Open Web Application Security Project® (OWASP), DSOMM is specifically designed to support security within the Agile development framework by prioritizing and enhancing security measures in DevOps. DSOM defines four maturity levels (Level 1=Least Mature to Level 4=Most Mature), with each maturity level containing four areas for evaluation:

- **Static Depth**—The level or degree of static code analysis.
- **Dynamic Depth**—The extent to which dynamic scans are executed.
- **Intensity**—The level of strength and power in most executed attacks.
- **Consolidation**—The extent to which findings are managed and processed.

Every activity in the DSOMM model includes representative control mapping back to both ISO and ISO/IEC 27001 (Information Security Management). DSOMM is not proprietary, and because it is an industry-accepted framework, it allows organizations to make “apples to apples” comparisons when it comes to defining the maturity of their product versus that of other products on the market. DSOMM can also be integrated into the entire culture of all teams involved in the software development lifecycle.

Integrating AppSec in SDLC—Best Practices

Organizations can achieve high levels of AppSec maturity by integrating a few key approaches into development activities.

BEST PRACTICE #1

Identify Documentation Resources

Many organizations are good at developing documentation and then putting the documentation on the proverbial shelf and forgetting about it. Documentation related to SDLC methodology, security policies, and developer guidance and resources can provide tremendous insight into not only potential security threats, but also how to improve the overall security model.

BEST PRACTICE #2

Ensure Transparency

Transparency means ensuring everyone involved in the SDLC knows goals, objectives, and the resources available, so an AppSec framework can integrate seamlessly into overall processes and priorities. Implementing AppSec on a development team without first being clear about the end goal often results in resistance or avoidance of the AppSec processes.

BEST PRACTICE #3

Foster a Security-driven Culture

Reframing the cultural mindset to make security a priority, alongside key SDLC components such as innovation, design, and implementation, can go a long way to supporting a seamless AppSec integration.

BEST PRACTICE #4

Treat Standards as Beneficial Tools

Make standards meaningful by discussing security models and frameworks within the context of a 'beneficial tool' instead of a 'dictatorial document'. Once the development team understands the usefulness of AppSec frameworks, it will be easier to integrate them into full policies and procedures.

BEST PRACTICE #5

Make Everyone Responsible for Application Security

Make everyone on the team responsible for security and encourage and reward SDLC team members that quickly embrace security standards and apply security tools during software development.

BEST PRACTICE #6

Bi-directional Communications

Not every process in a security maturity model framework will be the right fit during the software development lifecycle. Enable the development team to provide feedback to both AppSec practitioners and development management on what works and what doesn't. By the same token, allow AppSec practitioners to engage and communicate with developers to provide recommendations on how best to apply security methods.

AppSec Integration Best Practices

- ✓ Conduct a resource review to identify all existing relevant documentation.
- ✓ Be transparent with your development team about your security goals, objectives, and the resources available.
- ✓ Recognize and promote "security" as a key component of the SDLC
- ✓ Help the team recognize that security frameworks are highly useful and beneficial tools to be applied to the SDLC.
- ✓ Make application security everyone's responsibility.

GUIDEPOINT SECURITY'S APPSECAAS

Learn more about GuidePoint Security's AppSec as a Service offerings.

AppSec as a Service (AppSecaaS)

Application security as a service can extend an organization's security capabilities by providing added security support with applications or by assuming day-to-day AppSec activities.

AppSecaaS Benefits

- **Cost Savings**—AppSecaas offers customized solutions and services which can be scaled up or down depending on an organization's own unique needs.
- **Experience & Expertise**—AppSec-as-a-service practitioners can offer a level of expertise and experience in key areas, such as testing, validation of findings to identify false positives, and remediation verification to ensure a robust resolution for any vulnerabilities identified.
- **Current Security Features and Updates**—AppSec-as-a-service practitioners make it their job to stay on top of the latest security tools, threats, resources, and updates.
- **Resource Availability**—Hiring security practitioners can be challenging from both a cost and availability perspective. AppSec as a service can provide those critical resources with the right expertise and experience when and where you need them.
- **Seamless Integrations**—AppSec as a service can reduce the friction associated with integration or automating security tooling directly into the CI/CD pipeline or SDLC.
- **Customization**—Integrations can include source code management, IDE, and other IT systems, such as ticketing, logging and monitoring, incident management, or GRC platforms.

Who Should Use AppSec as a Service?

Organizations that can most benefit from using an application-security-as-a-service model include organizations:

- That plan to purchase an application security testing tool.
- That own an application security testing tool and aren't getting value from the product.
- That have a small security team without an application security specialist.
- That have large application portfolios.
- With a large in-house development team.
- With applications that store or access sensitive information (PII, PHI, intellectual property).
- With regulatory compliance requirements for secure development.

Conclusion

The application security process has changed dramatically over the last few decades, shifting from one in which security was more reactive, taking place at the end of the SDLC, to one in which security is now more proactively integrated into the software development lifecycle. But keeping up with application security can still be a challenge, as the number of endpoints and applications grows exponentially. In fact, software vulnerabilities offer cybercriminals some of the easiest and most prolific attack vectors. Organizations can move forward to fully integrate security into their software development process by following a few critical steps during the SDLC, including employing an AppSec maturity model, engaging in several established AppSec best practices, and working with an application-security-as-a-service provider to offer added security support.

Security stakes are higher than ever. By embracing application security and integrating it fully into the software development process, organizations can help minimize the overall impact of software vulnerabilities, as well as threats and breaches.

About the Authors

Kristen Bell

Senior Manager of Application Security Engineering, GuidePoint Security

Kristen Bell is the Senior Manager of Application Security Engineering for GuidePoint Security. She has been in the Application Security industry for over 16 years, with prior experience as a developer. Before joining GuidePoint, she consulted for numerous companies performing application security services. Kristen's background includes work in the government sector, building application security programs, and providing guidance in secure application design.

Kristen's experience also includes conducting application security assessments and database security reviews, secure SDLC consulting, as well as working with clients to improve their enterprise vulnerability management. Her ability to bridge the gap between technical and non-technical people, coupled with her strong interpersonal skills, has made Kristen a strong champion for application security frameworks and controls for her customers.

Robert Darwin

Director of Security Architecture, GuidePoint Security

Robert Darwin is the Director of Security Architecture at GuidePoint Security and leads the AppSec and DevSecOps Technical Services for the North Central Region. His work includes both professional and managed services, product-centric engineering, and consulting, relating to both AppSec and DevSecOps. His professional experience covers program development specializing in Software Security, Open-Source Security, Threat & Vulnerability Management, and DevSecOps maturation.

Robert is a proponent of developer enablement and self-service, embedding security capabilities through automation directly into the development workflow and empowering engineering teams to build security in and optimizing the secure software development lifecycle.

GUIDEPOINT®
SECURITY

2201 Cooperative Way, Suite 225, Herndon, VA 20171
guidepointsecurity.com • info@guidepointsecurity.com • (877) 889-0132
05.2021