REAL-WORLD RETROSPECTIVE

Application Security Lessons Learned

VERACODE

As a Veracode customer, you already understand the importance of having a formal application security program in place. But as you continue to develop your program, you may not always know what the right next step should be.

Let someone who's already been through the process show you the way. Colin Domoney, a DevSecOps consultant, was part of the team that worked to build an application security program from the ground up at a global investment bank. In this guide, we've compiled some of his key takeaways and lessons learned from building out that program. Read on to learn what steps you can take to continue to develop a comprehensive application security framework at your organization.

LESSONS LEARNED IN EARLY STAGES OF APPSEC DEPLOYMENT

Get stakeholder buy-in

Application security isn't a solitary pursuit; it's a far-reaching initiative that affects the processes and priorities of a variety of different individuals and groups in your organization. Without getting stakeholders on board early, your program will never get off the ground. **Domoney recommends starting the process of getting stakeholders on board by:**

- Identifying the business owners of affected applications
- Identifying application and technical leads
- Socializing the program internally through such things as internal portals, events, instant messaging, etc
- Holding kickoff sessions in key locations

To achieve support and secure resources for your AppSec program, you need to provide the right metrics and demonstrate risk reduction to your enterprise leaders. At the bank, Domoney and his team found they needed to collect key performance indicators (KPIs), balanced scorecards, and metrics, and then demonstrate to the business units and stakeholders that their investment in an application security program was delivering a measurable ROI based on a demonstrable and continued risk reduction.

Learn more with our guide, *Everything You Need to Know About Getting Application Security Buy-In*.

Conduct an asset inventory

One of the first challenges when kicking off an AppSec program is understanding where the applications are hosted, where the codebases are, who's responsible for building and maintaining them, what development languages are used, how critical they are to the organization, and so on.

Domoney says that beginning his program without first getting a clear picture of the application landscape and its dependencies would have been inefficient, ineffective, and eventually insecure. He began constructing an application inventory by:

- Determining how the bank kept track of its application assets (who owns the apps, what their business unit/owner/criticality/regulatory requirements is, etc.)
- Getting access to software source control systems
- Understanding business criticality classification
- Aligning with risk functions to agree on scoping

Domoney also recommends that, in order to ensure you're focused on the most critical applications, you should consider a number of indicators:

- Does the application require an application penetration test?
- Is the application external-facing?
- Has the application been the subject of recent incidents?
- Does the application have any particular regulatory requirements?

In this last instance, for example, the Monetary Authority of Singapore (MAS) guidelines mandate a code review process that may be fulfilled in part by static code analysis, so Domoney used the MAS compliance as an immediate inclusion criterion.

Learn more about prioritizing applications in this blog post.

Secure a mandate

Domoney points out that this was an area where his program fell short. “We should have done this earlier," he says. "In the first three years, we relied on the co-operation of development teams to participate; they could have easily refused, and we would have had no rights to enforce participation. Don’t allow application teams to be de-scoped too easily. Once applications were excluded from the program, it was hard to bring them back into scope.”

He found that his program failed to achieve an initial foothold due to the lack of a clear mandate, which enabled recalcitrant application teams an easy opt-out clause.

Learn more about the best ways to launch your AppSec program in our blog post, "Key Components to Consider When Kicking Off Your Veracode AppSec Program."

LESSONS LEARNED IN LATER STAGES OF APPSEC DEPLOYMENT

Start with an achievable AppSec policy

While a high standard of security is vital, unrealistic expectations and requirements too often lead to people looking for ways to circumvent policies so they can get their work done. A basic fact of application security is that any policy should be only as complicated as it needs to be in order to deliver the necessary results.

A more realistic approach starts with attainable goals — such as the eradication of a specific high-risk threat — and expands from there. As his program expanded, Domoney notes, "One key step that kept the program moving forward smoothly was to set the application security policy to a very achievable level: no XSS, SQLi, or CMDi."

Although his policy was set to a manageable level, again, however, his team had poor governance over the scanning process.

Learn more in our guide, *Everything You Need to Know About Application Security Policies.*

Identify security champions on development teams

A big lesson learned in this phase was the importance of creating and nurturing security champions. A security champion is a developer with an interest in security who helps amplify the security message at the team level. Security champions don't need to be security professionals by training; they just need to act as the security conscience of the team, keeping their eyes and ears open for potential issues. Once the team is aware of these issues, it can then either fix the issues in development or call in your organization's security experts to provide guidance.

Domoney's team learned that the best ways to create these champions were to:

- Coach developers who self-identify as security experts
- Incentivize within teams to encourage security champions

Learn more in our guide, *How to Turn Developers Into Security Champions*.

Develop a standard procedure for proposing, documenting, and processing mitigations

As Domoney and his team got deeper into remediating or mitigating vulnerabilities, they found success by developing a standard procedure for proposing, documenting, and processing mitigations. **This procedure required that each mitigation proposal include the following documentation:**

- **Technique:** type of mitigation in effect
- **Specifics:** specific compensating control in effect
- **Risk:** risks that the mitigation doesn't address
- **Verification:** how the mitigation effectiveness was verified

Use APIs to automate scanning

Domoney and his team eventually performed automatic scanning of all production releases in the CI/CD pipeline. Automation allows security testing to take place without human intervention, providing speed and scale while keeping developers focused on their most valuable tasks.

The key to automating security is to look at the tools you already have in place. Identify the ones you use for continuous integration, continuous deployment, defect tracking, and more to determine the best points to introduce automation. The more you can automate security to make it a part of the developers' process, the less work a separate security team will need to do later on.

Learn more in our guide, *Veracode Integrations*.

Use more than one testing type

When first overseeing AppSec at the bank, Domoney was shocked to discover that many risks were flying below the radar. He found that security teams hadn't been looking at such things as application business logic design, architecture design decisions, or addressing poor secure software development best practices. Instead, they'd been focused on "check-box compliance," which failed to address many of the underlying issues.

He felt that the organization needed to move beyond a legacy approach that relied on "bolt-on" solutions to manage application security. He eventually ensured that all developers had access to Veracode static analysis in their IDEs and that they regularly performed dynamic analysis of applications in production. In addition, his teams used software composition analysis to identify vulnerabilities in open source components.

There is no application security silver bullet. It's going to take more than one automated technique or manual process to secure your applications. Only by gathering the strengths of multiple testing techniques along the entire application lifetime will you drive down application risk in your organization.

Learn more about the different testing types in *Your Guide to Application Security Solutions*.

A mature application security program might seem intimidating to some organizations. But it's important to remember that there is an established series of steps most organizations take when developing an application security program. The keys are to start small, keep things simple, prove the value, and then mature the program over time. In addition, if you build security assessments into the development process, reaching maturity is less daunting.

Learn more about how your peers are tackling AppSec challenges and building out their programs in the **Veracode Community**.

VERACODE

Veracode gives companies a comprehensive and accurate view of software security defects so they can create secure software, and ensure the software they are buying or downloading is free of vulnerabilities. As a result, companies using Veracode are free to boldly innovate, explore, discover, and change the world.

With its combination of automation, integrations, process, and speed, Veracode helps companies make security a seamless part of the development process. This allows them to both find and fix security defects so that they can use software to achieve their missions.

Veracode serves more than 2,000 customers worldwide across a wide range of industries. The Veracode Platform has assessed more than 8 trillion lines of code and helped companies fix more than 36 million security flaws.

Learn more at www.veracode.com, on the Veracode blog and on Twitter.