How to Successfully Implement Shift-Left Security for Your Organization

According to the Systems Sciences Institute at IBM, it's about six times higher to fix a bug during the implementation compared to one that's identified during design. Yet, 86% of developers don't prioritize application security when writing code. The result? A higher risk of vulnerabilities making their way into production, leading to more expensive breaches and damage to reputation.

Now, if you want better quality software, where security is a core component of every feature and function, then it's about time to make a smart move for your organization's bottom line.

Table of Contents

What Stands in the Way of Adopting Shift-Left Security

Implementing Shift-Left Security is not without its challenges. There are some obstacles that organizations have to address before they can guarantee a smooth transition and reap the benefits of early-stage security.

1. **Cultural Resistance to Change**

Everybody finds it hard to break their established habits and workflows, and development teams are no different, especially when integrating new security practices. Introducing security from the beginning could look like you're already adding another task to their already full plates. Without cooperation from the teams, security will stay a second priority, and the chances of vulnerabilities being ignored until the final stages of development will become higher, right when they are harder and more expensive to fix.

2. **Lack of Security Skills**

Let's face it: many developers are not trained in security. Because of this, vulnerabilities could be baked into the code from the start without anyone noticing until it's too late. The consequences are severe: insecure applications, more frequent breaches, and a higher likelihood of expensive fixes or even worse, reputational damage after a breach.

3. **Complexity of Tooling and Integration**

Think about it: what's the process of integrating a tool, related to security or not, late into the development cycle? It would be messy. The complexity of adding new tools creates bottlenecks and confusion that eventually leads to delayed releases. Worse, teams might have to skip essential security checks just to stay on schedule.

4. **Balancing Speed and Security**

This happens to everybody, or at least to those who don't have security integrated into their CI/CD from the beginning. I'm talking about teams rushing to release features at a rapid pace, while security is being seen as a bottleneck. When security is left out until the last minute, it not only slows down the final push to production but also results in very expensive delays because of unforeseen vulnerabilities. There will be tension between shipping quickly and shipping securely, and often, speed wins out—leaving your product exposed to potential threats.

5. **Keeping Up with New Threats**

Like it or not, cybersecurity is evolving at a breakneck pace, and today's security measures are not enough tomorrow. As new threats show up, teams find it hard to keep their apps protected, which leaves gaps in their defenses. This constant struggle to catch up makes it difficult to maintain a strong security posture, increasing the likelihood of a breach that could result in significant financial and reputational damage.

Before we continue, let me just tell you that, yes, it could be challenging and tedious to adopt a Shift-Left approach to security, but these challenges that we discussed are small prices to pay. Keep on reading to know why.

How to Implement Shift-Left Security

Implementing shift-left security is more than just adding extra steps to your development process. Integrating security practices from the very beginning of the SDLC will drastically reduce vulnerabilities, minimize expensive last-minute fixes, and make sure that your product is secure by design. Now, let's talk about how to get your teams to catch issues before they become expensive problems, improve software quality, and speed up your time-to-market by preventing delays caused by late-stage security discoveries.

Embed security into every phase of SDLC.

To effectively minimize vulnerabilities, security checks must be done constantly from the very beginning—starting with the requirements phase, through design, and into coding. This will help you guarantee that security is a core part of the process instead of just something added at the end.

Adopt automation for security testing.

You know this: manual security checks slow down the process and are prone to human error. With Static Application Security Testing (SAST) and Dynamic Application Security Testing

(DAST) integrated into your CI/CD pipeline, you can continuously monitor for vulnerabilities without disrupting your team's workflow. Not only that, automation makes sure that important security checks are completed consistently and quickly.

Make threat modeling a continuous practice.

This is one of the most common mistakes that organizations make. Instead of making threat modeling a one-time exercise, integrate it into your development process as a continuous process. Identify potential vulnerabilities as the software evolves. By doing so, you catch risks early and proactively address them before they become serious problems.

Foster cross-team collaboration.

Collaboration between your development, security, and operations teams is very important. Having real-time communication between them will help address security concerns immediately and avoid expensive delays later in the development cycle. When teams work together, security becomes a shared responsibility, making your processes more efficient and secure. You're essentially covering all your bases here.

Upskill your teams with security training

Last but definitely not the least, your teams need the right skills to effectively implement Shift-Left Security. The security training that you'll provide them has to be tailored depending on the role they play in the process of developing your product. Developers, security engineers, and operations personnel each have unique responsibilities, and their training should reflect that. Here are some of the things that you might want to consider:

- **Role-Based Learning Paths**: Customized training paths that focus on the specific skills each team member needs, whether it's secure coding or cloud security.
- **Interactive Labs**: Hands-on labs where teams can practice secure coding, DevSecOps, and threat modeling in a safe, sandboxed environment.
- **Comprehensive Content Library**: Your teams have to have access to a broad range of resources that cover everything from fundamental security practices to the latest trends in cloud and application security.
- **Progress Tracking & Analytics**: This one's for you. Use admin tools to monitor course completion and assess team readiness with detailed progress tracking and analytics. This helps ensure that your teams stay on top of their training and are prepared to handle emerging security threats.
- **LTI (Learning Tools Interoperability) Support:** Look, if your teams are already using an LMS, you don't want them bouncing between platforms to do security training. With LTI, AppSecEngineer integrates directly into your LMS, so your team can access everything in one place without disrupting their workflow. It's seamless, and you keep everything under one roof.
- **SCORM Compliance:** Now, if you use SCORM for tracking e-learning, we've also got you covered. AppSecEngineer's SCORM packages plug right into your LMS, so you can

track progress and completion without any extra steps. It's simple, it's structured, and it fits right into the systems you already have.

Managing security training across diverse teams is challenging, especially in large organizations. To give you complete control and visibility over your organization's security training, AppSecEngineer's Admin Panel offers a solution to this complexity

These steps will help you create a sustainable security-first culture that's fully integrated into your development process while reducing vulnerabilities and keeping your organization secure.

Shift-Left Security = Staying Competitive and Secure

Nowadays, it's either you're prioritizing security or you're waiting for a data breach to ruin everything you and your team built. Ask yourself, which one are you?

Security in every phase of the SDLC reduces vulnerabilities, accelerates time-to-market, and protects your organization's reputation from very expensive breaches. For organizations, the best time to prioritize early-stage security integration is now. The risks of waiting until the final stages are too high, both in terms of cost and the potential for damage.

AppSecEngineer is here to support your journey. With comprehensive training programs, customizable learning paths, and tools like the Admin Panel to monitor and manage progress, your teams will have the knowledge and skills they need to effectively implement Shift-Left Security. Equip your teams, secure your software, and lead with confidence in your industry.