Is Your Development Team Ready to Meet New PCI-DSS Requirements?

Six steps to meet compliance and shift your culture

With the implementation of PCI DSS version 4.0, organizations face stricter guidelines concerning developing and maintaining secure systems and applications (Requirement 6). This paper examines the implications of PCI DSS 4.0 for development teams and outlines six actionable steps for organizations to ensure compliance.

As the **latest iteration of PCI DSS** comes into force over the next year, financial services, retailers, and select software vendors will be expected to embrace more secure practices to better protect payment card information - and for good reason. In 2023, the Federal Trade Commission received 1.036 million reports of identity theft, with 416,582 involving credit cards, highlighting the high prevalence of payment card data theft and its devastating impact on people's lives.[*]

While the mandatory elements of v4.0 came into force in March of this year, organizations have until March 31st, 2025, to adopt the changes marked as '**best practices**,' including much of requirement 6. Once that data has passed, organizations will not see a grace period. The best time to start is now.

These new PCI DSS requirements are a response to the current state of secure coding knowledge. According to recent research only **36% of developers** learning to write secure code on the job, and only 21% of organizations teaching vulnerability remediation. This stark reality underscores the need for immediate action, as we live in a world where **4 in 5 organizations** are unable to detect vulnerabilities in software during development.

So, what can you do to ensure your teams are on the right path before the 2025 deadline to implement PCI DSS v4.0 approaches?

1. Define Critical Roles

To effectively build an education program to meet compliance regulations, you'll need to understand who your learners are, what development projects they work on, and how their daily work affects PCI DSS compliance.

Start by conducting an internal audit of all roles within the software development lifecycle, or SDLC, to clearly define roles and responsibilities. Doing this first will ensure that your efforts are as effective as possible, creating a strong knowledge base within your organization without wasting your learners' valuable time.

Role-based training, as prescribed by v.4.0, ensures that education programs align with the specific needs of developers and the organization. Not every employee within the SDLC needs the same skills, so a role-based training program can effectively empower all employees in their respective positions.

[*] https://www.cpomagazine.com/cyber-security/a-third-party-breach-leaked-american-express-customers-credit-card-information/

2. Evaluate Your Knowledge Gap

Now that you have identified the roles that would benefit from secure coding education, you can assess the most relevant training.

> **PCI DSS v4.0, Requirement 6.2.2 mandates that:**
>
> "Software development personnel remain knowledgeable about secure development practices, software security, and attacks against the languages, frameworks, or applications they develop. Personnel are able to access assistance and guidance when required."

Where does the team require additional support? This step asks you to work with your SDLC to identify key languages, frameworks, applications, technologies, and the most significant threats you face. You can use exercises such as **Threat Modeling** to prioritize learning content.

Ensure that your secure coding strategy addresses your organization's most critical needs—don't rely on a one-size-fits-all approach such as OWASP Top 10 training for everyone.

3. Bolster Code Review Processes

> **PCI DSS v4.0, Regulation 6.2.3 states:**
>
> "Having code reviewed by someone other than the original author, who is both experienced in code reviews and knowledgeable about secure coding practices, minimizes the possibility that code containing security or logic errors that could affect the security of cardholder data is released into a production environment. Requiring management approval for the code to be reviewed limits the ability for the process to be bypassed."

Although most developers and organizations have some form of code review, these are often hyper-focused on functionality and delivering products to market as fast as possible. Look to implement security-focused code review checklists throughout the SLDC, ensuring that you cover the following bases:

- Ensure that code is developed according to secure coding guidelines
- Look for both existing and emerging vulnerabilities
- Ensure that appropriate corrections are implemented before the release

These checklists should be seen as a standard procedure for security best practices and developed specifically to fit industry and compliance needs.

But how do you ensure your code reviews are effective and that your reviewers can detect existing and emerging vulnerabilities? The answer is to train your employees tasked with code reviews continually. The **key to effective code reviews** is understanding that the review is only as good as the reviewer.

While AI tools can be useful in certain tasks, it's important to note that they lack human context. Although these tools can be highly efficient in evaluating small sections of code, they often struggle to assess how that code fits into a larger program.

4. Identify the Threats

The phrase "know thy enemy" will always be relevant in any security program. PCI DSS v4.0, Regulation 6.6 states:

"Ensure all public-facing web applications are protected against known attacks, either by performing application vulnerability assessment at least annually and after any changes or by installing an automated technical solution that detects and prevents web-based attacks (for example, a web-application firewall) in front of public-facing web applications, to check all traffic continually."

Resources like the **OWASP Top 10** can be helpful guides to the most common vulnerabilities, but they only offer a generalized list. Security teams and developers must remain up to date on language—and application-specific vulnerabilities and threats to ensure that their products are properly protected.

Combining regular threat assessments with expertly crafted, **language-specific** and hands-on developer training will ensure that all parties remain knowledgeable enough to keep security efforts effective.

5. Build A Habit

Continuous programs are the essence of the new regulation, with requirement v4 6.2.2 explicitly stating that developers must be trained at least every 12 months through tailored programs that meet the organization's specific needs. Moreover, these programs will need **continuous assessment**, to stay on top of emerging threats and address new priorities.

Proactive organizations should consider training year-round to stay at the forefront of secure application development. This will not only help developers and security teams feel prepared for new threats, but it will also improve retention and make security second nature.

6. Focus On Retention

The goal of every training program should be to deliver long-lasting knowledge that can support learners throughout their careers. With this, programs can save time and resources. Put simply, retention is the key to building more secure software and proving compliance.

Here are a few tips that can help you create a training program that sticks:

Tip 1 - Customized Training Programs

Our research found that over 50% of organizations say training is not customized to their needs.

Role-based training, as prescribed by v.4.0, ensures that education programs align with the specific needs of developers and the organization. Not every employee within the SDLC needs the same skills, so a role-based training program can effectively empower all employees in their respective positions.

Tip 2 - Deliver Small Training Lessons

Today, 61% of organizations **do not deliver training** in small consumable units, known as microlearning.

Knowledge retention is highest when courses are presented in bite-sized, delivered regularly throughout the year. A 'little and often' approach, in which developers are presented with short, frequent bursts of security training, is perfect for busy schedules and knowledge reinforcement.

Tip 3 - Measure Knowledge Gain

68% of lessons lack immediate and specific feedback, and 50% have no form of assessment to measure knowledge gain.

Immediate feedback and progress tracking can go a long way in keeping developers engaged with secure coding programs while giving training program leaders valuable insights into where their teams are succeeding and where they need more support.

Moving Beyond Compliance

The PCI DSS has mandated secure coding education, indicating a shift in the software industry. The White House, the SEC, and PCI DSS advocate for better software security standards. With high-profile breaches constantly hitting the news, improving software security has become a top priority.

With less than a year left until the deadline, organizations may face consequences for implementing poor or last-minute security measures too close to the deadline. However, a lot can be accomplished in 12 months. Implementing version 4.0 is an opportunity to proactively enhance the security of all future products, upskill your team, and build a true security culture.

Partnering with secure coding training experts will make the transition to proactive compliance even easier, ensuring maximum retention and ROI for your continuous secure coding training program. You can try Security Journey's secure coding training lesson now to learn more:
https://info.securityjourney.com/security-journey-try-our-training-access

SECURITY JOURNEY