

**Web Application Security Testing for PCI DSS**

**Course Number:** SEC-134  
**Duration:** 3 days

**Overview**

[PCI DSS](https://www.pcisecuritystandards.org/) is a mandatory security standard for all organizations working with systems that handle credit cards. To prevent any breach, developers must follow secure coding guidelines and apply the latest best practices.

This Web Application Security Testing for PCI DSS training course teaches attendees the common web application security issues, including those outlined in the OWASP Top Ten. Students learn how to identify core programming issues and security pitfalls of the Java language and framework.  In addition, participants learn about JSON testing and security testing techniques and tools. References to A1, A2, etc. in the outline reference specific vulnerabilities in the [OWASP Top 10](https://owasp.org/www-project-top-ten/).

To ensure ample one-on-one engagement with the instructor, this class is capped at 12 people, overriding Accelebrate’s default cap of 15**.**

**Note:** This 3-day course can be split into 1 day + 2 days. Day 1 is a broad overview intended for managers and staff.  Days 2 and 3 cover the technical aspects of implementing secure code and are intended for developers.  PCI DSS requirements mandate the *annual* training of staff. Your organization may take Day 1 during the first year, and then days 2 and 3 the next year, fulfilling the formal compliance requirement for *both years* with a single course.

**Prerequisites**

This intended audience for this security course is managers and developers working on Web applications in finance. Students must have general web application development and testing experience.

**Materials**

All Secure Coding training attendees receive comprehensive courseware.

**Software Needed on Each Student PC**

Attendees will not need to install any software on their computer for this class. The class will be conducted in a remote environment that Accelebrate will provide; students will only need a local computer with a web browser and a stable Internet connection. Any recent version of Microsoft Edge, Mozilla Firefox, or Google Chrome will be fine.

**Objectives**

* Understand essential cybersecurity concepts
* Gain an understanding of PCI DSS requirements
* Understand Web application security issues
* Analyze the OWASP Top Ten elements
* Put Web application security in the context of any programming language
* Go beyond the low hanging fruits
* Understand security testing methodology and approaches
* Be familiar with common security testing techniques and tools

**Outline**

* Introduction: Day 1
* Cyber Security Basics
  + What is security?
  + Threat and risk
  + Cyber security threat types
  + Consequences of insecure software
    - Constraints and the market
    - The dark side
  + Categorization of bugs
    - The Seven Pernicious Kingdoms
  + Cyber security in the finance sector
    - Threats and trends in fintech
  + PCI DSS
    - Overview
    - Requirements and secure coding (Requirements 1-5)
    - Requirement 6: Develop and maintain secure systems and applications
    - Requirement 6.5: Address common coding vulnerabilities
    - Requirements and secure coding (Requirements 7-12)
* OWASP Overview
  + A2: Broken Authentication
    - Authentication basics
    - Authentication weaknesses
    - Case study – Equifax Argentina
    - Spoofing on the Web
    - Case study – PayPal 2FA bypass
    - User interface best practices
    - Case study – Information disclosure in Simple Banking for Android
    - Password management
  + A3: Sensitive Data Exposure
    - Information exposure
    - Exposure through extracted data and aggregation
    - Case study – Strava data exposure
    - System information leakage
    - Information exposure best practices
  + A9: Using Components with Known Vulnerabilities
    - Using vulnerable components
    - Assessing the environment
    - Hardening
    - Untrusted functionality import
    - Importing JavaScript
    - Case study – The British Airways data breach
    - Case study – The Equifax data breach
    - Vulnerability management
  + A10: Insufficient Logging and Monitoring
    - Logging and monitoring principles
    - Insufficient logging
    - Plaintext passwords at Facebook
    - Logging best practices
    - Monitoring best practices
* Introduction: Days 2 and 3
* OWASP Top 10 for Developers
  + A1: Injection
    - Injection principles
    - Injection attacks
    - SQL injection
    - SQL injection best practices
    - Parameter manipulation
    - CRLF injection
    - Code injection
    - Injection best practices
  + A2: Broken Authentication
    - Password management
    - Session management
  + A4: XML External Entities (XXE)
    - DTD and the entities
    - Entity expansion
    - Attribute blowup
    - External Entity Attack (XXE)
  + Denial of service
    - Denial of Service
    - Resource exhaustion
    - Cash overflow
    - Flooding
    - Sustained client engagement
    - Infinite loop
    - Algorithm complexity issues
    - Regular expression denial of service (ReDoS)
    - Hashtable collision
  + A5: Broken Access Control
    - Access control basics
    - Failure to restrict URL access
    - Confused deputy
    - File upload
  + A6: Security Misconfiguration
    - Configuration principles
    - Configuration management
    - Server misconfiguration
  + A7: Cross-site Scripting (XSS)
    - Cross-site scripting basics
    - Cross-site scripting types
    - XSS protection best practices
  + A8: Insecure Deserialization
    - Serialization and deserialization challenges
    - Deserializing untrusted streams
    - Deserialization best practices
    - Creating a POP payload
    - Using the POP payload
* Web Application Security Beyond the Top Ten
  + Client-side security
  + Same Origin Policy
    - Relaxing the Same Origin Policy
    - Relaxing with Cross-Origin Resource Sharing (CORS)
    - Simple request
    - Preflight request
    - Tabnabbing
  + Frame sandboxing
    - Cross-Frame Scripting (XFS) attack
    - Clickjacking beyond hijacking a click
    - Clickjacking protection best practices
* Some further best practices
  + HTML5 security best practices
  + CSS security best practices
  + Ajax security best practices
* JSON Security
  + JSON injection
  + Dangers of JSONP
  + JSON/JavaScript hijacking
  + Best practices
  + ReactJS vulnerability in HackerOne
* Security Testing
  + Security testing techniques and tools
    - Code analysis
    - Dynamic analysis
* Conclusion
  + Secure coding principles
    - Principles of robust programming by Matt Bishop
    - Secure design principles of Saltzer and Schröder
  + And now what?
    - Software security sources and further reading