Faysal Aziz

Secure Coding Attack Lab

Hands-On Application Security and PCI DSS Workshop - 2 days

FA-0406 · 2 days

Software Development · DevOps, Cloud & Infrastructure · Cybersecurity · Data & Analytics

---

Introduction

Modern applications are assembled from APIs, databases, third-party components, browser code, authentication services and cryptographic libraries. A weakness in any one of these layers can expose payment data or provide a path into connected systems. This intensive workshop concentrates on the practical engineering work required to identify, reproduce and correct the application weaknesses covered by PCI DSS Requirement 6, with technical coverage aligned to current OWASP web and API security risks.

Participants work directly with deliberately vulnerable applications, intercepted requests, source code, API endpoints and security-testing tools. The instructor brings over 30 years of industry experience and delivers the course from a working practitioner's perspective, using realistic development and troubleshooting workflows rather than academic lectures.

Learning outcomes

Participants will be able to:

- Establish a controlled application-security testing environment
- Trace untrusted input through application code and data flows
- Reproduce SQL, LDAP, XPath, command and object-injection weaknesses
- Identify unsafe buffer, pointer, deserialization and shared-data handling
- Detect weak cryptographic implementations and insecure key handling
- Test APIs, communication channels and client-side functionality
- Identify XSS, CSRF and business-logic abuse cases
- Test authentication, authorisation and access-control enforcement
- Apply secure coding corrections to vulnerable components

- Validate fixes through repeatable negative and regression testing
- Perform focused manual secure-code reviews
- Produce actionable vulnerability findings and remediation evidence
- Integrate practical security checks into development and release workflows

Prerequisites

- Basic software-development or application-support experience
- Familiarity with HTTP requests, responses and web applications
- Basic understanding of APIs and database-backed applications
- Ability to read code in at least one common programming language
- Basic command-line skills
- Administrator access to the supplied lab workstation or virtual machine
- A laptop capable of running the designated lab environment
- No previous penetration-testing experience required

Training outline

1. Practical Application-Security Lab Environment

Isolated Testing Architecture

1. Deliberately vulnerable application deployment
2. Local application and database services
3. Test-user and role configuration
4. Application reset and recovery procedures
5. Safe handling of test data

Security Testing Toolchain

1. Browser developer tools
2. Intercepting proxy configuration
3. HTTP request repeater
4. API client configuration
5. Command-line request utilities
6. Source-code editor integration

7. Database inspection tools
8. Static analysis utilities
9. Dependency and secret scanning tools

Application Attack-Surface Mapping

1. Web routes and application functions
2. API endpoints and HTTP methods
3. Parameters, headers, cookies and request bodies
4. File uploads and data-import functions
5. Authentication and session boundaries
6. Role and permission boundaries
7. Database interaction points
8. External service integrations
9. Client-side trust boundaries

2. Practical Secure-Code Review Workflow

Source-Code Orientation

1. Application entry points
2. Controllers and route handlers
3. Service-layer functions
4. Data-access components
5. Authentication middleware
6. Authorisation checks
7. Cryptographic operations
8. Error-handling routines

Untrusted Data Tracing

1. Input sources
2. Data transformations
3. Validation boundaries
4. Encoding boundaries
5. Security-sensitive sinks
6. Cross-component data flow

7. Client-controlled state
8. Server-controlled state

Review Techniques

1. Entry-point analysis
2. Source-to-sink tracing
3. Security-control verification
4. Positive and negative path review
5. Abuse-case review
6. Diff-based security review
7. Tool-assisted code review
8. Manual validation of automated findings

3. Injection Attack Lab

SQL Injection

1. Query parameter manipulation
2. Form-input manipulation
3. Authentication query weaknesses
4. Numeric and string-context injection
5. Error-based behaviour
6. Blind response behaviour
7. Unsafe dynamic queries
8. Parameterised query remediation
9. Database-permission restrictions
10. Remediation verification

Command Injection

1. Operating-system command construction
2. Shell metacharacter handling
3. Argument injection
4. Unsafe process execution
5. Command allow-listing
6. Safe process APIs

7. Privilege reduction
8. Corrective testing

LDAP Injection

1. Directory-search filters
2. Authentication filter manipulation
3. Distinguished-name handling
4. Escaping and validation
5. Safe directory-query construction
6. Fix validation

XPath and XML Injection

1. XPath query construction
2. XML input handling
3. Query predicate manipulation
4. Safe expression handling
5. Parser configuration
6. Remediation testing

Object and Expression Injection

1. Dynamic property access
2. Expression-language evaluation
3. Template injection concepts
4. Reflection and dynamic invocation
5. Unsafe object binding
6. Restrictive binding models
7. Allow-listed property handling

Fault and Error Injection

1. Invalid input types
2. Boundary-value manipulation
3. Missing and malformed parameters
4. Exception-path testing

5. Fail-open behaviour
6. Information leakage through errors
7. Secure error-response handling

4. Data and Data-Structure Attack Lab

Buffer and Boundary Weaknesses

1. Fixed-size data structures
2. Length and bounds validation
3. Integer conversion risks
4. Integer overflow and underflow
5. Truncation and sign errors
6. Safe memory-handling functions
7. Boundary-focused testing

Pointer and Reference Handling

1. Null-reference conditions
2. Stale object references
3. Shared mutable state
4. Unsafe lifecycle assumptions
5. Resource ownership
6. Defensive reference validation

Unsafe Deserialization

1. Untrusted serialized input
2. Type reconstruction
3. Polymorphic object handling
4. Gadget-chain exposure concepts
5. Schema-restricted formats
6. Trusted-type allow-listing
7. Integrity verification
8. Secure replacement patterns

Mass Assignment and Object Binding

1. Automatic request-to-object mapping
2. Hidden and privileged properties
3. Over-posting
4. Data-transfer objects
5. Explicit property mapping
6. Property-level authorisation
7. Negative test cases

Shared Data and Concurrency

1. Race conditions
2. Time-of-check to time-of-use weaknesses
3. Duplicate transaction processing
4. Shared session state
5. Locking and atomicity
6. Idempotency controls
7. Concurrency test design

File and Data Processing

1. File-name manipulation
2. Path traversal
3. Content-type trust
4. File-size and structure validation
5. Archive extraction risks
6. Temporary-file handling
7. Secure storage boundaries
8. Malicious data-import scenarios

5. Cryptography Usage Attack Lab

Cryptographic Implementation Review

1. Approved algorithm selection
2. Deprecated algorithm identification

3. Cipher mode selection
4. Key-length verification
5. Random-number generation
6. Initialization-vector handling
7. Salt and nonce handling
8. Library and provider configuration

Password Protection

1. Plaintext password discovery
2. Reversible password storage
3. Fast-hash weaknesses
4. Password-hashing functions
5. Salt generation
6. Work-factor configuration
7. Credential migration strategies
8. Password-verification testing

Encryption at Rest

1. Sensitive-field identification
2. Key separation
3. Data-encryption boundaries
4. Authenticated encryption
5. Integrity protection
6. Decryption authorisation
7. Secure failure handling

Encryption in Transit

1. TLS configuration review
2. Certificate validation
3. Hostname verification
4. Trust-store handling
5. Insecure protocol fallback
6. Internal service communication
7. API transport protection

Key and Secret Management

1. Hard-coded credentials
2. Embedded API keys
3. Source-repository exposure
4. Configuration-file secrets
5. Environment-based secret injection
6. Central secret storage
7. Key rotation
8. Secret revocation
9. Log and diagnostic leakage
10. Automated secret scanning

Cryptographic Misuse Testing

1. Predictable tokens
2. Reused nonces
3. Static initialization vectors
4. Weak signing implementations
5. Missing integrity checks
6. Improper certificate validation
7. Secure remediation verification

6. Business-Logic Attack Lab

Business-Process Mapping

1. Intended transaction sequence
2. Required approvals
3. State transitions
4. Trust assumptions
5. Monetary and quantity boundaries
6. User and administrator functions
7. Server-side enforcement points

Workflow Manipulation

1. Step skipping
2. Request replay
3. Out-of-order operations
4. Duplicate submissions
5. State-parameter tampering
6. Transaction rollback abuse
7. Partial-completion abuse
8. Server-side state validation

Parameter and Value Manipulation

1. Price manipulation
2. Quantity manipulation
3. Discount and promotion abuse
4. Currency and decimal handling
5. Negative and extreme values
6. Hidden-field manipulation
7. Server-side value calculation
8. Boundary and invariant enforcement

Function and Feature Abuse

1. Undocumented functionality
2. Administrative function exposure
3. Feature-flag manipulation
4. Debug-function exposure
5. Bulk-operation misuse
6. Export-function misuse
7. Account-recovery abuse
8. Rate and quota bypass

Race and Replay Scenarios

1. Concurrent request submission
2. Duplicate payment operations

3. One-time action reuse
4. Token replay
5. Idempotency-key handling
6. Transaction locking
7. Replay-resistant design

7. API Security Attack Lab

API Discovery and Inspection

1. API documentation review
2. Endpoint enumeration
3. HTTP method testing
4. Request-body inspection
5. Authentication-token handling
6. Versioned endpoint discovery
7. Hidden and deprecated endpoints

API Object-Level Authorisation

1. Object-identifier manipulation
2. Horizontal access testing
3. Cross-account data access
4. Record update and deletion testing
5. Server-side ownership validation
6. Deny-by-default access
7. Negative authorisation tests

API Function-Level Authorisation

1. User-to-administrator function access
2. HTTP method switching
3. Restricted route exposure
4. Role-check placement
5. Centralised policy enforcement
6. Privilege regression testing

API Property-Level Authorisation

1. Sensitive property exposure
2. Excessive data response
3. Privileged property modification
4. Request-field allow-listing
5. Response-field filtering
6. Role-specific data models

API Resource Protection

1. Pagination limits
2. Request-size limits
3. Query-complexity limits
4. Rate limiting
5. Batch-request controls
6. Expensive operation protection
7. Timeout and cancellation handling

API Inventory and Lifecycle

1. Endpoint ownership
2. Version inventory
3. Deprecated API handling
4. Test and staging endpoint exposure
5. Documentation consistency
6. Route retirement
7. Security regression coverage

8. Communication Protocol and Channel Security

HTTP Security Behaviour

1. Method restrictions
2. Header trust
3. Proxy-header handling
4. Host-header manipulation

5. Content-type enforcement
6. Redirect handling
7. Caching controls

Service-to-Service Communication

1. Service identity
2. Mutual authentication
3. Certificate trust
4. Token audience and scope
5. Message integrity
6. Replay protection
7. Timeout and retry behaviour

Real-Time and Asynchronous Channels

1. WebSocket authentication
2. Message authorisation
3. Queue and topic permissions
4. Event payload validation
5. Subscription isolation
6. Message replay
7. Dead-letter data exposure

9. Client-Side Security Attack Lab

Browser Trust Boundaries

1. Client-side validation bypass
2. Hidden-field manipulation
3. Local storage exposure
4. Browser token storage
5. JavaScript-accessible secrets
6. Server-side validation requirements

Cross-Site Scripting

1. Reflected XSS
2. Stored XSS
3. DOM-based XSS
4. HTML-context injection
5. Attribute-context injection
6. JavaScript-context injection
7. Unsafe DOM operations
8. Context-aware output encoding
9. HTML sanitisation
10. Content Security Policy
11. Remediation validation

Cross-Site Request Forgery

1. State-changing request discovery
2. Cookie-based authentication
3. Forged form submissions
4. API request forgery
5. Anti-CSRF tokens
6. SameSite cookie controls
7. Origin and referer validation
8. Re-authentication for sensitive actions
9. Fix verification

Client-Side Functional Abuse

1. Disabled-control bypass
2. Front-end role-check bypass
3. JavaScript function invocation
4. Client-calculated values
5. Exposed configuration
6. Source-map exposure
7. Server-side control enforcement

Browser Communication Controls

1. Cross-origin resource sharing
2. Trusted-origin configuration
3. Credentialed cross-origin requests
4. Frame embedding
5. Clickjacking protection
6. Security-header validation

10. Authentication Attack Lab

Credential Handling

1. Hard-coded credentials
2. Default credentials
3. Credential disclosure
4. Username enumeration
5. Password reset weaknesses
6. Credential transport
7. Secure credential storage

Authentication Workflow Testing

1. Login request manipulation
2. Multi-factor workflow bypass
3. Recovery-flow abuse
4. Remember-me functionality
5. Account-lockout behaviour
6. Rate limiting
7. Re-authentication controls

Session Management

1. Session-token predictability
2. Session fixation
3. Token reuse
4. Logout invalidation

5. Idle and absolute timeout
6. Concurrent session handling
7. Cookie security attributes
8. Session rotation
9. Server-side revocation

Token-Based Authentication

1. Token structure review
2. Signature validation
3. Algorithm enforcement
4. Issuer and audience validation
5. Expiration handling
6. Scope enforcement
7. Refresh-token protection
8. Token revocation

11. Access-Control Attack Lab

Access-Control Model Mapping

1. Users, roles and permissions
2. Resources and operations
3. Ownership boundaries
4. Administrative boundaries
5. Tenant boundaries
6. Policy decision points
7. Policy enforcement points

Horizontal Privilege Escalation

1. User-to-user data access
2. Identifier substitution
3. Cross-account actions
4. Resource ownership checks
5. Tenant-isolation testing

Vertical Privilege Escalation

1. User-to-administrator access
2. Restricted function invocation
3. Role-parameter manipulation
4. Privileged route access
5. Server-side permission enforcement

Direct Object Reference Testing

1. Sequential identifiers
2. Predictable identifiers
3. Identifier discovery
4. Read, update and delete operations
5. Indirect reference patterns
6. Per-object authorisation

Enforcement Weaknesses

1. Missing middleware
2. Inconsistent endpoint checks
3. Client-side-only restrictions
4. Fail-open permission checks
5. Cached authorisation decisions
6. Default-allow policies
7. Deny-by-default remediation

Access-Control Regression Testing

1. Role-permission test matrix
2. Positive access tests
3. Negative access tests
4. Cross-role tests
5. Cross-tenant tests
6. Automated authorisation checks

12. Secure Remediation Workshop

Vulnerability Triage

1. Reproduction reliability
2. Attack preconditions
3. Data and function exposure
4. Exploitability
5. Business impact
6. Root-cause classification
7. Remediation priority

Secure Coding Corrections

1. Input validation
2. Parameterised operations
3. Context-aware output encoding
4. Explicit object binding
5. Centralised authorisation
6. Secure cryptographic APIs
7. Safe error handling
8. Least-privilege execution
9. Secure default configuration

Fix Validation

1. Original attack replay
2. Negative testing
3. Boundary testing
4. Alternate payload testing
5. Role-based regression testing
6. API regression testing
7. Static analysis reruns
8. Dependency scan reruns
9. Evidence capture

Disclaimer

This training outline is provided as a proposed instructional framework and may be refined, reordered, expanded or otherwise amended by the trainer to accommodate participant experience, technical constraints, organisational priorities, lab progress and emerging industry practices. Such adjustments may be made at the trainer's professional discretion without prior notice, provided that the overall learning objectives and intended practical character of the programme are preserved.

© 2026 Faysal Aziz · Kuala Lumpur