

Secure Coding Lab for Enterprise and Mobile Applications

Shared defensive patterns across web, mobile and data-processing examples

FA-0687 · 2 days

Cybersecurity · Software Development

Introduction

This two-day workshop examines secure coding across enterprise web, API, mobile and data-processing boundaries. Selected examples use .NET, Node.js, Java, Flutter, Go and SAS to illustrate shared principles; it is not a separate in-depth course for every technology.

Participants reproduce selected weaknesses in supplied lab applications, investigate root causes, apply defensive corrections and retest. PCI DSS Requirement 6 provides secure-development context, not a promise of certification or compliance assessment.

Learning outcomes

- Map web, API, mobile-client and data-processing trust boundaries.
- Reproduce selected injection, object-binding, file-handling and workflow weaknesses.
- Explain secret, password, token and transport-security risks across the supplied examples.
- Evaluate XSS, CSRF, client-side storage and authentication or authorisation controls.
- Apply selected defensive corrections and document negative and regression tests.

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 relevant language, such as C#, JavaScript, Java, Dart, Go or SAS
- 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. Day 1 — Lab and enterprise attack surfaces

1.1 Lab Environment

- Vulnerable application overview
- Local application and database services
- Test users and roles
- Application reset and recovery
- Overview of where .NET, Node.js, Java, Flutter, Golang and SAS commonly appear in enterprise application architectures

1.2 Security Testing Tools

- Browser developer tools
- Intercepting proxy
- API client
- Command-line request tools
- Source-code editor

1.3 Attack-Surface Mapping

- Web routes
- API endpoints
- HTTP methods
- Parameters
- Headers
- Cookies

- Request bodies
- Authentication boundaries
- Role and permission boundaries
- Client-side trust boundaries
- Mobile client-to-API boundaries relevant to Flutter
- Backend service boundaries relevant to .NET, Node.js, Java and Golang
- Data-processing and reporting boundaries relevant to SAS

2. Day 1 – Injection and input handling

2.1 SQL Injection

- Query parameter manipulation
- Form-input manipulation
- Authentication query weaknesses
- Unsafe dynamic queries
- Parameterised query remediation
- Remediation verification
- Examples relevant to .NET, Node.js, Java, Golang and SAS

2.2 Command Injection

- Unsafe operating-system command construction
- Shell metacharacter handling
- Argument injection
- Safe process execution
- Command allow-listing
- Corrective testing
- Examples relevant to .NET, Node.js, Java and Golang

2.3 LDAP and XPath Injection

- LDAP injection overview
- XPath injection overview
- Template injection overview
- Unsafe query construction
- Input escaping

- Safe query handling

2.4 Secure Coding Controls

- Input validation
- Parameterised operations
- Allow-listing
- Least-privilege database access
- Negative testing
- Regression testing

3. Day 1 – Data, object binding and files

3.1 Unsafe Input and Data Handling

- Missing validation
- Malformed input
- Boundary-value manipulation
- Invalid data types
- Truncation risks
- Secure error handling

3.2 Mass Assignment and Object Binding

- Automatic request-to-object mapping
- Hidden properties
- Privileged properties
- Over-posting
- Data-transfer objects
- Explicit property mapping
- Property-level authorisation
- Examples relevant to .NET, Node.js, Java and Golang

3.3 Insecure Deserialization

- Untrusted serialized input
- Unsafe object reconstruction

- Polymorphic object handling
- Trusted-type allow-listing
- Schema-restricted formats
- Secure replacement patterns

3.4 File and Data Processing

- File-name manipulation
- Path traversal
- Content-type trust
- File-size validation
- Temporary-file handling
- Secure storage boundaries

4. Day 1 — Business logic and client trust

4.1 Business-Process Mapping

- Intended transaction sequence
- Required approvals
- State transitions
- Trust assumptions
- Server-side enforcement points

4.2 Workflow Manipulation

- Step skipping
- Out-of-order operations
- Request replay
- Duplicate submissions
- Partial-completion abuse

4.3 Parameter and Value Manipulation

- Price manipulation
- Quantity manipulation
- Discount manipulation

- Negative values
- Extreme values
- Hidden-field manipulation
- Server-side value calculation
- Flutter mobile client trust boundaries

5. Day 2 — Passwords, secrets and cryptography

5.1 Password Protection

- Plaintext password storage
- Reversible password storage
- Weak password hashes
- Password-hashing functions
- Salt generation
- Work-factor configuration

5.2 Hard-Coded Credentials and Secrets

- Embedded passwords
- API keys in source code
- Configuration-file secrets
- Log and diagnostic leakage
- Secret scanning
- Secure secret injection

5.3 Encryption in Transit

- TLS configuration
- Certificate validation
- Hostname verification
- Trust-store handling
- Insecure protocol fallback

5.4 Token and Randomness Weaknesses

- Predictable tokens

- Weak session identifiers
- Improper signing
- Missing integrity checks
- Secure token generation

5.5 Cryptographic remediation

- Algorithms and parameters appropriate to the use case and applicable policy
- Secure cryptographic libraries
- Key separation
- Secret rotation
- Secure failure handling
- Remediation verification

6. Day 2— XSS, CSRF and client-side storage

6.1 Cross-Site Scripting

- Reflected XSS
- Stored XSS
- DOM-based XSS
- HTML-context injection
- Attribute-context injection
- Unsafe DOM operations
- Context-aware output encoding
- HTML sanitisation
- Content Security Policy as defence in depth, alongside encoding and sanitisation

6.2 Cross-Site Request Forgery

- State-changing request discovery
- Cookie-based authentication
- Forged form submissions
- Anti-CSRF tokens
- SameSite cookies as additional protection; select CSRF controls for the authentication model
- Origin validation
- Referer validation

- Re-authentication for sensitive actions

6.3 Client-Side Security

- Local storage exposure
- Browser token storage
- Source-map exposure
- Client-side secret exposure
- Flutter mobile local storage and token-handling risks
- Server-side validation requirements

7. Day 2 — Authentication and access control

7.1 Identification, Authentication and Authorisation

- User identification
- Credential validation
- Role verification
- Permission enforcement
- Policy decision points
- Policy enforcement points

7.2 Authentication Weaknesses

- Default credentials
- Username enumeration
- Password-reset weaknesses
- Multi-factor bypass
- Account-lockout weaknesses
- Missing rate limiting

7.3 Session and Token Weaknesses

- Session fixation
- Session-token predictability
- Token reuse
- Missing logout invalidation

- Missing session rotation
- Weak cookie attributes
- Improper token validation

7.4 Horizontal Privilege Escalation

- User-to-user data access
- Identifier substitution
- Cross-account actions
- Resource ownership checks
- Tenant-isolation testing

7.5 Vertical Privilege Escalation

- User-to-administrator access
- Restricted function invocation
- Role-parameter manipulation
- Privileged route access
- Server-side permission enforcement

7.6 IDOR and BOLA

- Sequential identifiers and per-object authorisation
- Predictable or opaque identifiers: neither replaces authorisation
- Object-identifier manipulation
- Read access testing
- Update access testing
- Delete access testing
- Per-object authorisation

7.7 Access-Control Remediation

- Centralised authorisation
- Deny-by-default policies
- Consistent middleware enforcement
- Role-permission matrices
- Negative authorisation testing

- Privilege regression testing

8. Day 2 — Remediation and retesting

8.1 Vulnerability Triage

- Reproduction reliability
- Attack preconditions
- Exploitability
- Data exposure
- Business impact
- Root-cause classification

8.2 Secure Coding Corrections

- Input validation
- Parameterised operations
- Context-aware output encoding
- Explicit object binding
- Centralised authorisation
- Secure cryptographic APIs
- Safe error handling
- Least-privilege execution

8.3 Fix Validation

- Original attack replay
- Negative testing
- Boundary testing
- Alternate payload testing
- Role-based regression testing
- API regression testing