CONCEPT

OWASP Top 10 and Secure Coding: Threat Modelling, Code Review and DevSecOps

8 – 12 August 2027
Jeddah - Corniche four-star

OWASP Top 10 and Secure Coding: Threat Modelling, Code Review and DevSecOps

Ref: 348_15675 **Date:** 8 – 12 August 2027 **Location:** Jeddah - Corniche four-star **Fees:** 19500 SAR

Introduction:

Most breaches of web applications trace back to flaws written into the code or the build: missing server-side authorisation, string-built queries, weak session handling, outdated libraries and pipelines that ship whatever compiles. The OWASP Top 10 names these recurring risks, but teams still need habits and gates that stop them reaching production. This Core Concept course trains developers and AppSec leads to prevent each OWASP Top 10 risk in code, threat model features with STRIDE, review pull requests for security and run SAST, DAST, SCA and secrets scanning in the pipeline. Participants build a Web Application Secure Coding Playbook.

Course Objectives:

- Map the OWASP Top 10 risk categories to the entry points, trust boundaries and data flows of a web application the team maintains
- Build a threat model for a new feature using the OWASP four-question method and STRIDE, and turn it into security requirements
- Apply secure coding patterns that prevent broken access control, injection, cryptographic failures and authentication failures in application code
- Configure pipeline security gates for SAST, DAST, SCA and secrets scanning, with SBOM generation and dependency triage rules
- Conduct a security code review of a pull request and rate and route findings for remediation
- Produce a Web Application Secure Coding Playbook aligned to the NIST SSDF practice groups

Target Audience:

- Developers who write and maintain server-side and front-end code for web applications
- Technical leads who approve pull requests and set coding standards for a development team
- Application security staff who define secure development requirements and triage scanner findings
- DevOps and build engineers who own CI/CD pipelines, artefact repositories and release gates
- QA and test engineers who add security test cases to functional and regression suites

Course Outline:

Day 1: Web Application Risk Landscape, the OWASP Top 10 and the Secure SDLC Baseline

- OWASP Top 10 Purpose, Risk Categories and Use as a Developer Awareness Standard
- Web Application Attack Surface Map: Entry Points, Trust Boundaries and Data Flows
- Secure Software Development Lifecycle Stages and Security Activity Mapping
- NIST SSDF Practice Groups: Prepare, Protect, Produce and Respond
- Current-State Secure Development Assessment Using OWASP SAMM Practices

Day 2: Threat Modelling, Security Requirements and Secure Design

- OWASP Four-Question Threat Modelling Method Applied to a Web Feature
- STRIDE Threat Categories on a Data Flow Diagram with Trust Boundaries
- Insecure Design Prevention: Abuse Cases, Secure Design Patterns and Design Review Gates
- OWASP ASVS Verification Levels as a Source of Testable Security Requirements
- Lab: Threat Model and Mitigation Register for a Sample Web Application

Day 3: Secure Coding Patterns for Access Control, Injection, Cryptography and Authentication

- Broken Access Control Prevention: Deny by Default, Server-Side Authorisation and SSRF Allow-Lists
- Injection Prevention: Parameterised Queries, Contextual Output Encoding and Allow-List Input Validation
- Cryptographic Failures Prevention: Data Classification, TLS Configuration, Key Handling and Password Hashing
- Authentication Failures Prevention: Multi-Factor Authentication, Session Lifecycle and Credential Storage
- Lab: Refactoring Flawed Code Samples into Secure Equivalents with Unit Tests

Day 4: Supply Chain, Configuration, Integrity, Logging and the DevSecOps Pipeline

- Security Misconfiguration Controls: Hardened Framework Defaults, Security Headers and Environment Parity
- Software Supply Chain Failures: Dependency Inventory, SBOM Generation and Vulnerable Component Triage
- Software or Data Integrity Failures: Signed Artefacts, Protected Build Pipelines and Safe Deserialisation
- Security Logging and Alerting Failures and Mishandling of Exceptional Conditions: Audit Events and Fail-Safe Error Handling
- Lab: Pipeline Gates for SAST, DAST, SCA and Secrets Scanning with Break-the-Build Rules

Day 5: Security Code Review, Defensive Testing and the Secure Coding Playbook

- Security Code Review Checklist Mapped to the OWASP Top 10 Categories
- Lab: Peer Security Review of a Pull Request with Severity-Rated Findings
- Defensive Security Testing Basics: Security Unit Tests, ASVS-Based Test Cases and Baseline DAST Scans
- Remediation Prioritisation, Fix Verification and Vulnerability Handling Workflow for Development Teams
- Web Application Secure Coding Playbook Build and Peer Review

Skills You Will Gain:

- Secure Coding Pattern Selection
- STRIDE Threat Modelling
- Security Requirements Definition
- Security Code Review
- Pipeline Security Gate Configuration
- Dependency and SBOM Management
- Scanner Finding Triage
- Secure Error Handling and Logging

Why Attend This Course:

- Return with a Web Application Secure Coding Playbook containing a team coding standard, threat model template, pipeline gate settings and review checklist
- Cut rework found late in penetration tests by catching access control, injection and dependency flaws at pull request and build time
- Practise refactoring, reviewing and scanning real code samples hands on in lab exercises rather than reading slides about risks
- Compare secure development practices with developers and AppSec leads from finance, government services, retail and technology teams

Conclusion:

Secure web applications come from repeatable engineering habits, not from a single test before go-live. The week moves from the OWASP Top 10 risk categories, the attack surface and the secure SDLC, through threat modelling with STRIDE and security requirements, to coding patterns for access control, injection, cryptography and authentication. It then covers configuration, supply chain, integrity, logging and pipeline scanning, and ends with security code review, defensive testing and a Web Application Secure Coding Playbook for each team.