Specialized - Secure .Net Web Application Development Lifecycle (SDL)

Code:	TT8325-N
Length:	5 days
URL:	View Online

Secure .Net Web Application Development Lifecycle (SDL) is a lab-intensive, hands-on .Net security training course, essential for experienced enterprise developers who need to produce secure .Net-based web applications. In addition to teaching basic programming skills, this course digs deep into sound processes and practices that apply to the entire software development lifecycle. In this course, students thoroughly examine best practices for defensively coding .Net web applications, including XML processing and web services. Students will repeatedly attack and then defend various assets associated with a fully-functional web application. This hands-on approach drives home the mechanics of how to secure .Net web applications in the most practical of terms. Security experts agree that the least effective approach to security is "penetrate and patch". It is far more effective to "bake" security into an application throughout its lifecycle. After spending significant time trying to defend a poorly designed (from a security perspective) web

Skills Gained

- Understand potential sources for untrusted data
- Understand the consequences for not properly handling untrusted data such as denial of service, cross-site scripting, and injections
- Be able to test web applications with various attack techniques to determine the existence of and effectiveness of layered defenses
- Prevent and defend the many potential vulnerabilities associated with untrusted data
- Understand the vulnerabilities of associated with authentication and authorization
- Be able to detect, attack, and implement defenses for authentication and authorization functionality and services
- Understand the dangers and mechanisms behind Cross-Site Scripting (XSS) and Injection attacks
- Be able to detect, attack, and implement defenses against XSS and Injection attacks
- Understand the concepts and terminology behind defensive, secure, coding
- Understand the use of Threat Modeling as a tool in identifying software vulnerabilities based on realistic threats against meaningful assets
- Perform both static code reviews and dynamic application testing to uncover vulnerabilities in web applications
- Design and develop strong, robust authentication and authorization implementations within the context of .Net
- Be able to detect, attack, and implement defenses for XML-based services and functionality
- Understand techniques and measures that can used to harden web and application servers as well as other components in your infrastructure
- Understand and implement the processes and measures associated with the Secure Software Development (SSD)
- Acquire the skills, tools, and best practices for design and code reviews as well as testing initiatives
- Understand the basics of security testing and planning
- Work through a comprehensive testing plan for recognized vulnerabilities and weaknesses

Who Can Benefit
.Net Developers

Prerequisites
This is an intermediate -level .Net secure programming course, designed for developers who wish to get up and running on developing well defended software applications. Familiarity with C# is required and real world programming experience is highly recommended. Ideally students should have approximately 6 months to a year of .Net development practical experience.

Course Details

Introduction: Misconceptions
- Security: The Complete Picture
- Seven Deadly Assumptions
- Anthem, Sony, Target, Heartland, and TJX Debriefs
- Causes of Data Breaches
- Meaning of Being Compliant
- Verizon's 2015 Data Breach Report
- 2015 PCI Compliance Report

Session: Foundation

Lesson: Security Concepts
- Motivations: Costs and Standards
- Open Web Application Security Project
- Web Application Security Consortium
- CERT Secure Coding Standards
- Assets are the Targets
- Security Activities Cost Resources
- Threat Modeling
- System/Trust Boundaries

Lesson: Principles of Information Security
- Security Is a Lifecycle Issue
- Minimize Attack Surface Area
- Layers of Defense: Tenacious D
- Compartmentalize
- Consider All Application States
- Do NOT Trust the Untrusted

Session: Vulnerabilities

Lesson: Unvalidated Input
- Buffer Overflows

- Integer Arithmetic Vulnerabilities
- Unvalidated Input: From the Web
- Defending Trust Boundaries
- Whitelisting vs Blacklisting

Lesson: Overview of Regular Expressions

- Regular Expressions
- Working With Regexes in .Net
- Applying Regular Expressions

Lesson: Broken Access Control

- Access Control Issues
- Excessive Privileges
- Insufficient Flow Control
- Unprotected URL/Resource Access
- Examples of Shabby Access Control
- Session and Session Management

Lesson: Broken Authentication

- Broken Quality/DoS
- Authentication Data
- Username/Password Protection
- Exploits Magnify Importance
- Handling Passwords on Server Side
- Single Sign-on (SSO)

Lesson: Cross Site Scripting (XSS)

- Persistent XSS
- Reflective XSS
- Best Practices for Untrusted Data

Lesson: Injection

- Injection Flaws
- SQL Injection Attacks Evolve
- Drill Down on Stored Procedures
- Other Forms of Injection
- Minimizing Injection Flaws

Lesson: Error Handling and Information Leakage

- Fingerprinting a Web Site
- Error-Handling Issues
- Logging In Support of Forensics

- Solving DLP Challenges

Lesson: Insecure Data Handling

- Protecting Data Can Mitigate Impact
- In-Memory Data Handling
- Secure Pipes
- Failures in the SSL Framework Are Appearing

Lesson: Insecure Configuration Management

- System Hardening: IA Mitigation
- Application Whitelisting
- Least Privileges
- Anti-Exploitation
- Secure Baseline

Lesson: Direct Object Access

- Dynamic Loading
- Direct Object References

Lesson: Spoofing, CSRF, and Redirects

- Name Resolution Vulnerabilities
- Fake Certs and Mobile Apps
- Targeted Spoofing Attacks
- Cross Site Request Forgeries (CSRF)
- CSRF Defenses are Entirely Server-Side
- Safe Redirects and Forwards

Session: Best Practices

Lesson: .NET Issues and Best Practices

- Manage Code and Buffer Overflows
- .Net Permissions
- ActiveX Controls
- Proper Exception Handling

Lesson: Understanding What's Important

- Common Vulnerabilities and Exposures
- OWASP Top Ten for 2013
- CWE/SANS Top 25 Most Dangerous SW Errors
- Monster Mitigations
- Strength Training: Project Teams/Developers
- Strength Training: IT Organizations

Session: Defending XML, Services, and Rich Interfaces

Lesson: Defending XML

- XML Signature
- XML Encryption
- XML Attacks: Structure
- XML Attacks: Injection
- Safe XML Processing

Lesson: Defending Web Services

- Web Service Security Exposures
- When Transport-Level Alone is NOT Enough
- Message-Level Security
- WS-Security Roadmap
- Web Service Attacks
- Web Service Appliance/Gateways

Lesson: Defending Rich Interfaces and REST

- How Attackers See Rich Interfaces
- Attack Surface Changes When Moving to Rich Interfaces
- Bridging and its Potential Problems
- Three Basic Tenets for Safe Rich Interfaces
- OWASP REST Security Recommendations

Session: Cryptography

Lesson: Cryptography Overview

- Strong Encryption
- Message digests
- Keys and key management
- Certificate management
- Encryption/Decryption

Lesson: .NET Cryptographic Services

- The role of cryptographic services
- Hash algorithms and hash codes
- Encrypting data symmetrically
- Encrypting data asymmetrically

Session: Secure Development Lifecycle (SDL)

Lesson: SDL Process Overview

- Software Security Axioms
- Security Lifecycle – Phases

Lesson: Applying Processes and Practices

- Awareness
- Application Assessments
- Security Requirements
- Secure Development Practices
- Security Architecture/Design Review
- Security Code Review
- Configuration Management and Deployment
- Vulnerability Remediation Procedures

Lesson: Risk Analysis

- Threat Modeling Process
- 1. Identify Security Objectives
- 2. Describe the System
- 3. List Assets
- 4. Define System/Trust Boundaries
- 5. List and Rank Threats
- 6. List Defenses and Countermeasures

Session: Security Testing

Lesson: Testing Tools and Processes

- Security Testing Principles
- Black Box Analyzers
- Static Code Analyzers
- Criteria for Selecting Static Analyzers

Lesson: Testing Practices

- OWASP Web App Penetration Testing
- Authentication Testing
- Session Management Testing
- Data Validation Testing
- Denial of Service Testing
- Web Services Testing
- Ajax Testing
