Who's watching your back?

Foundstone®
Professional Services A DIVISION OF McAFEE

Training Datasheet

Writing Secure Code: ASP.NET (C#)

2-Day Special Edition Course

DURATION
- Two (2) Days

WHAT YOU'LL LEARN
- The process and techniques of writing secure code
- Effective authentication and authorization techniques
- The most frequent web application vulnerabilities and how to avoid them
- Data validation strategies
- Effective error handling and exceptions management
- Software security testing techniques

COURSE MATERIALS
- Student manual
- Class handouts
- Foundstone t-shirt
- Free Tools CD with course tools and scripts
- "FsLive" a customized bootable Knoppix distribution CD

SUGGESTED NEXT COURSE(S)
- Writing Secure Code – Java (J2EE)
- Building Secure Code

Understand the key security features of the .NET platform, the common web security pitfalls developers make, and how to build secure and reliable web applications using ASP.NET. Students are lead through hands on code examples that highlight issues and prescribe solutions.

Who Should Take This Class

Software developers or software security auditors who have been working with the .NET framework for at least one year and developing ASP.NET C# code for at least one year. A comprehensive knowledge of the .NET framework, the C# language, and web technology is required.

Exercises

All topics are supported by hands-on exercises specifically designed to increase knowledge retention. Classroom exercises provide the basic hands-on experience needed to write secure code.

Course Outline

Day 1

Introduction
- Overview of course content and format
- Secure Design Principles
- Introduction to Hacme Bank
- *Demonstration* (Hacme Bank Penetration Test): Students observe (and may optionally participate) as the instructor exploits numerous common vulnerabilities in Hacme Bank, an ASP.NET web application that is designed to function as a real-world online bank.

.NET Platform Security
- .NET Language Security Features
- .NET Common Language Runtime (CLR) Security Mechanisms
- Code Access Security (CAS)
- Strong Name Signing
- ASP.NET Security Architecture

Data Protection
- Common Mistakes
- .NET System.Cryptography Namespace
- .NET Algorithm Recommendations
- Data Protection API (DPAPI)
- Advanced Cryptographic Features in .NET
- *Lab* (Data Protection):Encrypt sensitive data with the DPAPI

Authentication
- Common Mistakes
- ASP.NET Authentication
- Impersonation and Delegation
- Code Signing (Authenticode)
- Windows Card Space

User Management
- Common Mistakes
- Password Storage and Quality
- Strategies for Password Reset
- Account Lockout Schemes
- ASP.NET Membership API

Authorization
- Common Mistakes
- .NET Authorization Models
- ASP.NET Authorization
- RoleManager
- Session Management
- Cross-Site Request Forgery
- *Lab* (Authorization): Fix privilege escalation vulnerability in Hacme Bank

Error Handling and Exception Management
- Common Mistakes
- Safe Exception Handling
- Error Handling
- Event Logging and Audit Trails

Day 2

Data Validation
- Common Mistakes
- Data Validation Attacks
- Preventing Data Validation Attacks
- Regular Expressions
- .NET Validation Frameworks & APIs
- Canonicalization Issues
- *Lab* (Data Validation): Implement a variety of validation routines

Configuration Management
- Common Mistakes
- Securing Sensitive Configuration Data
- Securing Communications
- Resource Exhaustion and Other Denial of Service Conditions
- Application Secure hardening
- Secure Build and Deployment
- Partial Trust ASP.NET

Logging and Auditing
- Common Mistakes
- Logging Best Practices
- Data Retention
- Popular Logging Frameworks

Software Security Analysis
- Threat Modeling
- Secure Code Review Methodology
- Automated Code Scanning Tools
- Practical Strategies for Conducting Code Reviews

Q & A
- *Extended Lab* (ASP.NET Architecture Analysis and Secure Code Review): Students perform threat modeling and code review for the Hacme Bank web application to identify flaws and locate defective code, then re-engineer and implement a secure design.