

Who's watching your back?

**Foundstone**<sup>®</sup>  
Professional Services A DIVISION OF McAfee

## Training Datasheet

# Writing Secure Code: ASP.NET (C#)

### DURATION

- Four (4) 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
- Secure user management systems
- Data validation strategies
- Effective error handling and exceptions management
- Web services security and remoting
- Software security testing techniques

### COURSE MATERIALS

- Student manual
- Class handouts
- Foundstone t-shirt
- Free Tools CD with course tools and scripts

### SUGGESTED NEXT COURSE(S)

- Writing Secure Code – Java (J2EE)
- Writing Secure Code – C/C++

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. The class covers both the current version of the .NET framework (v2.0) as well as covers security relevant features in the .NET 3.0 and .NET 3.5 updates.

### Who Should Take This Class

Software developers or software security auditors who have been working with the .NET framework for at least a year and have experience developing ASP.NET web application using C#. 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. Exercises are uniquely intertwined into the lecture material both by way of instructor-led demonstrations as well as hands-on labs. This is intended to keep the students engaged at all times. Classroom exercises provide the 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 Architecture

#### Advanced .NET Security

- Advanced .NET Security Concepts
- Code Access Security
- Partial Trust ASP.NET
- Windows CardSpace

#### Cryptography

- .NET System.Cryptography Namespace
- Common Cryptographic Mistakes
- .NET Algorithm Recommendations
- DPAPI
- Other Cryptographic Features in .NET
- *Lab* (Cryptography): Protecting Data in ASP.NET

## Day 2

### Authentication

- ASP.NET Authentication
- Windows Authentication/Kerberos
- Impersonation and Delegation
- Forms Authentication
- Code Signing (Authenticate)

### User Management

- Password Storage and Quality
- Strategies for Password Reset
- Account Lockout Schemes
- Membership API

### Authorization

- Access Control Models Explained
- Common Authorization Flaws
- ASP.NET/IIS Authorization
- Session Management
- RoleManager
- *Lab* (Authorization): Implementing Authorization in ASP.NET

## Day 3

### Data Validation

- Input and Output Validation
- SQL Injection, Cross-Site Scripting (XSS), and Other Attacks
- Preventing Data Validation Attacks
- Regular Expressions
- Data Validation Controls and Libraries
- Canonicalization Issues

### Client Side Security

- Client Side Security Mistakes
- Security Objectives for Thick Clients
  - Protect the client from the user
  - Protect the user from the client
  - Protect the server from the client
- Reverse Engineering
- Byte Code Manipulation
- Licensing Schemes
- Source Code Protection
- Code Access Security on the Client
- Secure Design Patterns

## Security Testing

- White Box Techniques
  - Threat Modeling
  - Code Review
- Black Box Techniques
  - Functional Testing
  - Penetration Testing
- Unit Testing

## Web Services

- Web Service Risks
- Web Service Attacks
- Web Service Defense Techniques
- Survey of Security Technologies
- Windows Communication Foundation
- Web Services Security Patterns
- *Lab* (Data Validation): Performing Efficient Data Validation in ASP.NET

## Day 4

### Error Handling and Exception Management

- Exception Handling Patterns and Anti-patterns
- ASP.NET Exception Frameworks
- Best Practices for Handling User Errors

### Logging and Auditing

- Common Mistakes with Logging
- Logging Best Practices

### Secure Code Review

- Threat Modeling
- Secure Code Review Methodology
- Manual Code Review
- Automated Code Scanning Tools
- Practical Strategies for Conducting Code Reviews
- *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. Primary vulnerabilities include SQL Injection and Cross-Site Scripting.