Security Innovation Software & Information Security eLearning Curriculum

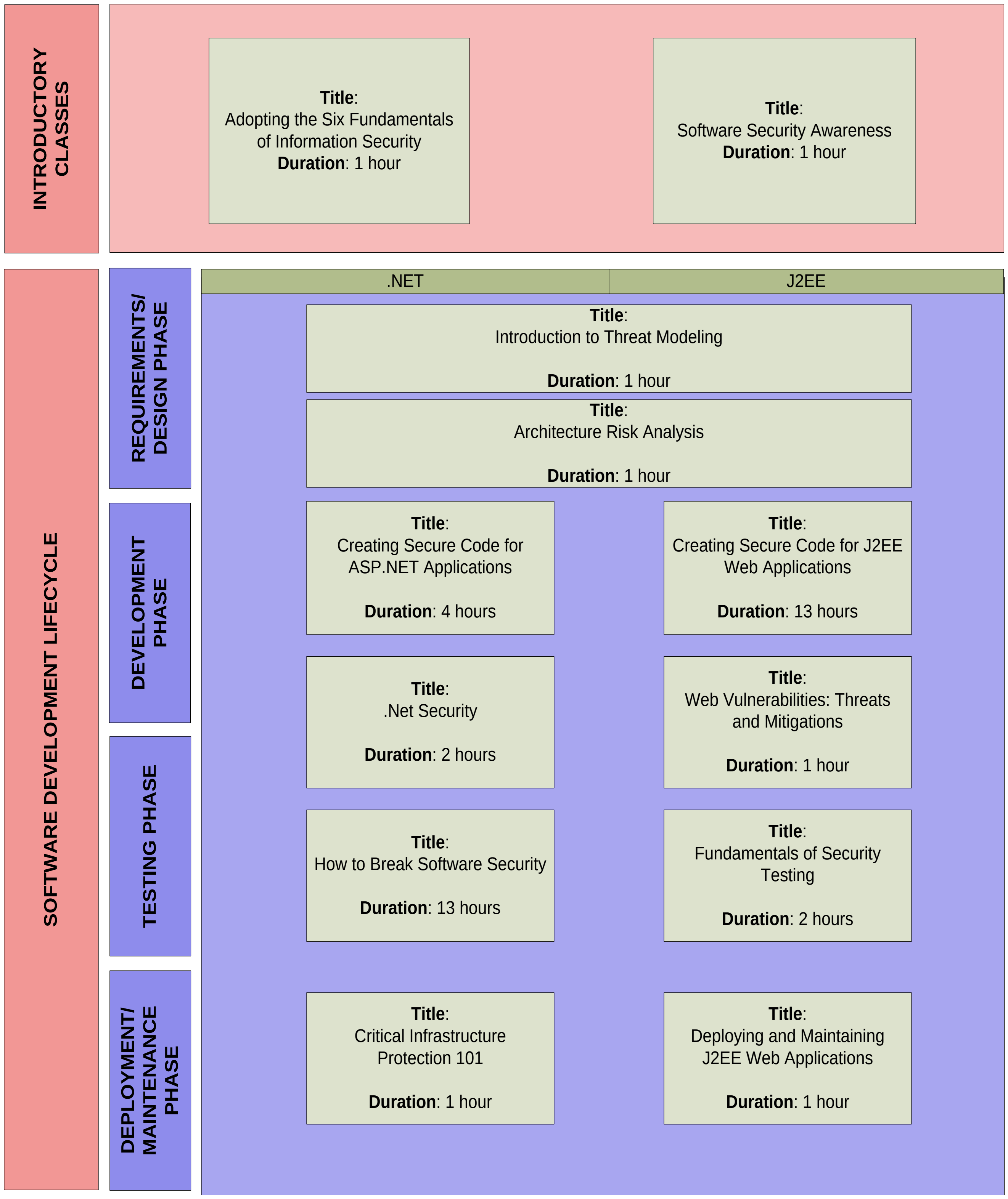
Courses to Help Build and Deploy more Secure Software and Information Systems

SecurityInnovation® BOSTON | SEATTLE www.securityinnovation.com

Security Innovation, Inc. 187 Ballardvale Street Suite A195 Wilmington, MA 01887 • Ph: 1.978.694.1008

Table of Contents

1. Sample eLearning Course Track by Technology



INTRODUCTORY CLASSES

Title:
Adopting the Six Fundamentals of Information Security
Duration: 1 hour

Title:
Software Security Awareness
Duration: 1 hour

SOFTWARE DEVELOPMENT LIFECYCLE

.NET | J2EE

REQUIREMENTS/ DESIGN PHASE

Title:
Introduction to Threat Modeling

Duration: 1 hour

Title:
Architecture Risk Analysis

Duration: 1 hour

DEVELOPMENT PHASE

Title:
Creating Secure Code for ASP.NET Applications

Duration: 4 hours

Title:
Creating Secure Code for J2EE Web Applications

Duration: 13 hours

Title:
.Net Security

Duration: 2 hours

Title:
Web Vulnerabilities: Threats and Mitigations

Duration: 1 hour

TESTING PHASE

Title:
How to Break Software Security

Duration: 13 hours

Title:
Fundamentals of Security Testing

Duration: 2 hours

DEPLOYMENT/ MAINTENANCE PHASE

Title:
Critical Infrastructure Protection 101

Duration: 1 hour

Title:
Deploying and Maintaining J2EE Web Applications

Duration: 1 hour

2. eLearning Courses – grouped by training level

Awareness Classes

MANAGERS & STAFF

- Software Security Awareness
- Fundamentals of Information Security
- Microsoft SDL for Managers
- Introduction to the Microsoft SDL
- Introduction to Credit Card Security
- Intro to the PCI-DSS
- Introduction to HIPAA

Introductory Classes

ARCHITECTS, DEVELOPERS, TESTERS

- Fundamentals of Application Security
- Fundamentals of Security Testing
- Introduction to the Microsoft SDL

INFORMATION SECURITY

- Critical Infrastructure Protection
- Introduction to Secure Printing

Advanced/Specialized Classes

ARCHITECT

- Introduction to Threat Modeling
- Architecture Risk Analysis

DEVELOPER

- Introduction to Integer Overflows
- .Net Security
- Creating Secure Code - ASP.Net
- Introduction to Buffer Overflows
- Creating Secure Code - Java Web Applications
- Web Vulnerabilities - Threats & Mitigations
- Windows Vista Security
- Defensive Programming C/C+
- Defensive Programming - C#

TESTER

- Exploiting Buffer Overflows
- How to Break Software Security
- Classes of Security Defects

3. Adopting the Six Fundamentals of Information Security

Duration: 1 hour

Audience

All audiences (from marketing managers to testers)

Prerequisite Knowledge/Skills

Basic computer skills

Course Description

This awareness course is intended to allow individuals to recognize information security concerns and respond accordingly using a set of best practices provided in this course.

Upon completion of this course, you will be able to:

- recognize the importance of protecting your organization's information
- follow appropriate security best practices

Course Modules

The need to protect your organization's information

This topic deals with the consequences of security vulnerabilities and provides real-world examples. It also provides basic information security definitions and principles.

The steps to adopt a secure behavior

This module provides the six fundamentals of information security that should be followed to minimize the risk of security breach:

1. Managing user accounts securely
2. Handling your organization's information securely
3. Protecting mobile devices and storage media
4. Preventing intrusions into corporate facilities
5. Defending against malware
6. Working safely away from the office

Assessment

A 10-question multiple-choice exam is taken at the end of the course.

4. Introduction to the PCI DSS

Duration: 1 hour

Audience

Information Technology, Security, Risk and Compliance Managers

Prerequisite Knowledge/Skills

None

Course Description

This course is designed for any individual or group needing to know about the PCI DSS (Payment Card Industry Data Security Standard) and best practices for achieving certification.

Course Objectives

Upon completion of this course, participants will understand:

- why PCI DSS was created and who the main actors are
- the high level requirements of PCI DSS
- controls and compensating controls
- how to build a PCI DSS team
- how to use best practice security to achieve PCI DSS compliance
- the Do's and Don'ts of PCI DSS Compliance

Course Modules

- Introduction to PCI DSS
- PCI Compliance levels
- Structure of PCI DSS
- The 12 Requirements of PCI DSS
- Creating a PCI Project Team

Assessment

Participants will complete a multiple choice exam at the end of course to test the participant's knowledge of PCI. Upon completion of the exam, successful participants will receive a certificate showing that they have passed the course.

5. Introduction to Secure Printing

Duration: 5 hours

Audience

Network, IT and Information System personnel

Prerequisite Knowledge/Skills

General understanding of network operations security

Course Description

Organizations are under constant pressure to increase the confidentiality of print jobs and data in transit, improve the flexibility associated with solutions and reduce overall printing costs. This course helps you understand the threats and legal issues surrounding printing and lays the foundation for a proactive, scalable and future-proof security strategy with secure printing being a key component.

Upon completion of this course, you will be able to:

- describe five key elements of any security strategy
- understand the value of data assets
- understand the legal issues surrounding the protection of corporate assets
- understand the threats to networked printers
- describe best practices in relation to secure printing

Modules Covered

General Security Principles

This module provides an introduction to the concept of security within an organization, which includes the CIA (confidentiality, integrity, availability) triangle and five key elements of sound security.

Secure Printing

This module introduces the concept of the secure printing environment and describes:

- how uncontrolled printing environments can pose a threat to your organization's assets
- how legislation such as the Data Protection Act can impact your organization's security strategy
- how secure printing solutions can add value to your organization's strategy
- how to deal with the security risk associated with printing environments

Security Incidents Related to Printing Environments

This module examines security incidents related to printing environments. You will learn that unsecured printers can be easily compromised by hackers and that there are many free tools available on the Internet that enable printer data to be read.

Secure Printing Network Example

This module looks at practical examples of secure printing in action. You will learn how the secure printing concept may be applied in practice and the benefits of implementing a secure printing environment.

10 Ways to Achieve a Secure Printing Best Practice

This module describes specific measures that you should put in place in order to achieve secure printing best practice.

Assessment

A 10-question multiple-choice exam is taken at the end of the course.

6. Introduction to the HIPAA Standard

Duration: 1 hour

Audience

This course is intended for employees who have responsibility for medical records at relevant organizations, including but not limited to health care providers, health care clearing houses or health plan

Prerequisite Knowledge / Skills

None

Course Description

Upon completion of this course you will be able to:

- demonstrate compliance to HSS if audited
- understand how the HIPAA standard came into existence
- recognize the main industry players involved
- understand the five rules which govern HIPAA, specifically the security rule
- understand the implications of non-compliance

Modules Covered

HIPAA Overview

The Security Rule – Administrative Safeguards

The Security Rule – Physical Safeguards

The Security Rule – Technical Safeguards

Complying with the Security Rules

Assessment

An interactive exam at end of course tests the participant's understanding of the "Administrative Simplification" provisions of HIPAA.

Upon completion of the exam, successful participants will receive a certificate showing that they have passed the course.

7. Introduction to Credit Card Security

Duration: 30 minutes

Audience

This course is intended for employees who deal with credit card transactions on a daily basis in shops or via the phone

Prerequisite Knowledge / Skills

None

Course Description

Upon completion of this course, you will be able to:

- understand the PCI standard and how it affects you
- identify the different components of credit card data
- understand what happens when a credit card is used
- safely handle credit cards and credit card data

Modules Covered

The PCI DSS Standard

Credit Card Transaction Flow

The Do's and Don'ts of Handling Credit Card Data

Assessment

An interactive exam at the end of the course to test how well the participants understand how to handle credit cards and credit card data.

Upon completion of the exam, successful participants will receive a certificate showing that they have passed the course.

8. Critical Infrastructure Protection 101

Duration: 1 hour

Audience

This course is intended for employees who work in a critical infrastructure environment, in particular power utilities

Prerequisite Knowledge / Skills

None

Course Description

Upon completion of this course, you will be able to:

- understand what critical infrastructure is
- understand what cyber-terrorism is and how it is a threat
- understand what SCADA systems are and know how they should be protected
- gain a realization of their importance in protecting critical Infrastructure
- know what the NERC is, and what its role is
- know what the NERC CIP reliability standards are

Modules Covered

Cyber Terrorism & Cyber Attacks

- Definition & description
- How it is carried out

NERC

SCADA

Assessment

An interactive exam at the end of course is used to test the participant's understanding of CIP and their role in it.

Upon completion of the exam, successful participants will receive a certificate showing that they have passed the course.

9. Software Security Awareness

Duration: 1 hour

Audience

Managers and Technical staff members involved in software development and IT/security personnel that handle software and want to better understand and reduce the risk it carries

Prerequisite Knowledge/Skills

Basic knowledge of software development processes and technologies

Course Description

Similar to functionality, performance, and reliability, security is another crucial component of an application's quality. Recognizing the risk that software vulnerabilities represent, understanding their root causes, and addressing these issues early in the software development lifecycle are essential for being able to help your organization build secure software.

By the end of this course you will be familiar with the main characteristics of a secure software development lifecycle and the activities that your organization should perform to develop secure software. Additionally, you will recognize the need to address software security in your everyday work.

Upon completion of this course, participants will be able to:

- understand why software security is critical
- list the primary motivations behind software security
- recognize the need for managing application security risk
- realize that every person in the organization plays an important role with respect to software security
- explain and implement general risk mitigation techniques
- identify security activities for every role within the development organization

Course Modules

Recognizing the importance of software security

This module presents recent trends in the attack landscape, the concept of software security risk and the need to manage this risk as an organization. Upon completion of this module, you will be able to:

- recognize that there are threats to assets
- define software security
- recognize the risk of software vulnerabilities
- evaluate the consequences of vulnerabilities

Managing software security risk in the development lifecycle

This module describes how a development organization can address software security risks by adopting a secure development lifecycle that includes establishing security requirements, designing security into software and leveraging risk-based testing. Upon completion of this module, you will be able to:

- identify the root causes of software vulnerabilities
- recognize that the cost of fixing vulnerabilities increases over time
- understand what security activities need to be conducted to reduce software security risk

Assessment

This module concludes with an assessment that contains 15 questions aimed at measuring the effectiveness of the training.

10. Application Security Fundamentals

Duration: 2 hours

Audience

Security and development practitioners that want to understand software security risk and seek specific implementation guidance on how to build and deploy more secure software applications

Prerequisite Knowledge/Skills

Understanding of the software development lifecycle and technologies; basic understanding of software security

Course Description

This course starts off describing the various risks that software vulnerabilities carry, and proceeds to lay the foundation for secure software development by presenting specific security controls and principles that teams can implement immediately to reduce software risk. Upon course completion, you will be able to:

- understand and recognize threats to applications
- leverage the OWASP top 10 list to create more secure Web applications
- conduct specific activities at each development phase to ensure maximum hardening of your applications

Course Modules

Introduction to Software Security

This module presents trends in the attack landscape, the attacker mindset, the concept of software security risk and the need to manage this risk as an organization.

Challenging Security Misconceptions

This module presents common and dangerous misconceptions that lead to a false sense of security, including:

- Client-side security does not exist
- QA is not security testing
- The application is not the network
- Tools are not solutions
- Patches do not guarantee security
- All software applications have bugs

The OWASP Top Ten List

This module presents the OWASP Top Ten Threats, how each works, its impact and the best way to mitigate.

Security Principles

This module describes specific principles that help guide design, coding and implementation decisions

- Layered security / defense in depth
- Segmentation
- Structural security
- Principle of least privilege
- Default to deny all
- Handling input and output

Security Goals and Controls

This module presents the goal of secure software design and security controls that will help mitigate software risk

- Confidentiality, Integrity, Availability
- Error/exception handling
- Authentication, Authorization/access control
- Cryptography and encryption

Security in the Software Development Lifecycle (SDLC)

This module prescribes specific activities for each phase of the SDLC: Requirements, Design, Development, Testing, Production, Maintenance

Assessment

The module concludes with an assessment containing 15 questions aimed at measuring the effectiveness of the training.

11. Introduction to the *Security Development Lifecycle* (SDL)

Duration: 1 hour

Audience

Anyone involved in the production of software applications that wants to adopt the Microsoft SDL, an industry-leading software security assurance process

Prerequisite Knowledge/Skills

Basic knowledge of the Software Development Lifecycle

Course Description

This course introduces the Security Development Lifecycle (SDL), a key security engineering process that was spawned from Microsoft's Trustworthy Computing Initiative - and shows you how to design and implement products that meet your organization's security needs.

Upon completion of this course, you will be able to:

- identify the benefits of the Security Development Life Cycle (SDL)
- recognize the importance of the Final Security Review (FSR)
- follow the necessary steps to meet SDL requirements
- identify the appropriate tools required by the SDL

Course Modules

Defining the Security Development Lifecycle

As part of any software product, development or engineering team, your goal is to help build software that meets your organization's security requirements. This module describes the importance of security to the products you are developing.

Applying the Security Development Lifecycle

In order to effectively integrate SDL principles into your daily tasks, you need to understand your role within your team, identify the steps you need to follow, and know what tools can help you with your activities.

Upon completion of this module, you will be able to:

- recognize how security fits into each phase of the product development lifecycle: requirements, design, implementation, verification, release, and response phases
- follow the SDL steps for each phase of the software development lifecycle
- select the most suitable tools that will help you to meet the security requirements

Assessment

Participants will complete various self-test questions throughout the course.

12. SDL for Management

Duration: 1 hour

Audience

Managers and Technical Leads

Prerequisite Knowledge / Skills

Knowledge of the Software Development Life Cycle

Course Description

This course introduces you to the Microsoft SDL, an industry leading software-security assurance process, developed by Microsoft to build trustworthy software products, The goal of this course is to help you understand and identify the *Security Development Life Cycle (SDL)* requirements for building and deploying secure software applications. The course demonstrates the benefits teams gain by following the SDL, and it provides you with information regarding your role and responsibilities in ensuring your team follows the SDL. Additionally, this course introduces you to the common problems that can delay or stop product shipping.

Course Objectives

Upon completion of this course, you will be able to:

- identify the benefits of the SDL
- outline the SDL process
- recognize the role of management in following the SDL
- identify whether the team is on track to ship secure products
- identify problems that can delay or stop product shipping

Modules Covered

Defining the Security Development Lifecycle

This module provides an overview of the SDL and the benefits of implementing the SDL. It also describes security requirements and activities for each phase of SDL that you will need to follow to ensure the security of your products, including:

Requirements Phase:

- Register for a security advisor
- Identify a security contact
- Configure your bug-reporting tool
- Establish an SDL bug bar

Design Phase:

- Complete a security design review
- Perform threat modeling
- Adhere to the firewall and APTCA policy
- Meet Microsoft minimum cryptographic requirements

Implementation Phase:

- Do not use banned APIs
- Build & analyze your code according to best practice
- Service /GS crashes
- Use the HeapSetInformation interface

Verification Phase:

- Review and update threat models
- Re-evaluate your project’s attack surface
- Identify the development and test owners
- Satisfy the AppVerifier testing requirements

Release Phase:

- Perform the Final Security Review (FSR)
- Establish the Emergency Response Plan (ERP)

Response Phase:

- Consult your team's ERP
- Reproduce the original vulnerability reports
- Re-evaluate your existing threat models
- Identify other attack vectors
- Develop & release a quality fix in timely manner
- Evaluate similar vulnerabilities in competitive products.

Managing the Security Development Life Cycle

This module explains the role of management in the success of the SDL implementation and provides you with the necessary information regarding the different tasks that managers should complete in order to ensure that their products pass the FSR. This module focuses on what managers must do to lead their teams to build more secure and reliable products. After completing this module you will be able to:

- recognize the role of management in following the SDL
- identify whether the team is on track to ship secure products

Assessment

Participants will complete various self-test questions throughout the course.

13. Introduction to Threat Modeling

Duration: 1 hour

Audience

Architects and/or those involved in the design of secure software applications

Prerequisite Knowledge/Skills

Basic knowledge of the Software Development Lifecycle

Course Description

This course introduces the technique of Threat Modeling, its primary goals, and its role within software development. Once you are familiar with the concepts behind Threat Modeling, the entire Threat Modeling process is demonstrated – giving you the knowledge you need to apply Threat Modeling to your own products and design/develop more secure code.

Upon completion of this course, you will be able to:

- identify the goals of Threat Modeling and the corresponding SDL requirements
- identify the roles and responsibilities involved in the Threat Modeling process
- use the Threat Modeling process to accurately identify, mitigate, and validate threats
- leverage various tools that help with Threat Modeling

Course Modules

Defining Threat Modeling

This module equips you with the necessary information to help you understand the importance of Threat Modeling and the role it plays in identifying and mitigating threats.

Upon completion of this module you will be able to:

- identify the goals of Threat Modeling
- recognize the relation between Threat Modeling and the SDL
- identify the roles involved in the Threat Modeling process
- understand what and when to Threat Model

Applying the Threat Modeling Process

This module identifies in detail each step in the Threat Modeling process, outlines each step's purpose, and demonstrates the procedure to follow in order to apply each step. This module includes a lab to help you apply what you have learned in a real-world scenario.

Upon completion of this module you will be able to:

- describe the application using diagrams
- identify Threat Types by using STRIDE
- identify appropriate mitigation techniques
- recognize the role of the Threat Model document
- understand the various threat modeling tools available to you

Assessment

Participants will complete various self-test questions throughout the course.

14. Architecture Risk Analysis

Duration: 1 hour

Audience

Program/Project Managers, architects, developers, engineers

Prerequisite Knowledge/Skills

Basic understanding of software architecture

Course Description

This course highlights methods of finding design-level security flaws in software. Techniques include accurately capturing application architecture, threat modeling with attack trees, attack pattern analysis, and enumeration of trust boundaries.

Learning objectives:

- Extract architecture views of a software system suitable for security analysis
- Apply a number of complementary techniques to find security vulnerabilities that cannot be easily discovered through tools
- Weigh the comparative impact of design-level security

Assessment

A multiple-choice exam is taken at the end of the course.

15. Creating Secure Code for ASP.NET Applications

Duration: 4 hours

Audience

Developers and testers with ASP.NET/C# programming experience

Prerequisite Knowledge/Skills

ASP.NET/C# programming knowledge and experience

Course Description

This course examines in depth the development of secure web applications in C#. It provides an overview of common web application vulnerabilities and presents ways to avoid those vulnerabilities in C# code. In the hands-on section, students will discover the vulnerabilities for themselves and find ways to address them, greatly enhancing the security of their code.

Upon completion of this class, participants will be able to:

- recognize the need to follow secure coding best practices
- follow secure coding best practices
- locate additional resources on secure coding best practices for ASP.NET

Course Modules

The need for secure coding best practices

This topic explains the consequences of not following best practices; it also provides an overview of the advantages of addressing security issues at the implementation phase, rather than later in the software development lifecycle.

Ten best practices

This topic details each of the ten best practices listed below; for each, it defines the best practice, outlines the vulnerabilities that could arise should the best practice not be followed, and provides the students with multiple hands-on exercises. The ten best practices covered include:

1. Perform Input and Data Validation
2. Err and Fail Securely
3. Handle Sensitive Data with Care
4. Follow Secure Account Management Policies
5. Follow Secure Auditing and Logging Procedures
6. Implement Proper Authorization
7. Do not Reveal Too Much Information
8. Practice Defense in Depth
9. Do Not Reinvent the Wheel
10. Locating Additional Resources

Assessment

A 15-question multiple-choice exam is taken at the end of the course.

16. Creating Secure Code for J2EE Web Applications

Duration: 13 hours

Audience

Developers and testers with Java web programming experience

Prerequisite Knowledge/Skills

Java programming knowledge and experience

Course Description

This course examines in depth the development of secure web applications in Java. It provides an overview of common web application vulnerabilities and presents ways to avoid those vulnerabilities in Java code. In the hands-on section, students will discover the vulnerabilities themselves and find ways to address them, greatly enhancing the security of their code.

Upon completion of this class, participants will be able to:

- identify why software security matters to their business
- recognize the root causes of the more common vulnerabilities
- identify the symptoms of common vulnerabilities
- use security best practices to prevent common vulnerabilities

Topics Covered

Best Practices

This topic details each of the best practices listed below; for each, it defines the best practice, outlines the vulnerabilities that could arise should the best practice not be followed and provides the students with many hands-on exercises. The best practices covered include:

1. Course Overview and Objectives
2. Recognizing the Need for Secure Coding Best Practices
3. Perform Input and Data Validation
4. Err and Fail Securely
5. Handle Sensitive Data with Care
6. Follow Secure Account Management Policies
7. Follow Secure Auditing and Logging Procedures
8. Implement Proper Authorization
9. Do not Reveal Too Much Information
10. Practice Defense in Depth
11. Do Not Reinvent the Wheel
12. Locating Additional Resources

Assessment

A 15-question multiple-choice exam is taken at the end of the course.

17. Introduction to Cryptography

Duration: 1 hour

Audience

Software Application Architects, Developers, Engineers

Prerequisite Knowledge/Skills

Strong software development skills

Course Description

This course provides you with the knowledge to understand cryptography and an opportunity to investigate the threats that affect two communicating parties. It also helps you understand how these threats can be mitigated using a proper cryptographic solution. Upon completion of this course, you will be able to:

- identify the problems that cryptography can address
- recognize threats that apply to two communicating parties
- select appropriate cryptographic solutions to mitigate these threats
- describe the mechanisms behind cryptographic protocols
- follow cryptographic best practices
- locate cryptography resources

Course Modules

The Foundations of Cryptography

- Explaining the Foundations of Cryptography
- Identifying the Aspects of Security That Cryptography Addresses
- Authentication
- Symmetric Ciphers
- Asymmetric Cryptographic Ciphers
- Explaining Hash Functions

Choosing Appropriate Cryptographic Methods

- Choosing Appropriate Cryptographic Methods
- Considering Symmetric Ciphers for Message Encryption
- Considering HMAC to Provide Tamper Detection
- Considering Asymmetric Ciphers for Message Encryption
- Considering Asymmetric Ciphers to Encrypt Cryptographic Keys
- Considering Asymmetric Cryptography to Provide Authentication

Applying Cryptographic Best Practices

- Applying Cryptographic Best Practices
- Selecting the Proper Cryptographic Algorithm
- Designing Cryptographically Agile Applications
- Applying Cryptographic Agility
- Using Proper Cryptographic Solutions
- Selecting a Proper Key Management Scheme
- Securely Storing, Exchanging and Using Secret Keys

Assessment

A 15-question multiple-choice exam is taken at the end of the course.

18. Introduction to Buffer Overflows

Duration: 2 hours

Audience

All development/engineering team members

Prerequisite Knowledge / Skills

Basic knowledge of Windows programming and memory management in Windows

Course Description

This course provides all the required information to understand, avoid and mitigate the risks posed by buffer overflows. The students are first provided with a detailed background on the mechanisms of exploit of stack-based and heap-based buffer overflows. The course then delves into the protections provided by the Microsoft compiler and the Windows operating system, such as the /GS flag and Address Space Layout Randomization (ASLR), followed by practical advice on how to avoid buffer overflows during the design, development, and verification phases of the software development life cycle. Practical examples are provided throughout the course to help students understand and defend against buffer overflows.

Course Objectives

Upon completion of this course, the student will be able to:

- identify the dangers posed by buffer overflows
- describe the exploitation techniques for stack-based and heap-based buffer overflows
- leverage built-in system defenses that protect against buffer overflows
- apply best practices to avoid buffer overflows
- perform testing that detects buffer overflows

Modules Covered

Recognizing the Threats Posed by Exploitation

In order to successfully mitigate buffer-overflow vulnerabilities it is important for you to understand what exploitation is, why it occurs, and what its consequences are to overall application and system security.

Explaining Exploitation Techniques

This module describes various aspects of the development of exploit code, such as common exploitation techniques, typical exploit payloads, and exploitation frameworks. It also provides you with an overview of the practices employed by exploit code to overcome operating system and application defenses. After completing this module you will be able to:

- outline the challenges faced by exploit code during execution
- cite actual exploits targeting Microsoft products and their modes of operation
- describe common exploit payloads
- outline common exploitation tools

Mitigating Exploitation

This module describes techniques aimed at avoiding and mitigating exploitation of buffer overflows, including built-in mechanisms in the Windows family of operating systems whose purpose is to make it difficult to exploit a buffer overflow.

Assessment

Participants will complete various self-test questions throughout the course.

19. Exploiting Buffer Overflows

Duration: 2 hours

Audience

Developers, engineers, testers, and QA personnel

Prerequisite Knowledge/Skills

Basic knowledge of the C/C++ programming language, Windows programming, and memory management

Course Description

This course provides you with all the required information to help you understand and mitigate buffer-overflow exploits. It first introduces the concepts necessary to recognize the threats posed by these exploits and to comprehend the mechanisms behind exploitation of stack-based and heap-based buffer overflows. The course then delves into the different challenges faced by exploit code and how different exploitation techniques overcome environmental limitations.

Upon completion of this course, you will be able to:

- recognize the threats posed by exploitation of vulnerable programs
- describe how buffer-overflow vulnerabilities are exploited
- outline the challenges faced by exploit code during execution
- describe common exploitation techniques and exploit payloads
- cite actual exploits targeting Microsoft products and their modes of operation
- outline common exploitation tools
- use existing mitigations to defend against exploitation

Course Modules

Recognizing the Threats Posed by Exploitation

Buffer overflows are one of the most commonly known types of security vulnerabilities. While they can be easily remediated, past events have shown that they have often been overlooked and widely exploited. In order to successfully mitigate buffer-overflow vulnerabilities it is important for you to understand what exploitation is, why it occurs, and what its consequences to overall application and system security are. This module will help you understand buffer overflow exploitation and will provide you with the necessary information to identify the risks posed by exploitation.

Exploitation Techniques

Exploitation techniques vary in complexity, from simple attacks that spawn processes to complex, multi-stage attacks. This module describes various aspects of the development of exploit code, such as common exploitation techniques, typical exploit payloads, and exploitation frameworks. It also provides you with an overview of the practices employed by exploit code to overcome operating system and application defenses.

Mitigating Exploitation

This module describes techniques aimed at avoiding and mitigating exploitation of buffer overflows, including built-in mechanisms in the Windows family of operating systems whose purpose is to make it difficult to exploit a buffer overflow. In addition, you will be provided with a list of additional resources on exploitation, which will enable you to keep up to date with the latest advancements and countermeasures in the field. After completing this module you will be able to use existing mitigations to defend against exploitation, including:

- Address Space Layout Randomization
- the /GS compiler flag
- Data Execution Prevention

Assessment

Participants will complete various self-test questions throughout the course.

20. Introduction to Integer Overflows

Duration: 1 hour

Audience

Testers, Developers, Engineers and QA personnel

Prerequisite Knowledge/Skills

Basic understanding of the C, C++ and C# programming languages

Course Description

An integer overflow is a programming error that can severely impact a computer system's security. Due to the subtlety of this bug, integer overflows are often overlooked during development. This course covers the security concepts, testing techniques, and best practices that will enable you to develop robust applications that are secure against integer overflow vulnerabilities.

Upon completion of this class, participants will be able to:

- recognize integer overflows in your existing code base
- apply best practices in order to prevent integer overflows
- perform specialized testing that detects integer overflows

Course Modules

Introduction to Integer Overflows

This module will cover all the information required to understand the mechanisms that lead to integer overflows, while showing the potential damage that they can cause to an application. The goal of this module is to enable you to explain why integer overflows occur and to correctly identify the threats posed by them.

Preventing Integer Overflows

This module will provide you the necessary information and best practices to prevent integer overflows. Specifically, this module will illustrate how to use integer overflow countermeasures such as checked expressions, how to choose correct integral types, and how to use existing libraries and classes that have been designed to be safe from integer overflows. Additionally, this module will provide you with proper code review and testing techniques that will enable you to identify and eliminate existing integer overflows.

Assessment

Participants will complete various self-test questions throughout the course.

21. Web Vulnerabilities – Threats & Mitigations

Duration: 1 hour

Audience

Developers, engineers, testers and those that handle Web applications

Prerequisite Knowledge/Skills

Basic knowledge of web technologies and regular expressions

Course Description

This course provides all the information you need to understand, avoid, and mitigate the risks posed by Web vulnerabilities. You are first provided with a detailed background on the most common and recent attacks against Web-based applications, such as cross-site scripting attacks and cross-site request forgery attacks. The course then delves into practical recommendations on how to avoid and/or mitigate Web vulnerabilities. Real-world examples are provided throughout the course to help students understand and defend against Web vulnerabilities.

Upon completion of this course, you will be able to:

- understand and identify the most common and recent attacks against Web-based applications
- describe the mechanisms of exploitation of Web vulnerabilities
- apply best coding practices to avoid Web vulnerabilities
- perform code reviews that detect Web vulnerabilities
- locate additional Web vulnerabilities resources

Course Module

Recognizing the Dangers of Web Vulnerabilities

In order for you to effectively mitigate Web vulnerabilities, it is important that you understand the impact of these dangers. This module provides a historical perspective on the damage these types of vulnerabilities have caused and presents detailed mechanics of major web vulnerabilities and how they lead to serious security issues.

Upon completion of this module you will be able to:

- describe the origins and impact of Web vulnerabilities
- recognize the dangers of ActiveX control misuse
- recognize the dangers of cross-site scripting, canonicalization, SQL Injection, HTTP response splitting, and cross-site request forgery vulnerabilities

Challenging Security Misconceptions

This module introduces several protections and best practices which if implemented properly, help mitigate the risk of web vulnerabilities in applications. Topics covered include the limitations of common mitigations, truly effective mitigations such as allow lists and frame restrictions, and SDL requirements aimed at mitigating Web vulnerabilities.

Upon completion of this module you will be able to:

- recognize the limitations of common mitigations for Web vulnerabilities
- recognize effective mitigations for Web vulnerabilities
- recognize the SDL requirements aimed at mitigating Web vulnerabilities

Assessment

Participants will complete various self-test questions throughout the course.

22. Defensive Programming - C/C+

Duration: 1 hour

Audience

Software Application Architects, Developers, Engineers

Prerequisite Knowledge/Skills

Strong software development skills

Course Description

This course focuses on common mistakes made in C/C++ applications in client/server and distributed systems on Windows and Unix systems and presents effective approaches to writing secure code. The language-specific prescriptive guidance covers data validation, avoiding denial-of-service, secure error handling and logging paradigms, and engineering common security features.

Upon completion of the course, students will be able to:

- implement common functional features using known-good techniques that provide built-in security assurance
- conduct low-level design such that robustness and resilience are first-class considerations
- understand the details and rationale behind properly avoiding common mistakes using defensive programming techniques

Assessment

A 15-question multiple-choice exam is taken at the end of the course.

23. Defensive Programming – C#

Duration: 1 hour

Audience

Software Application Architects, Developers, Engineers

Prerequisite Knowledge/Skills

Strong software development skills

Course Description

This course focuses on common mistakes made when building .NET applications with C# and presents effective approaches to writing secure code. The language-specific prescriptive guidance covers data validation, avoiding denial-of-service, secure error handling and logging paradigms, and engineering common security features.

Upon completion of the course, students will be able to:

- implement common functional features using known-good techniques that provide built-in security assurance
- conduct low-level design such that robustness and resilience are first-class considerations
- understand the details and rationale behind properly avoiding common mistakes using defensive programming techniques

Assessment

A 15-question multiple-choice exam is taken at the end of the course.

24. .NET Security

Duration: 2 hours

Audience

Development team members building applications on the .NET framework

Prerequisite Knowledge / Skills

Basic knowledge of Windows Programming and the .NET framework

Course Description

This course describes .NET security features, including concepts such as Code Access Security (CAS) and .NET cryptographic technologies. It also provides secure coding best practices that will enable you to build more secure applications in .NET.

Course Objectives

Upon completion of this course, you will be able to:

- identify the differences between managed and unmanaged code
- describe how access control functions in Windows
- describe how Code Access Security functions in .NET
- recognize the interactions between Windows access control and CAS
- describe how cryptography is handled in .NET
- recognize the main aspects of ASP .NET security
- recognize the security improvements brought by .NET 2.0
- write defensive code that protects your application from common threats
- recognize the usefulness of the FxCop tool

Modules Covered

Explaining .NET Security Features
In order to build secure applications in .NET, it is important that you first understand the .NET Framework and the security features it offers. This module provides you with the knowledge you need to understand the foundation of .NET, the CLR's native security infrastructure (Code Access Security), cryptographic technologies, and the ASP.NET security infrastructure.

Applying .NET Secure Coding Best Practices

This module provides you with secure coding techniques and best practices that will prevent you from falling victim to common security flaws and ultimately help you build secure applications in .NET.

Assessment
Participants will complete various self-test questions throughout the course.

25. Classes of Security Defects

Duration: 3 hours

Audience

Testers and other development team members who want to better understand the types of defects that lead to security vulnerabilities

Prerequisite Knowledge / Skills

Basic knowledge of client-server applications, web applications, the Software Development Lifecycle, cryptography and the STRIDE model

Course Description

This course equips you with the knowledge you need to create a robust defense against common security defects. You will learn why and how security defects are introduced into software and be presented with common classes of attacks, which will be discussed in detail. Along with examples of real life security bugs, you will be shown techniques and best practices that will enable you and your team to identify, eliminate, and mitigate each class of security defects. Additional mitigation techniques and technologies are described for each class of security defect.

Course Objectives

Upon completion of this course, you will be able to:

- understand and outline the common classes of security defects
- recognize the potential impact that common security defects can have
- identify the programming errors that are responsible for common security defects
- apply coding best practices in order to avoid common security vulnerabilities
- find common security defects in an application's source code
- map common security defects with specific technologies
- test software in order to detect common security bugs
- locate additional resources on common security defects

Modules Covered

Classes of Security Defects

This module presents the underlying root causes of security defects, explains the difference between functional and security bugs, and describes the inherent insecure nature of software.

Defending against Common Security Defects

This module offers best practice tips for defending against common security defects such as:

- buffer and integer overflows
- format string problems
- integer overflow
- SQL and command injection
- improper error handling
- cross-site scripting
- unprotected network traffic
- lack of server-side authorization
- poor usability
- improper use of SSL and TLS
- weak authentication and data protection
- information leakage
- improper file access
- spoofing
- race conditions
- unauthenticated key exchange
- weak random number generation

Assessment

Participants will complete various self-test questions throughout the course.

26. Windows Vista Security

Duration: 2 hours

Audience

Development team members building applications targeting Microsoft Windows Vista

Prerequisite Knowledge / Skills

Basic knowledge of Windows programming and memory management, and knowledge of basic security features of Windows versions prior to Vista

Course Description

This course provides you with knowledge and skills you need to understand Windows Vista security features and build applications that leverage Windows Vista's built-in security mechanisms.

Course Objectives

Upon completion of this course, you will be able to:

- describe the Windows Authorization Model
- describe Windows Vista User Account Control (UAC)
- implement Integrity Control in Windows Vista
- apply Service Hardening Techniques
- describe and leverage techniques that protect against Buffer Overflow exploits
- describe and leverage techniques that protect against network-level attacks
- locate additional Windows Vista security resources

Modules Covered

Describing Microsoft Windows Authorization Model

In order to fully appreciate and leverage Window Vista security features, it is important that you are familiar with the Microsoft Windows Authorization Model which serves as the basis for many of the new security features introduced in Windows Vista. After completing this module you will be able to:

- define Security Principals
- define Securable Objects
- describe Access Control Lists
- describe Access Tokens and Write-Restricted Tokens

Explaining Windows Vista Security Features

This module introduces you to the new security features of Windows Vista and provides you with information you need to build applications that leverage Windows Vista's built-in security mechanisms.

Assessment

Participants will complete various self-test questions throughout the course.

27. How to Break Software Security

Duration: 15 hours

Audience

Testers and developers

Prerequisite Knowledge/Skills

Functional testing knowledge as well as a basic understanding of how applications work

Course Description

This course is designed to give testers and developers the tools and techniques they need to help find security problems before their application is released. It lays the foundation you need to effectively recognize and expose security flaws in software and It introduces a fault model to help testers conceptualize these types of bugs.

Upon completion of this class, participants will be able to:

- recognize threats to your software applications
- identify the symptoms of security vulnerabilities
- conduct specialized attacks to uncover vulnerabilities in any type of software application
- recognize whether or not an attack was successful

Course Modules

Introduction to Software Security

Learn why security bugs are different and to recognize symptoms of insecure behaviors

Assessing Risk

Learn to master the art of translating threats into malicious abuse cases

Attacking Dependencies

- Discover 5 techniques that test that your application responds securely if a dependency were to fail
- Learn how memory, network, files, registry and other resources can cause your application to behave insecurely
- Learn how to simulate dependency failures in your application's environment using Fault Injection tools

Attacking through the User Interface

- Learn about SQL injection, buffer overflows, escape characters, executable data and much more
- Learn about the most common security vulnerability in software and how to test for it
- Learn the 3 testing techniques to expose security vulnerabilities in your software through the user interface

Attacking Design

- Learn 7 testing techniques to expose vulnerabilities that can creep into an application at the design stage
- Understand why legacy code can create huge security holes
- Learn how inappropriate uses of temporary files and the registry can be manipulated to force insecure behavior

Attacking Implementation

- Learn 4 techniques that can be used to expose vulnerabilities that exist because of implementation errors
- Recognize error messages that reveal sensitive information
- Learn about how timing related vulnerabilities work and how to expose them during testing

Assessment

A multiple-choice exam is taken at the end of the course.

28. Fundamentals of Security Testing

Duration: 2 hours

Audience

Testers as well as development team members who want to understand how to test for security vulnerabilities

Prerequisite Knowledge / Skills

Basic knowledge of the software development life cycle, software testing and computer programming

Course Description

This course introduces security-testing concepts and processes that will help you analyze an application from a security perspective and to conduct effective security testing. The course focuses on the different categories of security vulnerabilities and the various testing approaches that target these classes of vulnerabilities. Several manual and automated testing techniques are presented which will help you identify common security issues during testing and uncover security vulnerabilities.

Course Objectives

Upon completion of this course, the student will be able to:

- recognize how security testing fits in the Security Development Life Cycle (SDL)
- understand how security testing differs from functional testing
- categorize and test for common and dangerous security vulnerabilities
- leverage automated tooling for security testing

Modules Covered

Analyzing Applications with a Security Mindset

This module introduces you to the role security testing plays within the Security Development Life Cycle (SDL) and the specific requirements and activities that need to be adhered to. Information sources for security testing and functional testing are contrasted and you will learn how the main sources of information for security testing can be leveraged to conduct testing. Inputs and input entry points are then analyzed from a security perspective. Security test automation and risk quantification, two important topics for security testers, finalize the module.

After completing this module you will be able to:

- identify information sources for security testing
- assess inputs and data from a security perspective
- consider automating security testing
- define stopping criteria for security testing

Applying Security Testing Approaches

Security vulnerabilities can originate from different sources—from the environment, from the input, and from data and logic within the application. This module describes the different types of vulnerabilities and provides examples of each.

You will be shown various testing approaches designed to identify the different types of vulnerabilities. For each approach, you will learn what problem is being looked for, how to conduct testing, and how to identify symptoms that may indicate that vulnerability is present in the software.

This module also presents two important testing techniques, fuzz testing and fault injection, that help security testers maximize their efficacy and efficiency.

Assessment

Participants will complete various self-test questions throughout the course.