Instructor-led Training
Course Catalog

Effective January 2017
Introduction

Cigital's instructor-led courses are developed and taught by experts at the forefront of the software security field. Our instructors are certified security professionals who have hands-on experience working directly with clients on their security challenges.

Our curriculum includes training modules for professionals just starting out with software security as well as those who are looking to develop more advanced skills. Cigital continuously develops its courses to accommodate the rapid changes in software security.

Our Curriculum

Cigital's curriculum is a series of complementary courses designed to meet your organization's needs. You can select the courses that best complement your level of experience, your role, and the development platforms in your organization. We offer courses at 5 levels:

2. **Platform Security** – Understand the fundamentals of security for .NET, Java, and Mobile, and explore common cloud technologies and platforms.
3. **Secure Design** – Learn how to identify common vulnerabilities and essential strategies to defend your applications from external threats.
4. **Secure Development** – Learn defensive programming skills in context, in specific languages targeted to specific development platforms, so you can prevent coding errors.
5. **Security Testing** – Use your knowledge to test your applications for security vulnerabilities.

We can work with you to select the curriculum that is right for you.
Wondering what classes are right for your organization?

Locate the roles relevant to your organization, then build your custom curriculum using the course suggestions in the provided categories.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Security Foundations: Mobile</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Clarity and Security Foundations: Mobile
Sample Developer Curriculum

**Platform Security**
- Cloud Security: AWS DevOps Workshop
- Security Foundations: .NET
- Security Foundations: Java
- Security Foundations: Mobile

**Secure Design**
- Secure Design: Attack and Defense
- Secure Design: OWASP Top Ten
- Secure Design: Web Services

**Secure Development**
- Defensive Programming in C# .NET (Desktop Apps)
- Defensive Programming in C# .NET (Web Apps)
- Defensive Programming in C and C++
- Defensive Programming in HTML5
- Defensive Programming in Java for Android Apps
- Defensive Programming in Java for EE Web Apps
- Defensive Programming in Java for EJB Apps
- Defensive Programming in Java SE Apps
- Defensive Programming in JavaScript
- Defensive Programming in Objective C for iOS Apps
- Defensive Programming in PHP for the Web
- Defensive Programming in Python and Django

**Security Testing**
- Security Testing: Android
- Security Testing: iOS
- Security Testing: Web Apps
In-Person or Virtual – Your Choice

If you choose traditional instructor-led training, our certified instructors will travel to the location of your choice. Our instructors are trained to engage your audience through group discussion and interactive hands-on labs designed to simulate real-world environments. On-site instructors can make course adjustments to better complement the needs, interests, and experience-level of your participants.

If you have a distributed workforce, your participants can avoid travel and time away from the office using our Virtual Instructor-led Training (VILT). VILT is separated into smaller sessions to optimize participant engagement. VILT can be delivered over consecutive working days or on a weekly basis, depending on your team’s preference. Virtual training is a cost-conscious alternative for supporting your employees’ professional development.

Instructor-led courses are held on your schedule in the format that best works for you.
# Listing of Courses

**Introductory**

- Principles of Software Security ................................................................. 8

**Platform Security**

- Cloud Security: AWS DevOps Workshop ...................................................... 9
- Security Foundations: .NET ........................................................................... 10
- Security Foundations: Java ........................................................................... 11
- Security Foundations: Mobile ................................................................. 12

**Secure Design**

- Secure Design: Attack and Defense ........................................................... 13
- Secure Design: OWASP Top Ten ................................................................. 14
- Secure Design: Threat Modeling ................................................................. 15
- Secure Design: Web Services ............................................................... 16

**Secure Development**

- Defensive Programming in C# .Net (Desktop Applications) ......................... 17
- Defensive Programming in C# .Net (Web Applications) ................................ 18
- Defensive Programming in C and C++ ......................................................... 19
- Defensive Programming in HTML5 ............................................................. 20
- Defensive Programming in Java for Android Applications ......................... 21
- Defensive Programming in Java for EE Web Applications ......................... 22
- Defensive Programming in Java for EJB Applications ................................ 23
- Defensive Programming in Java SE Applications ....................................... 24
- Defensive Programming in JavaScript ......................................................... 25
- Defensive Programming in Objective C for iOS Applications ....................... 26
- Defensive Programming in PHP for the Web ............................................... 27
- Defensive Programming in Python and Django ............................................ 28

**Security Testing**

- Security Testing: Android ........................................................................... 29
- Security Testing: iOS .................................................................................... 30
- Security Testing: Red Team Workshop ....................................................... 31
- Security Testing: Web Applications ............................................................ 32
Principles of Software Security

Description
The key to proactive computer security involves getting a risk management handle on the software security problem. This one-day course—created by the experts who literally wrote the book on software security—encompasses software security awareness and best practices using a combination of lecture and interactive exercises for a general audience. The course explains some of the problems in software security and describes an approach to infusing software security into the development process through risk management, software security touchpoints, and historical knowledge of software security vulnerabilities. Everyone involved in software production requires baseline knowledge of software security problems and risks, along with an overall understanding of approaches for producing better software.

Objectives
After successfully completing this course, the student will be able to:

• Recognize why software developers, architects, and managers need to make software security an integral part of their jobs
• Recognize how best practices in the Software Security Touchpoints can augment development processes to improve software security
• Recognize the importance of identifying software requirements
• “Think like an attacker” when conceiving, building, and testing software
• Use knowledge of common software errors to improve the way software is built

Principles of Software Security

Intended Audience
• Developer
• QA and Testing
• IS Security Team

Delivery Format
✓ Traditional Classroom
✓ Virtual Classroom

Class Duration
• Traditional: 8 hours
• Virtual: 4 hours
• Virtual: 8 hours
Cloud Security: AWS DevOps Workshop

Description
Cloud computing has grabbed the world’s attention not only for its pervasive, on-demand, convenient usage, but for its ability to be vulnerable to data breaches and novel forms of attack. Since most software uses the cloud in various shared capacities (development, hosting, or integration with third-party code), threats from hackers are inevitable. This hands-on workshop equips students to understand this new landscape of converged infrastructure and shared services, its existing and emerging threats, and provides them with secure mitigation methods.

The Cloud Security: AWS DevOps Workshop course is a deep dive into cross-discipline information and perspective among developers, operations, and information security personnel. The course enables students to identify areas for cross-pollination between development and operations that enhance application, infrastructure, and network security.

This course assumes the following baseline student knowledge:
- Conceptual familiarity with:
  - Common AWS services: EC2, VPC, RDS, KMS, and IAM
  - Docker
  - Chef, or another infrastructure-as-code tool such as Puppet, SaltStack, or Ansible
- Operational familiarity with:
  - Linux CLI environment

Objectives
After successfully completing this course, the student will be able to:
- Harden public cloud infrastructure using native platform security controls, including: AWS IAM, KMS, EC2, and VPC
- Harden a cloud environment using common tooling, including: Docker and Chef

Intended Audience
- Developer
- Architect
- QA and Testing
- IS Security Team

Delivery Format
- Traditional Classroom
- Virtual Classroom

Class Duration
- 8 hours
Security Foundations: .NET

Description

The .NET platform serves as a powerful framework for developing a wide range of applications, from rich web sites and desktop applications to versatile shared libraries and embedded systems. The platform’s specific architecture and unique security model set it apart from other environments. While these traits enhance the capabilities of applications, they also introduce specific risks from an application security perspective. This course examines the platform’s security model and describes fundamental components that have security implications. The course first allows students to appreciate and understand common security issues inherent in many of the key features of the platform. Mitigation strategies are discussed next to help students understand how to minimize the risks exposed by these issues. Finally, the course describes several key security features provided by the platform along with how these can be leveraged to strengthen the security posture of .NET applications.

Objectives

After successfully completing this course, the student will be able to:

• Understand the .NET framework components and related concepts
• Understand the security features of the .NET Framework
• Describe the limitations of each feature and apply the best practices to create secure .NET applications
Security Foundations: Java

Description

The Java platform offers a powerful, versatile, and robust foundation for creating distributed applications. The platform’s specific architecture and security model set it apart from other environments. On the one hand, the platform provides developers and architects with a multitude of security features that can be leveraged to create resilient applications. On the other hand, some aspects of the Java platform have negative security implications that software practitioners must be aware of to avoid significant security issues.

This course teaches the Java platform security model along with fundamental aspects of the platform that have positive and negative security implications. It enables students to understand common security issues inherent to the Java platform, and then teaches them how to leverage built-in features to design and develop secure Java applications.

Objectives

After successfully completing this course, the student will be able to:

• Comprehend the security-related aspects of the Java platform
• Use the Java platform security model to sandbox Java applications
• Describe the risks inherent to Java platform features and apply best practices to mitigate them
Security Foundations: Mobile

Description

This modular course can be delivered as a one-day or two-day training. The following topics are available:

Mobile First AppSec: Mobile First AppSec (application security) describes what application security teams must consider when testing mobile applications. The name is derived from the development trend of Mobile First Development—a trend that many teams are adopting.

As an introduction, this module covers what is different and what is the same about mobile security when compared to regular web application security. It explains how the application architecture of a mobile application drives the various activities in Mobile First AppSec. In conclusion, it relates the different application architectures and frameworks to differences in their threat model and their consequences on activities, primarily security testing.

Overview of Mobile Platforms: This module provides an overview of the architecture and security controls in today’s two most popular mobile operating systems: Google’s Android and Apple’s iOS. It describes the runtime environment, types of mobile applications, and available inter-process communication (IPC) mechanisms. Security-related topics include application sandboxing, memory management, and application permissions. This information is necessary for architects, developers, and security practitioners who deal with mobile applications.

Protecting Client-side Code: Attackers target mobile applications for many reasons. The code running on mobile devices may contain sensitive intellectual property that attackers may want to reverse engineer. Attackers may also want to modify your application slightly and re-publish it (e.g., to steal credentials from users who download the modified application). This module presents the techniques and tools for making reverse engineering and tampering with mobile applications more difficult, as well as the limitations of these techniques.

Mobile Payments: Cigital has expertise in mobile payments. This module explores the attack surfaces of mobile payment architectures, including NFC-based (Near Field Communication) wallets. It examines the design of the mobile payment clients and the back-end applications, reviews the known attacks against these systems, and explores countermeasures against these attacks that system architects, developers and security practitioners can adopt to harden their systems.
Secure Design: Attack and Defense

Description

Building security in is about building software right the first time, and this course teaches students to do just that. Organized around a few major themes (e.g., data at rest, data in motion, input validation, output encoding), this course teaches some common use cases we want to support, and how to design and implement them securely. This course is not tied to any particular language or domain. Different use cases come from different contexts (e.g., web, embedded, thick client, mobile). Each is presented with its standard attacks and the standard solutions that defend against those attacks. Rather than follow industry-standard security taxonomies that categorize mistakes, this course is organized around common software user stories, and how to do them securely. Topics include proper use of encryption, and handling of data across module boundaries, validation and encoding, and authentication and authorization issues.

At the end of this course, students will have the foundational knowledge to expand their software security and learn specific engineering techniques such as defensive programming, threat modeling, and penetration testing.

Objectives

After successfully completing this course, the student will be able to:

- Recognize security needs around common software use cases
- Match the standard attacks to common software use cases
- Choose standard defenses that are appropriate, given the software’s use case

Intended Audience

- Developer
- Architect
- QA and Testing
- IS Security Team

Delivery Format

- Traditional Classroom
- Virtual Classroom

Class Duration

- 8 hours
Secure Design: OWASP Top Ten

Description
This course focuses on the most important security defects found in web applications, covering all issues in the latest Open Web Application Security Project (OWASP) Top 10 (2013). Each topic describes a vulnerability and provides practical guidance for remediation. This course also demonstrates some of these vulnerabilities and provides hands-on exercises where students learn what impact these security issues can have on web applications. Developers with experience in any programming language can benefit from this course.

Objectives
After successfully completing this course, the student will be able to:

• Understand the role of security in the software development lifecycle and how best to create secure web applications
• Recognize the details of, and the causes behind, common secure coding errors and mistakes in web applications
• Understand how these software security defects are exploited
• Understand the practices that help prevent the most common mistakes and lead to more secure software

Intended Audience
- Developer
- Architect
- QA and Testing
- IS Security Team

Delivery Format
- Traditional Classroom
- Virtual Classroom

Class Duration
- 8 hours
Secure Design: Threat Modeling

Description

Penetration testing and secure code review can uncover many types of security issues in an application; however, there are gaps that simply cannot be found with these traditional analysis techniques. Discovering weaknesses in the design of a system is the specific goal of threat modeling. Organizations benefit from this software design analysis because you can perform it without code to discover potential vulnerabilities early in the development cycle.

This course details Cigital’s threat modeling process and methodologies to teach students how to identify the assets, security controls, and threat agents for a given system. The course goes on to show how this information can be used to create a prioritized list of attacks and propose appropriate mitigations. First, system threat models are described and used to build a holistic view of the security posture of the system based on the application and its associated infrastructure. This is followed by closer analysis of component interaction using protocol/sequence/API threat models. The course is also supported by hands-on lab exercises that allow students to learn by actually going through the threat model process.

Objectives

After successfully completing this course, the student will be able to:

- Describe the Cigital threat modeling process and methodology
- Use Cigital’s threat modeling approach for analyzing applications and systems
  - Identify different types of threat models
  - Describe how to model the software for each type of threat model
  - Describe how to relate assets, security controls, and threat agents
  - Produce a report describing potential attacks and mitigations

Intended Audience

- Architect

Delivery Format

- Traditional Classroom
- Virtual Classroom

Class Duration

- 8 hours
- 16 hours (2 days) available in Q2 2017
Secure Design: Web Services

Description
Web services are the backbone of today’s integrated information technology (IT) systems for accessing web services, such as SOAP, REST, and XML. Web services-based applications represent a new approach and require new security strategies to deal with the risks introduced by these new architectures. This course takes a pragmatic approach towards identifying web services’ security risks and selecting and applying countermeasures to the application, code, web servers, and databases as well as identity servers and related software.

Objectives
After successfully completing this course, the student will be able to:

• Recognize how web application risks (e.g., OWASP Top Ten) apply to web services including SOAP and REST authentication, authorization and auditing
• Recognize specific web services and XML attack patterns
• Provide data and XML security using WS-Security, Security Assertion Markup Language (SAML), XML Encryption, XML Digital Signature, and identity services and federation with SAML and WS-Federation
• Recognize the need in web services for hardening servers, input validation, integrating securely with backend resources and applications, and secure exception handling
• Apply XML Security Gateways in a decentralized web services security architecture
• Recognize the key security issues in service-oriented architecture (SOA) and web services, and how to leverage standards and security protocols to proactively build security into SOA and web services systems

Intended Audience
• Developer
• Architect
• QA and Testing

Delivery Format
✓ Traditional Classroom
✓ Virtual Classroom

Class Duration
• 8 hours
Defensive Programming in C# .Net (Desktop Applications)

Description

This course gives a comprehensive overview of the security issues and common pitfalls affecting C# applications. .NET is the platform for all Windows-based systems and provides a powerful tool to develop applications. However, its architecture is peculiar and requires an understanding of specific design patterns and development techniques. The platform can be used to write clients or standalone DLLs, but is susceptible to many kinds of attacks. Building on the Security Foundations: .NET course, this course teaches students defensive programming techniques to mitigate common risks in C# applications.

The course first describes risks and mitigations associated with UI programming, how to deal with DLLs and unmanaged code securely, and safe techniques to deal with error handling. It then introduces the problems inherent to processing input and output securely, how to protect against XML attacks, secure handling of files, and problems associated with concurrency. Finally, the course explains methods to generate secure random numbers and how to securely use Windows Communication Foundation (WCF) functionality as well as inter-process communication.

Objectives

After successfully completing this course, the student will be able to:

- Recognize risks inherent to the .NET platform
- Describe methods to process input securely
- Describe security risks related to determinism and concurrency
- Describe security risks related to file access, inter-process communication (IPC), and UI programming
- Recognize risks with loading and executing binaries
- Describe methods for safe error handling
Defensive Programming in C# .Net (Web Applications)

Description

Microsoft ASP.NET provides powerful tools for developing robust and dynamic web applications. However, its architecture is peculiar and requires an understanding of specific design patterns and development techniques. Not unlike other web applications and software systems, inherent risks exist in ASP.NET applications that could leave the system susceptible to many kinds of attacks. Building upon the Security Foundations: .NET course, this course provides a comprehensive overview of the security issues and common developer pitfalls affecting ASP.NET applications written in C#. This course covers risks common to most web applications as well as typical .NET-specific risks. Each module concentrates on areas related to defensive programming for ASP.NET C# applications and includes code analysis and remediation exercises. The course is also supported by several interactive demonstrations and hands-on lab exercises.

Objectives

After successfully completing this course, the student will be able to:

• Comprehend the overall approach to securing web applications
• Recognize security risks common to .NET web applications
• Identify security vulnerabilities in .NET web applications
• Apply defensive programming techniques to write secure .NET web applications

Intended Audience

• Developer

Delivery Format

✓ Traditional Classroom
✓ Virtual Classroom

Class Duration

• 8 hours
Defensive Programming in C and C++

Description
This course provides developers with a strong foundation in software security as it relates to the implementation of applications developed in C or C++. It includes detailed examples and focuses on the correct way to think through security problems by combining structured theory, animated demonstrations, technical deep-dives, and illustrated explanations. This course connects the habit of building security in through proven programming practices and explains common security-related problems in detail so that the students can avoid them in their own work.

Objectives
After successfully completing this course, the student will be able to:

- Understand how C and C++ can be exploited in order to enable them to build secure code
- Identify common C and C++ coding mistakes that impact application security
- Apply best practices when developing software to avoid common security coding errors
- Recognize more security errors when reviewing source code manually or using automated code scanning tools
- Identify multiple secure alternatives for fixing common security bugs
- Understand the state of the art in secure coding best practices, and how to apply them to their organization
- Describe methods to eliminate or mitigate security coding errors in products faster

Intended Audience
- Developer

Delivery Format
- Traditional Classroom
- Virtual Classroom

Class Duration
- 8 hours
- 16 hours (2 days) available in December 2016
Defensive Programming in HTML5

Description

HTML5 is the fifth revision of the HTML standard. HTML5 and its integration with JavaScript introduce new security risks that need careful consideration when writing web front-end code. Modern web-based software, including mobile web front-end applications, make heavy use of innovative JavaScript and HTML5 browser support to deliver advanced user experiences. Front-end developers focus their efforts on creating this experience and are generally not aware of the security implications of the technologies they use.

This course helps web front-end developers understand the risks involved with manipulating the HTML Document Object Model (DOM) and using the advanced features of JavaScript and HTML5 (e.g., cross-origin resource sharing, local storage, and content security policy). The course reinforces some important security aspects of modern browser architecture and presents defensive programming techniques that can be immediately applied to prevent the introduction of common vulnerabilities. Additionally, the course provides a detailed description of typical JavaScript sources and sinks, and explains how they can be used to detect problems in code.

Objectives

After successfully completing this course, the student will be able to:

- Identify the attack surface of a web application
- Apply HTML5 defensive programming techniques
- Apply JavaScript defensive programming techniques
- Apply JSON defensive programming techniques

Intended Audience

- Developer

Delivery Format

- Traditional Classroom
- Virtual Classroom

Class Duration

- 8 hours
Defensive Programming in Java for Android Applications

Description

This course explores the security fundamentals of Google’s Android platform including its overall architecture and security model, the Dalvik virtual machine, Android RunTime, permission model, and inter-process communication (IPC) mechanisms. The course also focuses on the software security risks inherent to this specific platform and provides solutions for designing and developing Android applications that are resilient to a broad range of issues including (but not limited to) information disclosure, insecure storage, injection attacks and authentication/authorization problems. Throughout the course, a special emphasis is given on methods to leverage the Android architecture to follow principles such as defense-in-depth and least privilege in order to effectively mitigate the risks associated with your mobile applications and related assets.

Objectives

After successfully completing this course, the student will be able to:

- Comprehend the Android mobile platform
- Describe the risks affecting Android applications
- Implement secure inter-process communications
- Describe methods to prevent sensitive information disclosure
- Describe methods to limit the mobile application’s attack surface
- Describe methods to prevent eavesdropping and man-in-the-middle attacks
- Use generic defensive programming to write secure mobile applications
Defensive Programming in Java for EE Web Applications

Description

The Java Enterprise Edition (JEE) platform provides powerful tools for developing robust distributed applications. Although not web-centric, the JEE platform includes a significant number of web-specific specifications, including Java Server Pages (JSP) and Java Server Faces (JSF). No surprise, it is one of the most popular platforms for implementing large-scale, web-enabled, enterprise applications.

Not unlike other types of web applications, JEE web applications are affected by common problems such as SQL injection, cross-site scripting (XSS), cross-site request forgery (CSRF), and session management issues. Building upon the OWASP Top 10 course, this course provides a comprehensive overview of the security issues and developer pitfalls that affect web applications written in Java. This course teaches students to identify and mitigate vulnerabilities included in, but not limited to, the Open Web Application Security Project (OWASP) Top Ten taxonomy. Alternative remediation advice is also provided for the popular Model-View-Controller (MVC) frameworks such as Struts, Spring, and JSF. Additionally, the course teaches students secure configuration best practices to further harden web applications.

Objectives

After successfully completing this course, the student will be able to:

- Comprehend the overall approach to securing Web applications
- Describe security risks common to JEE Web applications
- Identify security vulnerabilities in JEE Web applications
- Apply defensive programming techniques to write secure JEE Web applications

Intended Audience

- Developer

Delivery Format

- Traditional Classroom
- Virtual Classroom

Class Duration

- 8 hours
Defensive Programming in Java for EJB Applications

Description

Enterprise JavaBeans (EJB) technology powers the back-end code behind complex, scalable, multi-tiered, reliable, and secure software systems.

Defined in the Java Enterprise Edition (JEE) specifications, EJB provides specifications to standardize common business functions. As the functionality of modern applications skyrockets, Java EE developers are often unaware of the security implications of such complex, interconnected systems.

This course helps students understand the risks involved with building large, EJB-based applications. The course also reinforces some important security aspects of the JEE platform and presents defensive programming techniques that may be immediately applied to prevent common vulnerabilities. Additionally, the course provides a detailed overview of the relationships between EJB-based applications and the environment in which they exist.

Objectives

After successfully completing this course, the student will be able to:

- Comprehend the JEE application architecture
- Describe common security risks for JEE applications
- Identify security vulnerabilities in EJB-based applications
- Apply defensive programming techniques to secure EJB-based applications
Defensive Programming in Java SE Applications

Description

The Java Standard Edition (JSE) platform comes with a comprehensive feature set that allows developers to develop and deploy Java applications on desktops, servers, and embedded devices. It offers one of the richest environments for implementing user interface, network communications, performance, versatility, portability, and security. As the functionality of modern applications skyrockets, developers are often unaware of the security implications of such complex, interconnected systems.

This course helps students understand the risks involved with client-server and standalone applications as well as applications that leverage the standard functionality of the Java platform. The course examines specific security aspects of the desktop and client-server software architectures and aims to clear improper trust assumptions that developers commonly make. The core of this course teaches secure coding techniques designed to mitigate security vulnerabilities that affect software, in general, as well as issues that are specific to the Java platform.

Objectives

After successfully completing this course, the student will be able to:

• Describe methods to prevent injection attacks
• Mitigate excessive client-side trust issues
• Perform secure file and error handling
• Describe methods to handle concurrency securely
• Implement secure network communications
• Describe how to leverage cryptographic support built into the JSE platform
Defensive Programming in JavaScript

Description
JavaScript became programming language number one for all front-end development and is now winning server-side ground after the introduction of the Node.js framework running on the V8 JavaScript engine. The Defensive Programming in JavaScript course covers the questions of secure development in front-end, as well as back-end JavaScript. It helps attendees understand generic web application risks, as well as specific risks involved with manipulating JavaScript in the DOM, bypassing browser controls, like same origin policy and sandboxing, sending Ajax requests, analyzing JSON, and using client-side frameworks and libraries. The course also covers the risks present in the server-side code written in JavaScript. These risks are typical for any back-end frameworks such as different types of injections, frameworks misconfigurations, cross-site request forgery, and input validation. The JavaScript frameworks covered in this course include, on the client side, AngularJS, and on the server side, Node.js and Express.js. The lab includes exercises covering vulnerabilities and best practices in each framework.

Objectives
After successfully completing this course, the student will be able to:

- Recognize that client-side JavaScript code can introduce security vulnerabilities
- Describe the JavaScript risk landscape
- Recognize risks related to server-side JavaScript
- Apply defensive programming techniques in JavaScript and its various frameworks (AngularJS, Node.js, and Express.js)
- Identify and fix security vulnerabilities in JavaScript code

Intended Audience
- Developer

Delivery Format
- √ Traditional Classroom
- √ Virtual Classroom

Class Duration
- 8 hours
Defensive Programming in Objective C for iOS Applications

Description

The Apple iOS platform provides a comprehensive set of features for creating versatile mobile applications. The platform's specific architecture and security model sets it apart from other mobile operating environments. This introduces specific risks from a mobile application security perspective. This course teaches defensive programming techniques to mitigate common risks in iOS applications. First, the course introduces fundamental concepts about the platform including the iOS architecture and security model. A special emphasis is given to describing key security controls provided by the platform and how to use them correctly. In addition, this course gives a comprehensive overview of the security issues and common developer pitfalls affecting iOS applications—both generic ones and ones that are inherent to the iOS platform. The course teaches detailed techniques for mitigating risks affecting iOS applications, including:

- Preventing information disclosure
- Implementing proper access control
- Strong cryptography
- Secure input validation and data representation
- Mitigations against reverse engineering
- Performing secure inter-process communication
- Using the fingerprint recognition feature, TouchID, for authentication
- Failing securely
- Secure network communications
- Mitigating the risk of jailbreaking

Objectives

After successfully completing this course, the student will be able to:

- Comprehend the Apple iOS mobile platform
- Describe the risks affecting iOS applications
- Implement secure inter-process communications
- Prevent sensitive information disclosure
- Limit the mobile application's attack surface
- Prevent eavesdropping and man-in-the-middle attacks
- Use generic defensive programming to write secure mobile applications

Intended Audience

- Developer

Delivery Format

✓ Traditional Classroom

Class Duration

- 8 hours
Defensive Programming in PHP for the Web

Description

PHP (PHP: Hypertext Preprocessor) is a powerful and versatile programming language, frequently used in web applications. Building secure PHP apps requires both the platform configuration and secure coding practices. This course is a hands-on, lab-based course which presents risks and solutions and invites the student to edit some sample code to mitigate example risks. Students learn and practice both platform configuration (e.g., php.ini issues) and code-level techniques to find and fix security vulnerabilities in sample code.

This course covers platform features and specifics that can potentially introduce risks like unsafe PHP configuration, null-byte issues, dangerous APIs, cryptography, and dynamic file inclusion issues. After securing the platform, the course teaches standard PHP defensive programming techniques. Topics include safe file system access, session management, authentication, input validation/output encoding, cross-site request forgery, transport security, and injection attacks. For each of these concepts, the course covers common mistakes, subtle semantics that can surprise the unwary, and correct ways to invoke the right APIs. Students leave with a solid understanding of the fundamentals of building secure PHP applications.

Objectives

After successfully completing this course, the student will be able to:

• Comprehend the PHP platform
• Describe the risks affecting PHP applications
• Write secure web applications using PHP
• Design and architect secure PHP applications
• Describe steps to configure PHP applications securely

Intended Audience

• Developer

Delivery Format

✓ Traditional Classroom
✓ Virtual Classroom

Class Duration

• 8 hours
Defensive Programming in Python and Django

Description

Python is a powerful and versatile programming language, frequently used in web applications. Building secure Python applications requires both platform configuration and secure coding practices. This course is a hands-on, lab-based course which presents risks and solutions and invites the student to edit some sample code to mitigate example risks. Students learn and practice both platform configuration and code-level techniques to find and fix security vulnerabilities in sample code.

The course teaches Python developers standard defensive programming techniques framed in a Django web application. Topics include safe file system access, session management, authentication, input validation/output encoding, cross-site request forgery, and injection attacks. For each of these concepts, the course covers common mistakes, subtle semantics that can surprise the unwary, and correct ways to invoke the right APIs. Students leave with a solid understanding of the fundamentals of building secure Python applications.

Objectives

After successfully completing this course, the student will be able to:

- Comprehend the Python platform
- Describe risks affecting Python applications
- Write secure applications using Python
- Design secure Python applications
- Describe steps to configure Python applications securely
Security Testing: Android

Description
The ubiquity of Google’s Android mobile platform and growing threats to mobile applications call for increased vigilance on the part of organizations developing Android software applications. This hands-on one-day course introduces students to the Android application framework and architecture. The course also includes a case study to highlight common misconceptions and security issues of the Android mobile platform. A significant portion of the course is dedicated to lab exercises where students are provided the opportunity to model attacks and perform penetration testing and reverse engineering of an Android application.

Objectives
After successfully completing this course, the student will be able to:

• Understand the Android ecosystem and application architecture
• Understand components of the Android data storage and security models
• Identify specific threats and risks associated with the Android mobile platform
• Perform a hands-on penetration test and reverse engineer an Android application

Intended Audience
• Developer
• QA and Testing
• IS Security Team

Delivery Format
✓ Traditional Classroom
✓ Virtual Classroom

Class Duration
• 8 hours
Security Testing: iOS

Description
The ubiquity of the Apple iOS mobile platform and growing threats to mobile applications call for increased vigilance on the part of organizations developing iOS software applications. This hands-on course introduces students to the iOS application architecture and how to perform security analysis of iOS applications and devices. The course also includes hands-on exercises where students are provided the opportunity to model attacks and perform penetration testing and reverse engineering of an iPhone/iPad application.

Objectives
After successfully completing this course, the student will be able to:

- Understand the iOS ecosystem and application architecture
- Understand components of the iOS data storage and security models
- Identify specific threats and risks associated with the iOS mobile platform
- Perform a hands-on penetration test and reverse engineer an iOS application

Intended Audience
- Developer
- QA and Testing
- IS Security Team

Delivery Format
- Traditional Classroom

Class Duration
- 8 hours
Security Testing: Red Team Workshop

Description

Organizations are continually faced with growing and evolving threats against their digital assets and infrastructure. Red Teaming is a goal-based assessment approach which allows organizations to gain insight into how their security posture is when faced with a real threat. This hands-on, two day course introduces students to the concepts of Red Teaming and how it is different than traditional vulnerability testing. The course will also include guidance for the organization on creating and maintaining their own internal Red Teams. Lastly, students in this course will be introduced to the physical, social, and electronic testing methods which can be utilized during Red Team engagements.

Objectives

After successfully completing this course, the student will be able to:

• Understand the benefits Red Teaming provides
• Create internal documentation to support an internal Red Team
• Understand the concepts for social engineering
• Understand the concepts for physical penetration testing
• Perform hands-on network penetration testing in a controlled lab environment
Security Testing: Web Applications

Description

Web applications are ubiquitous and plentiful. In fact, the web is the *de facto* delivery mechanism for both consumer-grade and business-critical functionality these days. As such, the web is also the most common target for application-level attacks.

This hands-on one-day course describes the goals, processes and risks with Web Security Testing. It introduces students to the basics of Web Application architecture and web security testing including the OWASP Top 10 vulnerabilities. A portion of the course is dedicated to lab exercises where students are provided the opportunity to test for the most commonly occurring web based vulnerabilities.

The course also discusses other aspects of security testing including risk rating of findings, communicating findings to different groups and creating test plans.

Objectives

After successfully completing this course, the student will be able to:

- Comprehend the basics of the HTTP protocol and other web-related technologies and standards
- Use tools for intercepting and modifying HTTP traffic
- Develop test strategies and execute tests to uncover the most important types of web application vulnerabilities
- Communicate findings to developers and management to ensure that relevant findings are properly addressed

Intended Audience

- Developer
- QA and Testing
- IS Security Team

Delivery Format

- Traditional Classroom
- Virtual Classroom

Class Duration

- 8 hours
About Cigital

Cigital is one of the world’s largest application security firms. We go beyond traditional testing services to help organizations find, fix and prevent vulnerabilities in the applications that power their business. Our holistic approach to application security offers a balance of managed services, professional services and products tailored to fit your specific needs. We don’t stop when the test is over. Our experts also provide remediation guidance, program design services, and training that empower you to build and maintain secure applications.

Our proactive methods help clients reduce costs, speed time to market, improve agility to respond to changing business pressures and threats, and focus resources where they are needed most. Cigital’s managed services maximize client flexibility, while reducing operational friction and cost. Cigital gives organizations of any size access to the scale, security expertise, and practices needed to build a successful software security initiative.

For more information, visit us at www.Cigital.com.