+33 (0)1 48 10 09 46
formation@ambient-it.net
140 Avenue Jean Lolive 93500 Pantin - France

Updated on 14/09/2023

Sign up

Cybersecurity conversion training: DevSecOps

10 days (70 hours)

PRESENTATION

Both cybersecurity and DevOps professions are facing a talent shortage. DevSecOps enables companies to accelerate the pace of development by integrating good security practices. This approach uses continuous integration and automation tools to improve application deliverability and security.

This cyber security conversion course will train you in best practices in the field of development security. We'll base our training on the best standards and practices on the market, and disseminate them consistently to your development teams.

Our cybersecurity retraining course alternates theory and practical case studies to ensure that the training is put into practice. An assessment of the acquired knowledge will be required to measure the assimilation of the concepts taught.

OBJECTIVES

- Understand the main concepts, principles and standards for development security
- Develop and test applications that meet compliance and certification requirements

TARGET AUDIENCE

- Developers
- Technical architects
- Project managers

Prerequisites

- Good knowledge of Windows and Linux/UNIX
- Good knowledge of TCP/IP
- Good knowledge of HTTP, JavaScript and development

PROGRAMME OF OUR cybersecurity retraining course: DEVSECOPS

Before you start

- Web security fundamentals
  - Web security: Overview
  - Web server security
  - Web applications and security
  - Risk reduction
  - Web applications, common targets
  - Who are the pirates?
  - Designing secure applications
  - Testing applications
  - OWASP (Open Web Application Security Project)
- .NET & C# Secure Coding Rules
- Java Secure Coding rules with CERT Oracle Java Secure Coding
- Exercise 1.1
  - Course configuration
  - Course web application
  - Microsoft IIS and Apache Tomcat
  - Altovo Mutual .NET application
  - Java application OWASP WebGoat
  - Exercise 1.1: Examine the course Web application
  - Chapter summary
  - Summary questions

Web security fundamentals

- Application security requirements
  - Safety requirements
  - Meeting safety requirements
  - Coding: Best practices
  - Developer's responsibility

Improving web server security

- Input validation
  - Input validation
  - User input
  - Editing non-editable fields
  - Validation strategy
  - Web application trust boundary
  - Disable client-side validation
  - Validation techniques
  - Validation logic
  - Regular expression elements
  - Validate, not just identify
  - .NET request validation
- Exercise 4.10
  - Exercise 4.1: Securing input validation
- Injection faults2
  - Injection: Definition
  - Injection faults
  - Control injection
  - SQL injection
  - SQL injection in action
  - Prevent SQL injections
  - Strongly typed parameterized queries
  - Prevent SQL injections with prepared Java statements
  - Preventing SQL injections using named parameters in .NET
- Exercise 4.25
  - Exercise 4.2: Avoiding SQL injections
- Cross site scripting (XSS)
  - Cross site scripting (XSS)
  - XSS in action
  - Protecting against cross-site scripting
  - HTML entity coding : Java
  - HTML entity coding: .NET
- Exercise 4.35
  - Exercise 4.3: Avoiding XSS vulnerabilities
- Secure access
- Authentication and session management
  - Maintain user sessions
  - Weak session identifiers
  - Intercept session identifiers
  - Session hijacking
  - Strong session identifiers
- Insecure storage and communication
  - Store and send data
  - Secure data storage
  - Do not display confidential data completely
  - Securing communications
- URL access restriction
  - Control access to pages
  - Low URL access restrictions
  - Highly restricted access to URLs
  - Implement URL access restriction
  - Java servlet filters
- Exercise 4.4
  - Exercise 4.4: Restricting access to URLs
- Securing information

- Unsecured direct object reference
  - Unsecured object reference
  - Avoid unsecured object references
  - Falsify a user identity
  - Unvalidated redirections and transfers
  - Preventive measures against redirects and non-validated transfers
- Cross-site request forgery
  - Cross-site request forgery
  - How can cross-site requests be falsified?
  - How cross-site request forgery works
  - Reduce cross-site request forgery vulnerabilities
  - Avoid cross-site request forgery for any application
  - Avoiding cross-site request forgery : Java
  - Avoiding cross-site request forgery: .NET
- Information leakage and error handling
  - Information leakage
  - Error and exception handling
  - Always expect the unexpected
  - Global exception handling with Java
  - Global exception handling with .NET
- Exercise 4.5
  - Exercise 4.5: Information leaks
  - Chapter summary
  - Summary questions

Improving Ajax security

- Ajax principles
  - Ajax: Definition
  - Ajax action
  - How Ajax works
  - Example of XHR code
  - Ajax in action
  - Firebug in action
  - Ajax data transfer
  - JSON (JavaScript Object Notation)
- Identifying increased risk exposure
  - Ajax/Web 2.0 vulnerabilities
  - Increased attack surface
  - Network recognition
  - Safety restrictions XHR
  - Ajax transfer proxy
  - JavaScript on demand
  - Can Ajax hijack a session?
  - Cross-site request forgery with JavaScript and Ajax
  - Stealing sensitive JSON data
  - Avoiding JSON data theft
- Exercise 5.19
  - Exercise 5.1: Preventing data theft with Ajax
  - Chapter summary
  - Summary questions

Web services security

Identify application weaknesses

- Web application analysis tools
  - Action of an application scanner
  - False positives and false negatives
  - IBM AppScan (and/or tool used by Cegid)
  - OWASP ZAP
- Exercise 7.26
  - Exercise 7.2: Automated analysis of a Web application
  - Chapter summary
  - Summary questions
  - Notes

Microsoft .NET & C# Secure Coding Rules

- Best practices for secure development of .NET & C# applications
  - Permission Requests
  - Securing State Data
  - Securing Method Access
  - Securing Wrapper Code
  - Security and Public Read-only Array Fields
  - Securing Exception Handling
  - Security and User Input
  - Security and Remoting Considerations
  - Security and Serialization
  - Security and Race Conditions
  - Security and On-the-Fly Code Generation
  - Dangerous Permissions and Policy Administration
  - Safety and installation issues
  - How do you launch partially reliable code in Sandbox?

Java Secure Coding rules

- Best practices for secure development of Java applications
  - Securing the JVM
    - Java's natural limitations: memory management
    - Bytecode control by the virtual machine
    - Implementing the Security ClassLoader
  - Performance protection
    - Protected execution : SecurityManager, ClassLoader
    - Overloading of access methods: reading, writing, execution, socket opening, connection authorization, etc.
  - Control
    - Reminder on LCDs
    - The java.security.acl package
    - Add entry, check access
  - Obfuscation
    - Principle
    - Obfuscation techniques
    - Commercial solutions
  - JAAS
    - Presentation
    - Operation and implementation
  - CERT Oracle Secure Coding Rules :
    - Input Validation and Data Sanitization (IDS)
    - Declarations and Initialization (DCL)
    - Expressions (EXP)
    - Numeric Types and Operations (NUM)
    - Object Orientation (OBJ) Methods (MET)
    - Exceptional Behavior (ERR)
    - Visibility and Atomicity (VNA)
    - Locking (LCK)
    - Thread APIs (THI)
    - Thread Pools (TPS)
    - Thread-Safety Miscellaneous (TSM)
    - Input Output (FIO)
    - Serialization (SER)
    - Platform Security (SEC)
    - Runtime Environment (ENV)
    - Miscellaneous (MSC)

Microsoft PHP Secure Coding rules

- Best practices for secure development of PHP applications
  - Client-side Security
    - Javascript security
    - AJAX security
    - HTML5 security
  - PHP Security Services
    - Cryptography extensions in PHP
    - Input validation APIs
  - PHP Environment
    - Server Configuration
    - Securing PHP configuration
    - Environment security
    - Hardening
    - Configuration management
  - Advice and Principles
    - Matt Bishop's principles of robust programming
    - The security principles of Saltzer and Schroeder
  - Input validation
    - Input validation concepts
    - Remote PHP code execution
    - MySQL validation errors - beyond SQL Injection
    - Variable scope errors in PHP
    - File uploads, spammers
    - Environment handling
  - Improper use of security features
    - Problems related to the use of security features
    - Insecure randomness
    - Weak PRNGs in PHP
    - Stronger PRNGs we can use in PHP
    - Password management - stored passwords
    - Some usual password management problems
    - Storing credentials for external systems
    - Privacy violation
    - Improper error and exception handling
    - Classification of security flaws
  - Time and State problems
    - Concurrency and threading
    - Concurrency in PHP
    - Preventing file race condition
    - Double submit problem
    - PHP session handling
    - A PHP design flaw - open_basedir race condition
    - Database race condition
    - Denial of service possibilities
    - Hashtable collision attack
    - Classification of security flaws
  - Using Security Testing Tools
    - Web vulnerability scanners
    - SQL injection tools
    - Public database
    - Google hacking
    - Proxy servers and sniffers

Companies concerned

This course is aimed at both individuals and companies, large or small, wishing to train its teams in a new advanced IT technology, or to acquire specific business knowledge or modern methods.

Teaching methods

Practical course: 60% Practical, 40% Theory. Training material distributed in digital format to all participants.

Organization

The course alternates theoretical input from the trainer, supported by examples, with brainstorming sessions and group work.

Validation

At the end of the session, a multiple-choice questionnaire verifies the correct acquisition of skills.

Sanction

A certificate will be issued to each trainee who completes the course.