Fundamentals of Secure Software Development Training

Course: **00084**
Filter: **Beginner**
Duration: **2 days**
Category:: **Software Application Security**
Price: **2500,00 €**

About Course

From proactive requirements to coding and testing, this secure software development training course covers the best practices any software developer needs to avoid opening up their users, customers and organization to attack at the application layer. We teach only constantly updated best practices, and our experts answer your questions live in class. Even with good information security policy and staff, the reality is that software developers are often underserved when it comes to security strategy. If their applications get built without attention to good software security practices, risk gets passed downstream and by the time an incident occurs it’s too late to be proactive. To mitigate these risks, attend this secure programming training course and return to work ready to build higher quality, more robustly protected applications. There are no formal prerequisites for this course.

What you'll learn

- Best practices any software developer needs to avoid opening up their users, customers and organization to attack at the application layer.

Pre-requisites

- None

Curriculum

Module 1: Secure Software Development

- Assets, Threats & Vulnerabilities
- Security Risk Analysis (Bus & Tech)
- Secure Dev Processes (MS, BSI…)
- Defense in Depth
- Approach for this course
- The Context for Secure Development
- Assets to be protected
- Threats Expected
- Security Imperatives (int&external)
- Organization's Risk Appetite
- Security Terminology
- Organizational Security Policy
- Security Roles and Responsibilities
- Security Training for Roles
- Generic Security Goals & Requirements

Module 2: Security Requirements

- Project-Specific Security Terms
- Project-Related Assets & Security Goals
- Product Architecture Analysis
- Use Cases & MisUse/Abuse Cases
- Dataflows with Trust Boundaries
- Product Security Risk Analysis
- Elicit, Categorize, Prioritize SecRqts
- Validate Security Requirements

Module 3: Designing Secure Software

- High-Level Design
- Architectural Risk Analysis
- Analyze Attack Surface
- Threat Modeling

- Trust Boundaries
- Detail-Level Design Sec
- Secure Design Principles
- Use of Security Wrappers
- Input Validation Eliminate Race Objects
- Design Pitfalls
- Validating Design Security
- Pairing Mem Mgmt Functinos
- Exclude User Input from format strings
- Canonicalization
- TOCTOU
- Close Race Windows
- Taint Analysis

Module 4: Writing Secure Code

- Coding
- Developer guidelines & checklists
- Compiler Security Settings (per)
- Tools to use
- Coding Standards (per language)
- Common pitfalls (per language)
- Secure/Safe functions/methods
- Stack Canaries
- Encrypted Pointers
- Memory Initialization
- Function Retrun Checking (e.e. malloc)
- Dereferencing Pointers
- Integer type selection
- Range Checking
- Pre/post checking
- Synchronization Primatives
- Early Verification
- Static Analysis (Code Review w/tools)
- Unit & Dev Team Testing

- Risk-Based Security Testing
- Taint Analysis

Module 5: Testing for Software Security

- Assets to be protected
- Threats Expected
- Security Imperatives (int&external)
- Organization's Risk Appetite
- Static Analysis
- Dynamic Analysis
- Risk-Based Security testing
- Fuzz Testing (Whitebox vs Blackbox)
- Penetration Testing (Whitebox vs Blackbox)
- Attack Surface Review
- Code audits
- Independent Security Review

Module 6: Making Software Development More Secure

- Incident Response Planning
- Final Security Review
- Release Archive
- OS Protections:
- Address Space Layout Randomization
- Non-Executable Stacks
- W^X
- Data Execution Prevention
- /ul>
- Monitoring
- Incident Response
- Penetration Testing
- Process Review
- Getting Started
- Priorities