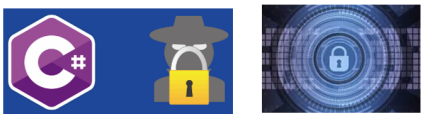
Secure C# Web Development

Duur: 3 dagen **Prijs: € 2299**

Open Rooster

Secure C# Development



Doelgroep Cursus Secure C# Web Development

De cursus Secure C# Web Development is bedoeld voor C# Developers die willen leren hoe je C# web applicaties beschermt met optimale security.

Voorkennis Cursus Secure C# Web Development

Om aan deze cursus te kunnen deelnemen is ervaring met de fundamentals van de C# taal vereist. Affiniteit met security concepten helpt bij de begripsvorming.

Uitvoering Training Secure C# Web Development

De cursus Secure C# Web Development heeft een hands-on karakter. De theorie wordt afgewisseld met praktische oefeningen en besproken aan de hand van presentatie slides.

Certificering cursus Secure C# Web Development

De deelnemers krijgen na het goed doorlopen van de training een certificaat van deelname aan de cursus Secure C# Web Development.

Inhoud Cursus Secure C# Web Development

De cursus Secure C# Web Development behandelt hoe C# web applicaties optimaal kunnen worden beveiligd en de best practices daarin. Aandacht wordt besteed aan de top 10 van de OWASP security vulnerabilities, hun gevolgen en ook hoe security programmatisch kan worden geïmplementeerd.

Intro Security

De cursus Secure C# Web Development gaat van start met bespreking van de voornaamste security risico's zoals vastgesteld door het Open Worldwide Application Security Project (OWASP).

Broken Access Control

Vervolgens wordt ingegaan op security risico's bij access control middels authenticatie en autorisatie. Role Based Access Control, Access Control Lists en de implementatie van Session Management zijn daarbij onderwerpen die aan de orde worden gesteld.

Cryptographic Failures

Ook security problemen met encryptie passeren de revue. Hierbij wordt aandacht besteed aan problemen bij het gebruik van weak keys, het hard coderen van secrets, het onvoldoende verifiëren van signatures en mogelijke side-channels attacks.

Injectie risico's

Een belangrijke security bedreiging vormen de diverse vormen van injectie die op de loer liggen. Hierbij worden onder andere SQL Injection, Cross-Site Scripting en XPath injection besproken evenals de maatregelen ter preventie.

Onjuist Design

Vervolgens wordt ingegaan op security problemen die het gevolg zijn van een verkeerd design zoals het gebrek aan input validatie en onveilig session management. Ook onvoldoende bescherming tegen Cross Site Request Forgery komt dan aan de orde.

Configuratie Fouten

Eveneens wordt besproken hoe fouten in de configuratie tot security problemen kunnen leiden. Diverse voorbeelden hiervan zoals het gebruik van default credentials en weak password policies komen hierbij ter sprake.

Verouderde Componenten

Ook wordt aandacht besteed security risico's die het gevolg zijn van componenten die niet meer up to date zijn, verkeerd geconfigureerde of kwaadaardige packages en cryptografische zwakheden.

Authenticatie fouten

Vervolgens komen veel voorkomende fouten bij authenticatie zoals weak password policies, overly permissive access controls en het gebrek aan multi-factor authenticatie aan bod. En tenslotte worden security gebreken bij monitoring en logging besproken.

Modules Cursus Secure C# Web Development

Module 1 : OWASP Top 10	Module 2 : Broken Access Control	Module 3 : Cryptographic Failures
Secure Coding practices Never trusting Input SQL injection and NoSQL injection OS command injection Broken Authentication Session Fixation Cross Site Scripting Cross Site Request Forgery Insecure Deserialization	Implement Proper Authentication Role Based Access Control (RBAC) Implement Use Session Management Session Timeout Access Control Lists (ACLs) Principle of Least Privilege (PoLP) URL and API Authorization Error Handling Regular Security Testing	Weak Key Generation Insecure Storage of Keys Using Outdated Algorithms Hardcoding Secrets Insufficient Key Management Using Homegrown Cryptography Failure to Verify Signatures Side-Channel Attacks Lack of Forward Secrecy
Module 4 : Injection Flaws	**Module 5 : Insecure Design**	**Module 6 : Configuration Failures**
SQL Injection (SQLi) Cross-Site Scripting (XSS) Command Injection XML Injection LDAP Injection XPath Injection: SSI Injection (Server-Side Includes) Object Injection Template Injection CRLF Injection	Inadequate Authentication Lack of Input Validation Excessive Data Exposure Insecure Session Management Hardcoding Secrets Insufficient Authorization Insecure Data Storage Cross-Site Request Forgery (CSRF) Insecure File Uploads Improper Error Handling	Improper Access Control Unsecured APIs Open Database Ports Default Credentials Unused or Unnecessary Features Weak Password Policies Missing Security Updates Improper Access Permissions Insecure Session Management Excessive Error Detail
Module 7 : Outdated Components	**Module 8 : Authentication Mistakes**	**Module 9 : Logging and Monitoring**
Known Vulnerabilities Malicious Packages License Compliance Misconfigured Packages Dependency Chains Cryptographic Weaknesses Data Privacy and Compliance Resource Exhaustion Insecure Configuration Defaults	Weak Password Policies No Account Lockout Mechanism Inadequate Password Storage Hardcoding Credentials Lack of Multi-Factor Authentication Missing CAPTCHA or Rate Limiting Overly Permissive Access Controls Insecure Remember-Me Functionality Hostname Verification	Insufficient Logging Lack of Centralized Logging Logging Sensitive Information Inadequate Log Retention Unencrypted Logging Insufficient Access Controls Failure to Monitor Logs in Real-Time No Alerts or Notifications Ignoring Anomalous Activity