Security in C# .NET Development

Duur: 3 dagen **Prijs: € 2250**

Open Rooster


Secure C# .NET Development


Doelgroep Cursus Security in C# .NET Development

De cursus Security in C# .NET Development is bedoeld voor C# Developers die willen leren hoe je C# applicaties kunt beschermen tegen de vele risico's qua veiligheid.

Voorkennis Cursus Security in C# .NET Development

Om aan deze cursus te kunnen deelnemen kennis van en ervaring met C# en het .NET Platform vereist.

Uitvoering Training Security in C# .NET Development

De cursus Secure C# .NET Development is een hands-on cursus. Theorie uitleg aan de hand van demos en presentaties en praktijk aan de hand van exercises wisselen elkaar af.

Certificaat cursus Security in C# .NET Development

De deelnemers krijgen na het goed doorlopen van de training een certificaat van deelname aan de cursus Security in C# .NET Development.

Inhoud Cursus Security in C# .NET Development

De cursus Security in C# .NET Development voorziet C# developers van de essentiële kennis en praktische vaardigheden om security problemen met web applicaties effectief aan te pakken. Veel voorkomende security problemen, zoals beschreven in de OWASP Top Tien, worden behandeld evenals best practices voor het omgaan met security uitdagingen in .NET C# code.

Intro Secure Coding

De cursus Security in C# .NET Development gaat van start met een overzicht van het landschap van applicatie security, waaronder veel voorkomende attack vectors en mogelijke risico's bij het ontwikkelen van C# code.

Broken Access Control

Vervolgens wordt ingegaan het voorkomen van kwetsbaarheden als gevolg van broken access control. Daarbij wordt aandacht besteed aan Role Based Access Control (RABC), de juiste implementatie van session management en Access Control Lists.

Cryptographic Failures

Ook cryptografische zwakheden zoals gebrekkige encryptie algoritmen en onjuist gebruik van cryptografische functies komen aan bod.

Injection Flaws

Dan wordt worden de gevaren van injectie belicht zoals SQL injection en cross-site scripting (XSS) en cross-site request forgery (CSRF). Ook wordt dan ingegaan op secure coding practices om injectie te voorkomen zoals input validation, output encoding en geparametriseerde queries.

Insecure Design

Ook onveilig design komt aan de orde met gebrekkige input validatie, onjuiste error handling en onveilige authenticatie.

Misconfiguration Failures

En wordt belicht hoe configuratie fouten kunnen leiden tot security risks, zoals het gebruik van standaard instellingen en het onvoldoende beschermen van gevoelige data.

NuGet Packages

Eveneens wordt ingegaan op de risico's van externe NuGet packages, hoe NuGet packages te beoordelen en best practices voor het veilig integreren van NuGet packages.

Logging and Monitoring

Tenslotte komt aan de orde hoe logging en monitoring de security van C# applicaties kunnen verbeteren. Aandacht wordt besteed aan het belang van logging en monitoring voor het detecteren en reageren op security incidenten.

Modules Cursus Security in C# .NET Development

Module 1 : Intro Secure Coding	Module 2 : Broken Access Control	Module 3 : Cryptographic Failures
Secure Coding practices Never trusting Input SQL injection and NoSQL injection OS command injection Session Fixation Cross Site Scripting and CSRF Sensitive Data Exposure Insecure Deserialization Security Misconfiguration Using Unsafe Components	Implement Proper Authentication Broken 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	Sensitive Data Exposure Weak Key Generation Insecure Storage of Keys Using Outdated Algorithms Hardcoding Secrets Insufficient Key Management Avoid Homegrown Cryptography Verify Signatures Side-Channel Attacks Lack of Forward Secrecy
Module 4 : Injection Flaws	**Module 5 : Insecure Design**	**Module 6 : Misconfiguration Failures**
SQL Injection (SQLi) Cross-Site Scripting (XSS) Command Injection XML Injection LDAP Injection XPath Injection SSI Injection Object Injection Template Injection CRLF Injection	Inadequate Authentication Inadequate Authorization Lack of Input Validation Excessive Data Exposure Insecure Session Management Hardcoding Secrets Insufficient Logging and Monitoring Insecure Data Storage Cross-Site Request Forgery Improper Error Handling	Improper Access Control Unsecured APIs Open Database Ports Default Credentials Unused or Unnecessary Features Weak Password Policies Missing Security Updates Improper File Permissions Insecure Session Management Excessive Error Detail
Module 7 : NuGet Packages	**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 (MFA) Insufficient Session Management Missing CAPTCHA or Rate Limiting Overly Permissive Access Controls Improper Handling Forgotten Passwords	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