SC470 Secure Coding

Kurzbeschreibung:

Secure Coding - d.h. Computersoftware so zu entwickeln, dass diese vor dem Eindringen von Sicherheitslücken geschützt ist! Defekte, Bugs und Fehler (Logik, Out of Bound, Systemebene, Workflow, Funktion, etc.) sind die Hauptursache für ausgenutzte Software-Schwachstellen. Doch die meisten Schwachstellen sind auf eine relativ kleine Anzahl häufiger Softwareprogrammierfehler zurückzuführen. Wie Sie diese vermeiden können und was wichtig zu beachten ist, lernen Sie in diesem 4-Tage-Training.

Zahlreiche **praktische Übungen** helfen Ihnen dabei, das erworbene Wissen direkt anzuwenden und zu festigen.

Zielgruppe:

Das Training **SC470 Secure Coding** ist ideal geeignet für:

- Software Entwickler (Web)
- Software Architekten
- Tester
- Cloud Architekten
- DevOps

Voraussetzungen:

Um den Kursinhalten und dem Lerntempo im Workshop **SC470 Secure Coding** gut folgen zu können, sind allgemeine Programmierkenntnisse und Grundlagenkenntnisse der Webentwicklung nötig.

Sonstiges:

Dauer: 4 Tage

Preis: 2790 Euro plus Mwst.

Ziele:

Der Kurs **Secure Coding (SC470)** bietet:

- Vermittlung des Security-Gedankens über den gesamten Lebenszyklus eines Software-Produkts
- Erkennen und Vermeiden von Schwachstellen bei der Software-Entwicklung
- Aufzeigen von häufig gemachten sicherheitsrelevanten Fehlern
- Vermittlung von Best Practices zur Vermeidung sicherheitsrelevanter Fehler

Im Workshop **SC470 Secure Coding** werden Grundlagen für die Entwicklung von sicherer Software sowie das Aufzeigen und das Vermeiden von Sicherheitslücken beim Software-Development vermittelt. Sie erhalten einen guten Ein- und Überblick in Secure Coding.

Inhalte/Agenda:

- **Intro: Warum sichere Entwicklung wichtig ist?**
  - Vorstellung von großen Sicherheitslücken und Data Breaches
  - Risiken für Unternehmen

- **Einführung in Risk Management**
  - Identify risks
  - Classify risks
  - Plan
  - Monitor
  - Action
  - Communicate

- **Essentials Do's and Don'ts in der Softwareentwicklung**

- **Software Development Lifecycle (SDL)**
  - Secure Software Concepts
  - Secure Software Requirements
  - Secure Software Design
  - Secure Software Implementation/Programming
  - Secure Software Testing
  - Software Lifecycle Management
  - Software Deployment, Operations and Maintenance
  - Supply Chain and Software Acquisition

- **Source Code Review**
  - Best Practices im Source Code Review

- **Web and Embedded Developer Threats and Vulnerabilities**
  - OWASP Top 10
    - ◊ Kurzeinführung in die häufigsten Klassen von Vulnerabilites
  - Introduction to Web Interception Proxies
    - ◊ Kurze Einführung in Burp Suite in der vorinstallierten Umgebung
  - Tools of the trade (sqlmap, dirbuster,)
    - ◊ Was können bestimmte "Hacking"-Tools und wie nutze ich sie (erforderlich für "Hands on"-Teile)
  - Hands on Hacking (Insecure Example Application)
    - ◊ Juice Shop Introduction, Challenges und CTF (teilweise frei von den Teilnehmern gestaltbar aber auch geführte Angriffe mit dem Instructor)
  - Finding Bugs in Open Source Applications (Real World Hacking)
    - ◊ 0day Hunting in Open Source Application
  - Enumeration techniques (DNS, application mapping,)
    - ◊ Möglichst viel über eine Anwendung herausfinden
    - ◊ "Geheime" Anwendungsteile finden
  - OSINT (Open Source Intelligence) Using public information to attack software
    - ◊ Beispiele von geleakten Secrets (github commits), Repos auf Webservern, Fragen in Foren und vieles mehr
  - Using insecure dependencies to attack secure software
    - ◊ Third Party Module/Libraries
  - Supply Chain attacks in the wild
    - ◊ Manipulation von OpenSource Projekten um sichere Anwendungen anzugreifen, demonstriert anhand von Beispielen aus der echten Welt

- **Penetrationstest**
  - Vorstellung einiger anonymisierter Penetrationtest Reports
  - Analyse der gefunden Lücken und Empfehlungen für Fixes

- **Web Developer - Prevention**
  - Best Practices for secure web applications (Abhängig von Programmiersprache)
  - Static code analysis
  - Dynamic code analysis
  - Spotting bugs in application logic
  - Third Party Libraries and dependency management
  - Keeping the impact low

- **Viele praktische Übungen zu den einzelnen Modulen.**