

Code responsibly with generative AI in Java (CRWGAIJ)

ID CRWGAIJ **Preis CHF 2'250.– (exkl. MwSt.)** **Dauer 3 Tage**

Zielgruppe

Java-Entwickler, die Copilot oder andere GenAI-Tools verwenden

Voraussetzungen

OWASP, SEI CERT, CWE und Fortify Taxonomien

Kursziele

- Die Grundlagen der verantwortungsvollen KI verstehen
- Vertrautmachen mit grundlegenden Konzepten der Cybersicherheit
- Verstehen, wie Kryptographie die Sicherheit unterstützt
- Lernen, wie man kryptografische APIs in Java richtig verwendet
- Verständnis von Sicherheitsfragen bei Webanwendungen
- Detaillierte Analyse der OWASP Top Ten Elemente
- Die Sicherheit von Webanwendungen im Kontext von Java
- Über die niedrig hängenden Früchte hinausgehen
- Verwaltung von Schwachstellen in Komponenten von Drittanbietern
- All dies im Kontext von GitHub Copilot

in Apache OFBiz

- Verwirrter Abgeordneter
- Unsichere direkte Objektreferenz (IDOR)
- Pfadüberquerung
- Übung - Unsichere direkte Objektreferenz
- Bewährte Verfahren zur Pfadüberquerung
- Übung - Experimentieren mit der Pfadverfolgung in Copilot
- Berechtigungsumgehung durch benutzergesteuerte Schlüssel
- Fallbeispiel - Fernübernahme von NEXX Garagentoren und Alarmanlagen
- Labor - Horizontale Genehmigung (Erkundung mit Copilot)
- Hochladen von Dateien
 - Uneingeschränkter Datei-Upload
 - Bewährte Praktiken
 - Übung - Uneingeschränkter Datei-Upload (Erkundung mit Copilot)
 - Fallstudie - Sicherheitslücke beim Hochladen von Dateien in Netflix Genie

A02 - Kryptographische Ausfälle

- Kryptographie für Entwickler
- Grundlagen der Kryptographie
- Die kryptografische Java-Architektur (JCA) in Kürze
- Elementare Algorithmen
- Hashing
 - Grundlagen des Hashings
 - Hashing in Java
 - Übung - Hashing in JCA (Erkundung mit Copilot)
- Erzeugung von Zufallszahlen
 - Pseudo-Zufallszahlengeneratoren (PRNGs)
 - Kryptografisch sichere PRNGs
 - Schwache und starke PRNGs in Java

Kursinhalt

Tag 1

Verantwortungsbewusst kodieren mit GenAI

- Was ist verantwortungsvolle KI?
- Was ist Sicherheit?
- Bedrohung und Risiko
- Arten von Cybersicherheitsbedrohungen - die CIA-Triade
- Folgen von unsicherer Software
- Sicherheit und verantwortungsvolle KI in der Softwareentwicklung
- GenAI-Werkzeuge für die Kodierung: Copilot, Codeium und andere
- Die OWASP Top Ten aus der Sicht von Copilot
 - Die OWASP Top Ten 2021
 - A01 - Defekte Zugangskontrolle
 - Grundlagen der Zugangskontrolle
 - Fallstudie - Fehlerhaftes authn/authz

Code responsibly with generative AI in Java (CRWGAIJ)

- Übung - Verwendung von Zufallszahlen in Java (Erkundung mit Copilot)
 - Fallstudie - Equifax-Kontosperrung
 - Schutz der Vertraulichkeit
 - Symmetrische Verschlüsselung
 - Blockchiffren
 - Betriebsarten
 - Betriebsarten und IV - bewährte Verfahren
 - Symmetrische Verschlüsselung in Java
 - Symmetrische Verschlüsselung in Java mit Streams
 - Übung - Symmetrische Verschlüsselung in JCA (Erkundung mit Copilot)
 - Asymmetrische Verschlüsselung
 - Kombination von symmetrischen und asymmetrischen Algorithmen
 - Schlüsselaustausch und Vereinbarung
 - Austausch von Schlüsseln
 - Diffie-Hellman-Schlüsselvereinbarungsalgorithmus
 - Die wichtigsten Fallstricke beim Austausch und bewährte Verfahren
- Injektion
- Bewährte Praktiken zur SQL-Einschleusung
 - Überprüfung der Eingaben
 - Parametrisierte Abfragen
 - Übung - Verwendung vorbereiteter Erklärungen
 - Übung - Experimentieren mit SQL-Injection in Copilot
 - Datenbankverteidigung in der Tiefe
 - Fallstudie - SQL-Injektion in Fortra FileCatalyst
 - Code-Einspritzung
 - OS-Befehlsinjektion
 - Bewährte Praktiken der OS-Befehlsinjektion
 - Verwendung von Runtime.exec()
 - Fallstudie - Shellshock
 - Labor - Shellshock
 - Fallstudie - Befehlsinjektion in VMware Aria
 - HTML-Injektion - Cross-Site-Scripting (XSS)
 - Grundlagen des Cross-Site-Scripting
 - Cross-Site-Scripting-Typen
 - Anhaltendes Cross-Site-Scripting
 - Reflektiertes Cross-Site-Scripting
 - Client-seitiges (DOM-basiertes) Cross-Site-Scripting
 - Übung - Gespeicherte XSS
 - Labor - Reflektiertes XSS
 - Bewährte Praktiken zum Schutz vor XSS
 - Schutzprinzipien - Flucht
 - XSS-Schutz-APIs in Java
 - Labor - XSS fix / gespeichert (Erkundung mit Copilot)
 - Labor - XSS-Behebung / reflektiert (Erkundung mit Copilot)
 - Zusätzliche Schutzschichten - Verteidigung in der Tiefe
 - Fallstudie - XSS-Schwachstellen in DrayTek Vigor-Routern
 - A04 - Unsicheres Design
 - Das STRIDE-Modell der Bedrohungen
 - Sichere Gestaltungsprinzipien von Saltzer und Schroeder
 - Wirtschaftlichkeit des Mechanismus
 - Ausfallsichere Voreinstellungen
 - Vollständige Mediation
 - Open design
 - Trennung der Privilegien

Tag 2

Die OWASP Top Ten aus der Sicht von Copilot

- A03 - Injektion
 - Injektionsprinzipien
 - Injektionsangriffe
 - SQL-Einschleusung
 - Grundlagen der SQL-Injektion
 - Übung - SQL-Injektion
 - Angriffsmethoden
 - Inhaltsbasierte blinde SQL-Injektion
 - Zeitbasierte blinde SQL-

- A04 - Unsicheres Design
 - Das STRIDE-Modell der Bedrohungen
 - Sichere Gestaltungsprinzipien von Saltzer und Schroeder
 - Wirtschaftlichkeit des Mechanismus
 - Ausfallsichere Voreinstellungen
 - Vollständige Mediation
 - Open design
 - Trennung der Privilegien

Code responsibly with generative AI in Java (CRWGAIJ)

- Geringstes Privileg
- Am wenigsten verbreiteter Mechanismus
- Psychologische Akzeptanz
- Client-seitige Sicherheit
- Rahmen-Sandboxing
- Cross-Frame-Scripting-Angriffe (XFS)
- Labor - Clickjacking
- Clickjacking geht über die Entführung eines Klicks hinaus
- Bewährte Praktiken zum Schutz vor Clickjacking
- Übung - Verwendung von CSP zur Verhinderung von Clickjacking (Erkundung mit Copilot)
- A05 - Fehlkonfiguration der Sicherheit
 - Grundsätze der Konfiguration
 - XML-Entitäten
 - DTD und die Entitäten
 - Erweiterung der Entität
 - Angriff auf externe Entitäten (XXE)
 - Einbeziehung von Dateien mit externen Stellen
 - Server-Side Request Forgery mit externen Entitäten
 - Labor - Angriff einer externen Einheit
 - Verhinderung von XXE
 - Labor - Verbot der DTD
 - Fallstudie - XXE-Schwachstelle in Ivanti-Produkten
 - Labor - Experimentieren mit XXE in Copilot

Tag 3

Die OWASP Top Ten aus der Sicht von Copilot

- A06 - Anfällige und veraltete Komponenten
 - Verwendung anfälliger Komponenten
 - Import von nicht vertrauenswürdigen Funktionen
 - Fallstudie - Der Angriff auf die Lieferkette von Polyfill.io
 - Management von Schwachstellen
 - Übung - Auffinden von Schwachstellen in Komponenten von Drittanbietern
 - Sicherheit von KI-generiertem Code
 - Praktische Angriffe auf Tools zur Codegenerierung
 - Abhängigkeits-Halluzination durch generative KI
 - Fallstudie - Eine Geschichte der Schwächen von GitHub Copilot (bis

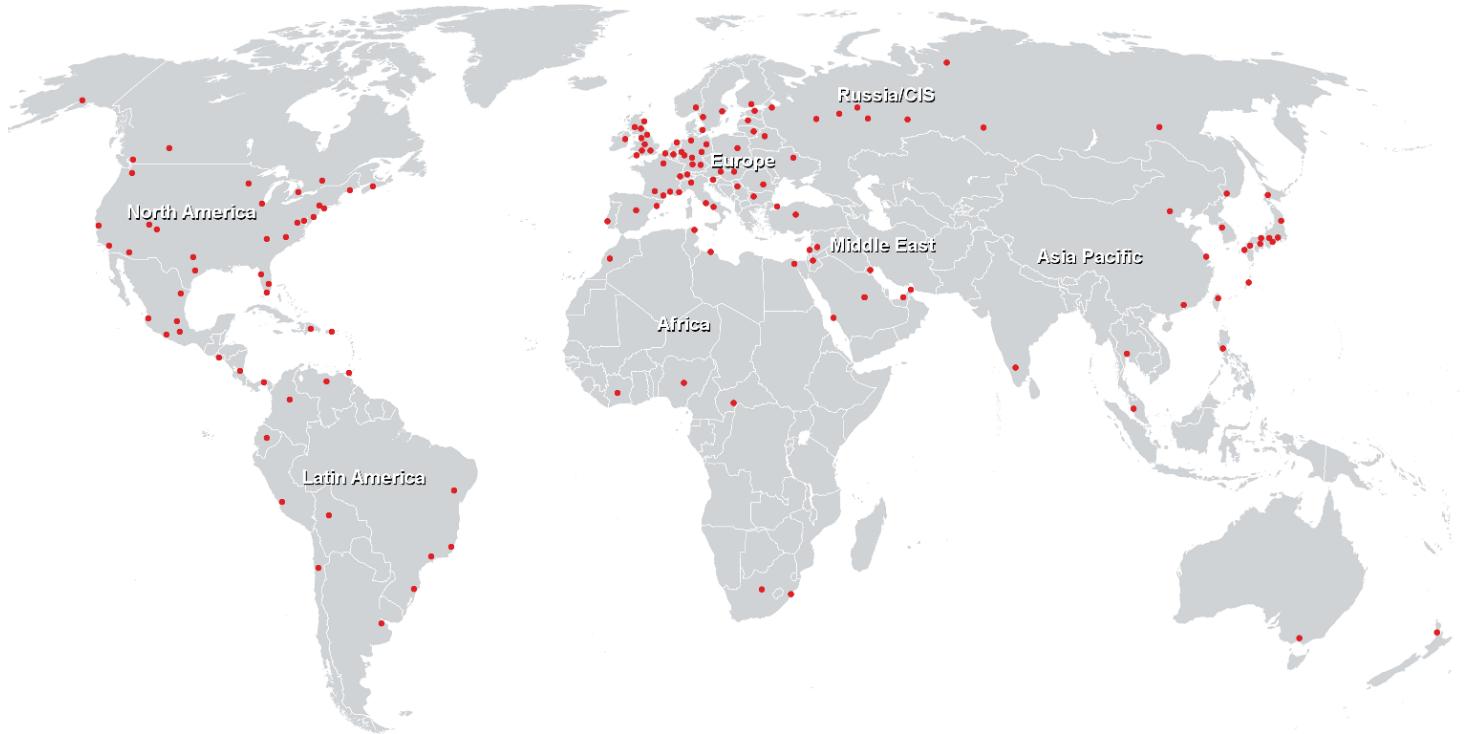
- Mitte 2024)
- A07 - Fehler bei der Identifizierung und Authentifizierung
 - Authentifizierung
 - Grundlagen der Authentifizierung
 - Multi-Faktor-Authentifizierung (MFA)
 - Fallstudie - Der InfinityGauntlet-Angriff
 - Passwortverwaltung
 - Verwaltung eingehender Passwörter
 - Speichern von Kontopasswörtern
 - Labor - Reicht das Hashing von Passwörtern aus?
 - Wörterbuchangriffe und Brute-Forcing
 - Salzen
 - Adaptive Hash-Funktionen für die Passwortspeicherung
 - Übung - Verwendung adaptiver Hash-Funktionen in JCA
 - Übung - Verwendung adaptiver Hash-Funktionen in Copilot
 - Passwort-Politik
 - NIST-Authentifikator-Anforderungen für gespeicherte Geheimnisse
 - A08 - Fehler in der Software und Datenintegrität
 - Schutz der Integrität
 - Nachrichten-Authentifizierungs-Code (MAC)
 - MAC-Berechnung in Java
 - Übung - MAC-Berechnung in JCA
 - Digitale Unterschrift
 - Elliptische Kurven Kryptographie
 - ECC-Grundlagen
 - Digitale Unterschrift mit ECC
 - Digitale Unterschrift in Java
 - Übung - Digitale Signatur mit ECDSA in JCA
 - Integrität der Subressource
 - JavaScript importieren
 - Übung - JavaScript importieren (mit Copilot erkunden)

Code responsibly with generative AI in Java (CRWGAIJ)

- Fallstudie - Die Datenschutzverletzung bei British Airways
- Unsichere Deserialisierung
 - Herausforderungen bei Serialisierung und Deserialisierung
 - Integrität - Deserialisierung nicht vertrauenswürdiger Datenströme
 - Integrität - bewährte Verfahren zur Deserialisierung
 - Vorausschauende Deserialisierung
 - Eigenschaftsorientiertes Programmieren (POP)
 - Erstellen einer POP-Nutzlast
 - Übung - Erstellen einer POP-Nutzlast
 - Übung - Verwendung der POP-Nutzlast
 - Fallstudie - Deserialisierungs-RCEs in NextGen Mirth Connect
- A09 - Fehler bei der Sicherheitsprotokollierung und -überwachung
 - Grundsätze der Protokollierung und Überwachung
 - Rundholz fälschen
 - Protokollfälschung - bewährte Verfahren
 - Fallstudie - Log-Interpolation in log4j
 - Fallstudie - Die Log4Shell-Schwachstelle (CVE-2021-44228)
 - Fallstudie - Log4Shell-Folgemassnahmen (CVE-2021-45046, CVE-2021-45105)
 - Übung - Log4Shell
- A10 - Server-seitige Anforderungsfälschung (SSRF)
 - Server-seitige Anforderungsfälschung (SSRF)
 - Fallbeispiel - SSRF in Ivanti Connect Secure
- Einpacken
 - Grundsätze der sicheren Kodierung
 - Grundsätze der robusten Programmierung von Matt Bishop
 - Und was nun?
 - Quellen zur Softwaresicherheit und weiterführende Literatur
- Java-Ressourcen
- Verantwortungsvolle KI-Prinzipien in der Softwareentwicklung
- Generative AI - Ressourcen und zusätzliche Anleitungen

Code responsibly with generative AI in Java (CRWGAIJ)

Weltweite Trainingscenter



Fast Lane Institute for Knowledge Transfer (Switzerland) AG

Husacherstrasse 3
CH-8304 Wallisellen
Tel. +41 44 832 50 80

info@flane.ch, <https://www.flane.ch>