Code responsibly with generative AI in Python (CRWGAIP)

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

Zielgruppe

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

Voraussetzungen

Allgemeine Python- und Webentwicklung

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 Python richtig verwendet
- Verständnis von Sicherheitsfragen bei Webanwendungen
- Detaillierte Analyse der OWASP Top Ten Elemente
- Die Sicherheit von Webanwendungen im Kontext von Python
- Über die niedrig hängenden Früchte hinausgehen
- Verwaltung von Schwachstellen in Komponenten von Drittanbietern
- All dies im Kontext von GitHub Copilot

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
            - Keine Beschränkung des URL-Zugriffs
            - 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)
        - A02 - Kryptographische Ausfälle
            - Kryptographie für Entwickler
            - Grundlagen der Kryptographie
            - Kryptographie in Python
            - Elementare Algorithmen
            - Hashing
                - Grundlagen des Hashings
                - Hashing in Python
                - Übung - Hashing in Python (Erkundung mit Copilot)
            - Erzeugung von Zufallszahlen
                - Pseudo-Zufallszahlengeneratoren (PRNGs)
                - Kryptografisch sichere PRNGs
                - Schwache PRNGs
                - Verwendung von Zufallszahlen
                - Übung - Verwendung von

Zufallszahlen in Python (Erkundung mit Copilot)
- Übung - Sichere PRNG-Verwendung in Copilot

- Schutz der Vertraulichkeit
  - Symmetrische Verschlüsselung
    - Blockchiffren
    - Betriebsarten
    - Betriebsarten und IV - bewährte Verfahren
    - Symmetrische Verschlüsselung in Python
    - Übung - Symmetrische Verschlüsselung in Python (Erkundung mit Copilot)
  - Asymmetrische Verschlüsselung
  - Kombination von symmetrischen und asymmetrischen Algorithmen

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-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-Injection gegen US-Flughafensicherheit
    - Code-Einspritzung
      - Code-Injektion über input()
      - OS-Befehlsinjektion
      - Übung - Befehlsinjektion
      - Bewährte Praktiken zur Injektion von OS-Befehlen
      - Vermeidung von Befehlseingaben mit den richtigen APIs
      - Übung - Bewährte Praktiken der Befehlseingabe
      - Übung - Experimentieren mit der Befehlsinjektion in Copilot
      - Fallstudie - Shellshock
      - Labor - Shellshock
      - Fallstudie - Befehlsinjektion in Ivanti-Sicherheitsanwendungen
    - 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
      - Fallstudie - XSS zu RCE in Teltonika-Routern
      - Bewährte Praktiken zum Schutz vor XSS
      - Schutzprinzipien - Flucht
      - XSS-Schutz-APIs in Python
      - XSS-Schutz in Jinja2
      - Lab - XSS fix / gespeichert (Erkundung mit Copilot)
      - Labor - XSS-Behebung / reflektiert (Erkundung mit Copilot)
      - 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 Standardwerte
      - Vollständige Mediation
      - Open design
      - Trennung der Privilegien
      - Geringstes Privileg
      - Am wenigsten verbreiteter Mechanismus
      - Psychologische Akzeptanz
    - Client-seitige Sicherheit
      - Politik der gleichen Herkunft
      - Einfacher Antrag

- Preflight-Anfrage
- Ursprungsübergreifende Ressourcennutzung (CORS)
- Labor - Demo zur Politik des gleichen Herkunftslandes
- 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)

Tag 3

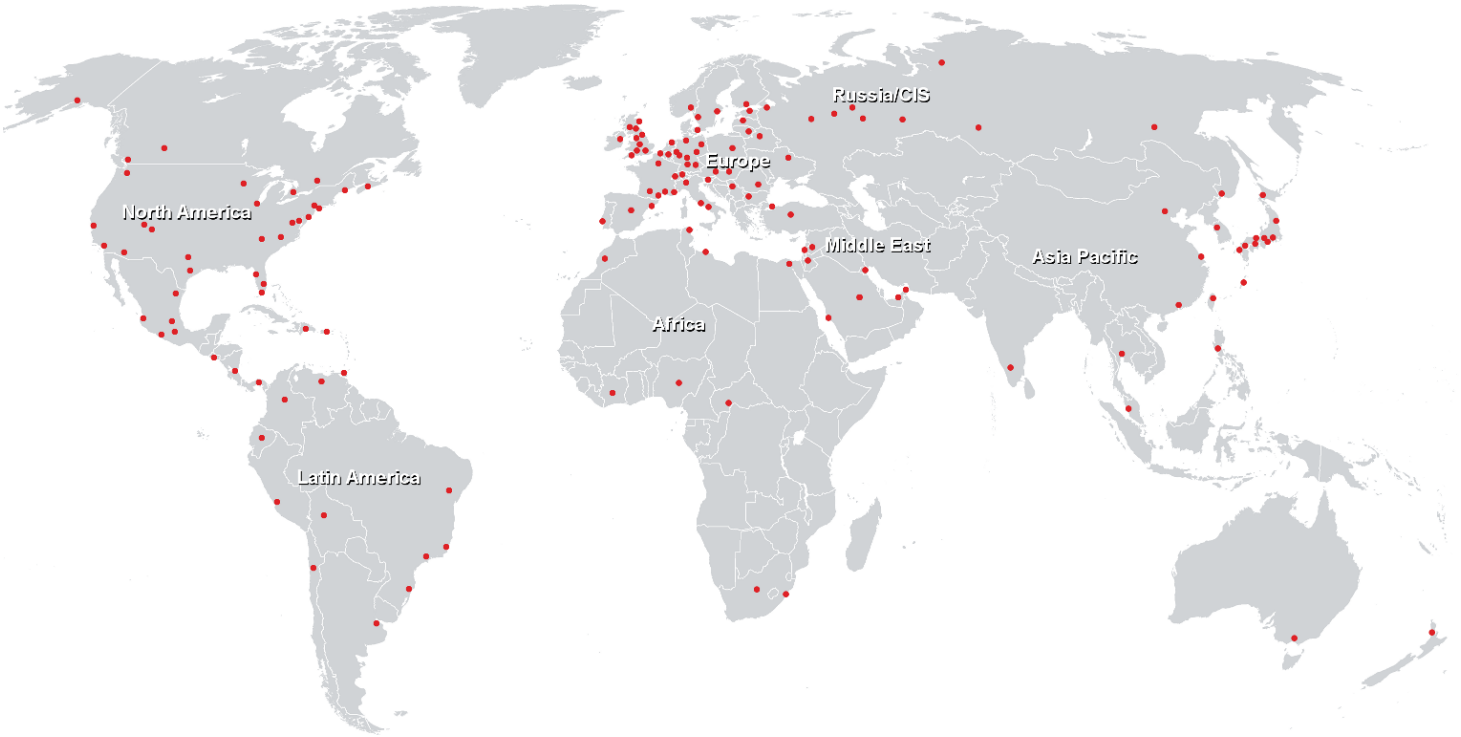
Die OWASP Top Ten aus der Sicht von Copilot

- A05 - Fehlkonfiguration der Sicherheit
  - Grundsätze der Konfiguration
  - Server-Fehlkonfiguration
  - Bewährte Verfahren für die Python-Konfiguration
  - Flask konfigurieren
  - Cookie-Sicherheit
    - Cookie-Attribute
  - 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
- A06 - Anfällige und veraltete Komponenten
  - Verwendung anfälliger Komponenten
  - Import von nicht vertrauenswürdigen Funktionen
  - Bösartige Pakete in Python
  - 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
    - Zeitbasierte Einmal-Passwörter (TOTP)
  - Passwortverwaltung
    - Verwaltung eingehender Passwörter
    - Speichern von Kontopasswörtern
    - Passwort im Transit
    - Labor - Reicht das Hashing von Passwörtern aus?
    - Wörterbuchangriffe und Brute-Forcing
    - Salzen
    - Adaptive Hash-Funktionen für die Passwortspeicherung
    - Übung - Verwendung adaptiver Hash-Funktionen in Python
    - Übung - Verwendung adaptiver Hash-Funktionen in Copilot
    - Passwort-Politik
    - NIST-Authentifikator-Anforderungen für gespeicherte Geheimnisse
    - Migration der Passwort-Datenbank
- A08 - Fehler in der Software und Datenintegrität
  - Schutz der Integrität
    - Nachrichten-Authentifizierungs-Code (MAC)
    - HMAC-Berechnung in Python
    - Übung - MAC-Berechnung in Python
  - Digitale Unterschrift
    - Digitale Unterschrift in Python
  - Integrität der Subressource
    - JavaScript importieren
    - Übung - JavaScript importieren (mit Copilot erkunden)
    - Fallstudie - Die Datenschutzverletzung bei British Airways
- 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
- Python-Ressourcen
- Verantwortungsvolle KI-Prinzipien in der Softwareentwicklung
- Generative AI - Ressourcen und zusätzliche Anleitungen

Code responsibly with generative AI in Python (CRWGAIP)

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