Johner Institut

INHOUSE-WORKSHOP IT SECURITY FÜR SOFTWARE-ENTWICKLER:INNEN VON MEDIZINPRODUKTEN UND IVD (SECHSTÄGIG)

Erlangen Sie als Software-Entwicklungsteam die erforderlichen Kompetenzen zur Entwicklung sicherer Software-Produkte – spezifisch für Ihre Programmiersprache und Plattform

Der maßgeschneiderte Inhouse-Workshop des Johner Instituts vermittelt Ihrem Software-Entwicklungsteam genau die Kompetenzen, die es für die Entwicklung sicherer Medizinprodukte und IVD benötigt.

Der Fokus liegt auf der praktischen Anwendung: von konkreten Coding Guidelines bis hin zu Tools für die Entwicklung. Die Inhalte sind speziell auf die Anforderungen der MDR, IVDR und IEC 81001-5-1 abgestimmt.

Das Besondere: Die Weiterbildung wird spezifisch für Ihre Programmiersprache und Plattform angeboten. So können Sie das Gelernte sofort in der Praxis umsetzen und von Anfang an sicheren, gesetzeskonformen Code entwickeln.

Lerninhalte

- Regulatorische Anforderungen an die Cybersecurity (z. B. aus der IEC 81001-5-1)
- Grundlagen der Cybersicherheit
- Schwachstellen in der Speicherverwaltung (z. B. Assembly Basics und Calling Conventions, Buffer Overflow)
- Härtung der Speicherverwaltung (z. B. Compiler Settings, Run-time Protection, ASLR)
- Häufige Sicherheitslücken in der Software (z. B. Authentifizierung, Autorisierung, Dateizugriffs-kontrolle, Passwortmanagement, Input-Validierung z. B. gegen Injections)
- Arbeiten mit Tools und Coding Guidelines

Lernziele

Nach dem Workshop

- verstehen Sie, was Secure Coding in Ihrem konkreten Kontext bedeutet und welchen Rolle Coding Guidelines dabei spielen.
- kennen und verstehen Sie die für Ihren Kontext relevanten Coding Guidelines.
- können Sie die Coding Guidelines in Ihrem Kontext anwenden, Schwachstellen im eigenen Code vermeiden, zuverlässig vorhandene Schwachstellen identifizieren und damit sichere Medizinprodukte entwickeln.
- können Sie den Einsatz der und die Konformität mit den Coding Guidelines nachweisen.
- sind Sie von der Sinnhaftigkeit der Coding Guidelines überzeugt und haben den Wunsch, sich eigenständig tiefer in das Thema einzuarbeiten.

Zielgruppe

Der Workshop richtet sich an Software-Entwickler:innen bei Medizinprodukte- und IVD-Herstellern, die:

- Software-Code entwickeln, der die gesetzlichen Anforderungen an die IT Security erfüllen muss.
- praktische Anleitungen für die Umsetzung von Secure Coding suchen.
- ihre Kompetenzen effizient und zielgerichtet erweitern möchten.
- einen anerkannten Nachweis ihrer Security-Kompetenzen benötigen.

Voraussetzung für die Teilnahme

Der Workshop wendet sich spezifisch an Software-Entwickler:innen bei Herstellern von Medizinprodukten und IVD. Er baut auf deren Erfahrungshintergrund auf, geht runter bis auf Code-Ebene und verwendet die Sprache und Terminologien von Programmierenden.

Ablauf (Agenda)

TAG	INHALT
Tag 1 wahlweise DE oder EN)	• **Begrüßung und Erwartungen:** Einstieg und Zielsetzung des Workshops • **Einführung und regulatorischer Rahmen:** Grundlagen und Anforderungen im Kontext relevanter Normen und Regularien • **Geteilte Verantwortung und Datenschutz:** Verantwortlichkeiten im Sicherheitskontext und Datenschutzanforderungen • **Lebenszyklusprozess nach IEC 81001-5-1:** Einführung in die Grundlagen des Lebenszyklusprozesses • **Risikomanagement:** Identifizierung, Bewertung und Planung von Maßnahmen zur Risikoabwehr • **Warum Planung entscheidend ist:** Fokus auf Sicherheitsplanung und deren Bedeutung für Produktsicherheit • **Abschluss und Zusammenfassung:** Offene Fragen, Rückblick, und Diskussionspunkte
Tag 2 (wahlweise DE oder EN)	• **Rückblick auf Tag 1:** Wiederholung der Kernthemen des Vortags • **Anforderungs-Management und Architektur:** Systematische Anforderungen, Softwarearchitektur und deren Sicherheitsaspekte • **Design, Implementierung und Testen:** Best Practices im Softwareentwicklungsprozess einschließlich Prüfansätzen • **Freigabe und Post-Market-Aktivitäten:** Normgerechte Freigabeprozesse sowie Überwachung nach der Markteinführung • **Legacy-Produkte:** Sicherheits- und Regulierungsherausforderungen für bestehende Produkte • **Diskussionen und praktische Auswirkungen:** Austausch zu realen Herausforderungen und deren Lösungen • **Abschluss und Zusammenfassung:** Offene Fragen, Rückblick und Verabschiedung

TAG	INHALT
Tag 3 (EN)	**Cybersecurity Foundations and Memory Management Vulnerabilities** **Cybersecurity Basics** • Introduction to security: What is security, understanding threats and risks • Threat models: CIA Triad and STRIDE • Consequences of insecure software: Real-world impacts on systems and end users • Regulations and standards: Key frameworks such as HIPAA, GDPR, MDR, ANSI/CAN/UL 2900, IEC 62443, and NIST guidelines for medical devices and industrial control systems • Cybersecurity in healthcare: Risk analysis, threats to medical devices, attacker motivations, and challenges with legacy systems **Memory Management Vulnerabilities** • Introduction to assembly basics and memory handling: Understanding x64 assembly, registers, stack frames, and calling conventions • Buffer Overflow vulnerabilities (BOF): Stack-based and heap-based buffer overflows, exploitation techniques (code injection, stack smashing, shellcode), and real-world case studies (Heartbleed, VxWorks DHCP vulnerabilities)
Tag 4 (EN)	**Memory Management Hardening and Secure Software Features** **Memory Management Hardening** • Best practices for secure memory handling in C/C++: Avoiding unsafe functions, dealing with terminated strings, and typical memory-related mistakes, labs include fixing common vulnerabilities such as buffer overflows • Compiler-based protections: FORTIFY_SOURCE, AddressSanitizer (ASan), and stack smashing protection, labs to demonstrate security tools in action • Runtime protections: Address Space Layout Randomization (ASLR), non-executable memory (NX), and mitigation techniques against common circumvention methods (e.g., Return-Oriented Programming (ROP) • Case studies: Analysis of key vulnerabilities in real-world systems to reinforce best practices **Common Software Security Features** • Introduction to secure software features: Fundamentals of authentication, authorization, and access control • Case studies: Broken authentication in Medtronic Conexus Protocol and improper file system access control • Basics of file system security: Preventing unrestricted file uploads and hardening Linux filesystems

TAG	INHALT
Tag 5 (EN)	**Common Weaknesses – Password Management and Input Validation** **Password Management** • Inbound password management: Proper storage techniques (salting, adaptive hashing, and policies), password protection during transit, and NIST guidelines for memorized secrets, hands-on Labs: Testing password hashing • Outbound password management: Avoiding hardcoded passwords, secure API usage, and reducing exploitation risks • Protecting sensitive data in memory: Challenges with heap inspection, compiler optimizations, and best practices • Case studies: Lessons learned from real breaches, such as the Ashley Madison attack and Abbott FreeStyle Libre NFC vulnerabilities **Input Validation** • Fundamentals of input validation: Allowlist vs denylist strategies, attack surface analysis, and principles of defense in depth • Dealing with input validation challenges: Unicode and regex issues, and Regular Expression Denial of Service (ReDoS), lab: Handling ReDoS attacks • Integer handling vulnerabilities: Overflow, truncation, and signed/unsigned confusion, hands-on labs: Debugging and mitigating integer-related weaknesses using toolchain-level solutions • Best practices for input validation: Upcasting, pre-condition and post-condition testing, and use of secure libraries
Tag 6 (EN)	**Injection Attacks, Errors, and Code Quality** **Input Validation (continued)** • Injection vulnerabilities: OS command injection, code injection, path traversal, and virtual resource manipulation, labs include demonstrations of injection risks and best practices for prevention, case studies: Shellshock, GE Healthcare MobileLink, and DLL injection attacks • File and stream security: Managing proper resource access and avoiding path traversal risks through canonicalization techniques, lab: Mitigation of path traversal **Time and State Vulnerabilities** • Race conditions: TOCTTOU (Time of Check to Time of Use) vulnerabilities with practical examples and lab exercises • Secure handling of insecure temporary files and managing time-sensitive data **Error and Exception Handling** • Principles of secure error handling: Avoiding misleading status codes and empty catch blocks, labs to identify and fix error-handling flaws • Information exposure: Preventing leaks through proper logging and exception management strategies **Code Quality** • Importance of secure coding in C++: Proper initialization, memory cleanup, and handling object-oriented pitfalls like inheritance and mutability, labs: Addressing type mismatches and resource leaks

Weitere Informationen und die Anmeldemöglichkeit finden Sie **HIER**

Dozent

Christian Rosenzweig

Als Ingenieur für biomedizinische Technik war Christian Rosenzweig viele Jahre lang zuständig für Grundlagen- und Softwareentwicklung bei komplexen aktiven Medizinprodukten. Als Verantwortlicher für die Durchführung und Betreuung von Konformitätsbewertungsverfahren in der EU und USA sammelte er Erfahrungen mit allen Audit-Formen und verschiedensten Märkten (insbesondere FDA und MDSAP) sowie als Qualitätsmanagementbeauftragter ISO 13485 in einem Großkonzern. Als gefragter Experte steht er unseren Kunden bei Strategiefragen zu Seite sowie bei der Umsetzung des Qualitäts- oder Regulatory Affairs Managements. Hierbei liegt sein Fokus auf der Sicherheit von Medizinprodukten, dem Risikomanagement und der IT Security.

Die **Software-Expert:innen, die Sie von Tag 3 bis 6 im Workshop begleiten**, werden flexibel ausgewählt, je nachdem, mit welcher Programmiersprache Sie arbeiten.