+33 (0)1 48 10 09 46
formation@ambient-it.net
140 Avenue Jean Lolive 93500 Pantin - France

Updated on 02/04/2026

Register

Secure by Design training for embedded systems

2 days (14 hours)

Overview

The Secure by Design concept is not just about complying with standards, but about integrating security into the very heart of the software development life cycle (SDLC). For embedded systems, which are often deployed for years without the possibility of easy patching, code quality is the first line of defense.

Our "Developing Secure by Design" training course is aimed at engineers who design and code embedded systems. It aims to transform development practices to anticipate threats from the design and implementation phase onwards. You will learn how to identify critical vulnerabilities in C/C++ (buffer overflows, race conditions), apply secure coding standards (CERT C, MISRA), and use static and dynamic analysis tools.

By the end of the training, you will be able to integrate security into your CI/CD pipelines and deliver robust code that complies with the requirements of the Cyber Resilience Act.

Objectives

- Understand the principles of Secure SDLC (Secure Development Life Cycle).
- Master secure coding rules in C/C++.
- Know how to identify and correct classic vulnerabilities (Buffer Overflow, Integer Overflow).
- Implement defense mechanisms (Secure Boot, Encryption, TEE).
- Use code audit tools (SAST) and testing tools (fuzzing).

Target audience

- Embedded C/C++ developers
- Software architects

- Lead Developers / Tech Leads

Prerequisites

- Good command of C or C++.
- Knowledge of Linux or RTOS environments is a plus.

Technical requirements

- Linux virtual machine provided with compilation chain (GCC/Clang) and analysis tools (Cppcheck, Flawfinder, AFL++).

Secure by Design training for embedded systems

[Day 1 - Morning]

Introduction and Secure SDLC

- The threat landscape for embedded systems (OWASP IoT/Embedded Top 10)
- The cost of technical security debt
- Integrating security into the V-model or Agile cycle (DevSecOps)
- Design principles: Least privilege, Defense in depth, Reduced attack surface
- Hands-on workshop: Analyzing vulnerable code (simplified bug bounty) to identify obvious vulnerabilities.

[Day 1 - Afternoon] Secure

coding in C/C++

- Memory management: the nightmare of C
- Understanding and preventing buffer overflows (stack & heap)
- Format string vulnerabilities
- Secure integer management (integer overflows/underflows)
- Coding standards: MISRA C and SEI CERT C
- Hands-on workshop: Exploiting a buffer overflow and correcting it using defensive code.

[Day 2 - Morning]

Architecture and Data Protection

- Secure storage: never hardcode credentials
- Principles of cryptography applied to embedded systems (AES, ECC, Hashing)
- Hardware mechanisms: Secure Boot, TrustZone (TEE), TPM
- Securing communications (TLS, secure MQTT)
- Hands-on workshop: Implementation of encrypted storage for sensitive configurations.

[Day 2 - Afternoon]

Tools and validation (testing)

- Static code analysis (SAST): Cppcheck, SonarQube, CodeQL
- Dynamic analysis (DAST) and instrumentation (Valgrind, Sanitizers)
- Introduction to Fuzzing (AFL++) for testing robustness
- Dependency management and SBOM (Software Bill of Materials)
- Hands-on workshop: Setting up an automatic analysis pipeline on a Git project.

Target companies

This training is intended for both individuals and companies, large or small, wishing to train their teams in a new advanced IT technology or to acquire specific business knowledge or modern methods.

Positioning at the start of training

The positioning at the start of the training complies with Qualiopi quality criteria. Upon final registration, the learner receives a self-assessment questionnaire that allows us to assess their estimated level of proficiency in different types of technologies, as well as their expectations and personal objectives for the upcoming training, within the limits imposed by the selected format. This questionnaire also allows us to anticipate certain connection or internal security issues within the company (intra-company or virtual classroom) that could be problematic for the monitoring and smooth running of the training session.

Teaching methods

Practical training: 60% practical, 40% theory. Training materials distributed in digital format to all participants.

Organization

The course alternates between theoretical input from the trainer, supported by examples and reflection sessions, and group work.

Validation

At the end of the session, a multiple-choice questionnaire is used to verify that the skills have been correctly acquired.

Certification

A certificate will be issued to each trainee who has completed the entire training course.