Secure Boot and Embedded Linux Hardening Training

The security architecture of an embedded Linux product, from the chain of trust at boot to the processes the Cyber Resilience Act requires.

4 days	Intermediate to advanced	On site or remote	English or French
Duration	Level	Format	Language

Overview

This course describes the security architecture of an embedded Linux product and how to put it in place, from the chain of trust at boot to the processes the Cyber Resilience Act now requires. It covers secure boot and measured boot, hardware key stores, storage integrity and encryption, kernel and user-space hardening, secure updates, vulnerability management and the SBOM, and assembles them into one Yocto-based product.

The course does not make a product compliant by itself, but it covers the technical requirements of the CRA and the processes behind them, so that a team knows what its product must provide and how.

What You Will Learn

- Build a threat model for a connected product
- Set up a chain of trust from the ROM code to the root filesystem
- Store keys in a TPM, a secure element or a TEE, and use them for measured boot and disk encryption
- Protect storage with dm-verity and dm-crypt, and harden the kernel and the services
- Run a vulnerability management process and produce an SBOM with Yocto
- Map these measures to the essential requirements of the CRA

Who should attend

Embedded Linux engineers and architects in charge of the security of a connected product, and technical leads preparing a product for the CRA.

Prerequisites

Good knowledge of embedded Linux, of its boot chain and of the Yocto Project.

Program

Cryptography is used throughout the course without going through its theory, which an optional module covers.

Day 1

- Threats, assets, attackers and threat modelling
- Secure boot: principles, signed FIT images, secure boot in Yocto, locking the bootloader
- Hardware key stores: TPM, secure elements and the trusted execution environment
- Measured boot and remote attestation

Day 2

- Physical attacks, debug ports and secure manufacturing
- Kernel hardening: attack surface, security modules, run-time settings
- Storage integrity: read-only root filesystem, dm-verity, dm-integrity, fs-verity

Day 3

- Encryption at rest: dm-crypt, LUKS2, unattended unlocking with a TPM or a secure element, fscrypt
- Users, privileges and access control: capabilities, seccomp, namespaces, systemd sandboxing
- Secure software updates: signed bundles, A/B slots, anti-rollback

Day 4

- Vulnerability management: CVEs, triage, VEX and the Yocto CVE tooling
- Software bill of materials: SPDX, CycloneDX, generation in Yocto, licence compliance
- The Cyber Resilience Act: scope, essential requirements, reporting duties and dates
- Putting it together in a Yocto-based product
- For x86 products: UEFI secure boot keys, shim and unified kernel images, detailed in the x86 boot chain and secure boot course

Labs and Hardware

Labs run on QEMU machines and on the boards of our lab: Arm 32-bit: STM32MP157C-DK2, BeagleBone Black; Arm 64-bit: Raspberry Pi 4 and 5, NXP i.MX8M Mini, Khadas VIM3; x86: QEMU x86, Generic x86_64 machines, Intel development kits. Other boards, or the hardware of the customer, can be used on request.

Sessions take place on site, in France, Belgium, Switzerland, Germany and beyond, with the hardware provided by LABCSMART, or remotely. Remote attendees work on our build server, each with their own SSH account and workspace, and run the labs on QEMU machines and on real boards. The exercises are drawn from our own client projects. Courses are given in English or French.

Optional Modules

These modules go further on topics that the course only introduces.

- SELinux in an embedded product (1 day)
- Network security: nftables, TLS 1.3, mutual TLS, SSH hardening (1 day)
- Secure development and security testing: static analysis, sanitizers, fuzzing (1 day)
- Cryptography and PKI for embedded engineers: primitives, signatures, X.509, OpenSSL, PKCS#11 (2 days)

Who Teaches This Course

The course is designed and given by the founder of LABCSMART, an embedded Linux and Linux kernel engineer with more than eleven years of industrial experience. He contributes upstream to the Linux kernel for Renesas RZ SoCs, in the PCIe, audio, Ethernet, thermal, DMA and clock subsystems, and wrote two books published by Packt, *Linux Device Driver Development, Second Edition* (2022) and *Mastering Linux Device Driver Development* (2021), presented on our publications page.

Plan a session for your team. Every course is made of independent modules. A team can drop the modules it does not need and add others in their place, and we also build courses on demand around a given SoC, board or project. Contact: contact@labcsmart.com, https://labcsmart.com/company/contact/. Course page: https://labcsmart.com/training/secure-boot-embedded-linux-hardening/