

Objectives

- Implementing secure boot
- Verifying the authenticity of system components before they are loaded and executed.
- Ensure the authenticity and integrity of the bootloader, kernel
- Implements the Trusted Boot
- Provides a secure environment for the secure monitor firmware
- Run OP-TEE on secure environment that runs alongside the main operating system

Labs are conducted on QEMU ARM-based board

Prerequisite

- C Language knowledge (see for example our L2 training course)
- Embedded Linux Build knowledge (see for example our D1 training course)
- You may be interested also by the SEC9 Advanced Embedded Linux Security course
- You may be interested also by the SEC1 Secure Development for Embedded System course
- You may be interested also by the SEC2 Advanced Embedded Systems Security course

Equipment

- Training manuals and software exercises
- One Linux PC for two trainees
- One target platform for two trainees

Duration

- Total: 2 days
- From 40% to 50% of training time is devoted to practical activities

Environnement du cours

- Cours théorique
 - Support de cours au format PDF (en anglais).
 - Cours dispensé via le système de visioconférence Teams.
 - Le formateur répond aux questions des stagiaires en direct pendant la formation et fournit une assistance technique et pédagogique.
- Activités pratiques
 - Les activités pratiques représentent de 40% à 50% de la durée du cours.
 - Elles permettent de valider ou compléter les connaissances acquises pendant le cours théorique.
 - Exemples de code, exercices et solutions.
 - Un PC Linux en ligne par stagiaire pour les activités pratiques.
 - Le formateur a accès aux PC en ligne des stagiaires pour l'assistance technique et pédagogique.
 - Certains travaux pratiques peuvent être réalisés entre les sessions et sont vérifiés par le formateur lors de la session suivante.
- Une machine virtuelle préconfigurée téléchargeable pour refaire les activités pratiques après le cours
- Au début de chaque session une période est réservée à une interaction avec les stagiaires pour s'assurer que le cours répond à leurs attentes et l'adapter si nécessaire

Audience visée

- Tout ingénieur ou technicien en systèmes embarqués possédant les prérequis ci-dessus.

Modalités d'évaluation

- Les prérequis indiqués ci-dessus sont évalués avant la formation par l'encadrement technique du stagiaire dans son entreprise, ou par le stagiaire lui-même dans le cas exceptionnel d'un stagiaire individuel.
- Les progrès des stagiaires sont évalués de deux façons différentes, suivant le cours:
 - Pour les cours se prêtant à des exercices pratiques, les résultats des exercices sont vérifiés par le formateur, qui aide si nécessaire les stagiaires à les réaliser en apportant des précisions supplémentaires.
 - Des quizz sont proposés en fin des sections ne comportant pas d'exercices pratiques pour vérifier que les stagiaires ont assimilé les points présentés
- En fin de formation, chaque stagiaire reçoit une attestation et un certificat attestant qu'il a suivi le cours avec succès.
 - En cas de problème dû à un manque de prérequis de la part du stagiaire, constaté lors de la formation, une formation différente ou complémentaire lui est proposée, en général pour conforter ses prérequis, en accord avec son responsable en entreprise le cas échéant.

Plan

First Day

Linux overview

- Linux history
- Linux architecture and modularity
- Linux system components
- The various licenses used by Linux (GPL, LGPL, etc)

Boot chain

- Low-level boot
 - Boot on NOR
 - Boot on NAND
 - Boot on SD/MMC/eMMC
 - Multistage Boot
 - Why do we need a trusted boot chain
- Security Concerns
 - Confidentiality and Integrity
 - Tampering Prevention
 - Compliance and Certification

Exercise : Boot the platform with the prebuilt image

Secure Boot

- Secure Boot concept
 - The chain of trust
 - Complete secure boot process
- Key Management
 - Introduction to key management
 - Cryptographic algorithms and key types
 - Key storage options: Hardware-based and software-based
 - Key management processes: Generation and revocation of keys
 - ARM-based platforms hardware features overview
 - Secure Monitor
 - Secure World
 - Trusted Execution Environment
 - Secure Boot on RISC-V and X86_64
 - Cryptographic Accelerators

- Software Solutions
 - Open source
 - Proprietary

Exercise : Generate keys that are going to be used for platform encryption

First and Second Stage Bootloaders

- U-Boot
 - Capabilities and features
 - Configuration, customization, and compilation
 - U-Boot SPL as First-Stage Boot Loader (SSBL)
 - Role of u-boot in the trusted boot chain
 - How U-Boot verifies the authenticity of the images it loads
 - Configuration options for securing the boot process
 - Interaction with the secure world and Trusted Execution Environment
 - Signing U-boot
- Arm Trusted Firmware (ATF)
 - Overview and features
 - ATF Boot flow
 - Services
 - Build and deploy
- Other platform specific components

Exercise : Build and boot the platform with U-boot as FSBL and SSBL

Exercise : Build and Boot the platform with ATF as FSBL and U-boot as SSBL

Secured Linux Image

- Introduction to Linux kernel
 - Source code
 - Configuration
 - Compilation
- FIT (Flattened Image Tree) Image
 - What is FIT and why is it used
 - Advantages of using FIT image
 - Configuration
 - Building a Secure FIT Image
- Kernel Configuration for a Secure Linux Platform
 - Configuration options for secure boot in the Linux kernel
 - Access Control Configuration overview

Exercise : Create a secured FIT Linux image

Second Day

Security Considerations when Creating a Root Filesystem

- Tips for hardening and securing a rootfs
 - Minimizing the rootfs
 - Strong authentication
 - Keep software updated
 - Using initramfs
- Read-only root filesystem
 - Introduction to read-only root filesystem
 - Purpose and benefits
 - Overview of the different solutions available
 - SquashFS
 - CramFS: Small memory footprint
 - OverlayFS-based read-only root filesystem
 - UnionFS-based read-only root filesystem
- Considerations when choosing a read-only root filesystem solution

- Evaluation based on use case, security, performance, and compatibility
- Encrypting Update Images
 - Securely update Linux platform using Mender

Exercise : Create a read-only file system using SquashFS

Open Portable Trusted Execution Environment (OP-TEE)

- Introduction to OP-TEE
- Key Features
- Hardware, software, and firmware requirements
- Architecture of OP-TEE
 - Components, modules, and communication channels
- Use Cases
 - Secure storage
 - Secure communication
 - Secure execution of applications
- OP-TEE build and deployment
 - Setting up the environment
 - Configuration of OP-TEE
 - Compilation of OP-TEE
- Comparison to other TEE solutions
- Trusted Applications (TA) on OP-TEE
 - The role of a TA in a secure system
 - Writing a Trusted Application
 - Loading and executing a Trusted Application within the OP-TEE runtime
 - Debugging and testing Trusted Applications
 - Communication between Trusted Applications and normal world applications
 - Best practices for creating secure Trusted Applications

Exercise : Build and install OP-TEE

Exercise : Write a TA application that communicates with a normal world application

Renseignements pratiques

Renseignements : 12 heures