Formation sécurité des systèmes Linux embarqués

Durée de la formation

3 jours – 24 h

Langue

Transparents	Anglais
Présentation	Français
	Anglais

Formateur

Un des ingénieurs suivants

- Mathieu Dubois-Briand
- Olivier Benjamin

Contact

training@bootlin.com

+33 4 84 25 80 96

bootlin

bootlin.com

Formation en présentiel

Public visé

Entreprises et ingénieurs qui conçoivent, développent et produisent des systèmes Linux embarqués ayant des exigences en matière de cybersécurité.

Objectifs opérationnels

- Être capable de comprendre les mécanismes de sécurité et d'isolation des systèmes Linux embarqués modernes : mémoire non exécutable, randomisation de l'espace d'adressage, niveaux de privilèges, contrôle d'accès obligatoire et discrétionnaire.
- Être capable de concevoir et de mettre en œuvre une chaîne de démarrage sécurisée, du chargeur de démarrage jusqu'à l'espace utilisateur, y compris dm-verity.
- Être capable de raisonner sur la cryptographie et de mettre en œuvre un stockage sécurisé des clés, à l'aide d'un HSM matériel ou logiciel.
- Être capable d'exploiter la technologie ARM TrustZone (monde sécurisé, TF-A, OP-TEE) afin d'isoler les opérations sensibles et de développer des Trusted Applications.
- Être capable de mettre en œuvre des mécanismes de sécurité en espace utilisateur : capacités Linux, namespaces, cgroups, SECCOMP, SELinux et fonctionnalités de durcissement de systemd.
- Être capable de configurer le chiffrement du système de fichiers à l'aide de dm-crypt et de l'intégrer à une gestion sécurisée des clés via PKCS#11.
- Être capable de gérer les vulnérabilités logicielles à l'aide de bases de données CVE, de générer et d'analyser des Software Bills of Materials (SBoM), et de comprendre les exigences de conformité, notamment le Cyber Resilience Act.
- Être capable de concevoir et de déployer des mécanismes de mise à jour sécurisés à l'aide de schémas de partitionnement A/B et d'outils tels que RAUC ou SWUpdate.
- Être capable de comprendre les concepts de measured boot et de mettre en œuvre la vérification de l'intégrité de la plateforme à l'aide de TPM et de IMA/EVM.

Prérequis

- **Connaissance et pratique des commandes UNIX ou GNU/Linux** : les participants doivent être à l'aise avec l'utilisation de la ligne de commande Linux. Les participants manquant d'expérience sur ce sujet doivent se former par eux-mêmes, par exemple en utilisant nos supports de formation.
- **Expérience minimale en développement Linux embarqué** : les participants doivent avoir une compréhension minimale de l'architecture d'un système Linux embarqué : rôle du noyau Linux par rapport à l'espace utilisateur, développement d'applications espace utilisateur en C. Suivre la formation Linux embarqué de Bootlin permet de remplir ce pré-requis.
- **Niveau minimal requis en anglais : B1**, d'après le *Common European Framework of References for Languages*, pour nos sessions animées en anglais. Voir la grille CEFR pour une auto-évaluation.

Méthodes pédagogiques

- Présentations animées par le formateur : 40% de la durée de formation
- Travaux pratiques réalisés par les participants : 60% de la durée de formation
- Version électroniques de supports de présentation, des instructions et des données de travaux pratiques. Les supports sont librement disponibles ici.

Modalités d'évaluation

Seuls les participants qui auront assisté à l'intégralité des journées de formation, et qui auront obtenu plus de 50% de réponses correctes à l'évaluation finale recevront une attestation individuelle de formation de la part de Bootlin.

Handicap

Les participants en situation de handicap qui ont des besoins spécifiques sont invités à nous contacter à l'adresse *training@bootlin.com* afin de discuter des adaptations nécessaires à la formation.

Équipement nécessaire

Pour les sessions en présentiel délivrées chez nos clients, notre client devra fournir :

- Projecteur vidéo
- Un ordinateur sur chaque bureau (pour une ou deux personnes), avec au moins 16 Go de RAM et Ubuntu Linux 24.04 installé dans une partition dédiée d'au moins 30 Go.
- Les distributions autres que Ubuntu Linux 24.04 ne sont pas supportées, et l'utilisation de Linux dans une machine virtuelle n'est également pas supportée.
- Connexion à Internet rapide et sans filtrage : au moins 50 Mbit/s de bande passante en téléchargement, et pas de filtrage des sites Web et protocoles.
- Les ordinateurs contenant des données importantes doivent être sauvegardés avant d'être utilisés dans nos sessions.

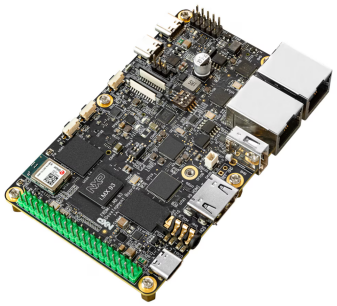
Pour les sessions en présentiel organisés dans les locaux de Bootlin, Bootlin fournit l'ensemble de l'équipement.

Plateforme matérielle pour les travaux pratiques

NXP i.MX93 FRDM

Carte de développement **NXP FRDM-IMX93**

- Processeur NXP i.MX93 (Dual Cortex-A55 + Cortex-M33)
- 2 GB LPDDR4X, 32 GB eMMC
- Double Ethernet Gigabit
- USB 2.0 Type-C + USB Type-A
- Interface CAN
- Slot microSD, EEPROM
- Wi-Fi 6 + Bluetooth 5.4 + 802.15.4 (MAYA-W276)
- Sortie HDMI (via LVDS), MIPI DSI et CSI
- Audio jack (MQS), buttons and LEDs
- debug via SWD et UART



Programme de la formation

Jour 1 - Matin

Cours	Concepts fondamentaux	▪ Modélisation des menaces ▪ Bases de la cryptographie : clé publique / clé privée ▪ Comprendre un handshake TLS du début à la fin ▪ Comprendre les infrastructures à clés publiques (PKI)
TP	Configuration d'une PKI simple	▪ Utilisation d'OpenSSL pour générer une hiérarchie de clés ▪ Expérimentation de la révocation de clés ▪ Comparaison des performances du chiffrement par clé publique/clé privée
Cours	Limites de sécurité assistées par le matériel	▪ Concepts généraux de sécurité : séparation noyau/utilisateur, CPL, protections mémoire ▪ Contre-mesures courantes : ASLR, NX/DEP, SMAP/SMEP, RELRO ▪ Fonctionnalités ARM : niveaux d'exception ▪ Fonctionnalités ARM : *secure world*, TF-A, OP-TEE ▪ Compréhension d'un flux de démarrage complet ARMv8, y compris le monde sécurisé ▪ Compréhension des Trusted Applications

Jour 1 - Après-midi

TP	Exploration du *secure world*	▪ Compilation de logiciel pour le *secure world* (TF-A, OP-TEE, TA) ▪ Ajout de journaux pour observer les transitions de niveaux d'exception ▪ Ajout de journaux pour observer les transitions entre mondes ▪ Écriture d'une petite Trusted Application dans OP-TEE ▪ Interaction avec notre Trusted Application depuis l'espace utilisateur
Cours	Secure boot : première partie	▪ Compréhension du concept de chaîne de démarrage sécurisée ▪ Exemples de mise en œuvre du secure boot : x86 et SoC ARM ▪ Secure boot : modèle de menace ▪ Être capable de concevoir et de mettre en œuvre une chaîne de démarrage sécurisée
TP	Secure boot sur i.MX93	▪ Compilation de tous les composants logiciels pour le secure boot (AHAB) sur i.MX93 ▪ Création d'un conteneur AHAB signé à l'aide de SPSDK ▪ Programmation des fusibles sur l'i.MX93 pour activer le secure boot (AHAB) ▪ Vérification que le secure boot (AHAB) est correctement activé

Jour 2 - Matin

Cours		▪ Analyse approfondie des composants d'U-Boot : TPL/SPL, U-Boot principal, images FIT ▪ Description matérielle : Device Tree ▪ Activation de la vérification des signatures dans le chargeur de démarrage U-Boot ▪ Vérification du RootFS : *dm-verity* et implications pour la conception du système
TP	Secure boot sur i.MX93 : étape du bootloader	▪ Intégration de la clé publique dans le DTB ▪ Ajout d'une signature à l'image FIT du noyau ▪ Configuration de U-Boot pour imposer la vérification des signatures ▪ (optionnel) Configuration du SPL pour imposer également la vérification des signatures

Jour 2 - Après-midi

Cours	Protection des données	▪ Chiffrement du système de fichiers, présentation de dm-crypt/cryptsetup ▪ Gestion des clés : module matériel de sécurité (HSM) ▪ Utilisation sécurisée des clés : introduction à PKCS#11 ▪ Utilisation sécurisée des clés : trousseau de clés du noyau et clés de confiance
TP	Gestion des clés	▪ Provisionnement des clés dans l'ELE du i.MX93 ▪ Utilisation d'OP-TEE comme HSM logiciel ▪ (optionnel) Chiffrement du système de fichiers à l'aide de la clé ELE ▪ (optionnel) Signature des images FIT de U-Boot avec intégration d'un HSM via PKCS#11
Cours	Mécanismes de sécurité en user-space	▪ Paradigmes de contrôle d'accès : MAC/DAC ▪ Linux *capabilities* : utilisation et exemples ▪ Isolation des processus : namespaces, cgroups, SECCOMP ▪ Modules de sécurité Linux : SELinux, AppArmor ▪ *Hardening* des applications via *systemd*

Jour 3 - Matin

TP	Restriction des applications user-space	▪ Empêcher une application simple d'exécuter du shellcode à l'aide de SECCOMP ▪ Manipulation des contextes SELinux ▪ Utilisation de systemd pour restreindre l'accès d'un démon aux ressources
Cours	Measured Boot	▪ Journaux des événements de démarrage ▪ Rôle du TPM dans l'intégrité de la plateforme ▪ Introduction à IMA/EVM
Cours	Maintenance, réglementations et conformité	▪ Introduction aux cadres de gestion des vulnérabilités : CVE, CWE, CVSS ▪ Présentation du Cyber Resilience Act (CRA) ▪ Définition d'une stratégie de mise à jour, basée sur les cycles de publication et de maintenance des projets open source importants ▪ Génération d'un Software Bill of Materials ▪ Analyse d'un Software Bill of Materials
TP	Prise en main des SBoM et des CVE	▪ Génération d'un SBoM pour une distribution Yocto ▪ Analyse du SBoM généré à l'aide de `sbom-cve-check` ▪ Correction d'une CVE présente dans le SBoM ▪ Analyse différentielle

Jour 3 - Après-midi

Cours	Mises à jour sécurisées	▪ Mises à jour A/B ▪ Présentation de SWUpdate ▪ Présentation de RAUC ▪ Intégration au chargeur de démarrage ▪ Conséquences sur la chaîne de démarrage sécurisée
TP	Mise en oeuvre de RAUC pour les mises à jour sécurisées	▪ Configuration de RAUC sur la cible ▪ Création, déploiement et installation d'un bundle RAUC ▪ Vérification de la signature du bundle à l'aide de PKCS#11 ▪ (optionnel) Chiffrement du bundle RAUC ▪ (bonus) Intégration d'un HSM