Formation temps-réel sous Linux avec *PREEMPT_RT*

Formation sur site, 2 jours

Dernière mise à jour : 26 April 2024

Titre	**Formation temps-réel sous Linux avec *PREEMPT_RT***
Objectifs opérationnels	• Être capable de comprendre et de maîtriser les caractéristiques d'un système d'exploitation temps-réel • Être capable de télécharger, compiler et utiliser le patch *PREEMPT_-RT* • Être capable d'identifier et de benchmarker la plateforme matérielle en terme de caractéristiques temps-réel • Être capable de configurer le noyau Linux pour un comportement déterministe • Être capable de développer, de tracer et de débugger des applications user-space temps-réel.
Durée	**Deux** jours - 16 h (8 h par jour)
Méthodes pédagogiques	• Présentations animées par le formateur : 50% de la durée de formation • Travaux pratiques réalisés par les participants : 50% de la durée de formation • Version électroniques de supports de présentation, des instructions et des données de travaux pratiques. Les supports sont librement disponibles sur https ://bootlin.com/doc/training/preempt-rt.
Formateur	Maxime Chevallier `https://bootlin.com/company/staff/maxime-chevallier/`
Langue	Présentations : Français Supports : Anglais
Public visé	Entreprises et ingénieurs intéressés dans le développement et le benchmarking d'applications et de drivers temps-réel pour un système Linux embarqué.

Pré-requis	• **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 disponible à l'adresse bootlin.com/blog/command-line/. • **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, disponible sur bootlin.com/training/embedded-linux/, 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 bootlin.com/pub/training/cefr-grid.pdf pour une auto-évaluation.
Équipement nécessaire	**Pour les sessions en présentiel dans les locaux de nos clients, notre client doit fournir** : • Projecteur vidéo • Un ordinateur sur chaque bureau (pour une ou deux personnes), avec au moins 8 Go de RAM et Ubuntu Linux 22.04 installé dans une **partition dédiée d'au moins 30 Go**. • Les distributions autres que Ubuntu Linux 22.04 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.
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.

Plateforme matérielle pour les travaux pratiques

Carte **STMicroelectronics STM32MP157D Discovery Kit 1**

- Processeur STM32MP157D (dual Cortex-A7) de STMicroelectronics
- Alimentation par USB
- 512 MB DDR3L RAM
- Ethernet Gigabit
- 4 ports USB 2.0 hôte
- 1 port USB-C OTG
- 1 slot Micro SD
- Debugger ST-LINK/V2-1
- Connecteurs compatibles Arduino
- Codec audio, boutons, LEDs

1ère journée - Matin

Cours - Introduction au comportement temps-réel et au déterminisme

- Définition d'un système d'exploitation temps-réel
- Spécificigés des systèmes multi-tâches
- Principaux patterns de verrouillage et de gestion des priorités
- Aperçu des systèmes temps-réel existants
- Approches pour apporter un comportement temps-réel à Linux

Cours - Le patch *PREEMPT_RT*

- Histoire et avenir du patch *PREEMPT_RT*
- Améliorations temps-réel provenant de *PREEMPT_RT* dans le noyau Linux officiel
- Fonctionnement interne de *PREEMPT_RT*
- Gestion des interruptions : interruptions threadées, softirqs
- Primitives de verouillage : mutexes et spinlocks, spinlocks avec sommeil
- Modèles de préemption

TP - Compiler un noyau Linux avec *PREEMPT_RT*

- Télécharger le noyau Linux et appliquer le patch *PREEMPT_RT*
- Configurer le noyau Linux
- Démarrer le kernel sur une plateforme matérielle

1ère journée - Après-midi

Cours - Configuration et limites du matériel pour le temps-réel

- Interruptions et firmware
- Interaction avec les fonctionnalités de gestion d'énergie : gestion dynamique de la fréquence du CPU et états de sommeil
- DMA

Cours - Outils : Benchmarking, Stress et Analyse

- Benchmarking avec *cyclictest*
- Stress du système avec *stress-ng* et *hackbench*
- L'infrastructure de *tracing* du noyau Linux
- Analyse de la latence et de l'ordonnancement avec *ftrace*, *kernelshark* ou *LTTng*

TP - Outils : Benchmarking, Stress et Analyse

- Utilisation des outils de benchmark et de stress
- Techniques classiques de benchmarking
- Benchmarking et configuration de la plateforme matérielle

2ème journée - Matin

Cours - Infrastructures du noyau Linux et configuration

- Bonnes pratiques pour le développement de drivers noyau Linux pour des systèmes temps-réel
- Politiques d'ordonnancement et priorités : *SCHED_FIFO*, *SCHED_RR*, *SCHED_DEADLINE*
- Affinité CPU et IRQ
- Gestion mémoire
- Isolation des CPUs avec *isolcpus*

Cours - Patterns de développement d'applications temps-réel

- API POSIX pour les applications temps-réel
- Gestion et configuration des threads
- Gestion mémoire : allocation mémoire et verouillage mémoire, gestion de la pile
- Patterns de verrouillage : mutexes, héritage de priorité
- Communication inter-processus (IPC)
- Signalisation

2ème journée - Après-midi

TP - Débugger une application de démonstration

- Créer une application de démonstration déterministe
- Utiliser l'infrastructure de *tracing* pour identifier la source de latence
- Apprendre à utiliser l'API POSIX pour gérer les threads, le verouillage, la mémoire
- Apprendre à utiliser l'affinité CPU et configurer la politique d'ordonnancement

Questions / réponses

- Questions / réponses avec les participants autour du noyau Linux
- Des présentations supplémentaires s’il reste du temps, selon les sujets qui intéressent le plus les participants.