logilin
Formations Linux

Temps-réel Linux et Xenomai

Référence formation : ILR

Filière : « Linux Industriel »

En fonction des contraintes temporelles qui lui sont imposées, le développeur dispose de plusieurs possibilités pour construire une application sur Linux. Depuis les processus ordonnancés en temps-réel souple par le noyau standard jusqu'aux extensions s'appuyant sur des micro-noyaux hard-realtime (comme Xenomai), ce cours vous propose un cheminement logique et illustré dans le panorama des solutions temps-réel disponibles avec Linux.

Tous les concepts théoriques présentés dans le cours font l'objet de mise en pratique par des exemples précis et des séances d'exercices complètes.

Organisation

Durée : 3 jours (21 heures).
Durée en formation individuelle accélérée : 2 jours (14 heures).
Public visé : développeurs et chefs de projets souhaitant utiliser Linux sur des projets à contraintes temporelles.
Objectifs :
- Comprendre le fonctionnement bas-niveau de l'ordonnanceur Linux.
- Connaître les rôles, avantages et inconvénients de PREEMPT_RT et Xenomai.
- Savoir développer du code métier répondant à des contraintes temps réel.

Pré-requis : Connaissance de Linux (utilisateur) et du langage C.
Moyens pédagogiques : les travaux pratiques, exemples et séances d'exercices se déroulent sur des postes PC et sur des systèmes embarqués à processeur ARM mis à disposition des participants (un PC et une carte à processeur ARM par participant).
Conseil cursus : Nos formations « Écriture de Drivers pour Linux » et « Linux embarqué » représentent de bons compléments à ce cours.

Thèmes abordés

- **Ordonnancement sous Linux** : introduction, multi-tâches, systèmes multi-processeurs, temps-partagé et priorité.
- **Temps-réel souple :** ordonnancements Fifo et Round-Robin sur un noyau standard *Vanilla*, fluctuations des timers, latence d'interruption, problèmes temps-réel classiques.
- **Temps-réel amélioré avec *PREEMPT_RT*** : application du patch et compilation d'un noyau *PREEMPT_RT*, mesure des améliorations de fluctuation ou de latence.
- **Extensions temps-réel strict** : principes de RT-Linux, RTAI, Adeos et Xenomai, installation et validation de Xenomai, écriture d'applications avec l'API Xenomai, handlers d'interruption.

Plan détaillé au verso .../..

Plan détaillé

I – Ordonnancement sous Linux

Introduction
Noyau et espace utilisateur, organisation du système, répartition des ressources.

Multi-tâche sous Linux
Processus et threads Posix. Synchronisation et communication entre tâches.

Systèmes multi-processeurs symétriques (SMP)
Multi-processeur, multi-coeur, *hyperthreading*. Affinités et migrations des tâches.

Temps-partagé
Ordonnanceurs O(1) et CFS, groupement automatique des tâches. Priorités, *nice*.

Travaux pratiques
Comparaison des temps de création et commutation des processus et des threads. Vitesse de communication par file de messages. Utilisation de la mémoire partagée. Synchronisation par mutex et par variable condition. Influence de la priorité temps-partagé.

II – Temps-réel souple

Fifo et Round-robin
Passage en temps-réel. Priorités. Garde-fou temps réel.

Timers
Création de timers Unix et Posix. Mesures temporelles.

Interruptions
Principe. Organisation des handlers. Préemptibilité du noyau. Latences.

Problèmes temps-réel classiques
Lancement de tâches en parallèle. Inversion de priorité. Reprise de mutex.

Travaux pratiques
Création de processus et de threads temps-réel. Mesure de précision des timers. Effet de la préemptibilité du noyau sur la latence des interruptions. Examen d'inversion de priorité. Héritage de priorité. Test de reprise de mutex.

III – Temps-réel amélioré avec *PREEMPT_RT*

Patch *PREEMPT_RT*
Principe. Patch d'Ingo Molnar et Thomas Gleixner. Compilation du noyau.

Préemption totale
Effets de l'option de préemptibilité totale. Activation à la compilation.

Threadeds interrupts
Configuration de la priorité des handlers d'interruptions.

Instrumentation et mesures
Outils *RT-test* et *cyclictest*.

Travaux pratiques
Compilation d'un noyau après application du patch *PREEMPT_RT*. Vérification de la préemptibilité. Utilisation de *cyclictest* et comparaison avec le noyau standard. Comparaison du comportement des exemples du chapitre précédent.

IV – Extensions temps-réel strict pour Linux

Principes du temps-réel strict (*Hard Realtime*)
Noyau standard et extensions *RT-Linux*, *RTAI*, *Xenomai*...

Installation et validation de *Xenomai*
Patch et bibliothèque utilisateur. Compilation du noyau et des outils de test.

Applications sous *Xenomai*
Tâches Xenomai. Timers et tâches périodiques. Communication et synchronisation.

Gestion des interruptions
Interface de programmation RTDM.

Travaux pratiques
Compilation et installation de *Xenomai*. Création de tâches. Synchronisation. Vérification des priorités par rapport aux tâches du noyau standard. Priorités par rapport au kernel Linux. Test de précision des timers. Mesure des latences d'interruption.

Conclusion

Discussions libres sur l'ensemble des thèmes abordés.
Travaux pratiques
Expérimentations libres suivant les demandes des participants.