SMB137 - Systèmes d'exploitation : principes, programmation et virtualisation

Valide le 20-05-2019

Code : SMB137

6 crédits

Responsabilité nationale :
EPN05 - Informatique / Samia BOUZEFRANE

Contact national :
EPN 05 Informatique
2 rue conté
31.1.79
75003 Paris
01 40 27 20 38
Agathe Froger
agathe.froger@lecnam.net

Présentation

Prérequis

Elèves avec les connaissances de la programmation, du langage C et des systèmes informatiques vues en premier cycle du Cnam ou équivalent

Objectifs pédagogiques

- **Comprendre les principaux concepts et paradigmes des systèmes d'exploitation modernes.**

- Etudier les mécanismes de base mis en oeuvre dans le noyau de systèmes tels que Linux ou Unix BSD.
- Obtenir des bases dans la compréhension des mécanismes de gestion du parallélisme utilisés dans les noyaux des systèmes.

- **Comprendre les nouvelles fonctions intégrées dans les architectures matérielles récentes** :

- processeurs multi-coeurs,
- architectures NUMA (Non Uniform Memory Architecture),
- support matériel de la virtualisation de systèmes.

- **Comprendre et maîtriser les objectifs et les principes de la virtualisation de systèmes**

- Etude des différentes techniques mises en oeuvre dans les hyperviseurs logiciels (VMWare, Xen, KVM)
- Etude du support de la virtualisation intégré dans les architectures matérielles récentes : processeurs Intel-VT, mécanismes de Direct I/Os, fonctions PCI virtuelles.

- **Mise en oeuvre de la virtualisation dans l'infrastructure des télécommunications et de l'Internet**

- Comprendre la mise en oeuvre de la virtualisation dans l'infrastructure des télécommunications et de l'Internet (Cloud Computing, réseaux d'entreprise, téléphonie mobile).
- Comprendre les notions de Network Function Virtualization (NFV) et de Virtual Switch (commutateur virtuel), et leur complémentarité. Appréhender la notion de Software Defined Network (SDN) et de réseau programmable, et leur application dans le contrôle des infrastructures réseau virtualisées.

A l'aide des exemples écrits en langage C sur lesquels sont basés les exercices dirigés, faire découvrir les méthodes de conception et de mise en forme de programmes selon les normes professionnelles en vigueur dans l'industrie du logiciel, et dans les projets open-source tels que le noyau Linux par exemple.

Faire connaître l'environnement du logiciel libre, par l'intermédiaire des logiciels disponibles gratuitement utilisés pour la réalisation du cours et des exercices dirigés :

- Noyau Linux
  https://www.kernel.org/
  http://www.tldp.org/LDP/tlk/tlk.html
- Distribution Ubuntu

http://www.ubuntu.com/
http://www.ubuntu-fr.org/
- Environnement de développement GNU
http://gcc.gnu.org/
- Gestionnaire de sources
http://git-scm.com/
- Suite bureautique LibreOffice
https://fr.libreoffice.org/
- Composition de documents en reStructuredText
http://docutils.sourceforge.net/docs/ref/rst/restructuredtext.html
http://docutils.sourceforge.net/docs/user/rst/quickref.html
http://sphinx-doc.org/
- Outil de documentation de logiciel Doxygen
http://www.doxygen.org
- Navigateur Firefox
http://www.mozilla.org/fr

Compétences

Obtenir des bases solides dans les domaines apparentés à la conception de systèmes d'exploitation et dans la mise en oeuvre de la concurrence dans les systèmes temps-réel et dans les applications parallèles (applications réparties, par exemple).
Comprendre l'architecture et le fonctionnement des systèmes d'exploitation tels que Unix et Linux pour maîtriser leur administration et le développement
d'applications.
Maîtriser les principes sous-jacents à la virtualisation de systèmes afin de faciliter l'intégration et l'administration de ce type de service dans un système informatique (Cloud Computing, Haute Disponibilité, Tolérance aux pannes, etc.)

Programme

Contenu

1. Principes de base d'un système d'exploitation
Rappel sur l'architecture des ordinateurs :
modes d'exécution, interruptions, gestion des périphériques, entrés/sorties en mode DMA, pile d'exécution, etc.
Environnement de développement et conception modulaire de programmes : compilation croisée, éditions de liens statique et dynamique, bibliothèques partagées. Exemple des "shared object" du système Linux
Présentation de la notion de processus et de contexte d'exécution indépendant, d'espace d'adressage séparé, de chargement dynamique de programme.
Introduction au parallélisme et à la programmation concurrente. Notions de thread et de processus multi-threads. Description des politiques d'ordonnancement pour l'attribution des processeurs.

2. Gestion mémoire
Introduction à la notion d'espace d'adressage (physique, virtuel).

Analyse des architectures de type NUMA (Non Uniform Memory Architecture) et de la prise en compte de leurs propriétés lors de la conception de logiciels critiques.

Etude du mécanisme matériel de protection de la mémoire physique de type MPU (Memory Protection Unit) introduit dans les processeurs dédiés aux systèmes enfouis (automobile, transport).

Etude des techniques d'allocation mémoire.
Tout d'abord des méthodes d'allocations classiques "historiques" : allocateurs first-fit, best-fit et worst-fit.
Puis des allocateurs mémoire modernes à 2 étages.

Etude de la pagination et de la gestion des espaces d'adressage paginés dans les systèmes Unix et Linux. Fonctionnement d'une MMU (Memory Management Unit) et d'un TLB (Translation Lookaside Buffer). Méthodes de gestion des défauts de page, principes du va-et-vient (swapping) et algorithmes de remplacement de pages.

Principes de fonctionnement des caches mémoire et de leur mise en oeuvre dans les architectures modernes à base de processeurs multi-coeurs. Techniques logicielles d'exploitation des caches mémoire pour l'optimisation des performances des systèmes.

3. Gestion du parallélisme dans un noyau de système

Etude du problème de l'exclusion mutuelle pour l'accès cohérent à des ressources partagées entre entités d'exécution concurrentes.
Exemple des mécanismes de synchronisation de type *mutex* dans le contexte des applications multi-threads.

Etude de réalisations d'un mécanisme de synchronisation de type *mutex* dans un noyau de système basées sur les méthodes suivantes :

- le masquage des interruptions sur un mono-processeur
- le masquage de la préemption de thread sur un mono-processeur
- l'utilisation d'instructions atomiques et d'instructions de type "test-and-set" dans les architectures multi-processeurs.

Etude comparée des politiques de synchronisation de type " coarse-grained locking " et de type " fine-grained locking ".
Problème de l'inversion de priorité et ses solutions.
Problème de l'inter-blocage ("deadlock" en anglais) et ses solutions.

Le cours sur la mise en oeuvre de la concurrence et des mécanismes de synchronisation est basé sur un exemple concret (écrit en langage C) d'un allocateur mémoire dans un noyau de système.

4. Virtualisation de Systèmes
Historique et objectifs de la virtualisation de systèmes : utilisation optimale des ressources, exécution simultanée de plusieurs systèmes hétérogènes sur une seule machine, etc.

Description des notions de machine virtuelle, de virtualisation hébergée ("hosted virtualisation") ou autonome ("standalone virtualisation"), de systèmes invités et de systèmes hébergeur, d'hyperviseur.

Virtualisation par émulation transparente du matériel - exemple de Qemu.

Virtualisation transparente du matériel (VMWare) :
- virtualisation du CPU - problème des instructions critiques
- virtualisation de MMU par la technique des “shadow page tables”
- virtualisation des entrées/sorties par émulation de périphériques.

Para-virtualisation (Xen) et entrées/sorties à travers des périphériques virtuels (Ethernet, disques).

Support matériel de la virtualisation de système.
Exemple de l'architecture Intel-VT intégrant le support de machine virtuelles (extension de la virtualisation VMX).
Virtualisation de périphériques PCI et fonctions PCI virtuelles partagées entre systèmes invités.
Exemple de Ethernet et des communications entre machines virtuelles et/ou des machines virtuelles avec l'extérieur.
Etude de la mise en œuvre optimisée de transferts de données entre les Machines

Virtuelles clientes et les serveurs d'un Data Center munis de disques SSD.

Etude de la virtualisation de systèmes dans les systèmes embarqués s'appuyant sur l'exemple des téléphones intelligents ("Smartphones") fonctionnant sous le système Android de Google.
Conclusion sur l'évolution des techniques de la virtualisation de système, des défis posés par sa diffusion (logiciel libre, open source, etc.)

5) **Mise en oeuvre de la virtualisation de systèmes dans le contexte de l'infrastructure des télécommunications et de l'Internet**
- Notion de Network Function Virtualization (NFV). Etude des avantages de la virtualisation de fonctions réseau, et des défis posés par leur mise en oeuvre dans des machines virtuelles. Exemple de la virtualisation d'un Broadband Remote Access Server (BRAS).
- Principe d'un commutateur réseau virtuel, intégration de la solution Open Virtual Switch (OVS) dans un hyperviseur.
- Notion de Software Defined Network (SDN) permettant le contrôle centralisé des infrastructures réseau, de la solution OpenFlow et de sa mise en oeuvre dans des infrastructures réseau
- Présentation des Smart NIC, cartes réseau intelligentes qui incluent des services de haut niveau comme la commutation et/ou le routage de paquets, services couramment intégrés dans l'hyperviseur des solutions de virtualisation de réseaux
- Présentation du DPDK (Data Plane Development Kit)
Projet open-source pour le développement de piles de protocoles réseau à haut-débit.
Mise en oeuvre du DPDK dans les infrastructures de réseau virtualisées pour en optimiser les performances.

Description des modalités de validation

Examen écrit

Bibliographie

Titre	Auteur(s)
Les systèmes d'exploitation, Unix, Linux et Windows XP, avec C et Java, Dunod 2003 (566 pages), ISBN : 2100071890	S. Bouzefrane
Modern Multithreading Wiley 2005 (465 pages)	R. H. Carver, Kuochung Tai
Unix, Linux et les systèmes d'exploitation, Dunod 2004 (429 pages)	M. Divay
Le noyau Linux, O' Reilly 2006 (1007 pages), ISBN 2841772438	D. Bovet, M. Cesati
Virtual Machines - versatile platforms for systems and processes, Elsevier	J.E. Smith, R. Nair
Hardware and Software Support for Virtualization - Morgan and Claypool Life Sciences ISBN: 1627056939	E. Bugnon, J. Nich, D. Tsafrir