LINGI 1113: Systèmes informatiques 2

Mission 7: Systèmes d'exploitation embarqués, Sécurité et Systèmes répartis

ÉCOLE
POLYTECHNIQUE
DE LOUVAIN

UCL

3 thèmes que nous ne ferons qu'introduire

- Systèmes d'exploitation embarqués: 13.1 à 13.3; pp 628-655
- Sécurité: 15.1; pp 660-666
- Systèmes répartis: 16.1 et 16.4 ; pp 698-709 et 715-718

Vous pourrez les approfondir dans des cours de master.

Comme nous n'irons pas en profondeur dans le cadre de cette mission, elle ne fait pas partie de la matière d'examen, mais je vous en recommande la lecture.

Systèmes d'exploitation embarqués

La majorité des systèmes informatiques sont des systèmes embarqués

Table 13.1 Examples of Embedded Systems and Their Markets [NOER05]

Market	Embedded Device
Automotive	Ignition system Engine control Brake system
Consumer electronics	Digital and analog televisions Set-top boxes (DVDs, VCRs, Cable boxes) Personal digital assistants (PDAs) Kitchen appliances (refrigerators, toasters, microwave ovens) Automobiles Toys/games Telephones/cell phones/pagers Cameras Global positioning systems
Industrial control	Robotics and controls systems for manufacturing Sensors
Medical	Infusion pumps Dialysis machines Prosthetic devices Cardiac monitors
Office automation	Fax machine Photocopier Printers Monitors Scanners

Systèmes d'exploitation embarqués

Vous en avez programmé 1, le PIC

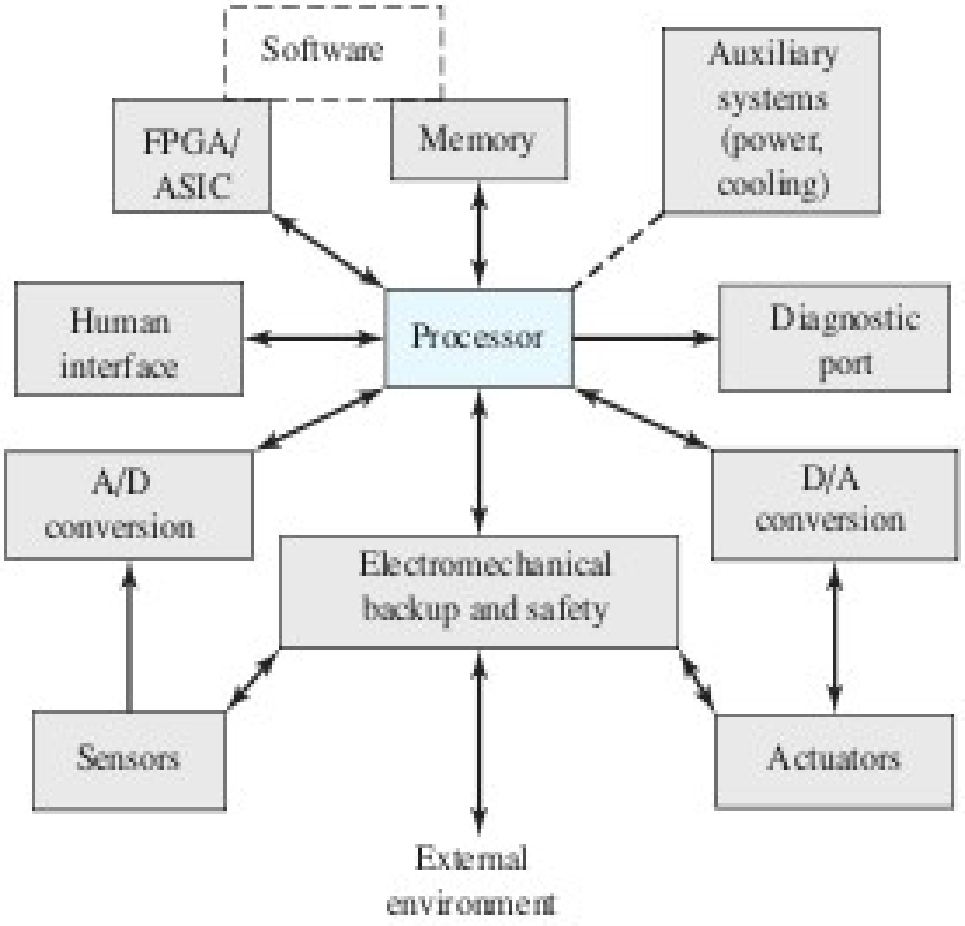


Figure 13.1 Possible Organization of an Embedded System

Systèmes d'exploitation embarqués

Vous en avez programmé un, mais sans OS

Il existe des OS adaptés ou spécifiques pour systèmes embarqués

Qualités spécifiques recherchées:

- Support pour applications temps réel
- Support pour applications réactives
- Configurabilité
- Support pour des dispositifs d'entrées/sorties variés
- Systèmes de protection simplifiés
- Usage des interruptions par les applications

Approches

- Pas d'OS
- OS « normal » adapté (Real-Time LINUX)
- OS spécifiquement conçu pour systèmes embarqués

Systèmes d'exploitation embarqués

Caractéristiques des OS conçus pour systèmes embarqués

- Commutation rapides des processus/threads
- Ordonnacement temps réel
- Petite taille
- Réponse rapide aux interruptions externes (10µs)
- Temps où les interruptions sont désactivées minimalisé
- Temps d'exécution prévisible et borné des fonctions de l'OS
- Inclut une horloge temps réel

parfois

- Gestion dynamique de la mémoire, par blocs de taille fixe ou variable (malloc/free)
- Fichiers séquentiels
- Généralement pas de swapping/mémoire virtuelle, mais s'il y en a possibilité de verrouilles les processus en mémoire.

Systèmes d'exploitation embarqués

Un exemple: eCos (embedded Configurable operating system)

- Open-source (ecos.sourceware.org)
- Environnement de développement « host-target »
- Porté sur de nombreuses architectures
- Configurable: ce n'est pas un tout monolithique mais des modules qu'on peut assembler en fonction des besoins; on compile les morceaux nécessaires et on les lie (avec l'aide de make)
- L'OS est lié avec l'application

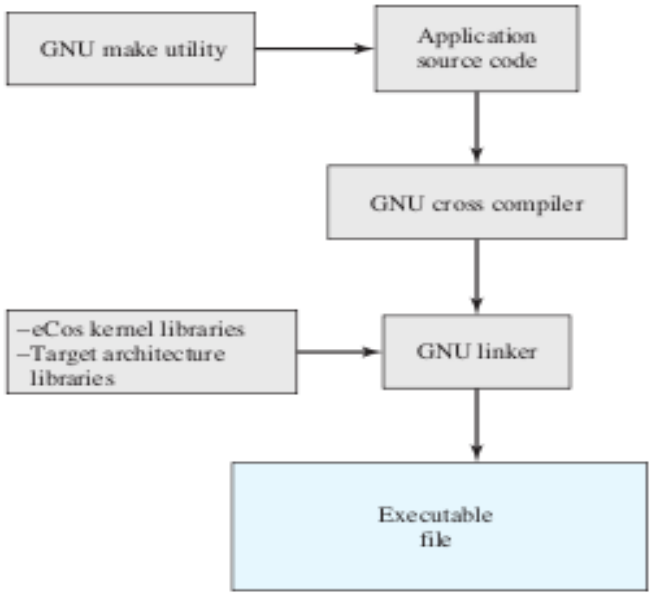


Figure 13.4 Loading an eCos Configuration

Systèmes d'exploitation embarqués

Les composants d'eCos

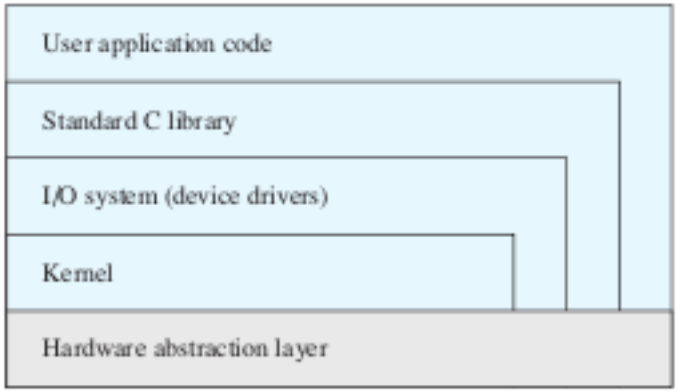


Figure 13.5 eCos Layered Structure

- HAL (Hardware abstraction layer)
  - Architecture (p.ex. ARM)
  - Variante (p. ex. ARM7TDMI)
  - Plate-forme: la carte utilisée

Systèmes d'exploitation embarqués

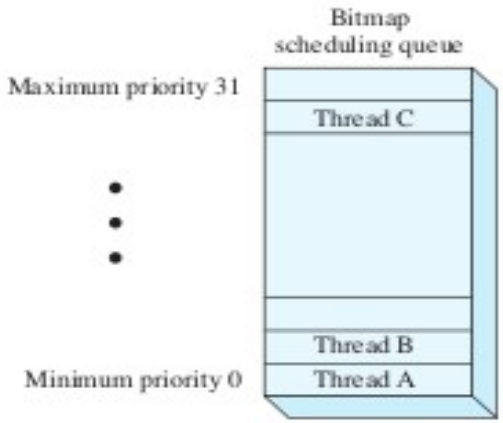
Les composants d'eCos (suite)

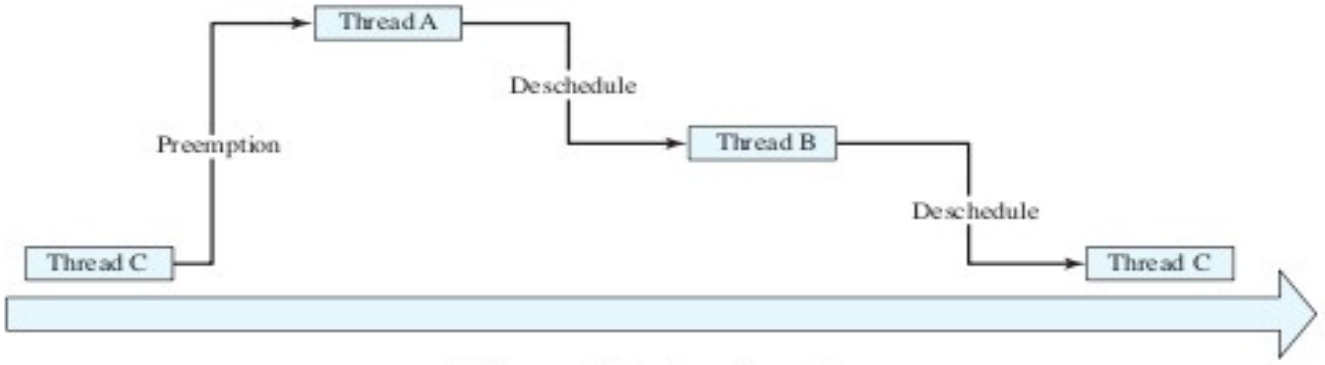
- Kernel: inclut les fonctions de base nécessaires à une application multithread
  - Création dynamique de threads
  - Gestion des threads (changement de priorité)
  - Choix d'ordonnanceurs
  - Primitives de synchronisation des threads
  - Intégration avec le mécanisme d'interruption
- Device drivers
  - ISR (Interrupt service routines): peuvent même interrompre l'ordonnanceur, peut désigner un DSR
  - DSR (Deferred Service routine)s'exécute quand l'ordonnanceur a fini son travail, peut réveiller des threads bloqués
  - Treads: ici se fait le vrai traitement, sous contrôle de l'ordonnanceur

Systèmes d'exploitation embarqués

Les composants d'eCos (suite)

- Les ordonnanceurs
  - bitmap



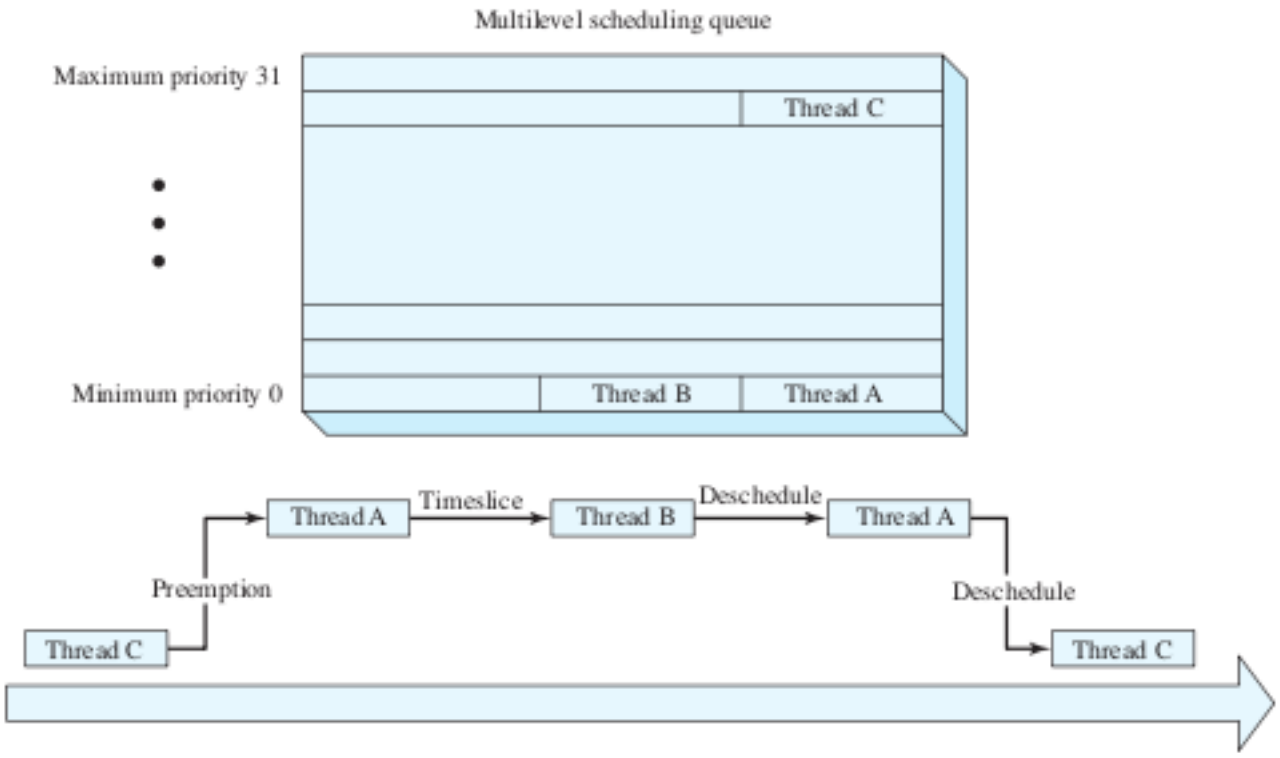


(a) Bitmap scheduler thread operation

Systèmes d'exploitation embarqués

Les composants d'eCos (suite)

- Les ordonnanceurs
  - Multilevel queue



(b) Multilevel queue scheduler thread operation

Figure 13.7 eCos Scheduler Options

Sécurité

Objectifs de la sécurité: CIA
(confidentiality, Integrity, Availability)

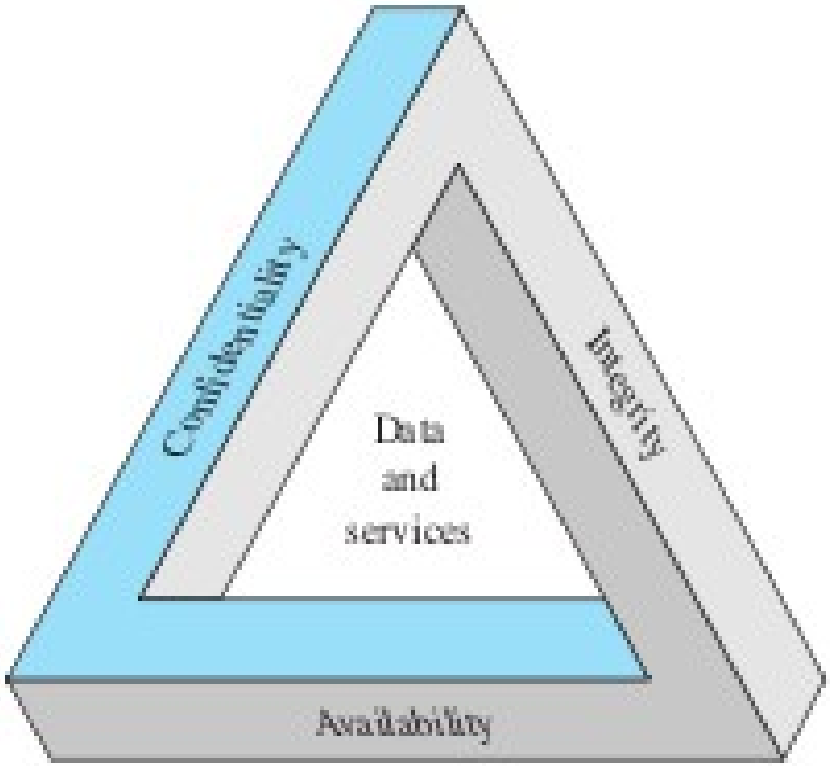


Figure 14.1 The Security Requirements Triad

Sécurité

Table 14.1 Threat Consequences, and the Types of Threat Actions That Cause Each Consequence. Based on RFC 2828

Threat Consequence	Threat Action (attack)
Unauthorized Disclosure A circumstance or event whereby an entity gains access to data for which the entity is not authorized.	**Exposure:** Sensitive data are directly released to an unauthorized entity. **Interception:** An unauthorized entity directly accesses sensitive data traveling between authorized sources and destinations. **Inference:** A threat action whereby an unauthorized entity indirectly accesses sensitive data (but not necessarily the data contained in the communication) by reasoning from characteristics or byproducts of communications. **Intrusion:** An unauthorized entity gains access to sensitive data by circumventing a system's security protections.
Deception A circumstance or event that may result in an authorized entity receiving false data and believing it to be true.	**Masquerade:** An unauthorized entity gains access to a system or performs a malicious act by posing as an authorized entity. **Falsification:** False data deceive an authorized entity. **Repudiation:** An entity deceives another by falsely denying responsibility for an act.
Disruption A circumstance or event that interrupts or prevents the correct operation of system services and functions.	**Incapacitation:** Prevents or interrupts system operation by disabling a system component. **Corruption:** Undesirably alters system operation by adversely modifying system functions or data. **Obstruction:** A threat action that interrupts delivery of system services by hindering system operation.
Usurpation A circumstance or event that results in control of system services or functions by an unauthorized entity.	**Misappropriation:** An entity assumes unauthorized logical or physical control of a system resource. **Misuse:** Causes a system component to perform a function or service that is detrimental to system security.

Sécurité

Table 14.2 Computer and Network Assets, with Examples of Threats

	Availability	Confidentiality	Integrity
Hardware	Equipment is stolen or disabled, thus denying service.		
Software	Programs are deleted, denying access to users.	An unauthorized copy of software is made.	A working program is modified, either to cause it to fail during execution or to cause it to do some unintended task.
Data	Files are deleted, denying access to users.	An unauthorized read of data is performed. An analysis of statistical data reveals underlying data.	Existing files are modified or new files are fabricated.
Communication Lines	Messages are destroyed or deleted. Communication lines or networks are rendered unavailable.	Messages are read. The traffic pattern of messages is observed.	Messages are modified, delayed, reordered, or duplicated. False messages are fabricated.

Sécurité

Table 14.3 Some Examples of Intruder Patterns of Behavior

(a) Hacker

1. Select the target using IP lookup tools such as NSLookup, Dig, and others.
2. Map network for accessible services using tools such as NMAP.
3. Identify potentially vulnerable services (in this case, pcAnywhere).
4. Brute force (guess) pcAnywhere password.
5. Install remote administration tool called DameWare.
6. Wait for administrator to log on and capture his password.
7. Use that password to access remainder of network.

(b) Criminal Enterprise

1. Act quickly and precisely to make their activities harder to detect.
2. Exploit perimeter through vulnerable ports.
3. Use Trojan horses (hidden software) to leave back doors for reentry.
4. Use sniffers to capture passwords.
5. Do not stick around until noticed.
6. Make few or no mistakes.

(c) Internal Threat

1. Create network accounts for themselves and their friends.
2. Access accounts and applications they wouldn't normally use for their daily jobs.
3. E-mail former and prospective employers.
4. Conduct furtive instant-messaging chats.
5. Visit Web sites that cater to disgruntled employees, such as f'dcompany.com.
6. Perform large downloads and file copying.
7. Access the network during off hours.

Sécurité

Table 14.4 Terminology of Malicious Programs

Name	Description
Virus	Malware that, when executed, tries to replicate itself into other executable code; when it succeeds the code is said to be infected. When the infected code is executed, the virus also executes.
Worm	A computer program that can run independently and can propagate a complete working version of itself onto other hosts on a network.
Logic bomb	A program inserted into software by an intruder. A logic bomb lies dormant until a predefined condition is met; the program then triggers an unauthorized act.
Trojan horse	A computer program that appears to have a useful function, but also has a hidden and potentially malicious function that evades security mechanisms, sometimes by exploiting legitimate authorizations of a system entity that invokes the Trojan horse program.
Backdoor (trapdoor)	Any mechanisms that bypasses a normal security check; it may allow unauthorized access to functionality.
Mobile code	Software (e.g., script, macro, or other portable instruction) that can be shipped unchanged to a heterogeneous collection of platforms and execute with identical semantics.
Exploits	Code specific to a single vulnerability or set of vulnerabilities.
Downloaders	Program that installs other items on a machine that is under attack. Usually, a downloader is sent in an e-mail.
Auto-rooter	Malicious hacker tools used to break into new machines remotely.
Kit (virus generator)	Set of tools for generating new viruses automatically.
Spammer programs	Used to send large volumes of unwanted e-mail.
Flooders	Used to attack networked computer systems with a large volume of traffic to carry out a denial-of-service (DoS) attack.
Keyloggers	Captures keystrokes on a compromised system.
Rootkit	Set of hacker tools used after attacker has broken into a computer system and gained root-level access.
Zombie, bot	Program activated on an infected machine that is activated to launch attacks on other machines.
Spyware	Software that collects information from a computer and transmits it to another system.
Adware	Advertising that is integrated into software. It can result in pop-up ads or redirection of a browser to a commercial site.

Sécurité

Virus et vers

Virus:

- Code qui se rajoute à un programme, composé de
  - Mécanisme d'infection
  - Mécanisme de déclenchement
  - Action

Exemple de mécanisme d'infection: se rajouter au début du code d'un programme, puis compresser le reste pour que la taille du fichier ne change pas. Quand le programme démarre, si pas déclenchement, décompresser et exécuter le programme

Sécurité

Virus:

- Peut infecter
  - Boot sector (facile à protéger)
  - Exécutable (facile à protéger sauf les siens propres)
  - Macros
- Mécanismes de camouflage
  - Chiffrement de chaque instance avec une clé secrète aléatoire différente
  - Stealth virus: on ajoute mécanismes pour le rendre invisible (mofifier ls)
  - Polymorphyque: mute à chaque infection (quelques changement pourne pas correpondre à la signature connue)
  - Métamorphique: idem mais se modifie entièrement

Sécurité

Vers:

- Programme qui se recopie sur des machines distantes

Bot: (robot)

- Mécanisme de propagation (comme virus et vers)
- Contrôlable à distance
- Utilisé, p. ex. pour déni de service (des milliers de sites (voire plus) attaquent simmultanément un même site)

Sécurité

Les premières lignes de défense d'un OS

- Sur les réseaux:chiffrement (intégrité et confidentialité; ne garantit pas disponibilité)
- Sur les ordinateurs eux-mêmes: authentification et protection des fichiers
  - Protection fichiers: p. ex exécutables non modifiables, répertoire courant pas dans PATH , donc non exécutable sauf si « ./ » (empêcher un intrus de placer un faux ls dans le répertoire courant)
  - Authentification: identifiant (~carte de visite) + vérification (~carte d'identité avec photo): mot de passe ou autre (carte à puce, biométrie,...)

Sécurité

Authentification unix:

hashed

(password +random salt)

Notes

- /etc/passwd
  + /etc/shadow
- Passwd: infos publiques
- Shadow: infos réservées à root (date création:jours depuis 1/1/70), durée de vie, etc)
- LINUX $hash$salt$encrypt

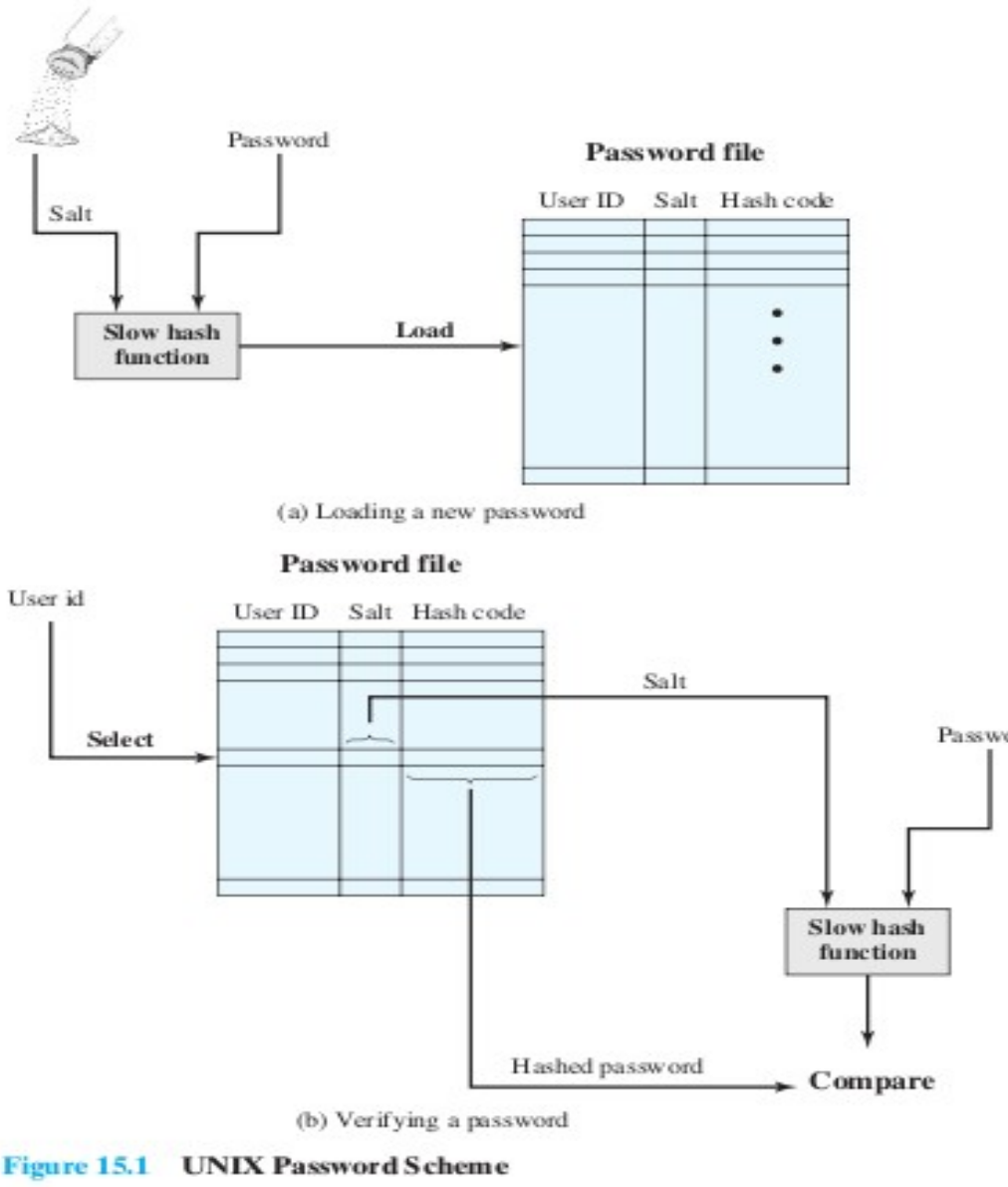


Figure 15.1 UNIX Password Scheme

Client/server computing

Le principe: certains services/fonctionalités/ressources sont sur des serveurs distants

- Base de donnée
- Imprimante
- Fichiers
- Application quelconque

Accès:

- à travers réseau: usage de ports divers, parfois bloqués par firewall
- Via port http ou https (web services)

Client/server computing

implémentation: appel de procédure à distance

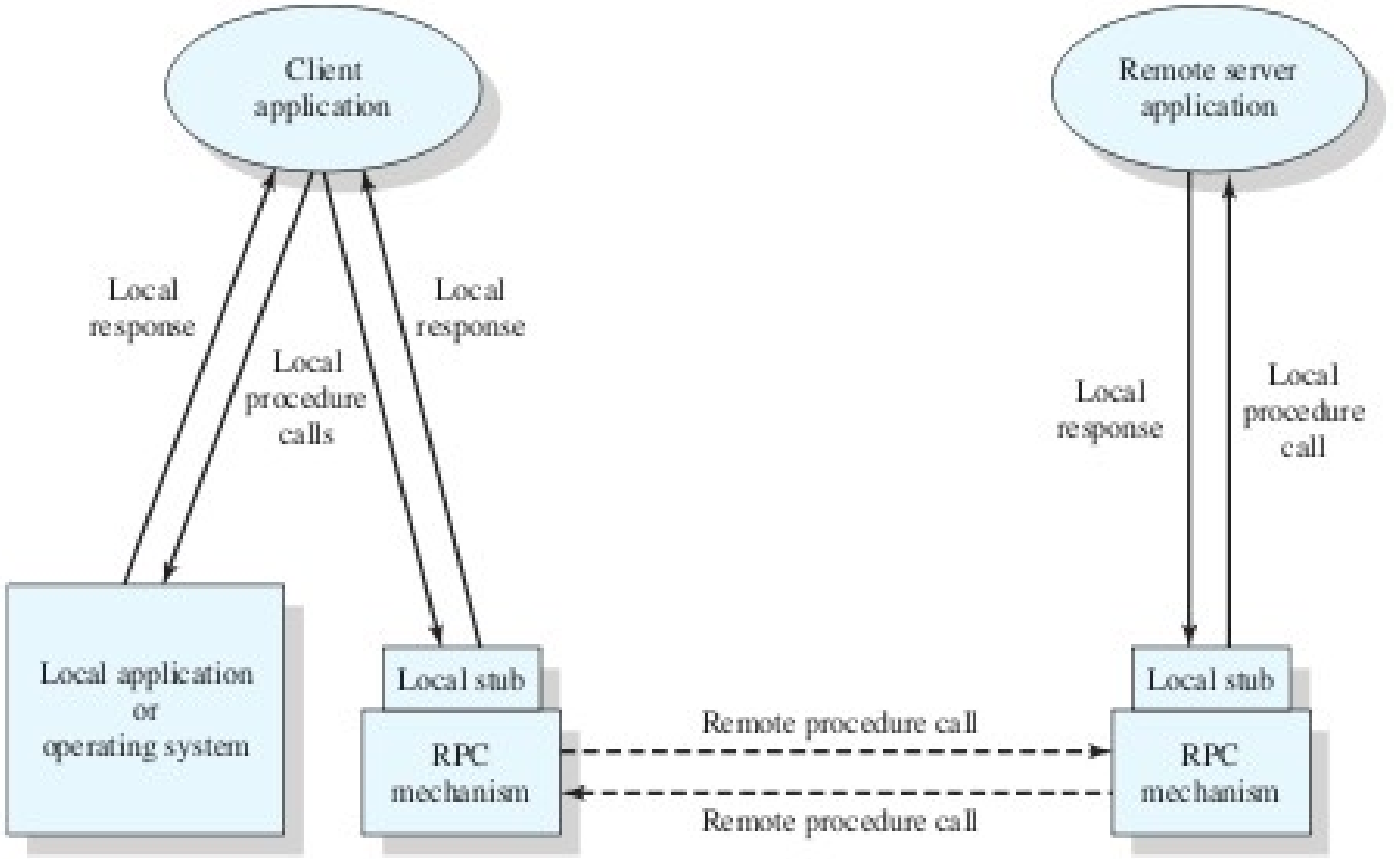
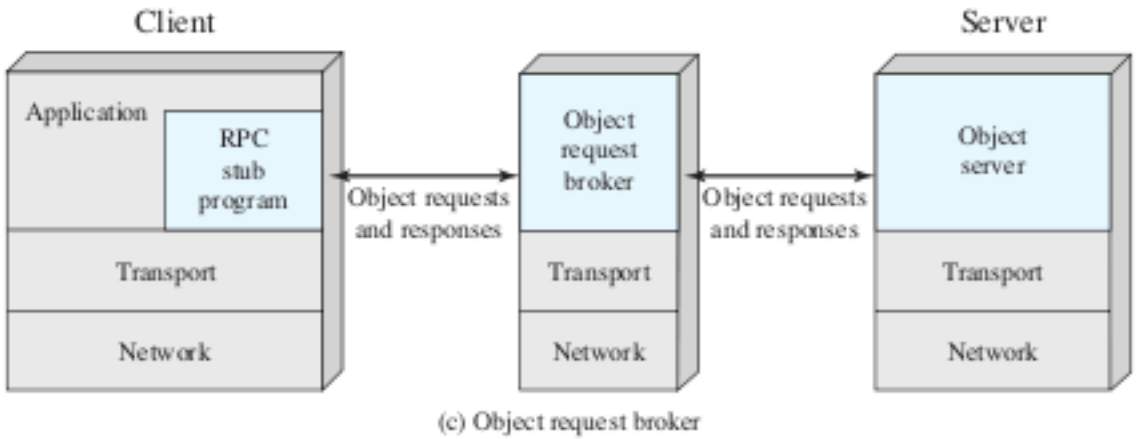


Figure 16.12 **Remote Procedure Call Mechanism**

Client/server computing

Problèmes techniques:

- Format des données dans ordinateurs différents: utilisation d'un format « réseau » normalisé et double conversion
- Identification du port destination (et parfois de l'adresse IP du serveur destination): serveur intermédiaire bien connu
  - (local à une machine (portmapper)
  - Local à un réseau (object request broker)



(c) Object request broker

Figure 16.10 **Middleware Mechanisms**

Entrées/sorties

Buffering pour performance

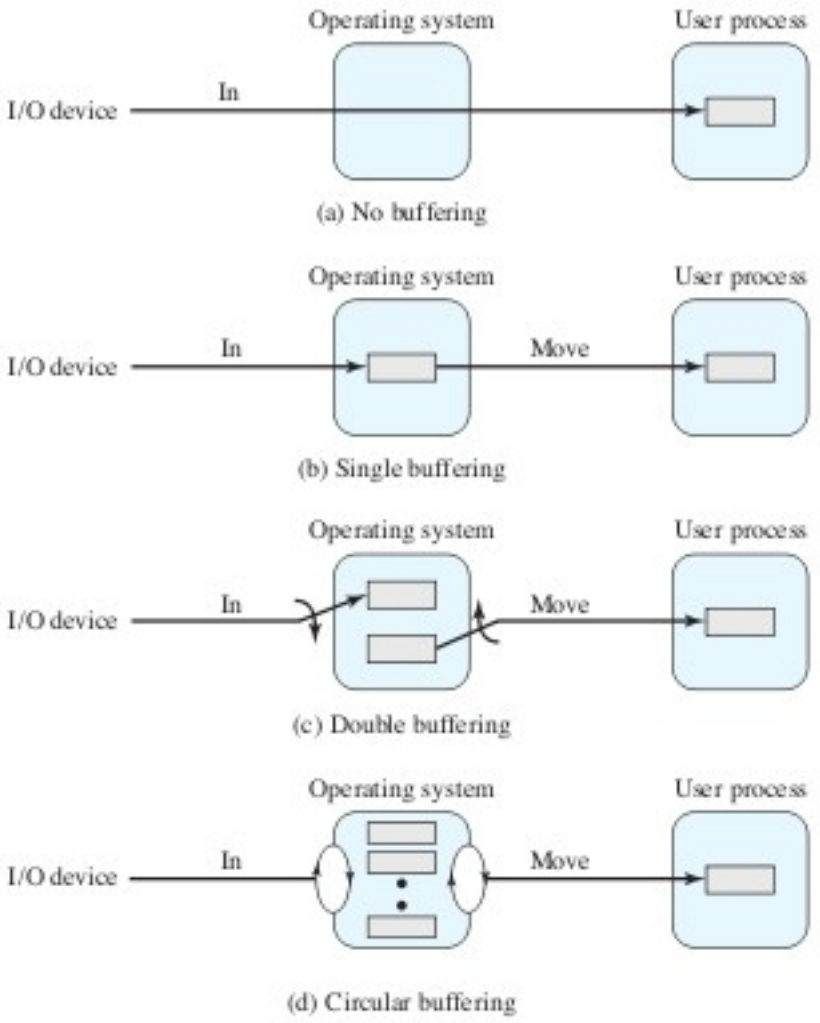


Figure 11.5 **I/O Buffering Schemes (input)**

Transfert disques

Timing

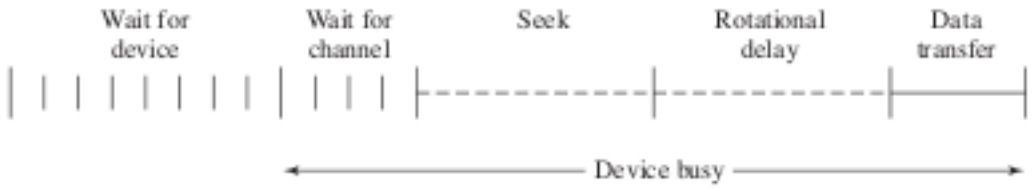


Figure 11.6 Timing of a Disk I/O Transfer

Ordonnancement des transferts:

- FIFO: mouvements du bras aléatoires, mais « fair »
- Shortest Service Time First (d'abord aux plus proches)
- Scan: comme ascenseur: ne change de sens que si plus de destinations dans le même sens
- C-scan: ne scanne que dans un sens

  Ces trois peuvent être injustes: si un processus à beaucoup d'activité sans movement, les autres n'auront rien

Transfert disques

Pour être efficace mais juste:

- Fscan:2 queues: on en scanne une et les nouvelles requetes vont dans l'autre puis on inverse
- N-step-scan: plusieurs queues de taille N, limitée: quand une est pleine, on la scanne et on en remplit une autre. S'il n'y en a plus qu'une, on la scanne sans ettendre qu'elle se remplisse.
  (NB: si N=1, c'est du FIFO!)

Autre façon d'améliorer les performances: parallelisme:

RAID: Redundant Array of Inexpensive Disks: 6 variantes dont 4 utilisées

RAIDS

Raid 0: disk striping - rapide mais si un disque crashe on perd toutes les données

Raid 1: mirroring – sûr mais cher: 2 fois plus de disques: souvent 2 disques mais tout nombre pair est possible; striping entre les disques d'une des copies

Raid 2: striping + redondance par code de Hamming (comme mémoires ECC) bit à bit : disques supplémentaires qui contiennet les bits de parité; permet de détecter 2 erreur et en corriger une

Raid 3: si on ne tient pas à se protéger d'erreurs ne concernant qu'un seul bit mais continuer à travailler si un des disques crashe et récupérer ses données, 1 seul bit de parité suffit

RAIDS

Raid 4: comme raid 3, mais parité au niveau du bloc, pas du bit

Raid 5: dans Raid 4 le disque de parité est sur-utilisé: les blocs de parité sont répartis entre tous les disques, chacun son tour

Raid 6: idem, mais avec 2 bits de parité dans 3 disques différents: il faut que 3 disques crashent pour perdre des données

cache disque

Et bien sûr, on peut accélérer les transferts disques en cachant des blocs en mémoire

Remplacement:

- Least recently used (on remet en tête de liste à chque usage et on sacrifie la queue de liste)
- Least frequently used: faut des compteurs, mais protège sans raison des blocs inactifs qui ont été actifs dans le passé
- Frequency based replacement: la file est divisée en 3 sections: les derniers arrivés sont traités en LRUet on ne les éjecte pas, les moyens en LFU mais on ne les éjecte pas, les plus vieux en LFU et on éjecte les moins utilisés

cache disque

- Frequency based replacement: la file est divisée en 3 sections: les derniers arrivés sont traités en LRUet on ne les éjecte pas, les moyens en LFU mais on ne les éjecte pas, les plus vieux en LFU et on éjecte les moins utilisés

  Explications

  - Première zone: pour ne pas défavoriser les nouveaux arrivants dont le compteur est encore bas (le compteur reste à 1, mais on les remet en tête de liste à chaque usage)
  - 2me zone idem car ceux là commence avec un compteur à 1: faut qu'ils aient le temps de le gonfler s'ils sont actifs
  - 3me zone, là, ils n'y a que des « vieux » et seuls les « vieux actifs » sont gardés

Exemple UNIX

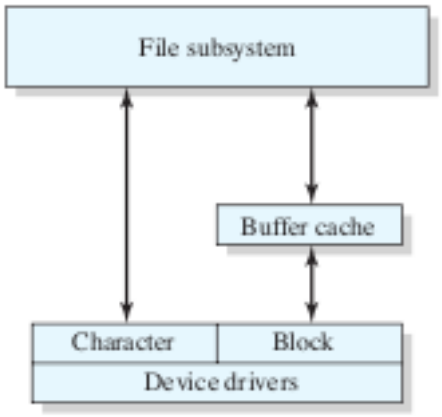


Figure 11.12 **UNIX I/O Structure**

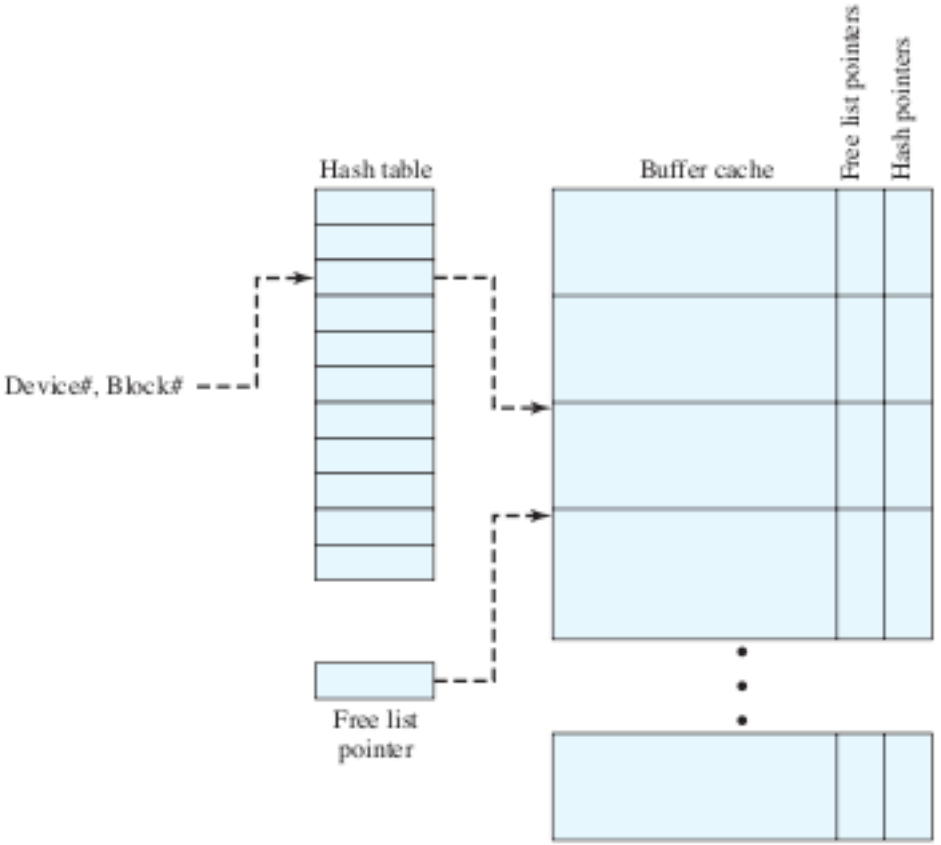


Figure 11.13 **UNIX Buffer Cache Organization**

Systèmes de fichiers

Deux approches:

- **Fichiers de caractères**; ce sont les applications (p. ex. bases de données) qui en définissent la structure et les protections (rwx) +setuid qui protègent contre accès sauvage
  (UNIX et autres OS récents)

- **Fichiers structurés en enregistrements**; les méthodes d'accès sont inclues dans l'OS
  (mainframes traditionnels)

  enregistrement = séquence de champs

Systèmes de fichiers

Les fichiers structurés

- **Empilement**: les entregistrements sont simplement insérés les uns à la suite des autres; ils peuvent être différents et doivent inclure des descripteurs permettant de les identifier; extraction d'un enregistrement par recherche exhaustive
- **Fichiers séquenciels**: tous les enregistrements sont de même type (mêmes champs de même longueur. Un champ (le 1er) sert de clé. Les enregistrements sont triés. Pour ajouter des enregistrements, on les met dans un fichier séparé qu'on intègre régulièrement dans le fichier principal

Systèmes de fichiers

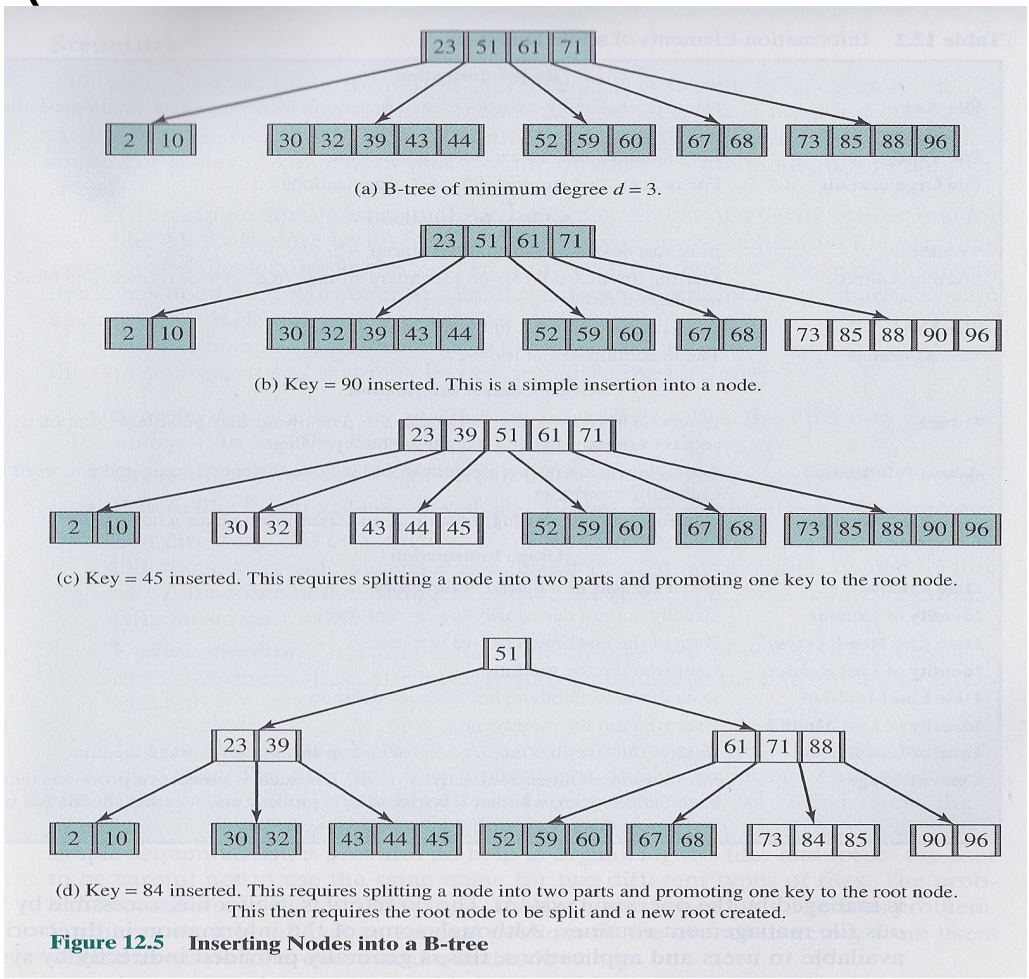
Les fichiers structurés (suite)

- **Fichiers séquenciels indexés**: idem mais on ajoute un index, qui permet d'arriver plus vite à l'enregistrement cherché (sans cela faut recherche dichotomique)
  pour ajouter des enregistrements, chaque enregistrement inclut un pointeur vers un fichier de débordement: on met donc le nouvel enregistrement dans fichier de débordement et on met à jour ce pointeur dans enregistrement précédent
- **Fichiers indexés**: pas triés; un index principal qui permet d'atteindre tous les enregistrements + autres index (partiels) sur autres champs
- **Fichiers en accès direct** : avec tâble de hachage

Systèmes de fichiers

Les fichiers structurés (suite)

- **B-Trees** *(cfr cours de structures de données)*



Systèmes de fichiers

Organisation d'un système de fichiers typiques

- Placé sur un volume (disque, partition, slice)
- Contient des répertoires et des fichiers

- Répertoire: lien entre nom du fichier et stucture de données sur le volume qui permet de retrouver le contenu du fichier et ses attributs (propriétaire, date de création, droits d'accès, ACL, etc.); souvent organisés hiérarchiquement; souvent gérés comme des fichiers de contenu particulier
- Volume: contient informations sur volume (blocs libres), sur les fichiers et le contenu des fichiers

Systèmes de fichiers

Allocation de l'espace aux fichiers

Table 12.3 File Allocation Methods

	Contiguous	Chained	Indexed	
Preallocation?	Necessary	Possible	Possible	
Fixed or variable size portions?	Variable	Fixed blocks	Fixed blocks	Variable
Portion size	Large	Small	Small	Medium
Allocation frequency	Once	Low to high	High	Low
Time to allocate	Medium	Long	Short	Medium
File allocation table size	One entry	One entry	Large	Medium

- indexé avec allocation de taille variable: on alloue plusieurs blocs contigus, avec pointeur vers 1er et Nblocs

Systèmes de fichiers

Blocs libres: se gèrent à peu près comme un fichier

- Zones libres chaînées entre elles
- Index vers zones libres (constituées de plusieurs blocs) de tailles variables
- Simple liste des numéros de blocs libres

Droits d'accès aux fichiers

Contrôle d'accès de base:

- RWX pour owner, group et autres
- Les attributs du fichier incluent les 9 bits, l'identification du propriétaire et d'un groupe
- Un utilisateur peut être membre de plusieurs groupes
- Bits setuid, setgid

Contrôle d'accès plus général

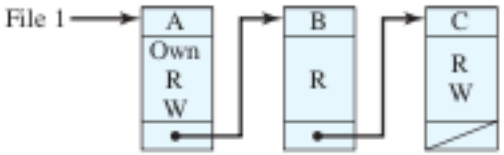
- Droits plus variés
- Structure logique matricielle: agents/objets, mais c'est une matrice creuse

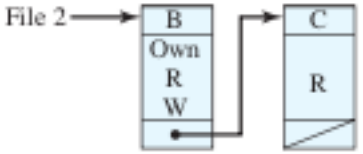
	File 1	File 2	File 3	File 4	Account 1	Account 2
User A	Own R W		Own R W		Inquiry Credit	
User B	R	Own R W	W	R	Inquiry Debit	Inquiry Credit
User C	R W	R		Own R W		Inquiry Debit

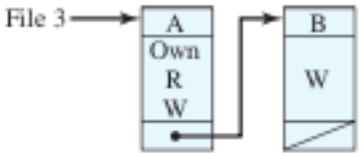
(a) Access matrix

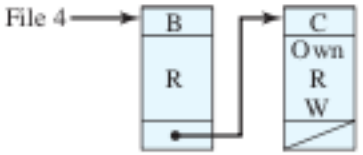
Droits d'accès aux fichiers

Structure logique matricielle: agents/objets, mais c'est une matrice creuse: représentation par colonnes (ACL) ou par lignes (capabilities)

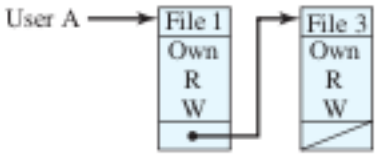


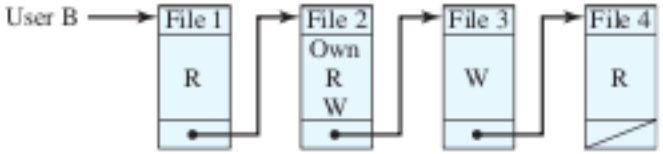


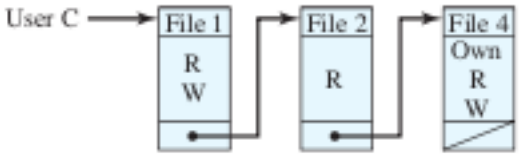




(b) Access control lists for files of part (a)







(c) Capability lists for files of part (a)

Figure 12.13 **Example of Access Control Structures**

Exemple UNIX

Stucture de données pour attributs des fichiers: **inodes**

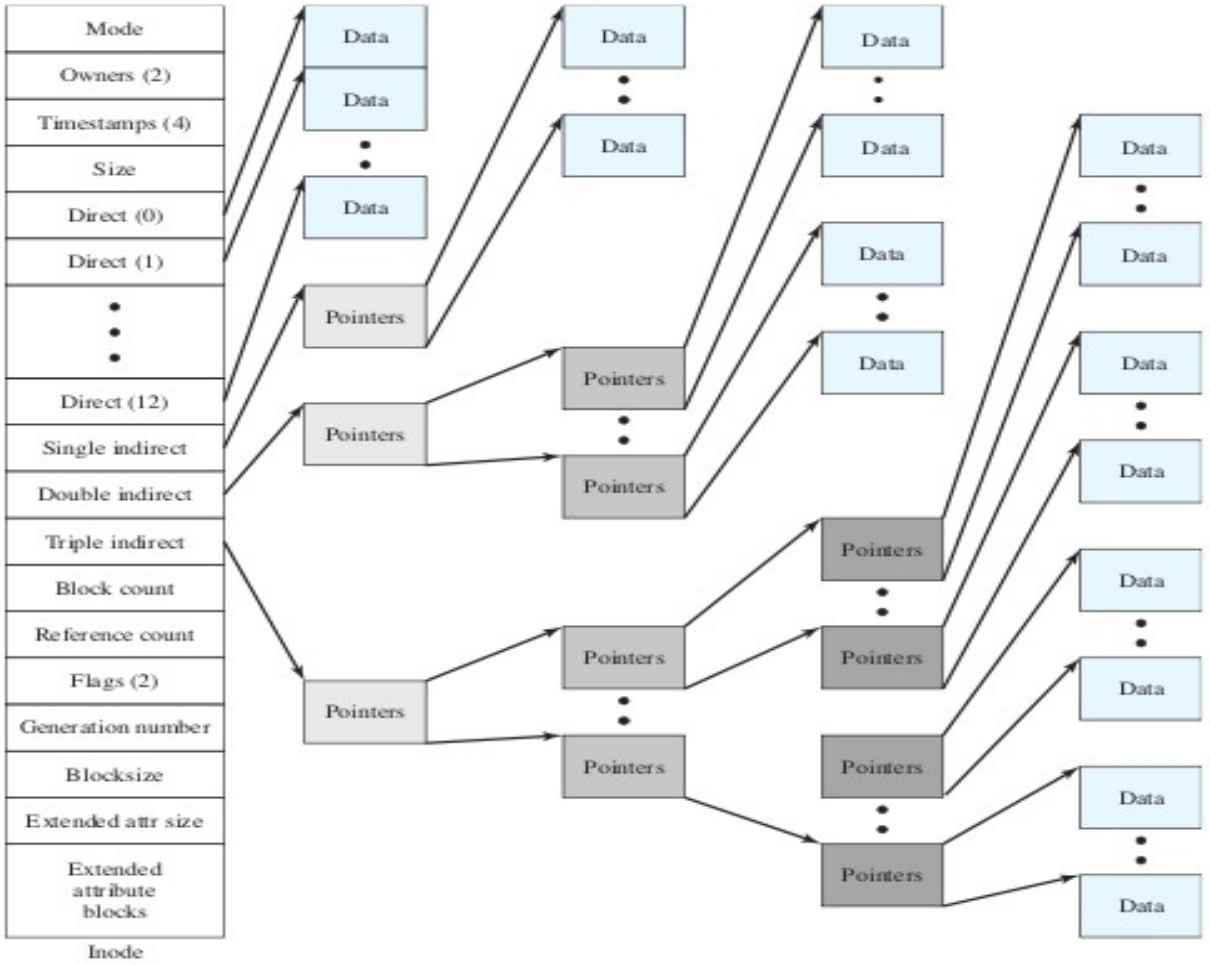


Figure 12.14 Structure of FreeBSD inode and File

Exemple UNIX

Inodes et répertoires

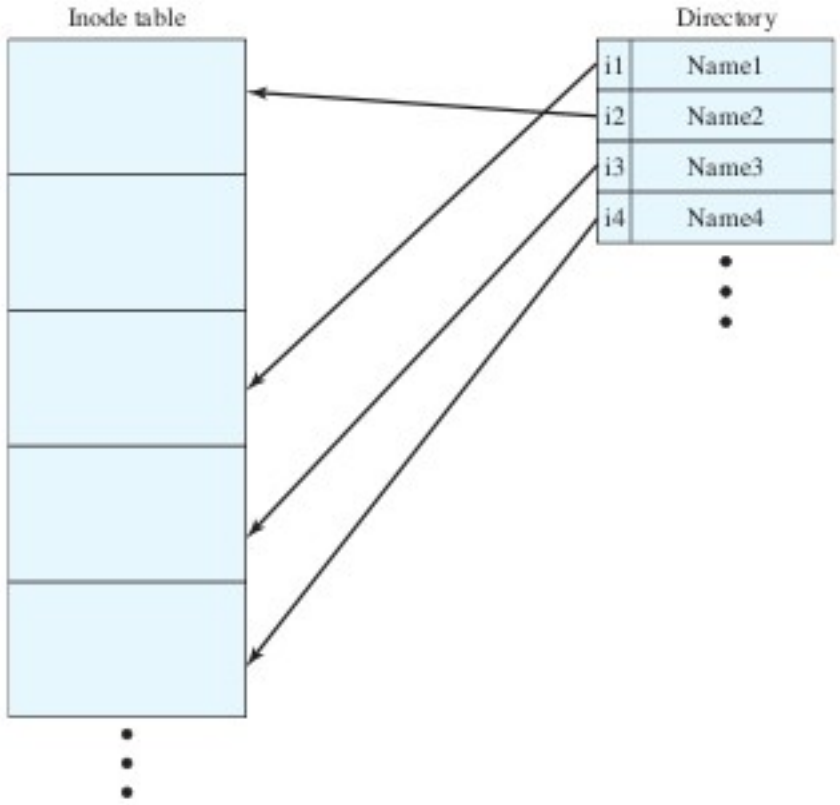


Figure 12.15 UNIX Directories and Inodes

Exemple UNIX

Systèmes de fichiers virtuels

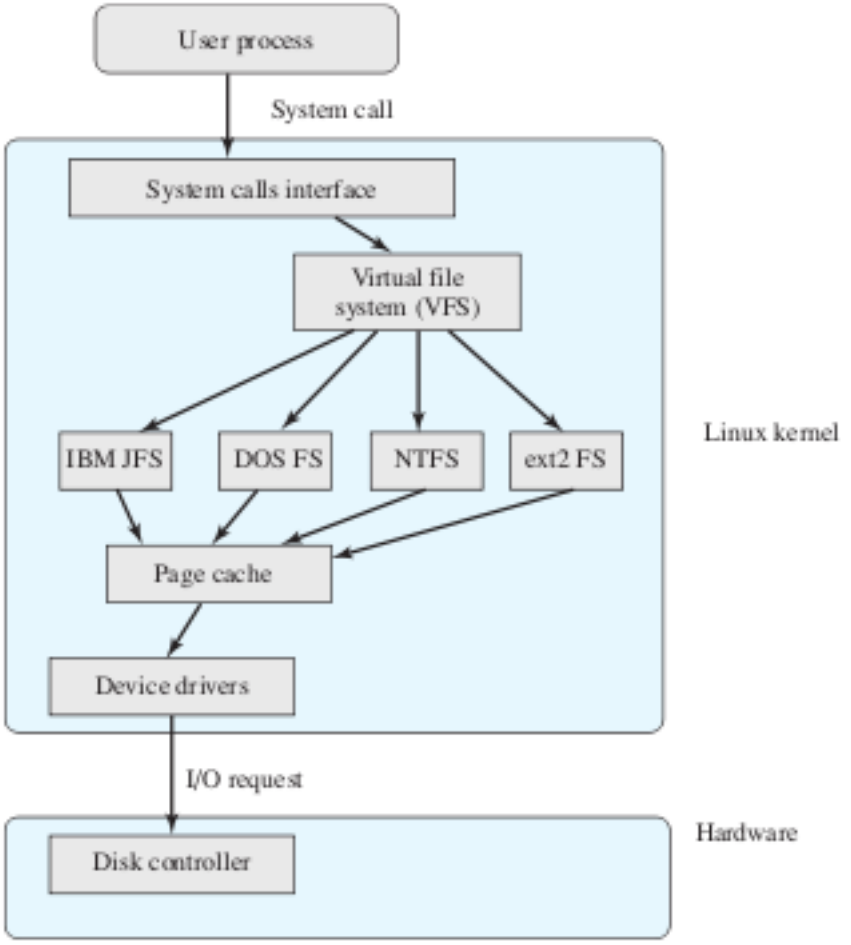


Figure 12.17 **Linux Virtual File System Context**

nouveautés

Systèmes de fichiers journalisés

- Éviter corruptions en cas de crashs
- Basés sur techniques de 2 phase commit
- Dans NTFS (windows), ZFS (solaris)

Systèmes de fichiers de grande taille

- ZFS= Zetabyte file system (données en 128 bits
- NTFS aussi

Raid software inclus (NTFS, ZFS)