Système d'Exploitation

Système de Fichiers
Machine virtuelle

Juan Angel Lorenzo del Castillo
juan-angel.lorenzo-del-castillo@cyu.fr

Contributions de :
Stefan Bornhofen
Florent Devin
Taisa Guidini Goncalves
Mariem ALLOUCH MAHDI

Plan

Système de gestion des fichiers

Partiellement tiré du cours de Stefan Bornhofen

Système de Gestion de Fichiers (SGF)

Un **Système de Gestion de Fichiers (SGF)** est une structure de données permettant de stocker les informations et de les organiser dans des fichiers sur des mémoires secondaires.

- Le SGF est la partie la plus visible d'un SE.
- Le SGF se charge de gérer le stockage et la manipulation des fichiers sur le matériel physique. Il offre également les primitives pour manipuler les fichiers.
- Les tâches effectuées par le SGF :
  - Fournit une interface "conviviale" pour faciliter la manipulation des fichiers.
  - Gestion de l'organisation des fichiers sur le disque (allocation de l'espace disque aux fichiers).
  - Gestion de l'espace libre.
  - Gestion des fichiers dans un environnement multi-utilisateurs.
- Quelques systèmes de fichiers connus :
  - Linux: Ext3, Ext4, XFS, Btrfs
  - Windows: FAT, NTFS

Fichiers

Un **fichier** est l'unité de stockage logique persistante, non volatile, mise à la disposition des utilisateurs pour l'enregistrement de leurs données : c'est l'unité d'allocation.

- Les fichiers sont gérés par le système d'exploitation.
- La manière dont ils sont
  - structurés
  - nommés
  - utilisés
  - protégés

  est à la charge du SE.

Remarque : Le SE fait la correspondance entre le fichier et le système binaire utilisé lors du stockage de manière transparente pour les utilisateurs.

Fichiers

Attributs des fichiers

- Exemple : UID, GID, droits d'accès, dates (dernière modification), taille, type (fichier, repertoire, lien symbolique, périphérique).
- Affichage des attributs : `ls -l` ou `stat` *`fichier`*.
- Sous Linux : numéro du fichier (unique) : numéro d'i-nœud (*i-node*)
  - Affichage de l'i-nœud : `ls -i` *`fichier`*.
- Concernant les droits d'accès, sous Unix, il existe 3 niveaux de confidentialité (modifiables avec la commande `chmod`):
  - propriétaire (user)
  - groupe (group)
  - autres (others)

Fichiers

Opérations sur les fichiers

- Création, suppression, lecture-écriture, modification des attributs.

Noms des fichiers

- Sous Windows : est un attribut du fichier
- Sous Linux : est associé à un lien sur le fichier
  - un fichier peut avoir plusieurs noms/liens

Fichiers

Attributs des fichiers

- Chaque fichier possède un nom et des attributes (meta-données + informations des droits et du propriétaire).
- Tous les SE y associent des informations complémentaires. Exemples:

Champ	Signification
Protection	Qui peut accéder au fichier et de quelle façon
Mot de passe	Mot de passe requis pour accéder au fichier
Créateur	Personne qui a créé le fichier
Propriétaire	Propriétaire courant
Indicateur lecture seule	0 pour lecture/écriture, 1 pour lecture seule
Indicateur fichier caché	0 pour fichier normal, 1 pour ne pas l'afficher dans les listages
Indicateur fichier système	0 pour fichier normal, 1 pour fichier système
Indicateur d'archivage	0 le fichier a été archivé, 1 il doit être archivé
Indicateur fichier ASCII/binaire	0 pour fichier ASCII, 1 pour fichier binaire
Indicateur d'accès aléatoire	0 pour accès séquentiel, 1 pour accès aléatoire
Indicateur fichier temporaire	0 pour fichier normal, 1 pour supprimer le fichier lorsque le processus se termine
Indicateur de verrouillage	0 pour fichier non verrouillé, 1 pour fichier verrouillé
Longueur d'enregistrement	Nb d'octets dans l'enregistrement
Position de la clé	Position relative de la clé dans chaque enregistrement
Longueur de la clé	Nb d'octets du champ clé
Date de création	Date et heure de création du fichier
Date du denier accès	Date et heure du dernier accès au fichier

... mais pas le nom !

Répertoires

Un répertoire (***directory*** ou dossier) est un fichier dont le contenu est une liste de descriptions de fichiers. Il contient les liens physiques vers d'autres fichiers ou répertoires.

- C.a.d., le répertoire est une *table de correspondance* entre le nom d'un fichier et sa localisation sur disque.
- Un répertoire a donc les même types de propriétés qu'un fichier.
- Les opérations possibles : création, suppression (si vide), ajout d'un lien vers un autre fichier/repertoire.
- Lorsqu'un fichier est effacé, il est effacé de la table de correspondance.

Arborescence

- Chaque fichier ou répertoire est référencé par un autre répertoire, ce qui forme une hiérarchie cohérente, appelée aussi arborescence, dont le point d'entrée est le répertoire racine. La racine est unique sur un système de type UNIX (noté comme « / »).

I-node (*Index Node*)

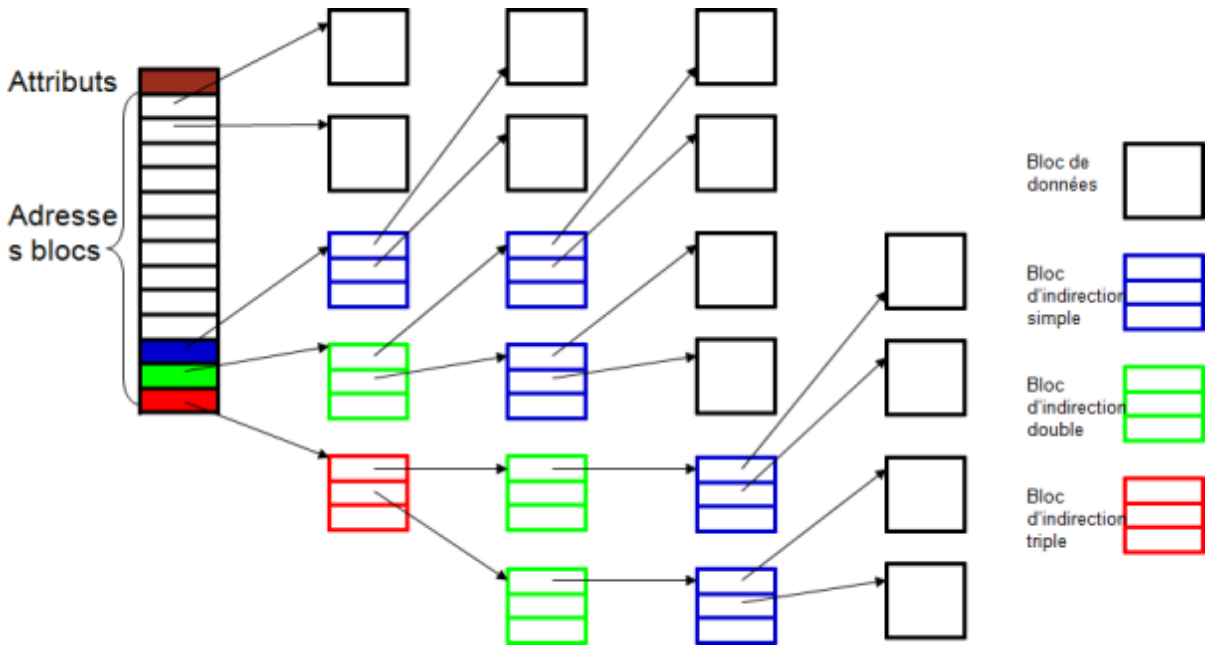
Structure de données contenant des informations à propos d'un fichier ou répertoire stocké dans certains systèmes de fichiers (notamment de type Linux/Unix).

- À chaque fichier correspond un numéro d'inode dans le système de fichiers dans lequel il réside, unique au périphérique sur lequel il est situé.
- Chaque fichier a un seul inode, même s'il peut avoir plusieurs noms (chacun de ceux-ci fait référence au même inode). Chaque nom est appelé lien.
- Les inodes peuvent, selon le système de fichiers, contenir aussi des informations concernant le fichier, tel que son créateur (ou propriétaire), son type d'accès (par exemple sous Unix : lecture, écriture et exécution), etc.

I-node (*Index Node*)

Les informations stockées dans un i-node disque sont :

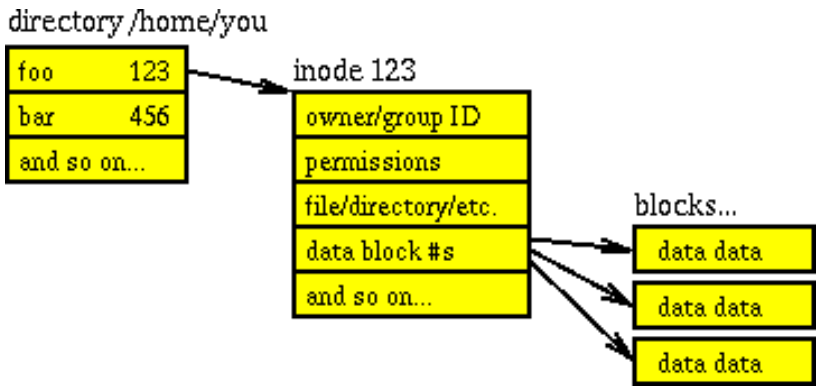
- Les attributs du fichier (type, propriétaire, droits, date de modification, etc.).
- 13 adresses de blocs contenant le fichier (10x direct, 3x indirect).



I-node

Gestion des i-nodes

- L'i-node ne contient pas le nom du fichier
  - Un fichier peut avoir plusieurs noms.
  - Les noms sont stockés dans la structure d'information des répertoires.



- Référence d'un fichier se fait par l'i-node (unique)
- I-node 0 : jamais utilisé, bloc de démarrage (BOOT), fichiers d'initialisation du SE
- I-node 1 : gestion des blocs (disques) défectueux
- I-node 2 : racine du disque (généralement)
- `.` : lien sur le répertoire lui-même
- `..` : lien sur le répertoire parent, numéro d'i-node

I-node

Question

Un système de fichiers utilise des blocs de données de taille fixe 1Ko. Chaque pointeur (numéro de bloc) est représenté sur 32 bits (4 octets, 1 octect = 8 bits). Quelle est la taille maximale d'un fichier ?

I-node

Question

Un système de fichiers utilise des blocs de données de taille fixe 1Ko. Chaque pointeur (numéro de bloc) est représenté sur 32 bits (4 octets, 1 octect = 8 bits). Quelle est la taille maximale d'un fichier ?

Réponse

On peut donc mettre 1Ko / 4 = 1024 /4 = 256 numéros de blocs dans un bloc.

- blocs directs : 10 blocs,
- bloc indirect_1 : 256 blocs,
- bloc indirect_2 : 256^2 blocs,
- bloc indirect_3 : 256^3 blocs.

Nombre maximum de blocs dans un fichier : $(10 + 256 + 256^2 + 256^3)$ * 1Ko $\sim$ la taille maximale est de 16 Go

Hiérarchie de fichiers

Système de fichiers virtuel (UNIX)

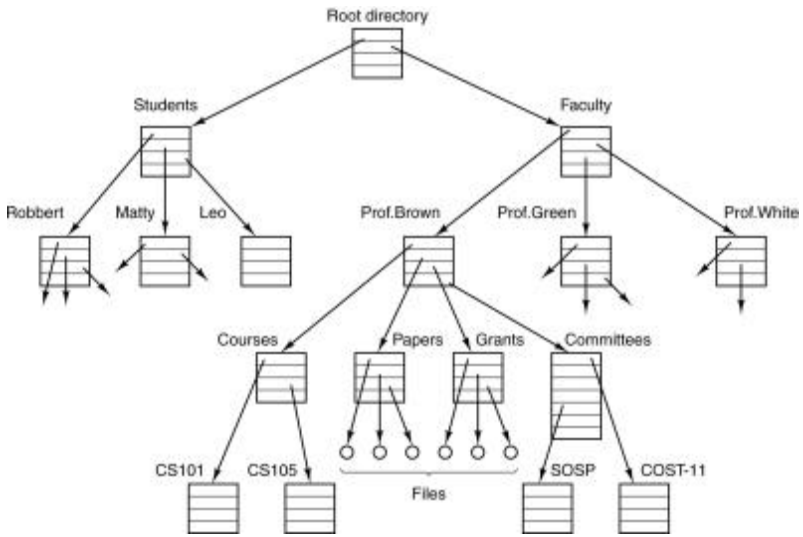
- Une seule hiérarchie arborescente (répertoires + fichiers).
- Le SE masque les spécificités des disques.
- Modèle de programmation "unique", indépendant, avec des appels systèmes pour la création, suppression, lecture écriture, ouverture, fermeture.

Hiérarchie de fichiers

Système de fichiers virtuel (UNIX)

- Notion de montage d'un système de fichier (`mount`, `umount`)
  - Greffer la racine d'une arborescence non encore montée en un point accessible depuis la racine absolue.
  - D'autres partitions, des clés USB, des CD-ROM, des périphériques, etc., peuvent être montés dans des répertoires.
  - Les périphériques sont des fichiers dans `/dev/` :

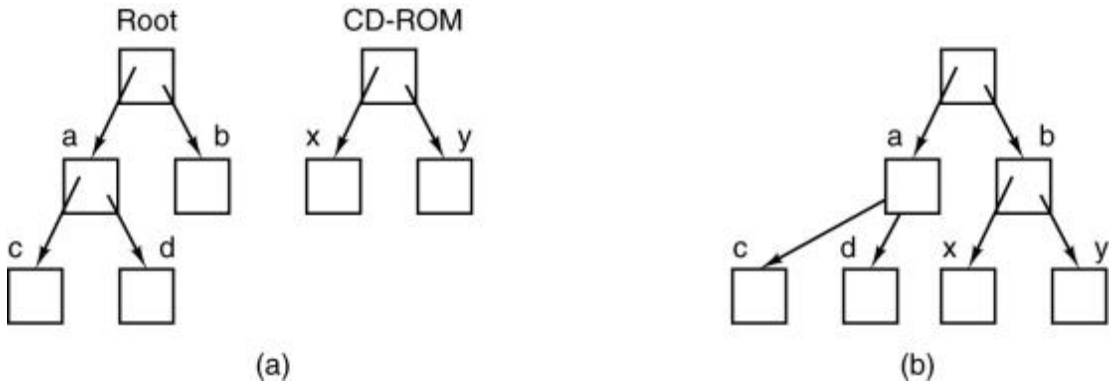
    `/dev/pts/X`, `/dev/hda`, `/dev/sda`, `/dev/mouse`, `/dev/keyboard`, ...



Hiérarchie de fichiers

Montage d'un système de fichiers

- Exemple de fonctionnement :
  - (a) Avant montage les fichiers du CD ne sont pas accessibles.
  - (b) Après montage, ils font partie de la hiérarchie des fichiers.



Hiérarchie de fichiers

Montage d'un système de fichiers

- Accès aux partitions en Unix/Linux :
  - Pointeurs stockés dans `/dev/xxyz`
  - `xx`: type du bus (`hd` : IDE (*Intelligent Drive Electronic*), `sd` : SCSI (*Small Computer System Interface*), `fd` : floppy (lecteur de disquette))
  - `y` : lettre de périphérique
  - `z` : numéro de partition
  - `hda1` : Première partition (`1`), du premier disque (`a`), du bus IDE (`hd`)
- Adressage par *Universally Unique IDentifier* (UUID)
  - Identificateur de 128 bits
  - À consulter en Linux avec `blkid`
- En linux, points de montage permanents sur `/etc/fstab`

Structure hiérarchique des répertoires Unix

/ ← racine
|
|__/bin ← fichiers binaires (exécutables).
|__/etc ← fichiers de configuration.
|__/media ← dispositifs CDROM, clé USB, etc.
|__/root ← home de l'administrateur du système.
|__/dev ← fichiers des périphériques.
|__/home ← répertoires home des utilisateurs.
|__/sbin ← fichiers binaires n'accessibles qu'au root.
|__/tmp ← fichiers temporaires.
|__/var ← fichiers qui peuvent varier (p.ex logs).
|__/usr ← logiciel pour l'utilisateur.
|__/lib ← librairies dynamiques.
|__/boot ← fichiers de démarrage du système.
|__/mnt ← dispositifs éjectables.

/usr
|
|__/bin ← fichiers binaires d'utilisateur.
|__/local ← programmes compilés d'utilisateur.
|__/src ← sources des utilisateurs.
|__/share ← fichiers indépendants de l'architecture matérielle.
|__/include ← fichiers *header* des applications.
|__/lib ← liens dynamiques à des librairies.

- `usr` : logiciel de seule lecture, disponible pour tout le système. Structure similaire à la racine.

Implantation des systèmes de fichiers

Sous LINUX : 12 Millions de lignes !

Organisation en couches :

1. Gestionnaire de système de fichiers virtuels.
2. Gestionnaires de systèmes de fichiers (FAT, Ext4, NTFS, ISO9660, NFS, Btrfs, . . .).
3. Gestion des buffers (mémoire tampon).
4. Pilotes disques/cdrom/clé USB — Pile réseau.
5. Pilotes bus IDE, SCSI, USB ...
6. Matériel (contrôleurs disques, réseau).

Gestionnaire de système de fichiers virtuels

Fournit :

- Fonctions de création/suppression de fichiers/répertoires.
- Fonctions de lecture/écriture dans un fichier/répertoire.
  - Ces fonctions utilisent des chemins (**absolu** : se base sur la racine de l'arborescence et commence par / ; ou **relatif** : relatif au répertoire courant où se trouve l'utilisateur et commence par autre chose que / ou ~) pour désigner les fichiers/répertoires.

Rôle :

- Contrôle les droits d'accès.
- Doit déterminer à partir des chemins sur quel périphérique se trouve un fichier donné et quel système de fichier est utilisé.
  - Contacte alors le bon gestionnaire de système de fichier.

Gestionnaire de système de fichiers

Fournit :

- Idem que pour le gestionnaire de système de fichiers virtuels, mais seulement pour un système de fichier particulier (ex. FAT, Ext4, ISO9660, NFS, Btrfs, . . .).
- Il y a donc plusieurs gestionnaires.

Rôle :

- Contrôle dans quels blocs du disque sont stockés les fichiers.

Quelques systèmes de fichiers actuels

- Microsoft FAT (*File Allocation Table*)
  - Format de Microsoft à l'origine.
  - L'un de plus simples.
  - Limitations :
    - Racine stockée sur un emplacement fixe de la partition.
    - Mise à jour de la table d'allocation non efficace.
    - Pas d'organisation de la structure de répertoire ⇒ fragmentation importante.
    - Taille des blocs variables en fonctions de la taille des partitions.
  - Évolutions : FAT12, FAT16 et FAT32.
- Microsoft NTFS (*New Technology File System*)
  - Peu documenté.
  - Respecte la norme POSIX (*Portable Operating System Interface*, et X pour UNIX). Gestion des droits.
  - Pas de taille maximum d'un fichier (taille de la partition).
  - Système journalisé [1]

1. Permet d'éviter la corruption d'un fichier afin de garantir l'intégrité des données en cas d'arrêt brutal de l'écriture.

Quelques systèmes de fichiers actuels

- Ext4 (*Extended File System Version 4*)
  - Système le plus commun en Linux.
  - Liste d'i-nodes.
  - Quatre catégories de fichiers :
    - Fichiers normaux (textes, exécutables).
    - Fichiers répertoires.
    - Fichiers spéciaux, contenus dans le répertoire `/dev`.
    - Fichiers liens symboliques qui font référence à d' autres fichiers.
  - Sensible à la casse.
  - Fichiers cachés commencent par "`.`"
  - Défragmentation inutile.
  - Système journalisé.
- Btrfs (*B-tree file system*)
  - Système de fichiers avec du copy on write (CoW).
  - Création de subvolumes, ou partitions racines différentes dans le même système de fichiers.
  - Création de snapshots : *photographie* à un instant donné pour sauvegarder le système de fichiers dans un état cohérent tout en continuant à travailler.
  - Compression.

Types d'accès au stockage

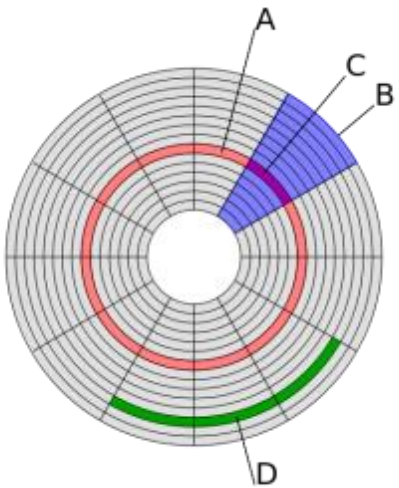
- **Accès séquentiel** (bande magnétique)
  - Les premiers SE n'offraient qu'un seul type d'accès aux fichiers.
  - Un processus pouvait lire tous les octets dans l'ordre à partir du début du fichier, mais pas dans le désordre.
  - Un fichier pouvait être rembobiné, donc être lu plusieurs fois.
  - Utilisé aujourd'hui pour la sauvegarde à long terme (*backup*).

- **Accès aléatoire «Random Access File»** (disque dur - DD)
  - L'arrivée des DD a autorisé la lecture dans le désordre des octets.
  - Les fichiers à accès aléatoire sont indispensables à de nombreuses applications
    Comme les systèmes de gestion de base de données (accès à un enregistrement sans parcourir tous les enregistrements mémorisés)

Organisation physique d'un disque

Structuration du disque

- Un enregistrement physique, aussi appelé bloc, est la plus petite unité de stockage manipulée par le système.
- Un fichier est un ensemble de blocs.
- Les blocs sont numérotés par le système.
- Le système doit connaître l'emplacement sur disque (numéro de cylindre, piste et secteur) de chaque bloc.



Méthodes possibles:

- contiguë
- chaînée
- indexée

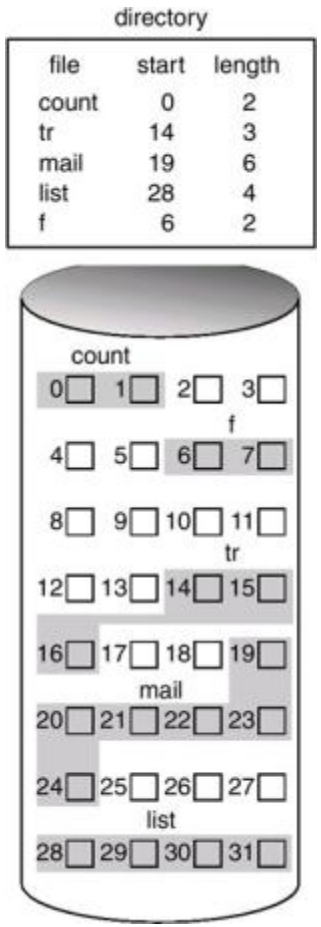
Structure d'un disque:

A) piste
B) secteur géométrique
C) secteur de piste
D) bloc

Allocation contiguë

Stocker chaque fichier dans une suite de bloc consécutifs.

- Un fichier de N Ko occupe N blocs consécutifs sur un disque dont la taille d'un bloc est de 1Ko.



Avantages

- Simple. Il suffit de mémoriser un seul nombre, l'adresse du premier bloc pour localiser le fichier.
- Performance excellente, car le fichier peut être lu en une seule opération.

Inconvénients

- La taille du fichier doit être connu au moment de leur création et peut difficilement grandir.
- Fragmentation du disque (le compactage peut y remédier mais il est coûteux).

Adapté aux systèmes de fichiers en lecture seule (CD-ROM)

Allocation chaînée

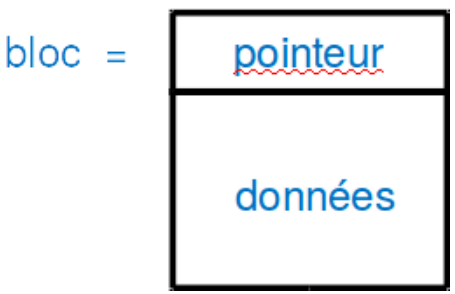
Le premier mot de chaque bloc est un pointeur sur le bloc suivant. Le reste contient les données.

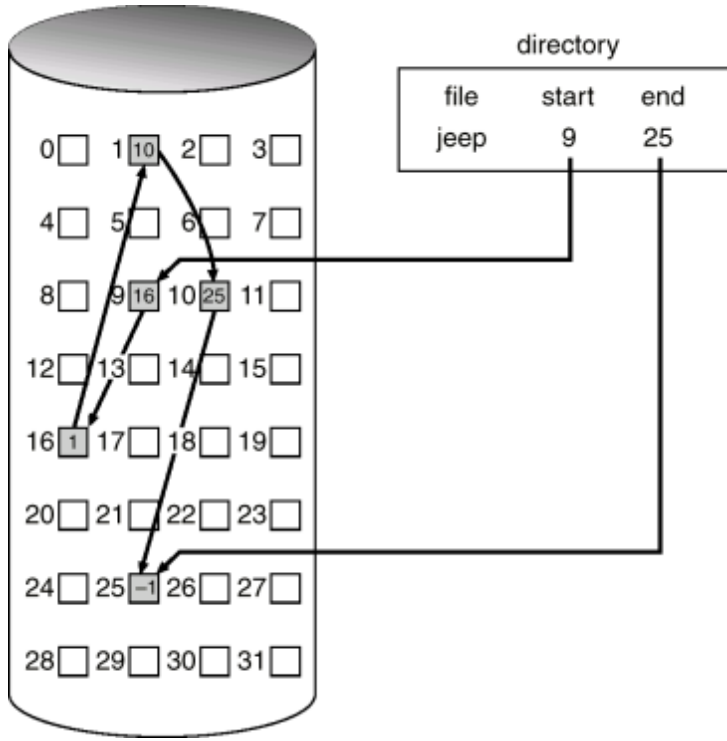
Avantages

- Facilite les modifications et l'accroissement.

Inconvénients

- Alourdit la recherche d'une donnée.
- Pas d'accès aléatoire mais séquentiel.





Allocation indexée

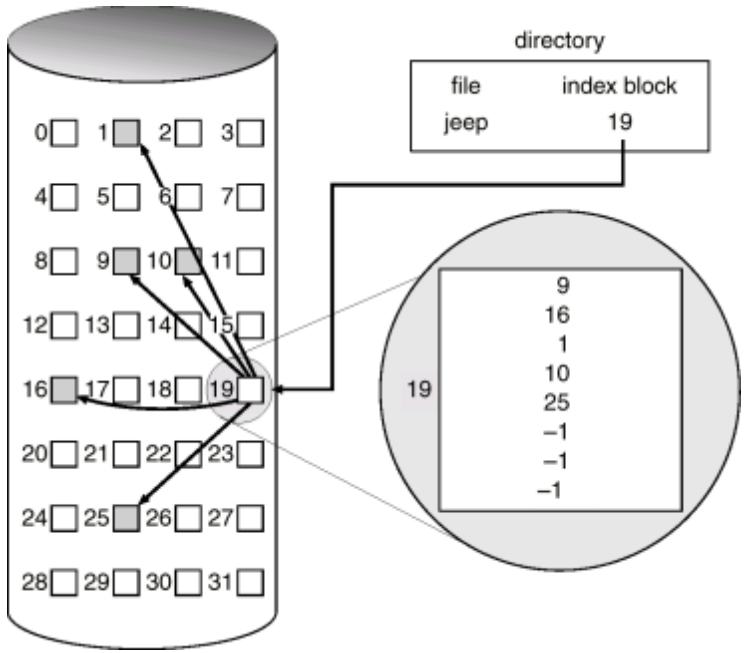
Tous les pointeurs sont regroupés dans un tableau (index block).

Avantages

- Facilite les modifications et l'accroissement.
- Accès aléatoire.

Inconvénients

- Les index prennent de l'espace.
- Taille du fichier limitée.



Organisation logique d'un disque

1. Un schéma de partitionnement
2. Des partitions

Vocabulaire :

- Partitionnement :
  - fractionnement d'un disque dur réel (matériel) en plusieurs disques virtuels (logiques).
  - Chaque partition contient un système de fichiers (FAT, Ext4, . . .).
  - C'est un "superbloc".
  - Une *Partition* a un type, emplacement de début et taille.
  - Le partitionnement est réversible.

Organisation logique d'un disque

- Formattage :
  - Pour qu'un système de fichiers puisse gérer des fichiers sur une unité de stockage, son unité doit être formatée (selon les spécificités du SGF).
  - Le formatage inspecte les secteurs, efface les données, crée le répertoire racine du système de fichiers et crée un "superbloc", pour stocker les informations nécessaires pour assurer l'intégrité du système.

Utilité :

- Permet d'installer plusieurs systèmes.
- Permet la sauvegarde des données.
- Permet une séparation logique des répertoires (système,utilisateur, log, . . .).

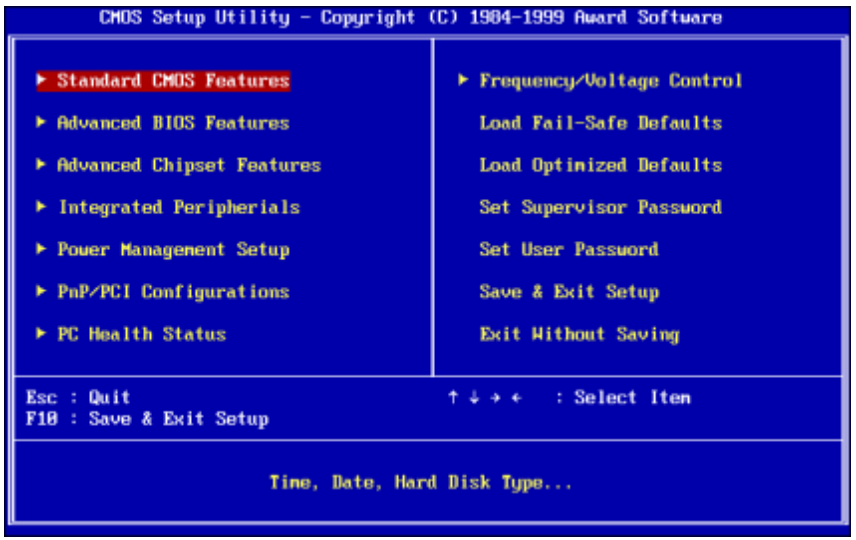
Organisation logique d'un disque

Vocabulaire :

- **BIOS** (*Basic Input Output System*) :
  - Préinstallé sur la carte mère.
  - C'est le premier logiciel qui est exécuté après avoir allumé l'ordinateur.
  - Il est utilisé pour effectuer l'initialisation du matériel.
- **UEFI** (*Unified Extensible Firmware Interface*) :
  - Nouveau micrologiciel. Remplacement du BIOS. Utilisation d'une nouvelle table de partitionnement : le GPT.
  - Renforcement de la sécurité.
- **MBR** (*Master Boot Record*) :
  - Le premier secteur physique d'un disque dur. Il contient la table de partitionnement du disque et un code d'amorçage (*boot*) permettant de démarrer le système d'exploitation. Il fonctionne de pair avec le BIOS.
- **GPT** (*Globally Unique IDentifier Partition Table*) :
  - nouveau standard pour la table de partitionnement qui fonctionne de pair avec l'UEFI.
  - Il remplace le MBR.

Organisation logique d'un disque

BIOS vs UEFI:



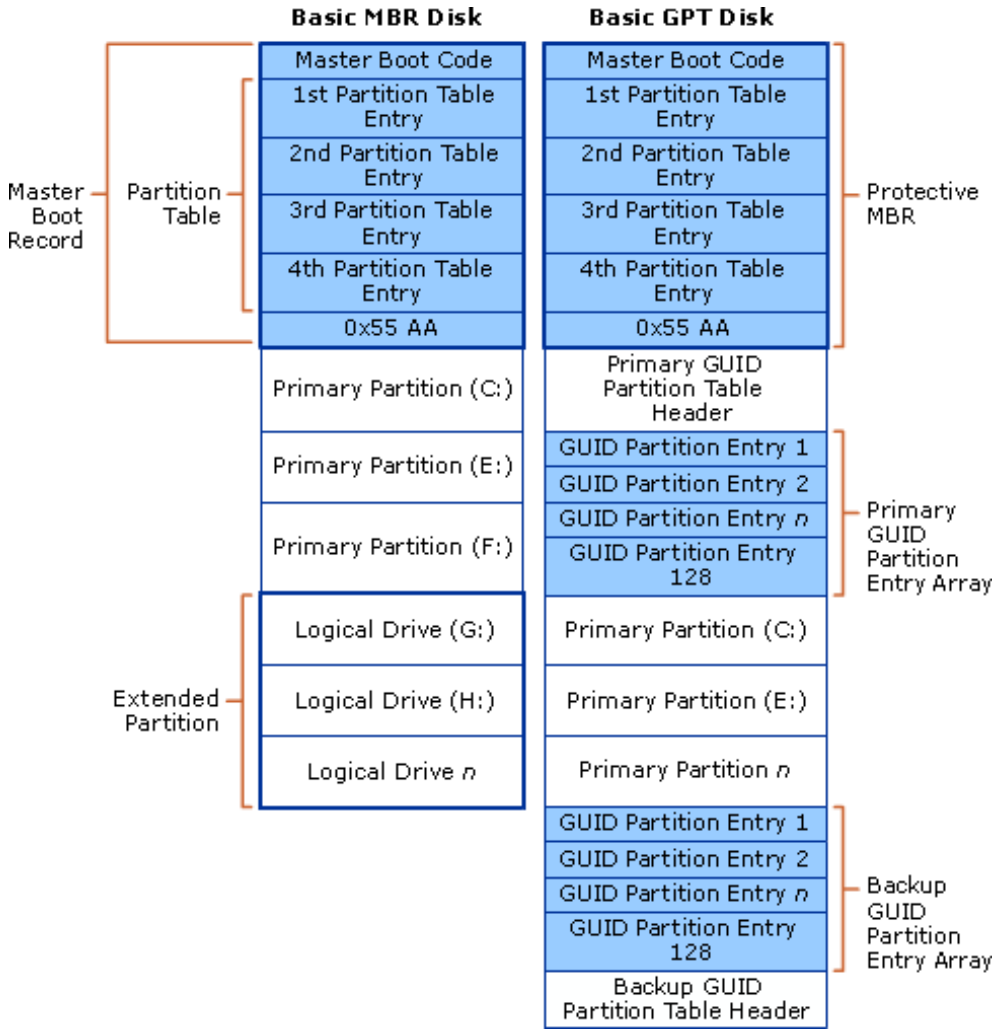
Organisation logique d'un disque

Schémas de partitionnement

- Schéma MBR
  - Quatre partitions maximum et taille maximale de 2.1 TBytes.
  - **partition principale :** au maximum quatre, accepte tout type de système de fichiers. Si nous avons moins de quatre partitions principales, possibilité de créer une partition étendue :
    - **partition étendue :** ne peut contenir que des partitions logiques, ne peut pas recevoir de système de fichiers, ne peut exister que s'il existe une partition principale
    - **partition logique :** contenue dans une partition étendue, pas de limitation en nombre, accepte tout type de système de fichiers
- Schéma GPT
  - Créé pour le standard *Unified Extensible Firmware Interface* (UEFI) : replacement de la BIOS traditionnelle
  - Supporté aussi par Linux dans la BIOS traditionnelle
  - Nombre illimité de partitions (Microsoft : 128).
  - Jusqu'à 256 TBytes par partition.
  - Taille maximale théorique de disque : 9.4 zettabytes (9.4 billion terabytes).

Organisation logique d'un disque

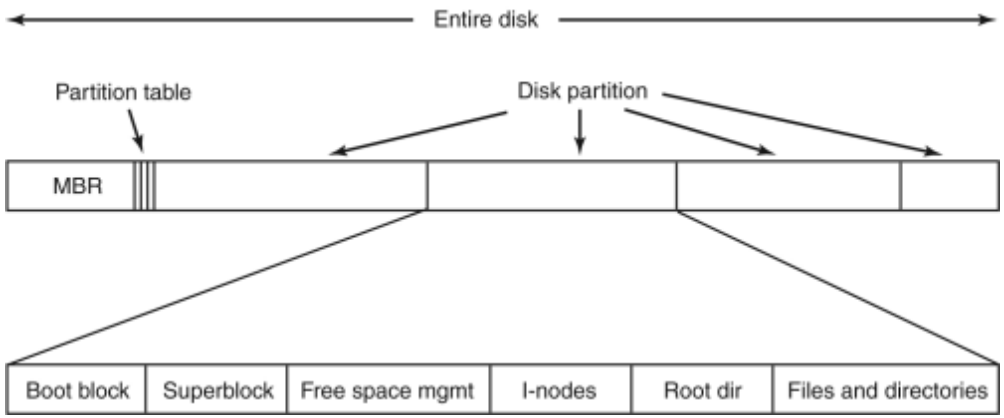
MBR vs GPT :



Source : technet.microsoft.com

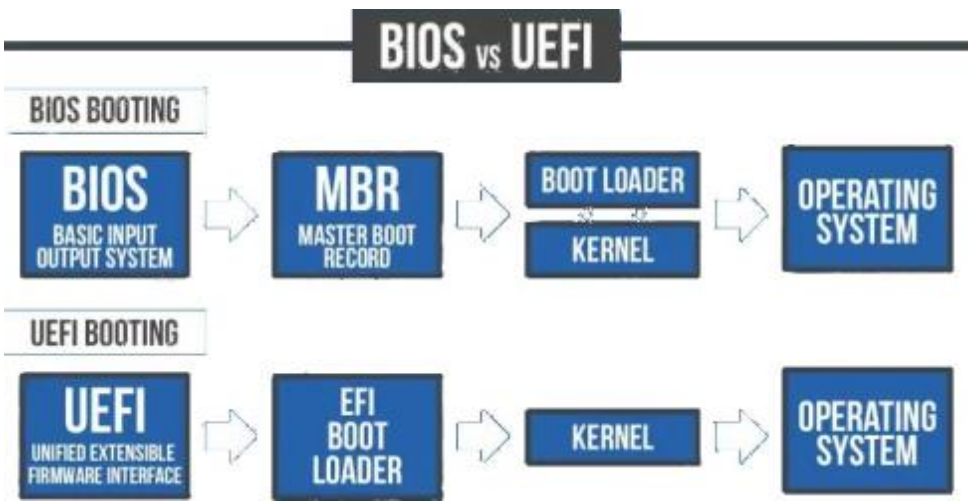
Démarrage de l'ordinateur

- BIOS contenu dans une mémoire ROM (*Read-Only Memory*), mémoire morte.
- Disque dur divisé en partitions, chacune avec un système de fichiers.
- Le secteur 0 ou MBR (*Master Boot Record*) est lu au moment du démarrage
  - La fin du MBR contient la table de partitions
  - L'une d'entre elles est marqué comme active



Démarrage de l'ordinateur

- Systèmes MBR :
  1. La BIOS cherche, lit et exécute le *first-stage boot loader* contenu dans le MBR du disque dur.
  2. Le MBR cherche la partition active, lit son premier bloc (*bloc de démarrage*, *Volume Boot Record* ou *Boot Block*) et l'exécute.
  3. Le *second-stage boot loader* dans le bloc de démarrage (GRUB2, par exemple) charge le SE contenu dans ce partition.
- Systèmes UEFI :
  - L'UEFI charge directement le chargeur d'amorçage (bootloader) du SE, enregistré sous la forme d'un fichier .efi sur la partition EFI du disque GPT.



Processus de démarrage *SysVinit*

Initialisation :

1. Décompression du noyau linux
2. Initialisation des périphériques (disque, réseau, son, . . .)
3. Configuration “automatique” des pilotes
4. Montage du système de fichier racine
   - Lecture seule : permet de vérifier s’il y a des erreurs
5. Lacement du processus `init` (PID 1)
6. Lecture du fichier `/etc/inittab`
7. Lancement des processus à partir des **scripts shell** contenus dans `/etc/rc<n>.d/`
8. Commutation vers le mode multi utilisateurs
9. Exécution de `getty` (La commande getty définit et gère les lignes et les ports du terminal) pour chaque console
10. Remontage du système de fichier en lecture-écriture
11. Exécution des processus de services (`daemon`)
12. Construction d’une arborescence de processus

Processus de démarrage *systemd*

- Simplicité d'utilisation
- Maîtrise du système moins éparpillée
- L'utilisation de scripts de lancement est découragée
- Services pilotés par des fichiers texte de configuration à la place des scripts de SysVinit

Apports de *systemd*

- Allocation fine des ressources (processeur, mémoire, E/S, etc) aux services (cgroups)
- Surveillance améliorée
- Log plus complet (au format binaire) et normalisation du format
- Possibilité de *démoniser* tout processus en le relançant automatiquement s'il s'arrête
- Séparation claire entre les services fournis par la distribution et les services créés par l'administrateur
- Simplification du processus d'empaquetage des services
- Centralisation du développement des briques de base du système

Source : `linuxfr.org`

Plan

Mise à jour du système

Mise à jour en Debian/Ubuntu

apt-get update

La commande update permet de resynchroniser un fichier répertoriant les paquets disponibles et sa source. Ces fichiers sont récupérés aux endroits spécifiés dans /etc/apt/sources.list (...). On doit toujours exécuter une commande `update` avant les commandes `upgrade` ou `dist-upgrade`.

apt-get upgrade

La commande `upgrade` permet d'installer les versions les plus récentes de tous les paquets présents sur le système en utilisant les sources énumérées dans `/etc/apt/sources.list`. Les paquets installés dont il existe de nouvelles versions sont récupérés et mis à niveau. **En aucun cas des paquets déjà installés ne sont supprimés** ; de même, des paquets qui ne sont pas déjà installés ne sont ni récupérés ni installés.

apt-get dist-upgrade (**à ne pas faire sur vos machines CY Tech !**)

La commande `dist-upgrade` effectue la fonction `upgrade` en y ajoutant une gestion intelligente des changements de dépendances dans les nouvelles versions des paquets ; apt-get possède un système « intelligent » de résolution des conflits et il essaye, quand c'est nécessaire, de mettre à niveau les paquets les plus importants aux dépens des paquets les moins importants. Le fichier /etc/apt/sources.list contient une liste de sources où récupérer les paquets désirés.

Plan

Virtualisation

Virtualisation

Infrastructure Matérielle

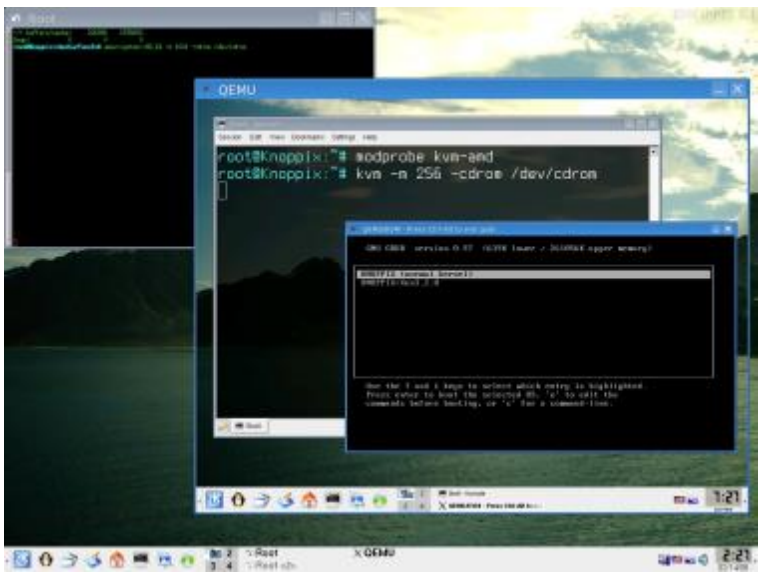
Guest OS

Environnement Virtualisé

Création d'une version virtuelle (par rapport à une réelle) de quelque chose, tel que une plate-forme matérielle, système d'exploitation, dispositif de stockage, ou ressources du réseau.

Hyperviseur

Un hyperviseur (moniteur de machine virtuelle) est un processus qui crée et exécute des machines virtuelles (VM). Il permet à un ordinateur hôte de prendre en charge plusieurs VM clientes en partageant virtuellement ses ressources, telles que la mémoire et la capacité de traitement.

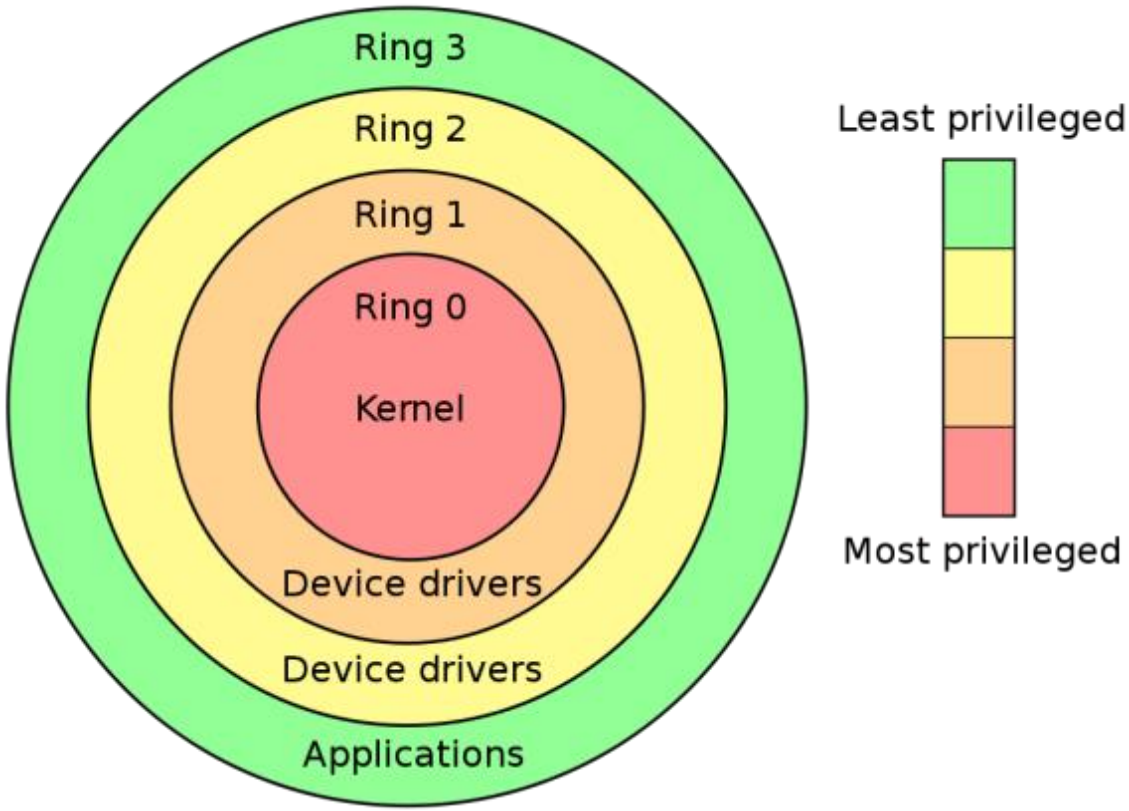


Types de Virtualisation

Traduction Binaire
Paravirtualisation
Virtualisation Matérielle

Types de Virtualisation

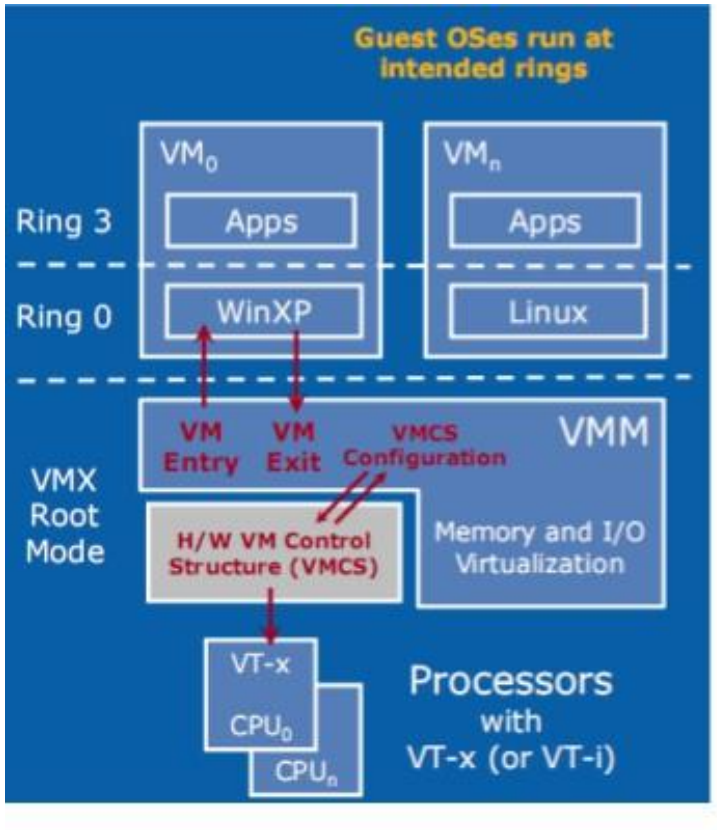
Anneaux de Privilèges



Anneaux de privilèges pour les processeurs x86 disponibles en mode protégé (source: Wikipedia).

Types de Virtualisation

Virtualisation Matérielle



(Source:: VmWare)

Types de Virtualisation

Types d'hyperviseurs

Hyperviseur de type 1 : hyperviseur de système bare metal ou natif, s'exécute directement sur le matériel de l'hôte pour gérer les SEs invités.

Types de Virtualisation

Types d'hyperviseurs

Hyperviseur de type 2 : est un logiciel qui s'installe et s'exécute sur un SE déjà en place. On parle d'hyperviseur hébergé. Par exemple, création des VMs.