Operating Systems

January 6, 2019

Contents

1 Quelles sont les différences de philosophie entre les systèmes d'exploitations de type mainframe et UNIX ?

1.1 Les Mainframes :

- Lourds
- L'OS Intègre toutes les fonctionnalités
- Monde Propriétaire (Hardware Propriétaire)

1.2 UNIX :

- Essentiel
- Gestion des processus et de fichiers + qq focntionnalités de base
- reste dans des Librairies de fonctions ou programmes
- Ouvert (peut tourner sur un autre hardware que celui prévu inititalment)

2 Qu'est-ce que l'on considère comme un fichier dans Unix?

Tout ce qui peut être organisé par une suite de bytes (mais sans voir la structure des données).

3 Quels sont les deux types de représentation d'un fichier et leurs buts?

Représentation Logique : les noms liés au fichier (pouvoir donner un nom ou plusieurs noms au fichier). **Représentation Physique** : Contient l'aspect de protection et petmet de retrouver les bytes.

4 Qu'est-ce qu'un I-node et que contient-il ?

Un **I-node** est le conteneur de la *représentation phyqique d'un fichier*.

- Le Propriétaire
- Le Groupe
- Les Protections
- Le type de fichier (+ sous-type)
- La date de création
- La date de modification
- La dernière date d'utilisation
- La taille
- L'endroit où on peut trouver les bytes et le nombre de links (nombre de noms logiques)

5 Quels sont les principaux types de fichiers dans Unix?

- Fichier Normal
- Répertoire
- Lien symbolique
- philosophie
- Hardware (Blocs par blocs ou bytes par bytes)

6 Quelles sont les techniques de protection des fichiers dans UNIX

Il en existe 3 types: Le **Propriétaire**, le **groupe** et le **reste**.

Permission	Fichier	Répertoire
X	exécuter	Permets l'utilisation du pathname
W	Ouvrir écriture	Modification(ajout,suppr et renommer un fichier)
R	Ouvrir Lecture	lister le directory

7 Puis-je mettre des protections différentes aux noms d'un même fichier logique ?

non, différents noms (représentation logique) se réfèrent à un même Inode Physique

8 Quelles sont les relations entre la date de création, de modification et de dernière utilisation d'un fichier ?

Date de dernière utilisation : date la plus récente. **Date de création** : de l'I-node. **Date de modification** : modification du contenu du fichier.

9 Puis-je avoir un fichier de taille plus grande que l'espace physique d'un système de fichier ?

Oui, si on se déplace d'un bout à l'autre d'un fichier et que l'on écrit un byte, la taille totale du fichier sera prise en compte. Les disques ignorent une partie des bytes (on écrit pas les bytes au dessus desquels on passe avec l'opération lseek).

10 A quoi sert

10.1 unlink

unlink détruit un nom dans l'aborescence des fichiers. Si ce nom était le dernier lien sur une représentation physique (i-node) ce dernier est effacé et l'espace mémoire qu'il occupait est rendu disponible. ## link **link** associe

un nouveau nom à un fichier (i-node) existant. ## mkfifo **mkfifo** crée un pipe nommé.

10.2 mount et umount

Ils servent à la **Construction** de l'arborescence des fichiers du système d'exploitation. La commande **mount** permet d'attacher un système de fichier trouvé sur un périphérique à l'arborescence du système. Et umount est l'inverse

10.3 dup

Duplique un descripteur de fichier dans la table des fichiers ouverts du processus. L'ancien et le nouveau descripteur peuvent être utilisés de manière interchangable. Ils partagent le même pointeur de position dans le fichier et si celui-ci est modifié sur l'un des descripteurs, la position sera également changée pour l'autre. Cela permet donc d'avoir le même pointeur dans le même fichier ouvert courant pour une série d'entrées ouvertes dans l'OFT.

10.4 fork

Le fork permet à un processus de créer un processus fils qui aura un nouveau pid et un status différent. Les processus fils héritent de tous les autres attributs du père : **open file table**, **programme**, **propriétaire de programme et processus**, **tableau de fonctions liés aux signaux**, **contenu des données**, **point d'exécution(P-Counter)**.

Retour : 0 fils , et Pid fils si on est le père.

11 Qu'est-ce que la table de fichier ouverts d'un processus (open file table) et à quel moment sont contenu change-t-il ? y-a-t-il dans cette table une nomenclature particulière ?

La **table des fichiers** ouverts d'un processus est une table de « file descriptors » que l'on peut utiliser dans les appels systèmes de lecture, d'écriture, déplacement, ...) Elle contient des références vers des entrées de la table des fichiers ouverts du système où est, entre-autre, stocké le pointeur courant du fichier ouvert). Le contenu de la table change lorsqu'on fait un open(), un close() ou un dup() On a un standard au niveau de la nomenclature des 3 premières entrées : 0 pour stdin, 1 pour stdout et 2 pour stderr

12 Unix accepte-il que l'on ouvre plusieurs fois le même fichier en même temps ?

Oui, c'est à l'utilisateur de gérer la synchronisation.

13 Quels sont les appels système pour la gestion des fichiers (data) (a l'exclusion des Propriétaires et de la sécurité)?

- create
- open
- read
- write
- seek
- lseek
- close

- delete
- unlink
- link
- dup

14 Qu'est-ce qu'un répertoire ou directory ?

C'est un fichier qui contient une liste de hardlink.
hardlink : lien entre un nom et un i-node. il permet de gérer la structure arborescente de la structure logique d'un système de fichiers

15 Quelle est la différence entre Windows et Unix en termes de représentation logique et de type de fichiers ?

15.1 Windows

chaque disque à sa propre arborescence

15.2 unix

On monte dans l'arborescence principale les autres fichiers hardware.