

Processus

Permissions: Lecture (R), Ecriture (W), Exécution (X), mprotect(): changer les permissions

Faute de page: interruption émise par le processeur pour ouvrir le système d'exploitation dans les situations suivantes:

- Adresse en dehors des espaces du processus
- Non respect des permissions
- Page non-présente en mémoire

Le système d'exploitation décide quoi faire (charger une page du disque, terminer le programme, etc.)

Cache de traduction: TLB

- Traduire les adresses virtuelles est complexe et très fréquent (chaque instruction)
- Solution: utiliser une **cache de la traduction**
- TLB: Translation Lookaside Buffer
- Doit être invalidée (au moins partiellement) après un changement de processus (ordonnancement)

Allocation de mémoire

Allocation de page, m-à-t de la liste de pages

Carte de la mém. virt

Algorithmme FIFO

On remplace la page qui est la plus ancienne en priorité

Procédure: maintenir une liste des pages chargées. Si la page n'est pas dans la liste, on l'ajoute à la fin, si la liste est pleine, évicte celle de début

On remplace la page qui est la plus ancienne en priorité

Procédure: maintenir une liste des pages chargées. Si la page n'est pas dans la liste, on l'ajoute à la fin et remettre au début; si la liste est pleine, évicte celle à la fin de la liste

Algorithmme Least Recently Used (LRU)

Temps	0	1	2	3	4	5	6	7	8	9	10	11
Accès	10	20	20	10	30	40	10	30	40	10	20	30
PAGE 1	10	10	10	10	30	30	10	40	40	10	20	30
PAGE 2	20	20	20	20	40							
PAGE 3												
PAGE 4												
Faute	✓	✓	✗	✗	✓							
autre	V	V	X	X	V							

Écroulement (trashing)

- Quand la mémoire disponible est très faible, alors le SE sauvegarde des pages sur disque
- Le système continue de fonctionner, mais cause un ralentissement important
- Stratégie pour atténuer le problème: mettre sur disque des pages AVANT de manquer de mémoire
- Paramètre noyau "vm.swappiness"

 - Valeur de 0: utilise la pagination en dernier recours
 - Valeur 100: utilise la pagination agressivement
 - Valeur par défaut sur Ubuntu: 60
 - cat /proc/sys/vm/swappiness" → il attend de 0% de swap (RPM) jusqu'à ce qu'il commence à écraser les fichiers

Fonctions d'un système de fichier

Détermine comment répartir les fichiers en blocs

Présente une vue hiérarchique de répertoires

Implémente les fonctions d'accès (read, write, create, delete, etc.)

Stock les permissions d'accès et d'autres attributs

- Permet au système d'exploitation d'imposer les permissions, quotas, etc.
- Assure l'intégrité des fichiers

Fonctions de sécurité: encryption et confidentialité

Monter un système de fichier

Accéder au deuxième disque SCSI (sda, sdb) par le répertoire backup avec le système de fichier ext4.

\$ mount -t ext4 /dev/sdb /media/backup

Inventaire des blocs libres

Tableau de bit: un bit par bloc indique s'il est occupé ou libre.

Espace libre chaîné: la fin d'un espace libre pointe vers le prochain emplacement disponible.

Index avec taille: un bloc possède le début d'un nombre maximal de fichiers par répertoire

Format du système de fichier FAT

Périphérique

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Table des fichiers

Fichier	Index
A	18
B	9
C	2

Table d'allocation des blocs

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

- La taille est indiquée à l'avance, peu importe le nombre de blocs utilisés.
- Temps d'accès moyen de 0(0) comme une liste chaînée.
- Taille et position proportionnelle au nombre de blocs.
- Exemple: déterminer les blocs pour un fichier:

 - A: 18, 19
 - B: 9
 - C: 2, 11, 14, 6, 1

Un ordinateur possède des adresses virtuelles de 32 bits et des pages de 8Ki. La taille de page est d'un seul niveau et chaque entrée de la table de pages fait 32 bits. Quelle est la taille maximale de la table de page?

La taille maximale de mémoire adressable avec 32 bit est de 4Go. La table de pages contient donc au maximum 4Gi / 8Ki = 524288 entrées. Si chaque entrée fait 4 octets, la taille est donc de 2Mi.

Un ordinateur utilise une table de pages à trois niveaux. Les adresses virtuelles sont donc décomposées en 4 champs (a, b, c, d), a contenant les bits les plus significatifs. Quoique déterminer la taille de chacun de ces champs? Il y a souvent une relation précise entre la taille relative de ces champs sur la plupart des systèmes.

Les champs d détermine la taille des pages (2^d). Les champs a, b et c déterminent le nombre d'entrées dans les sections de tables de niveau 1, 2 et 3 processus. La lecture dans la table de pages prend 5ns. Une cache de pré-préparation: Utilisuellement, une page est utilisée pour chaque section de table de traduction d'adresse (TLB) contient 32 entrées et peut être accédée en 1ns (auquel que soit le niveau). Le nombre d'entrée dans la section est donc la taille (en 5ns). Quel est le taux de succès requis afin d'avoir un temps d'accès d'une page divisée par la taille d'une entrée. De ce fait, a, b et c sont moyens de 2ns? Généralement égaux. La taille d'une entrée est usuellement la taille de l'adresse virtuelle (32 bits pour un système 32 bits et 64 bits pour un système 64 bits); ceci permet d'y stocker le numéro de page physique ainsi que quelques bits pour les permissions, le statut...

Défaut de cache v.s. Faute de page

Survient quand une donnée n'est pas dans la cache du processeur

Niveau L1 > L2 > L3 > RAM

Événement de la microarchitecture

Résolution sans l'intervention du SE

AKA Défaut de page

Type d'interruption

- Suite à un **accès invalide** à une adresse mémoire
- Le SE doit intervenir
 - Ajouter une page
 - Charger du disque
 - Terminer l'application

Appliquer des permissions

Donne l'illusion d'un plus grand espace mémoire qu'il n'existe en réalité

- Espace d'adresse plus grand que le **bus physique**
- Sauvegarder des pages peu utilisées sur disque et libérer de la mémoire

Deux processus peuvent utiliser les mêmes adresses (pas de conflit)

Charger une seule copie des **librairies partagées** (économie de mémoire)

Donne utile perte

Résumé des objectifs

- Petite taille (4 kio)
- Réduit les pertes de fragmentation
- Augmente le nombre de pages à gérer
- Diminue le nombre de pages à gérer

Quelle taille de page choisir?

- Grosses pages (4 MiB)
- Augmente les pertes par fragmentation
- Diminue le nombre de pages à gérer

Entrée dans la table de page (Intel)

Linear address: 31:24/23:16:15:8:7

page directory

page table

4K memory page

* 32 bits aligned to a 4-KByte boundary

Linear address: 31:24/23:16:15:8:7

page directory

page table

PS = 1

CR3

32 bit aligned to a 4-KByte boundary

Faute de page mineur v.s. Majeur

- Mineur (lent)
- La page demandée requiert un accès disque
- Processus mis à l'état bloqué, ordonnanceur invoqué
- Décalage de l'ordre de la milliseconde (dépend du périphérique)

Allocation mémoire avec brk/sbrk

- Utilisé pour le monceau du programme
- Agrandir et rétrécir une plage mémoire du processus
- Différence avec la pile: l'allocation persiste après le retour d'une fonction
- Agrandir vers des adresses plus élevées → ~~0x70000000~~ → 0x80000000
- brk(addr): adresse absolue
- sbrk(décalage): agrandissement ou rétrécissement

0xFFFF

brk file

monceau

0x0000

Choix des pages à évincer en réalité

- Algorithm "Pas utilisé récemment" pour une question de performance
- Approximation de LRU basé sur le bit d'accès de la page
- Le système d'exploitation met à zéro le bit d'accès des pages périodiquement
- Le processeur remet à 1 le bit lors d'un accès à la page (le système d'exploitation n'intervient pas)
- Si le bit d'accès à une page reste à zéro, alors il s'agit d'une page inutilisée récemment
- Le noyau maintient une liste de pages froides sujettes à être évincées sur disque

Ensemble de travail (Working Set Size)

Cache des pages

Cache des pages pour 2MiB est + rapide

Libère toutes les pages en cache

Gouille bavu qui a P dans le cache

Mais il faudra recharger les pages depuis le disque

Métriques sur la mémoire

- VIRT: Somme de toute la mémoire virtuelle
 - Surestime l'utilisation réelle de la mémoire
 - Certaines pages sont sur disque ou partagées
- RES: Resident (RAM occupée)
- SHR: mémoire du processus + proportion de pages partagées

francs libertés

total: 1MiB

used: 2.8GiB

free: 3.1GiB

shared: 0.1GiB

buff/cache: 1.6GiB

available: 3.4GiB

francs libertés

total: 1MiB

used: 2.8GiB

free: 5.4GiB

shared: 4.7GiB

buff/cache: 9.3GiB

available: 4.8GiB

francs libertés

total: 1MiB

used: 2.8GiB

free: 1.4GiB

shared: 0.2GiB

buff/cache: 4.6GiB

available: 4.8GiB

Allocation continue (1)

Consiste à enregistrer les fichiers en séquence

Exemple: Soit trois fichiers F1, F2, F3. Lorsque F2 est effacé, il laisse un trou qui n'est pas assez grand pour un nouveau fichier F4. Il faut déplacer F3 (copier) pour assembler l'espace libre, une opération coûteuse.

Allocation continue de l'espace libre → fragmentation externe

Même si l'espace libre est suffisant, il devient inutilisable

Méthode utilisée pour l'écriture unique (ex: céderom)

Allocation par blocs

Le système de fichier abstrait l'emplacement des blocs

Le fichier apparaît continu

Agrandir ou rétrécir un fichier se fait par bloc entier

Méthode généralement utilisée pour l'allocation dynamique (ex: disque dur)

Allocation par blocs indexés

Il ne faut pas oublier que chaque région a ses propres pages. En effet, la section text n'a pas d'index executable et a une protection lecture/exécution, alors que la pile croît vers le bas et vient avec lecture et écriture, par exemple. Dans le cas où il y a plusieurs entrées dans l'espace content 64KiB/4KiB = 16 pages, 8 prises par le texte, 5 pour la pile et 11 pour les données (16388 / 4 * 4096) et 4 requises pour la pile, ce qui est plus que l'espace disponible. L'exécution de ce programme pourrait fonctionner, mais sa performance sera réduite en raison des fautes de page pour la pagination.

Une instruction assembler charge une valeur de 32 bits dans un registre sur une architecture n'ayant pas de contrainte d'alignement. Combien de défauts de pages pourraient subvenir à cause de cette instruction?

Cette architecture n'ayant pas de contrainte d'alignement, il est possible que l'instruction et la valeur à charger soit à la fin d'une page et s'étende sur la suivante. Si cette situation survient, il se produit un total de 4 fautes de pages pour cette seule instruction.

Une page peut-elle faire partie de l'espace de travail de deux processus en même temps?

Bien sûr, il est tout à fait possible pour deux processus d'utiliser le même code exécutable (programme ou bibliothèque partagée), le même fichier attaché en mémoire ou la même zone de mémoire partagée.

Un ordinateur possède des adresses virtuelles de 32 bits et des pages de 4Ki. Un programme utilise un secteur de 32768 octets, data 16388 octets et pile 15870 octets. Est-ce que ce programme peut entrer dans l'espace disponible?

Nombre maximal de fichiers total

Nombre maximal de fichiers par répertoire

Taille maximale d'un fichier

Taille maximale du volume

Un ordinateur fournit à chaque processus un espace de 65536 octets divisé en pages de 4Ki. Un programme utilise un secteur de 32768 octets, data 16388 octets et pile 15870 octets. Est-ce que ce programme peut entrer dans l'espace disponible?

Index avec taille: un bloc possède le début d'un nombre maximal de fichiers par répertoire

Format du système de fichier FAT

Périphérique

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Table des fichiers

Fichier	Index
A	18
B	9
C	2

Table d'allocation des blocs

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Contraintes

Longueur maximale des noms de fichiers

Caractères autorisés dans les noms de fichiers

Sensibilité à la case

Nombre maximal de fichiers total

Nombre maximal de fichiers par répertoire

Taille maximale d'un fichier

Taille maximale du volume

Un ordinateur fournit à chaque processus un espace de 65536 octets divisé en pages de 4Ki. Un programme utilise un secteur de 32768 octets, data 16388 octets et pile 15870 octets. Est-ce que ce programme peut entrer dans l'espace disponible?

Index avec taille: un bloc possède le début d'un nombre maximal de fichiers par répertoire

Format du système de fichier FAT

Périphérique

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Table des fichiers

Fichier	Index
A	18
B	9
C	2

Table d'allocation des blocs

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Contraintes

Longueur maximale des noms de fichiers

Caractères autorisés dans les noms de fichiers

Sensibilité à la case

Nombre maximal de fichiers total

Nombre maximal de fichiers par répertoire

Taille maximale d'un fichier

Taille maximale du volume

Un ordinateur fournit à chaque processus un espace de 65536 octets divisé en pages de 4Ki. Un programme utilise un secteur de 32768 octets, data 16388 octets et pile 15870 octets. Est-ce que ce programme peut entrer dans l'espace disponible?

Index avec taille: un bloc possède le début d'un nombre maximal de fichiers par répertoire

Format du système de fichier FAT

Périphérique

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Table des fichiers

Fichier	Index
A	18
B	9
C	2

Table d'allocation des blocs

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Contraintes

Longueur maximale des noms de fichiers

Caractères autorisés dans les noms de fichiers

Sensibilité à la case

Nombre maximal de fichiers total

Nombre maximal de fichiers par répertoire

Taille maximale d'un fichier

Taille maximale du volume

Un ordinateur fournit à chaque processus un espace de 65536 octets divisé en pages de 4Ki. Un programme utilise un secteur de 32768 octets, data 16388 octets et pile 15870 octets. Est-ce que ce programme peut entrer dans l'espace disponible?

Index avec taille: un bloc possède le début d'un nombre maximal de fichiers par répertoire

Format du système de fichier FAT

Périphérique

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Table des fichiers

Fichier	Index
A	18
B	9
C	2

Table d'allocation des blocs

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Contraintes

Longueur maximale des noms de fichiers

Caractères autorisés dans les noms de fichiers

Sensibilité à la case

Nombre maximal de fichiers total

Nombre maximal de fichiers par répertoire

Taille maximale d'un fichier

Taille maximale du volume

Un ordinateur fournit à chaque processus un espace de 65536 octets divisé en pages de 4Ki. Un programme utilise un secteur de 32768 octets, data 16388 octets et pile 15870 octets. Est-ce que ce programme peut entrer dans l'espace disponible?

Index avec taille: un bloc possède le début d'un nombre maximal de fichiers par répertoire

Format du système de fichier FAT

Périphérique

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Table des fichiers

Fichier	Index
A	18
B	9
C	2

Table d'allocation des blocs

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Contraintes

Longueur maximale des noms de fichiers

Caractères autorisés dans les noms de fichiers

Sensibilité à la case

Nombre maximal de fichiers total

Nombre maximal de fichiers par répertoire

Taille maximale d'un fichier

Taille maximale du volume

Un ordinateur fournit à chaque processus un espace de 65536 octets divisé en pages de 4Ki. Un programme utilise un secteur de 32768 octets, data 16388 octets et pile 15870 octets. Est-ce que ce programme peut entrer dans l'espace disponible?

Index avec taille: un bloc possède le début d'un nombre maximal de fichiers par répertoire

Format du système de fichier FAT

Périphérique

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Table des fichiers

Fichier	Index
A	18
B	9
C	2

Table d'allocation des blocs

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Contraintes

Longueur maximale des noms de fichiers

Caractères autorisés dans les noms de fichiers

Sensibilité à la case

Nombre maximal de fichiers total

Nombre maximal de fichiers par répertoire

Taille maximale d'un fichier

Taille maximale du volume

Un ordinateur fournit à chaque processus un espace de 65536 octets divisé en pages de 4Ki. Un programme utilise un secteur de 32768 octets, data 16388 octets et pile 15870 octets. Est-ce que ce programme peut entrer dans l'espace disponible?

Index avec taille: un bloc possède le début d'un nombre maximal de fichiers par répertoire

Format du système de fichier FAT

Périphérique

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Table des fichiers

Fichier	Index
A	18
B	9
C	2

Table d'allocation des blocs

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Contraintes

Longueur maximale des noms de fichiers

Caractères autorisés dans les noms de fichiers

Sensibilité à la case

Nombre maximal de fichiers total

Nombre maximal de fichiers par répertoire

Taille maximale d'un fichier

Taille maximale du volume

Un ordinateur fournit à chaque processus un espace de 65536 octets divisé en pages de 4Ki. Un programme utilise un secteur de 32768 octets, data 16388 octets et pile 15870 octets. Est-ce que ce programme peut entrer dans l'espace disponible?

Index avec taille: un bloc possède le début d'un nombre maximal de fichiers par répertoire

Format du système de fichier FAT

Périphérique

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Table des fichiers

Fichier	Index
A	18
B	9
C	2

Table d'allocation des blocs

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Contraintes

Longueur maximale des noms de fichiers

Caractères autorisés dans les noms de fichiers

Sensibilité à la case

Nombre maximal de fichiers total

Nombre maximal de fichiers par répertoire

Taille maximale d'un fichier

Taille maximale du volume

Un ordinateur fournit à chaque processus un espace de 65536 octets divisé en pages de 4Ki. Un programme utilise un secteur de 32768 octets, data 16388 octets et pile 15870 octets. Est-ce que ce programme peut entrer dans l'espace disponible?

Index avec taille: un bloc possède le début d'un nombre maximal de fichiers par répertoire

Format du système de fichier FAT

Périphérique

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Table des fichiers

Fichier	Index
A	18
B	9
C	2

Table d'allocation des blocs

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Contraintes

Longueur maximale des noms de fichiers

Caractères autorisés dans les noms de fichiers

Sensibilité à la case

Nombre maximal de fichiers total

Nombre maximal de fichiers par répertoire

Taille maximale d'un fichier

Taille maximale du volume

Un ordinateur fournit à chaque processus un espace de 65536 octets divisé en pages de 4Ki. Un programme utilise un secteur de 32768 octets, data 16388 octets et pile 15870 octets. Est-ce que ce programme peut entrer dans l'espace disponible?

Index avec taille: un bloc possède le début d'un nombre maximal de fichiers par répertoire

Format du système de fichier FAT

Périphérique

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19

Table des fichiers

Fichier	Index
A	18
B	9

Modes d'exécution du processeur

- Mode privilégié
 - Accès global à la machine
 - Sélectionne les tâches à exécuter
 - Peut interrompre ou terminer une tâche
 - Gère les interruptions
 - Attribue la mémoire
 - Arbitrage des accès aux périphériques
 - Libère les ressources utilisées par une tâche
 - Écoute dans les files, s'exécute le noyau du système d'exploitation
 - Crash du noyau est (souvent) fatal

En C, la convention est d'utiliser la valeur de Si un appel système échoue, alors on retourne l'erreur à la fonction appelant (parent)

- Appel système: valeur < 0 signifie erreur

Si on ne vérifie pas le code de retour, le programme peut continuer dans un mauvais chemin!

Mémoire allouée, mais jamais libérée

- Si survient dans une boucle, alors peut épuiser la mémoire de l'ordinateur

- Instrumentation de malloc/free, bilan à la fin

- Analyse statique v.s. analyse à l'exécution

Appel système wait()

Retourne seulement quand l'enfant s'est terminé (bloque le parent)

Retourne immédiatement si aucun enfant

- Sinon, on attendrait un événement qui ne viendrait jamais!

Typiquement, fork() et wait() s'utilisent ensemble

Création: creat()

Ouvrir: open()

Fermer: close()

Lire: read()

Écrire: write()

Réinitialiser ou agrandir: truncate()

Exécution

Échéance (si temps-redd)

<p