Systèmes d'Exploitation

Timothy Bourke

21 février 2024

Première partie
Mémoire Virtuelle

Un OS est constitué d'un Noyau, de Bibliothèques et d'Applications pour partager la mémoire, le temps de calcul et les périphériques/composants, pour abstraire une base pour construire de plus gros systèmes avec des services communs, de la concurrence, de la communication et une indépendance matérielle des applications. Le noyau contient de nombreuses structures de données et des fonctions de pagination, des modules abstraits pour les services, et du contrôle matériel bas-niveau (drivers) pour la gestion de la concurrence et des évènements.

1 Espaces d'Adresses et Traduction d'Adresses

1.1 Espaces d'Adresses

Un OS doit partager des ressources finies entre plusieurs utilisateurs/applications et proposer des abstractions pour la construction d'applications. La mémoire physique doit être partagée, mais que faire si l'on en n'a pas assez ? Si un processus tent de lire ou d'écrire sur la mémoire d'un autre ? Les abstractions sont utiles pour les OS et les développeurs d'applications : on fait tourner chaque processus dans un espace d'adresses virtuel, on utilise de la mémoire physique (plus rapide) comme cache pour des fichiers sur disque (plus lent) et on peut communiquer entre les processus en sélectionnant de la mémoire à partager.

La mémoire virtuelle est une abstraction proposée par un mix sophistiqué et élégant de matériel et de logiciel :

Matériel Le matériel permet un système d'exceptions (interuptions synchronisées), de traduction d'adresses, d'avoir du cache et une mémoire principale ainsi que des fichiers sur disque. Il est nécessaire pour intervenir au plus bas niveau - chaque instruction `mov` - et pour la vitesse d'exécution.

Logiciel Dans le logiciel, le noyau va contenir un système de mémoire. Il est nécessaire pour implémenter des algorithmes flexibles et sophistiqués et pour une intégration dense dans un noyau.

La mémoire virtuelle est présente à tous les niveaux d'un système d'ordinateur, comprendre son fonctionnement revient à comprendre comment le système fonctionne. Elle offre des capacités puissantes qu'on peut exploiter dans des applications. Les développeurs d'OS ne peuvent pas éviter de connaître la mémoire virtuelle.

Les systèmes qui n'utilisent pas d'adressage virtuel sont utilisé dans des systèmes embarqués, tandis que l'adressage virtuel est utilisé partout ailleurs et utilise une unité de gestion de mémoire (MMU) entre le CPU et la mémoire physique.

1.2 Traduction d'Adresses

Définition 1.1: Espaces d'Adresses

- L'Espace d'Adresses Virtuel V est un ensemble de $N = 2^n$ adresses virtuelles utilisées dans les programmes
- L'Espace d'Adresses Physique P est un ensemble de $M = 2^m$ adresses physiques de la RAM.

Définition 1.2: Traduction d'Adresses

Une fonction de traduction d'adresses est une fonction $VMAP : V \to P$ `option` qui a une adresse virtuelle a va associer :

- $VMAP(a) =$ `Some` a' si la donnée à l'adresse virtuelle a dans V est à l'adresse physique a' dans P.
- $VMAP(a) =$ `None` si la donnée à l'adresse virtuelle a dans V n'est pas dans la mémoire physique.

On divise les espaces d'adresses en pages numérotées (blocs de taille spécifique) et on s'aligne toujours sur la taille d'une page, en empêchant les dépassements. On représente cette fonction de traduction par une table de pages. On approxime ensuite la correspondance par une liste d'entrées, une par page virtuelle. On spécifie alors `None` ou `Some` numéro de page physique. L'idée est alors que chaque processus va avoir son espace virtuel d'adressage pour lequel la mémoire est une simple array d'octets. Il faut alors bien choisir la fonction de traduction pour simplfier l'allocation mémoire. Chaque page virtuelle peut être map à n'importe quelle page physique et une page virtuelle peut être stockée dans différentes pages physiques à la fois. On a alors le processus suivant :

1. Le CPU envoie une adresse virtuelle à la MMU. (VA)
2. La MMU demande une table de pagination de la mémoire. (PTEA)
3. La Mémoire envoie une table de pagination à la MMU. (PTE)
4. La MMU envoie l'adresse physique au cache/à la mémoire. (PA)
5. Le cache/la mémoire envoie le mot de données au processeur. (DATA)

On peut accélérer la traduction avec un Buffer Lookaside de Traduction (TLB) : Un cache matériel dans la MMU qui contient une correspondance entre numéros virtuels et physiques de pages et qui contient même les entrées complètes de la table de pagination pour un petit nombre de pages. Ceci permet d'éliminer un accès mémoire en cas de hit, mais coûte un accès mémoire supplémentaire sinon. Les miss sont rares grâce à la localité des programmes.

1.3 Protection

Chaque processus (utilisateur) a son espace virtuel défini par une table de pagination distincte, qu'il ne devrait pas être autorisé à modifier directement. Un processeur a donc au moins 2 modes d'exécution : Mode Noyau/Superviseur et mode Utilisateur. Des instructions spéciales ou des évènements permettent de changer d'un mode d'exécution à l'autre. On étend alors les PTEs avec des bits de permissions (Sup, Read, Write). Les permissions de page sont vérifiées à chaque instruction de déplacement vers ou depuis la mémoire et en cas de problème, l'OS envoie le processus SIGSEGV - SEGFAULTTTTTTTTTTTTT.

1.4 Pagination Multi-Niveaux

Si on se donne un espace d'adresses virtuel de 48 bits, une taille de page de 4kB (2^{12}) et des PTE de 8 octets, on aurait besoin d'une table de pagination de $512GB$. On utilise alors des tables de pagination multi-niveaux : chaque PTE pointe vers une table de pagination du niveau suivant, ce qui permet de réduire l'usage de mémoire. On peut créer les tables de pagination de niveau supérieur à la volée, la plupart de l'espace étant généralement non-alloué. Grâce à des TLB à

chaque niveau ou presque, on peut faire de sorte que la recherche d'une donnée soit presque aussi rapide qu'avec un seul niveau de pagination.

2 L'Architecture x86

2.1 Adressage par Décalage et Mode Réel

Le premier processeur Intel 16-bit utilisait des mots de 16 bits, des adresses de 20 bits, de la segmentation mémoire et de l'adressage physique. Pour accéder à un espace d'adresses physique sur 20 bits avec des registres 16 bits, on utilise un registre spécial de segments pour gagner les 4 bits manquants : L'adresse physique est l'adresse relativement à l'adresse de segments : on shift left de 4 bits.