INFO3203 - ARCHITECTURE & SYSTÈME 2023-2024

TD 5 - GESTION DE LA MÉMOIRE

GESTION DE LA MÉMOIRE

Dans ce TD, vous allez commencer à étudier le fonctionnement de la gestion de la mémoire d'un système d'exploitation. Dans la première partie du TD, vous examinerez l'utilisation de la mémoire de votre machine. Dans la seconde partie, vous implémenterez quelques fonctions qui reproduisent les fonctionnalités offertes par votre système.

GESTION DE LA MÉMOIRE EN BASH

Dans cet exercice, nous utiliserons deux outils fournis par le système d'exploitation qui peuvent aider à comprendre l'utilisation et l'administration du système.

`top`

Lancez la commande `top` et identifiez le processus qui utilise le plus de mémoire. Répondez aux questions suivantes :

- Quelle quantité de mémoire le processus utilise-t-il ?
- Combien de mémoire virtuelle le processus a-t-il alloué ?
- Quelle quantité de mémoire virtuelle se trouve actuellement dans la mémoire physique ?

`/proc/meminfo`

Nous avons vu en classe que le fichier `/proc/meminfo` fournit des informations détaillées sur les ressources mémoire de la machine utilisée. Affichez le fichier et répondez aux questions suivantes :

- Quelle est la quantité de mémoire physique de votre système ?
- Combien de mémoire virtuelle votre système peut-il allouer ?
- Combien de pages sont actuellement en mémoire ?

IMPLÉMENTATION DE LA PAGINATION

La gestion de la mémoire dans les systèmes d'exploitation comporte deux éléments.

1. Le premier composant est un allocateur de mémoire physique pour le noyau, afin que ce dernier puisse allouer de la mémoire et la libérer par la suite.

2. Le second composant est la mémoire virtuelle, qui fait correspondre les adresses virtuelles utilisées par le noyau et le logiciel utilisateur aux adresses de la mémoire physique. L'unité de gestion de la mémoire (MMU) effectue le mappage lorsque les instructions utilisent la mémoire, en consultant un ensemble de tables de pages.

Dans cet exercice, vous allez mettre en œuvre un certain nombre de fonctions qui reproduisent le premier composant décrit précédemment. Votre "allocateur" fonctionnera par unités de 4096 octets, appelées pages. Votre tâche consistera à maintenir des structures de données qui enregistrent les pages physiques libres et celles qui sont allouées, ainsi que le nombre de processus qui partagent chaque page allouée. Vous écrirez également les routines permettant d'allouer et de libérer des pages de mémoire.

Détails

Le système d'exploitation doit savoir quelles parties de la mémoire physique sont libres et lesquelles sont en cours d'utilisation. Le système d'exploitation gère la mémoire physique du PC avec une granularité de page afin de pouvoir utiliser la MMU pour mapper et protéger chaque partie de la mémoire allouée.

Vous allez maintenant écrire l'allocateur de pages physiques. Il garde la trace des pages libres grâce à une liste chaînée de `struct PageInfo`, chacun correspondant à une page physique. Vous devez écrire l'allocateur de pages physiques avant de pouvoir penser/écrire le reste de l'implémentation de la mémoire virtuelle, parce que votre code de gestion des tables de pages devra allouer de la mémoire physique dans laquelle stocker les tables de pages.

Pour ce faire, implémentez les fonctions suivantes :

- `struct mmap* init()` : cette fonction initialise l'allocateur de mémoire. Elle génère les structures de données nécessaires pour gérer la mémoire. De plus, elle alloue un "grand" espace mémoire que vous utiliserez pour émuler votre mémoire physique (évidemment, vous ne travaillerez pas avec votre vraie mémoire physique 😉).
- `struct PageInfo* page_alloc(struct mmap* mmap)` : cette fonction alloue une page physique. NOTE : la fonction doit remplir la totalité de la page physique retournée avec des octets `'\0'`.
- `struct PageInfo* page_free(struct mmap* mmap)` : renvoie une page dans la liste des pages libres.
- `int check_page_free_list(struct mmap* mmap)` : renvoie le nombre de pages libres

(vous l'avez deviné) `swap`. Vous pouvez stocker ce fichier n'importe où dans votre système de fichiers (je suggère de le stocker dans le dossier `/tmp/`).

- `struct PageInfo* read_from_swap(struct mmap* mmap)` : Lit une page du swap et la supprime.

NOTEZ : Ceci est supposé être un TD, ce qui signifie que vous devriez l'utiliser comme une excuse pour vérifier que vous travaillez correctement vers l'objectif de l'exercice. C'est particulièrement important cette semaine car vous allez réutiliser ce code dans un futur proche. N'hésitez pas à poser des questions !

Modifié le: mercredi 7 février 2024, 18:04

◄ TD4 - Raisonnement sur les OS

Aller à…

TD 1 ►

ADMINISTRATION

› Administration du cours

PRATIQUE

Connecté sous le nom « Augustin Lucas » (Déconnexion)
INFO3203 - Architecture & Système 2023-2024

Enseignement et Formations
Étudiants
Enseignants et Personnels

Obtenir l'app mobile
Politiques

BESOIN D'AIDE ?

Vous êtes étudiant ? enseignant ? Vous avez une question concernai le portail ?

Écrivez à l'assistance