# Système d'exploitation

Examen – 28 mai 2008

2h

*Le barême est donné à titre indicatif. Les différentes parties sont totalement indépendantes.*
*L'examen est volontairement trop long,passez à une autre partie si vous êtes bloqués.*

## I. Questions d'échauffement *6 points*

*Les sous-questions suivantes sont indépendantes.*

a) Que pensez-vous de la phrase suivante ?
« L'ordonnanceur est un processus qui se réveille régulièrement et change les processus en cours d'exécution sur les différents processeurs. » *2 pts*

b) Quels sont les points communs et différences entre un appel-système et une interruption au niveau matériel et logiciel ? *2 pts*

c) Quel est l'intérêt de l'appel-système vfork ? Que pourrait-il se passer si le père reprenait son exécution avant que le fils n'ait fait exec ou exit ? *2.pts*

## II. Ordonnancement *8 points*

On suppose qu'aux temps T=0 puis 4, 8, 12 et 16, une tâche A de durée 1 et une tâche B de durée 5 sont soumises au système.

a) Donner l'ordonnancement de ces tâches avec la politique FIFO. *2 pts*

On suppose maintenant que les tâches de durée 1 ont une priorité supérieure aux autres.

b) Que devient l'ordonnancement si l'ordonnanceur FIFO est capable de privilégier les tâches prioritaires ? A quel autre politique sans priorité ce cas ressemble-t-il ? *3 pts*

c) Et si l'ordonnanceur FIFO avec priorités est capable de préempter la tâche en cours par une nouvelle tâche plus prioritaire ? Indiquer quelle tâche se fait préempter à quel moment. *3 pts*

## III. Gestion mémoire *9 points*

a) Expliquer rapidement à quoi sert le pseudo code suivant ? Quand est-il appelé ?

*4 pts*

```
1     traitant(adresse, processus, en_ecriture)
2           trouver la zone mémoire du processus contenant adresse
3           si zone invalide
4                 tuer le processus
5           si zone valide
6                 si page absente
7                       allouer une page
8                       si page non anonyme
9                             lire la page depuis le disque
10                      mettre page dans table de pages du processus
11                otherwise si page présente
12                      si page en lecture seule et en_ecriture
13                            tuer le processus
```

b) Combien d'entrées-sorties peuvent avoir lieu pendant son exécution ? Combien de temps cette exécution peut-elle durer sur une machine réelle ? En déduire la durée approximative du chargement d'un programme de 4Mo avec des pages de 4ko si un système d'exploitation utilise bêtement le code ci-dessus. Comment les systèmes d'exploitation font-ils pour réduire cette durée ? *3 pts*

c) Expliquer rapidement comment ajouter le support du *Copy-on-Write* dans ce pseudo-code. *2 pts*

## IV. Table de pages *9 points*

On suppose disposer d'une architecture 64bits avec 2 niveaux de table de pages matériel et des pages de 16ko ($2^{14}$ octets). La table de pages est remplie avec des tableaux de pointeurs vers les sous-pages, ou des PTE de taille 8 octets.

*Pour toutes les questions suivantes, on pourra arrondir les calculs.*

*Rappel:* $1\ k = 10^3 = 2^{10}$ $1\ M = 10^6 = 2^{20}$ $1\ G = 10^9 = 2^{30}$ $1\ T = 10^{12} = 2^{40}$

a) Combien de bits sont nécessaires pour décrire un décalage dans une page ? Combien de pointeurs ou PTE peut-on mettre par page ? En déduire le nombre de bits utilisé pour chaque niveau de la table. En déduire la taille maximale des adresses virtuelles manipulées sur cette architecture. *3 pts*

On suppose maintenant qu'une machine dispose de 256 Mo de mémoire physique et une quantité infinie de swap, ce qui permet de stocker l'intégralité des données des processus.

b) Si le système remplit l'intégralité de la table de pages même si le processus de n'utilise pas, quelle quantité de mémoire physique peut être gaspillée pour chaque processus ? Combien de tables de pages de différents processus pourrait-on stocker en mémoire physique ? Quel problème rencontre-t-on lors de l'exécution des processus si toute la mémoire physique est utilisée pour stocker les tables de pages ? *3 pts*

Les systèmes peuvent donc être amenés à limiter le nombre de processus pour garder de la mémoire physique disponible pour les données des processus et pas seulement stocker les tables de pages.

c) Quelles techniques peut-on utiliser pour réduire la mémoire physique nécessaire en permanence pour une table de pages ? Même si la table des pages doit être entièrement remplie car le processus utilise intégralement son espace d'adressage ? Montrer par un calcul rapide que cela permet d'augmenter grandement la limite sur le nombre de processus possibles simultanément. *3 pts*

## V. Entrées-sorties *8 points*

On considère le sous-système gérant les émissions réseau dans un système d'exploitation. On suppose disposer d'une carte réseau capable d'accéder à la mémoire centrale par DMA à 100Mo/s et d'envoyer sur le câble réseau à la même vitesse.

a) Rappeler rapidement ce qu'est un DMA et comment la carte va l'utiliser pour émettre des données vers le réseau. Comment le pilote peut-il indiquer à la carte quand démarrer l'émission et où se trouvent les données ? *2 pts*

b) Si on suppose vouloir envoyer 1 Mo de données, donner le coup approximatif des différentes étapes de l'entrée-sortie physique permettant l'émission vers le réseau. Quelle stratégie vaut-il mieux utiliser dans le pilote de la carte pour attendre la terminaison du traitement ? *2 pts*

c) Les systèmes copient souvent les données dans la mémoire noyau avant de les soumettre à la carte pour émission. Quel avantage cette méthode procure-t-elle pour les points suivants ?

- La durée du blocage de l'application pendant l'émission « logique »
- La retransmission éventuelle des paquets perdus un peu plus tard
- Le verrouillage mémoire

Quels sont les inconvénients ? *4 pts*