Examen IN201 - Cours de système d'exploitation

14 Janvier 2019

1. (1 point) Décrivez en quelques lignes les principaux rôles du système d'exploitation.
2. (1 point) Quels sont les principaux composants matériels d'un ordinateur?
3. (4 points) Décrivez la séquence des opérations faites par le processeur pour exécuter une instruction, lorsque la prochaine instruction à exécuter est de type `load r1, r2` (stocke contenu de la mémoire à l'adresse contenue dans `r2` dans le registre `r1`).
4. (3 points) Sous UNIX, l'appel système `read` permet de copier des données en provenance d'un file descriptor. Un file descriptor représente par exemple une connection réseau ou un fichier ouvert précédemment.
   (a) L'appel à `read` peut être bloquant, c'est à dire bloquer le thread qui l'appelle. Qu'est-ce que cela signifie?
   (b) À quelle famille de mécanisme de contrôle d'accès correspondent ces file descriptors?
5. (4 points) On vous donne le programme suivant:

```
void main(void){ int b = 3; f(b); return g(b); }
void g(int c){ int i = 1; int j = i + c; if(j == 11) print_secret();
return; }
void f(int d);
```

   (a) En supposant qu'il n'y a pas d'optimisation, dessinez l'état de la pile avant l'instruction `if;` de `g`.
   (b) On suppose maintenant que le programmeur a oublié d'initialiser `i` (on supprime la partie soulignée). Écrivez une fonction `f` de manière à ce que `g` affiche le secret.
6. (4 points) Vous avez 6 gros programmes à exécuter sur 3 processeurs: A prend 70 minutes, B, C et D 60, E et F 40.
   (a) Quel est le temps minimal requis pour exécuter toutes les tâches?
   (b) Quel algorithme permet d'exécuter toutes les tâches dans ce temps minimal?
   (c) Dessinez un plan d'ordonnancement permettant d'exécuter toutes ces tâches dans le temps minimum.
   (d) Est-ce que ce plan d'ordonnancement fait des préemptions et/ou des migrations? Si oui, dites à quels moment et les tâches concernées.
7. (3 points) Vous développez un site web permettant de catégoriser des musiques automatiquement à partir de fichiers MP3 qu'on vous envoie. Chaque requête d'un client est traitée en plusieurs étapes:
   1. Récupération du fichier du client à partir du réseau et mise en mémoire RAM.

2. Calcul de la signature du fichier.
3. Récupération des informations sur la chanson correspondant à la signature dans une grosse base de donnée située sur le disque dur.
4. Envoi du nom de l'artiste au client sur le réseau.

(a) Quel est l'intérêt de séparer chacune de ces étapes en plusieurs threads? Autrement dit: quel problème aurait-on en traitant les requêtes dans un code séquentiel?

(b) Expliquez par quel mécanismes du matériel et de l'OS, la séparation de ces différentes étapes en plusieurs processus, permet à une erreur de manipulation de pointeur dans l'étape 1 d'éviter d'affecter les autres processus.

(c) Est-ce que le processus correspondant à l'étape 1. doit avoir accès au système de fichier, et en particulier au fichier de base de donnée? Justifiez votre réponse.