TP 1 : Gestion de la mémoire

RSA : Réseaux et Systèmes Avancés
Deuxième année

★ Exercice 1: Informations générales sur le système

Pour avoir des informations sur le système, de nombreuses commandes sont disponibles sous linux.

▷ **Question 1:** Quelles sont les différentes options possibles de la commande `uname` ? Quelles sont les informations fournies ? Que fournit la commande `arch` ?

▷ **Question 2:** Que fournit la commande `uptime`.

Système de fichiers /proc (procfs). /proc est un pseudo-système de fichiers qui n'existe pas sur le disque et il ne semble exister qu'au moment où on référence un fichier ou un sous-répertoire. /proc est utilisé comme interface avec les structures de données du noyau. La plupart des fichiers sont en lecture seule, mais quelques uns permettent la modification de variables du noyau.
On peut utiliser les commandes habituelles comme `ls` pour connaitre le contenu d'un sous-répertoire du /proc, `cat` pour lire des informations à partir de fichiers ...
Pour la suite un `man proc(5)` vous sera utile.

▷ **Question 3:** Taper les commandes suivantes en interprétant l'essentiel des résultats à chaque fois :

```
cat /proc/sys/kernel/hostname
grep cpu /proc/stat
more /proc/loadavg
cat /proc/uptime
cat /proc/version
cat /proc/sys/fs/file-max
```

▷ **Question 4:** En se basant sur les informations fournies par `cat /proc/cpuinfo`, donner la commande ou la suite de commandes shell permettant de donner les informations suivantes :

- le nombre de processeurs ;
- le nombre de processeurs virtuels ;
- si les processeurs sont dual ou multi-core ;
- si les processeurs sont 64 bits.

▷ **Question 5:** Il est aussi possible de changer des paramètres (variables) du noyau. Exemple : `nouveauNom > /proc/sys/kernel/hostname`
Quel est le message obtenu ? Pourquoi ?
Afin de vérifier l'ensemble des droits (comme pour les fichiers et répertoires usuels), on peut utiliser `ls -l`. Vérifier les droits du fichier ***/proc/sys/kernel/hostname***.

★ Exercice 2: Informations sur la mémoire

▷ **Question 1:** En utilisant `ps`, afficher la liste de tous les processus avec le format
`commande tailleV tailleR Pmem`
où `tailleV` est la taille de la mémoire virtuelle du processus, `tailleR` est la quantité de la mémoire physique utilisée par le processus, et `Pmem` est le pourcentage de la mémoire correpondant.

▷ **Question 2:** Que fait la commande `free` ? Comment faire pour obtenir un nouvel affichage toutes les 5 secondes ?

▷ **Question 3:** Le programme `free` utilise ***/proc/meminfo***. Comment utiliser ce dernier pour afficher sur une seule ligne la quantité de mémoire disponible sur le système ?

▷ **Question 4:** Quelles sont les informations fournies par ***/proc/stat*** ? Même question pour ***/proc/<pid>/stat***, ***/proc/<pid>/statm*** et `cat /proc/swaps`

▷ **Question 5:** Quelles informations peut-t-on obtenir en utilisant la commande `vmstat` ? Noter en particulier celles relatives à la gestion de la mémoire.

★ **Exercice 3: Alignement des structures** Le but de cet exercice est d'étudier l'impact de l'organisation des structures de données sur les performances de la mémoire en termes de nombre de défauts de pages.

- Récupérer ***/home/depot/2A/RSA-Maimour/align.tgz***
- Compiler les sources à l'aide de `make`
- Lire useGetrusage.c et comprendre ce qu'elle fait

▷ **Question 1:** Comparer la structure tableElement des fichiers align.c et nonAlign.c.

▷ **Question 2:** Exécuter align et nonAlign. Que remarquez vous par rapport à la taille d'un élément du tableau dans les 2 programmes ? C'est dû à quoi d'après vous ?

▷ **Question 3:** Sachant qu'une page mémoire est de 4Ko, combien de pages mémoire sont nécessaires pour stocker le tableau dans les 2 programmes ? Retrouver ces valeurs par un calcul simple !

▷ **Question 4:** Quels sont vos remarques par rapport aux autres paramètres ?

▷ **Question 5:** Quelle règle de programmation pouvez-vous vous fixer par rapport aux structures ?

★ **Exercice 4: Écroulement**

- Récupérer ***/home/depot/2A/RSA-Maimour/ecroul.tgz***
- Compiler les sources à l'aide de `make`

▷ **Question 1:** gentil.c, gentil2.c et mechant.c initialisent un tableau de N*4096 caractères. Comparer leur manière de travailler.

▷ **Question 2:** Déterminer à l'aide de la commande free, la quantité de mémoire physique disponible et la taille de l'espace disque disponible pour le swap

▷ **Question 3:** Dans une fenêtre 1, exécuter la commande vmstat 1 et dans une fenêtre 2, taper :

```
/usr/bin/time ./gentil
/usr/bin/time ./gentil2
/usr/bin/time ./mechant
```

Commenter l'évolution des champs r, b, free, si, so, id (dans la fenêtre 1) et les différences de temps d'exécution entre les différents processus. Au vu de la valeur de NBMEG, à combien de défauts de pages mineurs pouvait-on s'attendre ? Comparer avec le résultat affiché par time.

▷ **Question 4:** Faire plusieurs exécutions en changeant la valeur de NBMEG dans constante.h jusqu'à la valeur maximale possible. N'oubliez pas le make à chaque fois.