TP 2 : Gestion des processus

RSA : Réseaux et Systèmes Avancés
Deuxième année

★ Exercice 1: Informations sur les processus

▷ **Question 1:** Quelle est la différence entre les commandes `ps` et `top` ? Que fait la commande `pstree` ?

▷ **Question 2:** Comment utiliser la commande `ps` pour obtenir la liste des processus en première colonne et leur état en 2ème colonne ? Quels sont les états possibles ?

Les commandes `ps` et `top` utilise le /proc pour récupérer leurs informations.

★ Exercice 2: procfs

Des informations sur les processus en cours d'exécution peuvent être trouvées à partir du /proc. Un processus est représenté avec un sous répertoire de /proc avec comme nom son pid.
En tapant `ls /proc`, un certain nombre de numéros s'affichent. Il s'agit des pid des processus du système.

▷ **Question 1:** Quelles sont les informations concernant un processus donné qu'on peut avoir ? `ls -l /proc/[pid]/`
En consultant les pages du manuel, déterminer la signification de chacune de ces informations.

▷ **Question 2:** Comment obtenir l'ID du processus parent d'un processus donné ?

★ Exercice 3: Gestion des Processus sous Linux 2.6

Pour cet exercice, on utilise LXR (Linux Cross-Referencing) disponible à ***http ://lxr.free-electrons.com***. C'est un outil puissant qui permet de naviguer dans le code source des projets de taille importante. La navigation se fait en utilisant un navigateur web avec des liens sur les différents fichiers, identificateurs, ...utilisés au niveau des sources.
Pour les besoins de ce TP, cliquer sur le lien qui correspond à une version quelconque dans la section `Versions`, ensuite spécifier la version 2.6.38 au niveau de l'url directement.

▷ **Question 1:** À l'aide de LXR, retrouver le nombre maximum de threads, la taille d'une page et la taille de la pile noyau d'un thread. **Indication** : `max_threads, PAGE_SIZE, THREAD_SIZE`

▷ **Question 2:** Comment peut-t-on modifier le nombre maximum de threads à l'aide du procfs ? Est-ce-possible de le faire sur votre machine ? Pourquoi ? **Indication** :/proc/sys/kernel/

▷ **Question 3:** En utilisant le procfs, déterminer le nombre maximum de fichiers que peut ouvrir un processus ?

▷ **Question 4:** Retrouver la structure de données qui représente le descripteur d'un processus (task_struct). Repérer les champs qui correspondent aux information suivantes : état, pid, tgid et la pile.

▷ **Question 5:** Retrouver la structure thread_union qui permet de stocker la *thread_info* dans la même zone mémoire avec la pile **noyau** du processus. Quelle est la taille de cette zone mémoire ?

★ Exercice 4: Priorités de processus

Sous Linux, du point de vue utilisateur, les priorités (*nice*) vont de -20 à 19, mais seul le super-utilisateur peut spécifier des priorités négatives. Plus le nombre est grand, plus la priorité est faible.

▷ **Question 1:** Comment, à l'aide de `ps`, on peut obtenir des informations sur la priorité des processus en cours ?

Ces informations sur la priorité peuvent être retrouvées aussi avec `top` en temps réel.

▷ **Question 2:** Retrouver les champs PRI et NI dans le procfs.

Pour changer la priorité d'un processus, on dispose de la commande `nice` pour modifier la priorité d'un processus à son exécution.

▷ **Question 3:** Taper la commande : `nice -5 ps -l`. Que remarquez-vous au niveau de la colonne NI ?

▷ **Question 4:** Essayer par exemple :

```
sleep 240 &
nice sleep 240 &
ps -l
```

Quelle est la différence par rapport à la question précédente ? Conclure !

Pour changer la priorité d'un processus au cours de son exécution, il y a la commande `renice`.

▷ **Question 5:** Donner un exemple sur le modèle de la question précédente, pour montrer l'utilisation de renice.

★ Exercice 5: Mémoire virtuelle d'un processus

Récupérer le fichier ***/home/depot/2A/RSA-Maimour/memVirtuelle.c***.

▷ **Question 1:** Compiler le programme. Appliquer la commande `size` sur l'exécutable résultant et interpréter les résultats affichés.

▷ **Question 2:** Exécuter le programme résultant. Pendant que le programme est en veille, afficher dans une autre fenêtre, la mémoire virtuelle du processus à l'aide de `cat /proc/<pid>/maps`. Analyser les différentes régions et faire la correspondance avec les différentes façons de déclarer les données.