Systèmes d'Exploitation

L3 Informatique & L3 MI-Info

TP 2 - Manipulation de threads

Léa Brunschwig <lea.brunschwig@univ-pau.fr>

Le but de ce TP est de se familiariser avec l'utilisation des appels systèmes de base pour la manipulation des processus légers.

On comparera les résultats entre :

- un programme employant deux processus lourds,
- un programme employant deux threads noyau,
- un programme employant deux threads utilisateur.

Contexte générale

Les processus, threads noyau et threads utilisateur créés vont tous exécuter la fonction `count`. Cette fonction va, dans tous les cas, incrémenter un milliard de fois la variable globale `a`, définie et initialisée à 0 avant la fonction `main`.

```
#define MAX 1000000000

void *count(void *arg) {
    volatile unsigned long long*var = (unsigned long
                                        long*) arg;
    volatile unsigned long long i;
    for (i = 0; i < MAX; i++)
        *var = *var + 1;
    return NULL;
}
```

Deux processus

1. Écrire un code qui crée deux processus et qui utilise la fonction `count` sur `a`. Afficher le `pid` des processus et la valeur de `a`.

Deux threads noyau

1. Écrire un code qui crée deux threads noyau et qui utilise la fonction `count` sur `a`. Afficher le `pid` des processus et la valeur de `a`. Vous utiliserez la librairie `pthread`.

Deux threads utilisateur

1. Écrire un code qui crée deux threads noyau et qui utilise la fonction `count` sur `a`. Afficher le `pid` des processus et la valeur de `a`. Vous utiliserez la librairie `pth`.
2. Corrigez le code précédent en utilisant `pth_sleep`.
3. Modifier le code précédent en utilisant `sleep`.