

Systèmes d'exploitation : Devoir 3

GROUPE 3

Thierno Sambegou DIALLO

Nathan LESNE

Décembre 2019

1 Bilan

Cette partie visait à implémenter l'utilisation de la mémoire virtuelle, et la gestion de processus utilisateurs.

Nous avons implémenté la mémoire virtuelle, avec `ReadAtVirtual` et `PageProvider`. Cette partie fonctionne correctement.

La partie de gestion de processus a été implémentée, néanmoins, il existe des bugs critiques à la terminaison, décrits dans la partie Limitations.

Pour ce qui est des choses non implémentées:

- Les verrous pris par les threads ne sont pas libérés à leur terminaison.
- Nous n'avons pas démontré que les processus libèrent bien leur ressources, à cause des bugs à la terminaison.
- Nous n'avons pas implémenté le bonus shell.

Enfin, nous avons étudiés et corrigés les problèmes présents dans le retour sur la Partie 2 (`snprintf`, `testAndResize`, le nombre de Threads dans `AddrSpace` et l'utilisation d'`AllocateUserStack` pour le thread principal).

2 Points délicats

Le point le plus délicat a été la désallocation des ressources.

En effet, il semblerait qu'il existe des race conditions quand on supprime un `AddrSpace` au même moment qu'un Thread est créé.

Nous nous sommes rendus compte de ce problème trop tard.

La deuxième soucis est qu'à l'appel de `ThreadExit`, celui-ci a été victime d'un débordement, ou d'un `delete`, et on ne peut pas libérer ses ressources.

Ce point délicat n'a donc pas été résolu.

3 Limitations

Il existe 2 limitations critiques sur cette partie.

On ne peut pas libérer l'AddrSpace quand on appelle Exit sur un processus, sinon le programme SEGFAULT.

Quand on appelle ThreadExit, le thread peut être corrompu, la désallocation fait alors planter le programme, car sa bitmapLocation est invalide.

4 Tests

Les tests effectués sont les ASSERT et les programmes de tests indiqués dans le sujet, en plus des programmes de tests avec les threads déjà présents dans les parties précédentes.

Nous avons jugés que cela était suffisant.