osj

9 avril 2025

Table des matières

0.1 Introduction

0.2 Création et terminaison de processus

Un processus, appelé le *parent*, peut en créer un autre, appelé le *enfant*. Pour cela, un nouveau PCB (Process Control Block) est alloué et initialisé. Par exemple, en exécutant la commande 'ps auxwww' dans le shell, vous pouvez observer le PID du parent (PPID).

0.2.1 Héritage des attributs du parent par le processus enfant

Dans POSIX, le processus enfant hérite de la plupart des attributs du parent. Cela inclut l'UID, les fichiers ouverts (qui devraient être fermés s'ils ne sont pas nécessaires, pourquoi ?), le répertoire de travail actuel (cwd), etc.

0.2.2 Déplacement du PCB entre différentes files d'attente

Pendant son exécution, le PCB se déplace entre différentes files d'attente, selon le graphique de changement d'état. Ces files d'attente peuvent être : exécutable, sommeil/attente d'un événement i (i=1,2,3...).

0.2.3 Processus zombie

Après qu'un processus meurt (sortie() / interrompu), il devient un zombie. Le parent utilise le syscall wait* pour effacer le zombie du système (pourquoi ?). La famille de syscall wait comprend : wait, waitpid, waitid, wait3, wait4. Par exemple :

0.2.4 Attente ou exécution en parallèle du parent

Le parent peut dormir/attendre que son enfant termine ou s'exécuter en parallèle. Le wait*() bloquera à moins que WNOHANG ne soit donné dans 'options'. Par exemple, vous pouvez lire 'man 2 wait' pour plus d'informations.

```

  \begin{tcolorbox}[    colback=yellow!10,    colframe=yellow,    title={\
    fontfamily{lmr}\selectfont \faBookmark À retenir},    fonttitle=\bfseries,
    fontupper=\fontfamily{lmr}\selectfont,    boxrule=1pt,    sharp corners,
    ]  Un processus parent peut créer un processus enfant, qui hérite de la
    plupart des attributs du parent. Pendant son exécution, le PCB du processus se
     déplace entre différentes files d'attente. Un processus devient un zombie
    après sa mort, et le parent doit utiliser le syscall wait* pour l'effacer du
    système. Le parent peut attendre que son enfant termine ou s'exécuter en
    parallèle.  \end{tcolorbox}

\section{Création d'un processus enfant avec fork()}

```