OS

7 avril 2025

Table des matières

0.1 Introduction

0.2 Création et terminaison des processus

Un processus (le « parent ») peut créer un autre processus (le « enfant »). Pour cela, un nouveau PCB est alloué et initialisé.

> **À retenir**
>
> Exécutez 'ps auxwww' dans le shell ; PPID est le PID du parent.

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, tels que l'UID, les fichiers ouverts (qui devraient être fermés si inutiles ; pourquoi ?), le cwd, etc.

0.2.2 Mouvement du PCB entre différentes files d'attente

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

0.3 Terminaison des processus

Après qu'un processus meurt (exit()s / interrompu), il devient un zombie. Le parent utilise le syscall wait* pour effacer le zombie du système (pourquoi ?).

0.3.1 Famille de syscall wait

La famille de syscall wait comprend : wait, waitpid, waitid, wait3, wait4. Par exemple :

```

\section{Fork - Création d'un processus enfant}
 La fonction fork() est utilisée pour créer un nouveau processus,
    appelé processus enfant, qui est une copie du processus qui l'a
    appelé, le processus parent. Après la création du processus
    enfant, les deux processus, le parent et l'enfant, s'exécutent
    en parallèle.

 \subsection{Initialisation d'un nouveau PCB}
 La fonction fork() initialise un nouveau Bloc de Contrôle de
    Processus (PCB) basé sur la valeur du parent. Le nouveau PCB est
     ensuite ajouté à la file d'attente des processus prêts à être
    exécutés.

 \subsection{Espace d'adressage du processus enfant}
 Le nouvel espace d'adressage du processus enfant est une copie
    complète de l'espace d'adressage du parent, avec une seule
    différence : la valeur de retour de la fonction fork().

 \subsection{Valeur de retour de fork()}
 La fonction fork() retourne deux fois : une fois dans le processus
     parent avec une valeur de pid>0 et une fois dans le processus
    enfant avec une valeur de pid=0.

 \subsection{Ordre d'impression}
 L'ordre d'impression des processus parent et enfant n'est pas
    déterminé et peut varier.

 \subsection{Gestion des erreurs}
 En cas d'échec de la fonction fork(), la variable globale 'errno'
    contient le numéro d'erreur de la dernière syscall.
\begin{lstlisting}[language=C]
int main(int argc, char *argv[])
{
  int pid = fork();
  if( pid==0 ) {
   //
   // child
      //
```

```
27        printf("parent=%d son=%d\n",
28               getppid(), getpid());
29    }
30    else if( pid > 0 ) {
31        //
32        // parent
33        //
34        printf("parent=%d son=%d\n",
35               getpid(), pid);
36    }
37    else { // print string associated
38           // with errno
39        perror("fork() failed");
40    }
41    return 0;
42 }
```

À retenir

La fonction fork() est utilisée pour créer un nouveau processus, qui est une copie du processus parent. Elle retourne deux fois : une fois dans le processus parent avec une valeur de pid>0 et une fois dans le processus enfant avec une valeur de pid=0. En cas d'échec, la variable globale 'errno' contient le numéro d'erreur de la dernière syscall.