Système partie 1

Pastille 4 - destruction

Vincent Colotte - N. de Rugy-Altherre

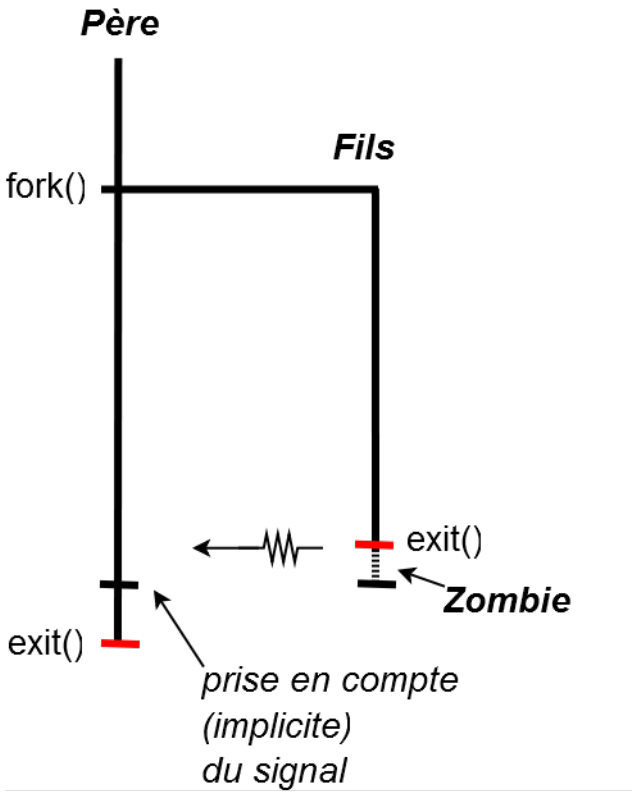
Destruction
Exit

```
#include <stdlib.h>
prototype : void exit(int status);
```

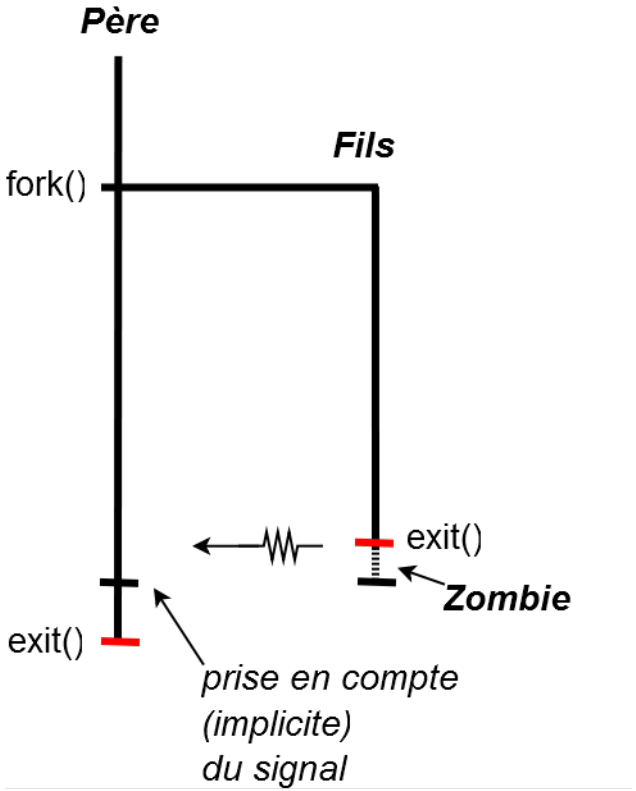
1. Lance l'arrêt de l'exécution du processus.
   - fermeture des fichiers ouverts
   - libération des ressources (verrous, mémoire partagées...)
   - rattachement des fils du processus au processus 1 (*).
2. Transmet un code de retour.
   - Pour une terminaison normale on envoie 0 (ou EXIT_SUCCCESS)
   - Sinon une autre valeur non-nulle (ou EXIT_FAILURE)
3. Passage en mode **Zombie** jusqu'à la prise en compte par son père.
4. Envoi du signal SIGCHLD à son père.

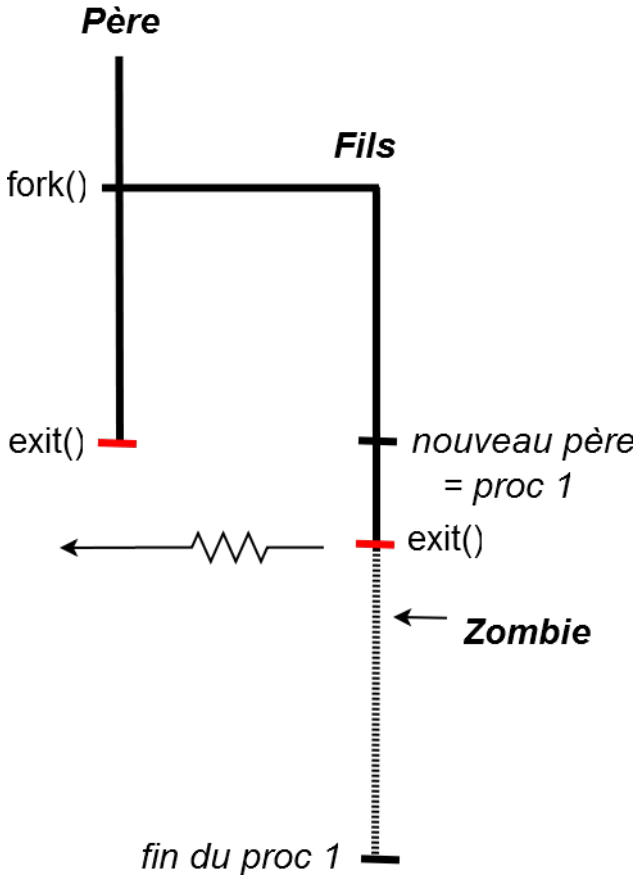
Un processus fils dans l'état **zombie** n'occupe plus la mémoire mais garde une entrée dans la table des processus permettant à son père de récupérer la valeur du code de retour. Une fois la prise en compte de la terminaison du fils par son père, le processus fils est totalement terminé.

Destruction
Exit



Destruction
Exit





Remarque

Si le père meurt avant le fils, le fils est rattaché au processus premier (de *pid* = 1*) de la machine.

Exercice

Exercice

Dans un fichier faites un fork. Dans la partie père, faites une boucle infinie. Dans la partie fils, faites un `sleep(5)` puis un `exit(0)`.

Compilez le programme, puis lancez-le en arrière fond (avec une esperluette à la fin `./prog &`). Juste après le lancement, écrivez dans le terminal `ps -l`. Vous devriez voir le fils et le père. Au bout de 5 secondes, refaites `ps -l`. Vous devriez voir que le père.

Écrivez un programme similaire mais où le `sleep(5); exit(0);` est dans le père et la boucle infinie dans le fils. Faites les deux `ps -l` et observez le ppid (pid du père du processus).

Pour tuer un processus en cours, utilisez la commande `kill` prenant en paramètre le pid du processus.