Info_sw
www.wuolah.com/student/Info_sw

APSO_apuntes_practica3.pdf

APSO. Parte 2: programación

2° Administración y Programación de Sistemas Operativos

Grado en Ingeniería Informática

Escuela Técnica Superior de Ingeniería
UHU - Universidad de Huelva

APSO. PROGRAMACIÓN DEL SISTEMA.

APUNTES PRÁCTICA 3:

LLAMADAS AL SISTEMA PARA LA CREACIÓN DE PROCESOS EN LINUX

********** fork() **********

Sin parámetros, devuelve un entero.

Cuando un proceso cualquiera hace un fork() desde código, se crea un nuevo proceso igual al que hace el fork() y tendrá el mismo código (se genera una copia del proceso original que hace el fork()). Esta copia tendrá su propio identificador (pid) distinto al del proceso original. Como ambos procesos tinen el mismo código, el proceso copia también llegará al fork(). Aquí está la diferencia entre el proceso original y la copia. El fork() del proceso original devolverá el pid asignado al proceso copia (en definitiva, un numero >0) y el fork() del proceso copia devolverá un 0. Para hacer que parte del código lo ejecute uno u otro tendremos que evaluar este entero que devuelve el fork().

```
proceso_original
...
main()
{
  int x;
  ...
  x=fork(); -> devuelve pid asignado al proceso_copia
  if(x==0)
  {
    //hago lo que sea para mi proceso_copia
    //Aquí va todo el código que ejecutará la copia
```

```
  }
  else
  {
    //hago lo que sea para el proceso_original
  }
}

proceso_COPIA (duplica al original, pero el fork() devuelve un 0)
...
main()
{
  int x;
  ...
  x=fork(); -> devuelve un 0
  if(x==0)
  {
    //hago lo que sea para mi proceso_copia
  }
  else
  {
    //hago lo que sea para el proceso_original
  }
}
```

********** execl() **********

Podremos crear procesos con código diferente al que hace el proceso original, pero todo proceso que ejecute execl() muere y desaparece del sistema.
Para que no desaparezca, debemos hacer un fork() y en la parte del cógigo de la copia ejecutamos el execl().

No podemos hablar de proceso padre e hijo hasta que no hagamos un execl(). Con el fork() solo podemos hablar de proceso original y proceos copia.

PARÁMETROS:

execl("param1","param2",NULL)

-> Para que el NULL funcione habrá que incluir la librería #include<unistd.h>

-> param1: ruta al fichero ejecutable que contiene el código que queremos ejecutar*
-> param2: nombre del fichero ejecutable que contiene el código que queremos ejecutar.

*NOTA: nosotros siempre pasaremos como param1 la ruta relativa (que será lo mismo que poner
el nombre del ejecutable)

EL proceso creado con execl() solo hereda las tres primeras posiciones de la tabla de canales del padre. Sin embargo, al hacer un fork() se hereda la tabla de canales COMPLETA.

********** REDIRECCIONAMIENTO **********

¿CÓMO PUEDO MODIFICAR LA TABLA DE CANALES DE UN PROCESO COPIA SIN MODIFICAR LA TABLA DEL ORIGINAL?

-> Habrá que hacerla dentro del if(vpid==0) pero antes de hacer el execl(-,-,-) -> Ver página 4 de los apuntes.

********** MODIFICAR TABLA DE CANALES **********

close(canal) //cierra una entrada de la tabla de canales y la deja vacía.

open(ver_sintaxis) //busca la primera posición libre de la tabla de canales y añade en esa posición de la tabla el fichero que sea (si hago un close(0) previamente, este comando añade en la entrada 0)

creat(ver_sintaxis) //Igual que open pero crea el fichero (no existe previamente)

dup(canal) //