

Benemérita Universidad Autónoma de Puebla
Facultad de Ciencias de la Computación
Sistemas Operativos 1



“Llamas al sistema fork(): Colaborativa”

Gallegos Melchor Angélica

Hernández Fernández Saúl

Muñoz Tehuitzil Jorge Leonel

Pedraza Celón Ian Yael

19/02/2020

En Unix, un proceso es creado mediante la llamada del sistema fork. El proceso que realiza la llamada se denomina proceso padre (parent process) y el proceso creado a partir de la llamada se denomina proceso hijo (child process).

La sintaxis de la llamada efectuada desde el proceso padre es:

valor=fork()

La llamada fork pero devuelve un valor distinto a los procesos padre e hijo: al proceso padre se le devuelve el PID del proceso hijo, y al proceso hijo se le devuelve el valor cero. Las acciones implicadas por la petición de un fork son realizadas por el núcleo (kernel) del S.O. Unix. Tales acciones son las siguientes:

1. asignación de un hueco en la tabla de procesos para el nuevo proceso (hijo).
2. asignación de un identificador único (PID) al proceso hijo.
3. copia de la imagen del proceso padre (con excepción de la memoria compartida).
4. asignación al proceso hijo del estado "preparado para ejecución".
5. dos valores de retorno de la función: al proceso padre se le entrega el PID del proceso hijo, y al proceso hijo se le entrega el valor cero.

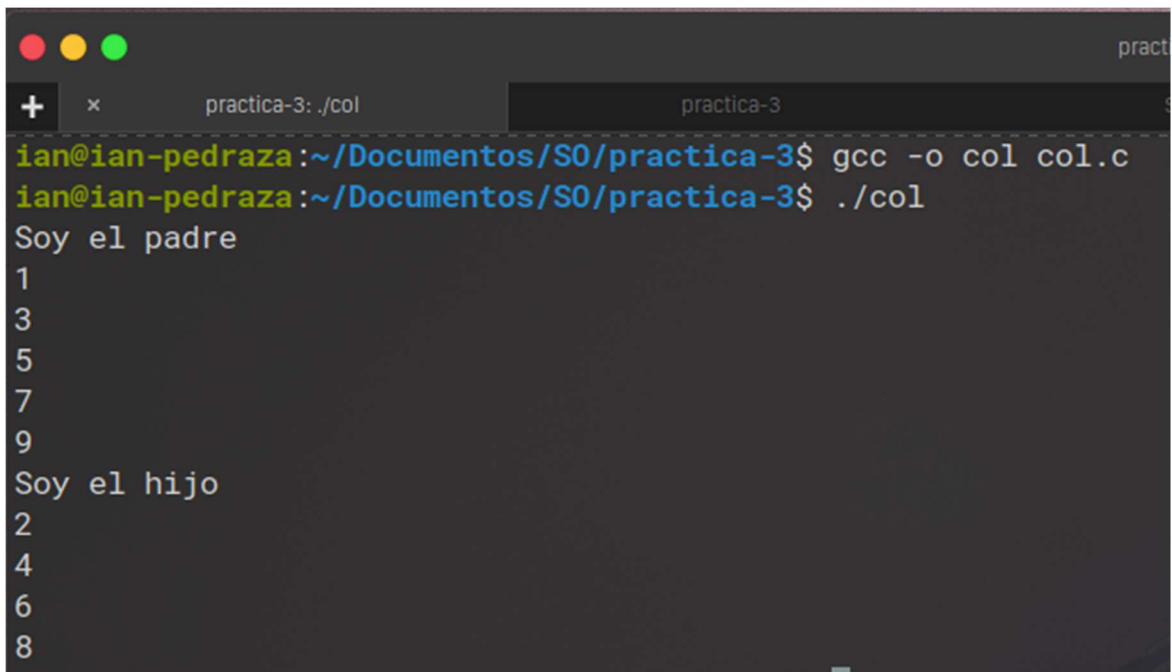
DESARROLLO DE LA PRÁCTICA

Escriba un programa que el proceso padre imprima los números impares y el proceso hijo los números pares.

```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>

int main(){
    pid_t pid;
    pid = fork();
    int p, i, num = 10;

    if(pid == 0) {
        printf("Soy el hijo\n");
        for(p = 2; p < num; p+=2) printf("%d\n", p);
    } else if(pid > 0) {
        printf("Soy el padre\n");
        for(i = 1; i < num; i+=2) printf("%d\n", i);
    } else if(pid == -1) {
        printf("Error al crear el proceso\n");
    }
}
```



```
practica-3: ./col
practica-3
ian@ian-pedraza:~/Documentos/S0/practica-3$ gcc -o col col.c
ian@ian-pedraza:~/Documentos/S0/practica-3$ ./col
Soy el padre
1
3
5
7
9
Soy el hijo
2
4
6
8
```

Conclusión

Un proceso puede generar hijos que heredan una cantidad de atributos de su padre. Es importante donde colocar las condiciones para identificar al padre o al hijo, ya que, si no hacemos esto, dichas instrucciones se ejecutarán en el proceso pesado.