

1.7. Bifurcación o fork

Una bifurcación, referenciada habitualmente por su término en inglés **fork**, es una copia idéntica de un proceso. El proceso original se denomina padre y sus copias, hijos, teniendo todos ellos diferentes identificadores de proceso (PID). La copia creada continua con el estado del proceso original (padre), pero a partir de la creación cada proceso mantiene su propio estado de memoria. En el siguiente código se realiza la creación de un nuevo proceso (proceso hijo) mediante la llamada a la instrucción **fork** de un proceso escrito en lenguaje C y ejecutándose en un sistema Linux.

```
1. #include <stdio.h>
2. #include <unistd.h>
3. #include <sys/types.h>
4. int main(void) {
5.     int contador=0;
6.     printf("Comenzando la ejecución\n");
7.     pid_t idHijo;
8.     pid_t idPadre;
9.     idPadre = getpid();
10.    printf("Antes de bifurcar\n");
11.    contador++;
12.    idHijo = fork();
13.    contador++;
14.    printf("Después de bifurcar\n");
15.    if (idHijo == 0) {
16.        printf("Id. hijo:%ld Id. padre:%ld Contador:%d \n ", (long)getpid(),
17.            (long)idPadre, contador);
18.    } else {
19.        printf("Id. padre:%ld Id. hijo:%ld Contador:%d \n", (long)getpid(),
20.            (long)idPadre, contador);
21.    }
22.    return 0;
23. }
```

La salida generada por la ejecución es la siguiente:

```
1 Comenzando la ejecución
2 Antes de bifurcar
3 Después de bifurcar
4 Después de bifurcar
5 Id. padre:143 Id. hijo:143 Contador:2
6 Id. hijo:144 Id. padre:143 Contador:2
```

Sin entrar en detalles sintácticos sobre el lenguaje C, los aspectos relevantes de la creación de la bifurcación son los siguientes:

- La variable *contador* se declara e inicializa a 0 en la línea 5.
- En la línea 6 se muestra el texto «Comenzando la ejecución» una única vez, ya que en este punto el proceso es único.
- En la línea 10 se muestra el texto «Antes de bifurcar» una única vez, ya que en este punto el proceso es único.
- En la línea 11 el valor de *contador* se incrementa en 1, tomando el valor 1.
- En la línea 12 se crea la bifurcación.

- En la línea 13 el valor de *contador* se incrementa en 1, una vez por cada proceso. No obstante, cada proceso dispone de su propio espacio de memoria por lo que la variable de ambos procesos es distinta.
- En la línea 14 se muestra el texto «Después de bifurcar» dos veces, una por cada copia del proceso.
- En la línea 16 se muestra la información referente al proceso hijo. Se puede observar que el PID es 144 y el valor de la variable *contador* es 2.
- En la línea 18 se muestra la información referente al proceso padre. Se puede observar que el PID es 143 y el valor de la variable *contador* es 2.

Explicación de como generar y ejecutar un programa en lenguaje C para un entorno Linux:

Si no tienes instalado lenguaje C, aquí te muestro una forma sencilla de hacerlo:

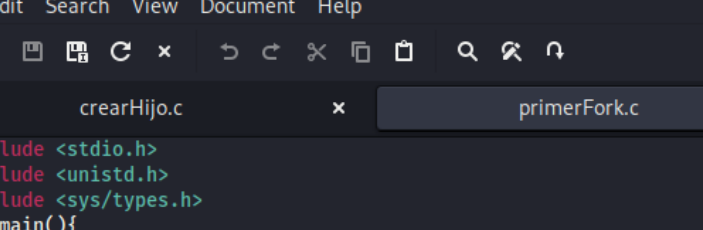
1. Primero asegúrate de actualizar el sistema.

```
(kali㉿kali)-[~/Desktop/lenguajec]
$ sudo apt-get update
[sudo] password for kali:
```

2. Luego instalamos el **GCC**, que es un compilador integrado del proyecto GNU para C, C++, Objective C y Fortran. La sigla GCC significa "GNU Compiler Collection".

```
(kali㉿kali)-[~/Desktop/lenguajec]
$ sudo apt install gcc
```

3. Crea un fichero de texto con el código fuente en C y no te olvides de colocarle al nombre la extensión “.c”. Ejemplo:



```
~/Desktop/lenguajec/primerFork.c - Mousepad
File Edit Search View Document Help

1 #include <stdio.h>
2 #include <unistd.h>
3 #include <sys/types.h>
4 int main(){
5
6     fork(); // Función que crea un proceso nuevo - tenedor
7     printf("Como me gusta la asignatura de procesos y servicios :)\n");
8     // getpid() → devuelve el PID del proceso
9     // getpid() → devuelve el PID de mi padre
10    int miPID = getpid();
11    int papa = getpid();
12    printf("Hola soy %d y mi papa es :) %d\n",miPID,papa);
13    sleep(120);
14    return 0;
15 }
16 |
```

4. Para compilar:

```
kali@kali: ~/Desktop/lenguajec
File Actions Edit View Help

(kali@kali)-[~/Desktop/lenguajec]
$ gcc primerFork.c -o primerFork
```

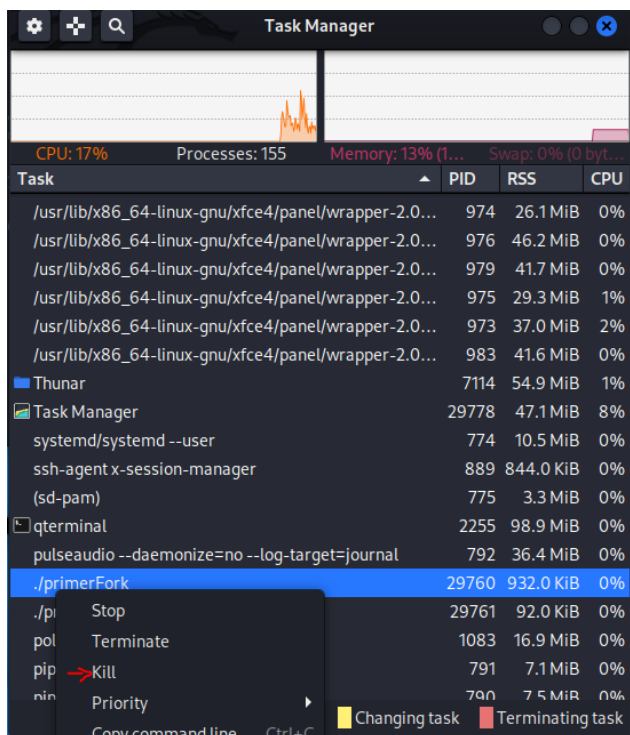
5. Ejecutar:

```
kali@kali: ~/Desktop/lenguajec
File Actions Edit View Help

(kali@kali)-[~/Desktop/lenguajec]
$ gcc primerFork.c -o primerFork

(kali@kali)-[~/Desktop/lenguajec]
$ ./primerFork
Como me gusta la asignatura de procesos y servicios :)
Hola soy 27573 y mi papa es :) 2262
Como me gusta la asignatura de procesos y servicios :)
Hola soy 27574 y mi papa es :) 27573
```

6. Vamos a ver los forks en la cola de tareas: y vamos a eliminarlos.



Que sucede aquí?