

## Examinación de la memoria

Se ejecutó el programa con `./donde_en_la_memoria`, resultando el PID 18800.

Para que el core incluyera el bloque de memoria correspondiente al texto, se modificó el archivo `/proc/18800/coredump_filter` (de 0x33 a 0x37). Con `gcore -o donde_en_la_memoria 18800` se generó el core `donde_en_la_memoria.18800`.

Para analizarlo, se utilizó `objdump -s donde_en_la_memoria.18800` para ver la memoria de cada sección, y `objdump -d donde_en_la_memoria` para ver el texto ensamblado con su respectivo opcode. Además, generando `a.out` con `gcc -g donde_en_la_memoria.c`, se utilizó `gdb a.out donde_en_la_memoria.18800` para la depuración.

En el archivo `donde_en_la_memoria.s` puede leerse el código generado por el ensamblador, comentado.

Los archivos utilizados se encuentran en el directorio `archivos`.

### Mapa de la memoria del proceso

Una ejecución del programa `proyecto3.py` dando como parámetro el PID 18800, nos da el siguiente resultado:

```
File Edit View Search Terminal Help
roel@pc-rpc:~/Documents/SisOp/sistop-2020-2/proyectos/3/PerezRoel$ python3 proyecto3.py 18800
VACIO 00000000-5583fbc6f 85.5 TB 22955408495 VACIO
texto 5583fbc6f-5583fbc70 4 KB 1 r-x /home/roel/Documents/SisOp/sistop-2020-2/proyectos/3/PerezRoel/donde_en_la_memoria
VACIO 5583fbc70-5583fbc70 2.0 MB 512 VACIO
datos 5583fbc70-5583fbc71 4 KB 1 r-- /home/roel/Documents/SisOp/sistop-2020-2/proyectos/3/PerezRoel/donde_en_la_memoria
datos 5583fbc71-5583fbc72 4 KB 1 rw- /home/roel/Documents/SisOp/sistop-2020-2/proyectos/3/PerezRoel/donde_en_la_memoria
VACIO 5583fbc72-5583fbc74 29.0 MB 7426 VACIO
Heap 5583fbc74-5583fbc95 132 KB 33 rw- [heap]
VACIO 5583fbc95-7f39a9717 41.7 TB 11196349314 VACIO
Btexto 7f39a9717-7f39a98fe 1.9 MB 487 r-x /lib/x86_64-linux-gnu/libc-2.27.so
7f39a98fe-7f39a9afe 2.0 MB 512 --- /lib/x86_64-linux-gnu/libc-2.27.so
Bdatos 7f39a9afe-7f39a9b02 16 KB 4 r-- /lib/x86_64-linux-gnu/libc-2.27.so
Bdatos 7f39a9b02-7f39a9b04 8 KB 2 rw- /lib/x86_64-linux-gnu/libc-2.27.so
[anon] 7f39a9b04-7f39a9b08 16 KB 4 rw- [anon]
Btexto 7f39a9b08-7f39a9b2f 156 KB 39 r-x /lib/x86_64-linux-gnu/ld-2.27.so
VACIO 7f39a9b2f-7f39a9d14 1.9 MB 485 VACIO
[anon] 7f39a9d14-7f39a9d16 8 KB 2 rw- [anon]
VACIO 7f39a9d16-7f39a9d2f 100 KB 25 VACIO
Bdatos 7f39a9d2f-7f39a9d30 4 KB 1 r-- /lib/x86_64-linux-gnu/ld-2.27.so
Bdatos 7f39a9d30-7f39a9d31 4 KB 1 rw- /lib/x86_64-linux-gnu/ld-2.27.so
[anon] 7f39a9d31-7f39a9d32 4 KB 1 rw- [anon]
VACIO 7f39a9d32-7ffc9df4a 779.8 GB 204423704 VACIO
Stack 7ffc9df4a-7ffc9df6b 132 KB 33 rw- [stack]
VACIO 7ffc9df6b-7ffc9df7b 64 KB 16 VACIO
[vvar] 7ffc9df7b-7ffc9df7e 12 KB 3 r-- [vvar]
[vdso] 7ffc9df7e-7ffc9df80 8 KB 2 r-x [vdso]
VACIO 7ffc9df80-fffffffff600 16.0 EB 4503565271176832 VACIO
[vsyscall] ffffffff600-fffffffff601 4 KB 1 r-x [vsyscall]
roel@pc-rpc:~/Documents/SisOp/sistop-2020-2/proyectos/3/PerezRoel$
```

Versión no final del programa

Los bloques de memoria que nos importan son:

5583fbc6f000-5583fbc70000. Texto

5583fbc70000-5583fbc71000. Datos (Lectura)

5583fbc71000-5583fbc72000. Datos (Lectura y escritura)

5583fbc72000-5583fbc95000. Heap

7ffc9df4a000-7ffc9df6b000. Stack

## Variables globales

Las variables globales `cadena1`, `cadena_total`, y `tamano` se encuentran en la sección de datos de lectura y escritura:

```
File Edit View Search Terminal Help
Contents of section .data:
5583fbe71000 00000000 00000000 0810e7fb 83550000 .....U..
5583fbe71010 596f2073 6f6c6f20 73c3a920 71756520 Yo solo s.. que
5583fbe71020 6e6f2073 c3a9206e 61646100 1e000000 no s.. nada.....
5583fbe71030 002ab0a9 397f0000 00000000 00000000 .*..9.....
5583fbe71040 d046b7fd 83550000 00000000 00000000 .F...U.....
5583fbe71050 00000000 00000000 00000000 00000000
```

En el segundo y tercer renglón podemos ver el arreglo `cadena1`: “Yo solo sé que no sé nada”. En la última parte del tercer renglón vemos a `tamano` ( $0x1e = 30$ ). El quinto renglón tiene al apuntador `cadena_total` (`5583fdb746d0`), apuntando a una dirección de memoria en el heap.

`cadena_total`:

```
File Edit View Search Terminal Help
5583fdb746d0 596f2073 6f6c6f20 73c3a920 71756520 Yo solo s.. que
5583fdb746e0 6e6f2073 c3a9206e 6164610a 596f2073 no s.. nada.Yo s
5583fdb746f0 c3b36c6f 2073c3a9 20717565 206e6164 ..lo s.. que nad
5583fdb74700 612073c3 a90a5065 726f2073 6920616c a s...Pero si al
5583fdb74710 67756965 6e207361 6265206d 656e6f73 quien sabe menos
5583fdb74720 0ac2a173 69656d70 72652070 75656465 ...siempre puede
5583fdb74730 20736572 20757374 6564210a 00000000 ser usted!.....
5583fdb74740 00000000 00000000 11040000 00000000 .....
```

Las variables globales se obtienen desde el texto por medio de desplazamientos relativos a `rip` (el apuntador a la siguiente instrucción).

## Variables locales

Las variables locales de cada función se encuentran en el stack. Ya que la función `construye_final` se terminó de ejecutar antes de hacer el volcado de memoria, no es posible ver sus variables locales, pero su resultado queda almacenado en el apuntador `cadena_total`.

```
File Edit View Search Terminal Help
#3 0x00005583fbc6fa33 in main () at donde_en_la_memoria.c:31
    cadena2 = 0x7ffc9df68cf0 "Yo sólo sé que nada sé"
---Type <return> to continue, or q <return> to quit---
    cadena3 = 0x5583fdb74260 "Pero si alguien sabe menos"
    cadena4 = 0x5583fdb746a0 "¡siempre puede ser usted!"
(gdb) █
```

Variables locales de main

Como se observa en `donde_en_la_memoria.s`, las variables locales de `main` están en el siguiente orden (en la dirección hacia la que crece la pila): `cadena4` (`5583fdb746a0`), `cadena2` (`7ffc9df68cf0`), `cadena3` (`5583fdb74260`). El texto accede a estas variables a través de un desplazamiento relativo a `rbp` (el apuntador a la base del stack frame).

```
File Edit View Search Terminal Help
(gdb) print $rsp
$14 = (void *) 0x7ffc9df68bf0
(gdb) print $rbp
$15 = (void *) 0x7ffc9df68c10
(gdb)
```

Registros `rbp` y `rsp` (límite inferior y superior del stack frame)

```
File Edit View Search Terminal Help
7ffc9df68bf0 50fbc6fb 83550000 6042b7fd 83550000 P....U..`B...U..
7ffc9df68c00 f08cf69d fc7f0000 a046b7fd 83550000 .....F...U..
7ffc9df68c10 50fbc6fb 83550000 978b73a0 307f0000 D..H..e.g
```

Variables en el stack frame.

La `cadena3` se encuentra en el heap porque se utilizó la función `malloc` para asignarle memoria. De igual forma, a la `cadena4` se le asignó el valor de retorno de `construye_final`, el cuál devuelve el apuntador completa, siendo este asignado por la función `malloc`. El único diferente es `cadena2`, el cual tiene un apuntador a un área más alta de memoria en el stack. Esto se debe a que este apuntador no se inicializó con `malloc`, y simplemente tomó el valor que había en el stack. En este caso, apuntaba a otra dirección en el stack. Se muestran las direcciones de los apuntadores a continuación:

`cadena4`:

```
File Edit View Search Terminal Help
5583fdb746a0 00000000 00000000 31000000 00000000 .....1.....
5583fdb746a0 c2a17369 656d7072 65207075 65646520 ..siempre puede
5583fdb746b0 73657220 75737465 64210000 00000000 ser usted!.....
5583fdb746c0 00000000 00000000 31000000 00000000 .....1.....
```

`cadena2`:

```
File Edit View Search Terminal Help
7ffc9df68cf0 e08cf69d fc7f0000 10000000 00000000 .....1.....
7ffc9df68cf0 596f2073 c3b36c6f 2073c3a9 20717565 Yo s..lo s.. que
7ffc9df68d00 206e6164 612073c3 a9000000 00000000 nada s.....
7ffc9df68d10 04a8f69d fc7f0000 41a8f69d fc7f0000 A
```

`cadena3`:

```
File Edit View Search Terminal Help
5583fdb74260 00000000 00000000 31000000 00000000 .....1.....
5583fdb74260 5065726f 20736920 616c6775 69656e20 Pero si alguien
5583fdb74270 73616265 206d656e 6f730000 00000000 sabe menos.....
5583fdb74280 00000000 00000000 31000000 00000000 .....1.....
```

## Cadenas de solo lectura

Si observamos `donde_en_la_memoria.s`, podemos ver que existe una tabla de símbolos asociados a cadenas. Esta se compone de las cadenas: “Proceso filósofo, PID %d\n\n”, “Yo sólo sé que nada sé”, “Pero si alguien sabe menos”, “%s\n%s\n%s\n%s\n” y “puede ser”. Éstas son dadas como parámetros a funciones que requieren de un apuntador a cadena (`printf`, `strncpy`, `sprintf`, o en asignaciones a apuntadores de cadenas). La forma en que se maneja es que se almacenan en memoria de solo lectura. En este caso, se encuentran dentro de la misma página que el texto. Cada vez que se necesite referir a una, de forma similar a las variables globales, se aplica un desplazamiento relativo a `rip` para obtener su dirección

Por otra parte, ya que las cadenas “¡siempre “ y “usted!” de `construye_final` son simplemente asignadas a un bloque fijo de memoria en el stack, estas se encuentran dentro del mismo código del programa. Como se puede apreciar más claramente en `donde_en_la_memoria.s`, simplemente se cargan a la dirección de memoria de las variables.

```
File Edit View Search Terminal Help
(gdb) print main
$6 = {int ()} 0x5583fbc6f93a <main>
(gdb) print construye_final
$7 = {char *} 0x5583fbc6fa61 <construye_final>
(gdb)
```

Direcciones de las funciones `main` y `construye_final`.

```
File Edit View Search Terminal Help
5583fbc6f930 554889e5 5de966ff ffff5548 89e54883 UH..].f...UH..H.
5583fbc6f940 ec208b05 e4162000 48984889 c7e8aeef . .... .H.H....
5583fbc6f950 ffff4889 45e8e855 feffff89 c6488d3d ..H.E..U....H.=
5583fbc6f960 70020000 b8000000 00e862fe ffff8b05 p.....b....
5583fbc6f970 b8162000 4863d048 8b45f048 8d356e02 .. .Hc.H.E.H.5n.
5583fbc6f980 00004889 c7e806fe ffff8b05 9c162000 ..H.....
5583fbc6f990 4863d048 8b45e848 8d356c02 00004889 Hc.H.E.H.5l...H.
5583fbc6f9a0 c7e8eafd ffff8b00 000000e8 b1000000 .....
5583fbc6f9b0 488945f8 8b057216 2000c1e0 02489848 H.E...r. ....H.H
5583fbc6f9c0 89c7e839 feffff48 89057216 20008b05 ..9...H..r. ...
5583fbc6f9d0 58162000 c1e00248 63f0488b 055f1620 X. ....Hc.H...
5583fbc6f9e0 00488b4d e8488b55 f04883ec 08ff75f8 .H.M.H.U.H...u.
5583fbc6f9f0 4989c949 89d0488d 0d131620 00488d15 I..I..H.... .H..
5583fbc6fa00 21020000 4889c7b8 00000000 e8cfdfff !...H.....
5583fbc6fa10 ff4883c4 10488b05 24162000 4889c7e8 .H...H..$. .H...
5583fbc6fa20 7cfdffff 488b0505 16200048 89c7e8dd |...H.... .H....
5583fbc6fa30 fdffff48 8b45e848 89c7e841 fdffff48 ...H.E.H...A...H
5583fbc6fa40 8b45f848 89c7e835 fdffff48 8b05ee15 .E.H...5...H....
5583fbc6fa50 20004889 c7e826fd ffff8b00 000000c9 .H...&.....
5583fbc6fa60 c3554889 e54883ec 5064488b 04252800 .UH..H..PdH..%(.
5583fbc6fa70 00004889 45f831c0 48b8c2a1 7369656d ..H.E.1.H...siem
5583fbc6fa80 7072ba65 20000048 8945c048 8955c8c7 pr.e ..H.E.H.U..
5583fbc6fa90 45d00000 000048b8 20757374 65642100 E....H. usted!.
5583fbc6faa0 ba000000 00488945 e0488955 e8c745f0 ....H.E.H.U..E.
5583fbc6fab0 00000000 488d0577 01000048 8945b08b ....H..w...H.E..
5583fbc6fac0 05671520 00489848 89c7e831 fdffff48 .g. .H.H...1...H
5583fbc6fad0 8945b88b 05531520 004863d0 488d4dc0 .E...S. .Hc.H.M.
5583fbc6fae0 488b45b8 4889ce48 89c7e801 fdffff8b H.E.H..H.....
```

Código de `main` y `construye_final`. Nótese las cadenas “¡siempre” y “usted!”.

```
File Edit View Search Terminal Help
5583fbc6fb10 004863d0 488d4de0 488b45b8 4889ce48 .Hc.H.M.H.E.H..H
5583fbc6fb20 89c7e8c9 fcffff48 8b45b848 8b75f864 .....H.E.H.u.d
5583fbc6fb30 48333425 28000000 7405e881 fcffffc9 H34%(...t.....
5583fbc6fb40 c3662e0f 1f840000 0000000f 1f440000 .f.....D..
5583fbc6fb50 41574156 4989d741 5541544c 8d250e12 AWAVI..AUATL.%..
5583fbc6fb60 20005548 8d2d0e12 20005341 89fd4989 .UH.-.. .SA..I.
5583fbc6fb70 f64c29e5 4883ec08 48c1fd03 e8cffbff .L).H...H.....
5583fbc6fb80 ff4885ed 742031db 0f1f8400 00000000 .H..t 1.....
5583fbc6fb90 4c89fa4c 89f64489 ef41ff14 dc4883c3 L..L..D..A...H..
5583fbc6fba0 014839dd 75ea4883 c4085b5d 415c415d .H9.u.H...[]A\A]
5583fbc6fbb0 415e415f c390662e 0f1f8400 00000000 A^A_..f.....
5583fbc6fbc0 f3c30000 4883ec08 4883c408 c3000000 ....H...H.....
5583fbc6fbd0 01000200 50726f63 65736f20 66696cc3 ....Proceso fil.
5583fbc6fbe0 b3736f66 6f2c2050 49442025 640a0a00 .sofo, PID %d...
5583fbc6fbf0 596f2073 c3b36c6f 2073c3a9 20717565 Yo s..lo s.. que
5583fbc6fc00 206e6164 612073c3 a9005065 726f2073 nada s...Pero s
5583fbc6fc10 6920616c 67756965 6e207361 6265206d i alguien sabe m
5583fbc6fc20 656e6f73 0025730a 25730a25 730a2573 enos.%s.%s.%s.%s
5583fbc6fc30 0a007075 65646520 73657200 011b033b ..puede ser....;
5583fbc6fc40 40000000 07000000 34fbffff 8c000000 @.....4.....
5583fbc6fc50 e4fbffff b4000000 f4fbffff 5c000000 .....\.
5583fbc6fc60 fefcffff cc000000 25feffff ec000000 .....%.
5583fbc6fc70 14ffffff 0c010000 84ffffff 54010000 .....T...
5583fbc6fc80 14000000 00000000 017a5200 01781001 .....zR..X..
5583fbc6fc90 1b0c0708 90010710 14000000 1c000000 .....
5583fbc6fca0 90fbffff 2b000000 00000000 00000000 ....+.....
5583fbc6fcb0 14000000 00000000 017a5200 01781001 .....zR..X..
5583fbc6fcc0 1b0c0708 90010000 24000000 1c000000 .....$.

```

Cadenas de solo lectura.



## Ejemplos.

Con el comando `objdump -d donde_en_la_memoria`, se muestra cada instrucción y su correspondiente opcode. Podemos compararlo con su contraparte cargada a memoria.

### Ejemplo 1.

```
File Edit View Search Terminal Help
93a: 55          push    %rbp
93b: 48 89 e5    mov     %rsp,%rbp
93e: 48 83 ec 20  sub     $0x20,%rsp
942: 8b 05 e4 16 20 00  mov     0x2016e4(%rip),%eax    # 20202c <tamano>
948: 48 98      cltq
94a: 48 89 c7    mov     %rax,%rdi
```

```
5583fbc6f930 554889e5 5de966ff ffff5548 89e54883
5583fbc6f940 ec208b05 e4162000 48984889 c7e8ae fe
5583fbc6f950 ffff4889 45e8e855 feffff89 c6488d3d
5583fbc6f960 70020000 b8000000 00e862fe ffff8b05
5583fbc6f970 b8162000 4863d048 8b45f048 8d356e02
```

En los valores resaltados, tenemos un desplazamiento de `0x2016e4`, con un `rip` de `0x5583fbc6f948`. El cálculo se muestra a continuación:

#### Hexadecimal Calculation—Add, Subtract, Multiply, or Divide

##### Result

Hex value:

`5583fbc6f948 + 2016e4 = 5583FBE7102C`

Revisando la dirección obtenida, comprobamos que corresponde a `tamano`.

```
5583fbe71010 596f2073 6f6c6f20 73c3a920 71756520  Yo solo s.. que
5583fbe71020 6e6f2073 c3a9206e 61646100 1e000000  no s.. nada.....
5583fbe71030 002ab0a9 397f0000 00000000 00000000  .*..9.....
```

### Ejemplo 2.

```
File Edit View Search Terminal Help
952: 48 89 45 e8    mov     %rax,-0x18(%rbp)
956: e8 55 fe ff ff  callq   7b0 <getpid@plt>
95b: 89 c6          mov     %eax,%esi
95d: 48 8d 3d 70 02 00 00  lea     0x270(%rip),%rdi    # bd4 <_IO_stdin_used+0x4>
964: b8 00 00 00 00  mov     $0x0,%eax
969: e8 62 fe ff ff  callq   7d0 <printf@plt>
```

```
5583fbc6f930 554889e5 5de966ff ffff5548 89e54883
5583fbc6f940 ec208b05 e4162000 48984889 c7e8ae fe
5583fbc6f950 ffff4889 45e8e855 feffff89 c6488d3d
5583fbc6f960 70020000 b8000000 00e862fe ffff8b05
5583fbc6f970 b8162000 4863d048 8b45f048 8d356e02
```

De forma similar al ejemplo anterior, encontramos un desplazamiento `0x270` con un `rip` de `0x5583fbc6f964`.

### Hexadecimal Calculation—Add, Subtract, Multiply, or Divide

#### Result

Hex value:

5583fbc6f964 + 270 = **5583FBC6FBD4**

Revisando esta dirección, observamos que corresponde a la cadena “Proceso filósofo, PID %d\n\n”.

```
5583fbc6fbd0 01000200 50726f63 65736f20 66696cc3 ....Proceso fil.
5583fbc6fbe0 b3736f66 6f2c2050 49442025 640a0a00 .sofo, PID %d...
```

### Ejemplo 3.

```
File Edit View Search Terminal Help
99e: 48 89 c7      mov    %rax,%rdi
9a1: e8 ea fd ff ff  callq  790 <strncpy@plt>
9a6: b8 00 00 00 00  mov    $0x0,%eax
9ab: e8 b1 00 00 00  callq  a61 <construye_final>
9b0: 48 89 45 f8      mov    %rax,-0x8(%rbp)
9b4: 8b 05 72 16 20 00  mov    0x201672(%rip),%eax # 20202c <tamano>
```

```
5583fbc6f980 00004889 c7e806fe ffff8b05 9c162000
5583fbc6f990 4863d048 8b45e848 8d356c02 00004889
5583fbc6f9a0 c7e8eafd ffff8b00 000000e8 b1000000
5583fbc6f9b0 488945f8 8b057216 2000c1e0 02489848
5583fbc6f9c0 89c7e839 feffff48 89057216 20008b05
```

En este caso tenemos un desplazamiento de 0xb1 con un rip de 0x5583fbc6f9b0.

### Hexadecimal Calculation—Add, Subtract, Multiply, or Divide

#### Result

Hex value:

5583fbc6f9b0 + b1 = **5583FBC6FA61**

Efectivamente, comprobamos que la dirección obtenida corresponde a la dirección de `construye_final`.

```
(gdb) print construye_final
$7 = {char *} 0x5583fbc6fa61 <construye_final>
(gdb)
```

Se trata de la instrucción `call construye_final`.

### Ejemplo 4.

Las llamadas a funciones de biblioteca son un poco diferentes. A continuación, se muestra la instrucción que llama a `malloc`.

```
File Edit View Search Terminal Help
948: 48 98      cltq
94a: 48 89 c7      mov    %rax,%rdi
94d: e8 ae fe ff ff  callq  800 <malloc@plt>
952: 48 89 45 e8      mov    %rax,-0x18(%rbp)
956: e8 55 fe ff ff  callq  7b0 <getpid@plt>
```

File	Edit	View	Search	Terminal	Help
5583fbc6f930	554889e5	5de966ff	ffff5548	89e54883	
5583fbc6f940	ec208b05	e4162000	48984889	c7e8ae	fe
5583fbc6f950	ffff4889	45e8e855	ffffff89	c6488d3d	
5583fbc6f960	70020000	b8000000	00e862fe	ffff8b05	

Tiene un desplazamiento de 0xfffffeae con un rip de 0x5583fbc6f952.

#### Hexadecimal Calculation—Add, Subtract, Multiply, or Divide

Result

Hex value:

5583fbc6f952 + fffffeae = **5584FBC6F800**

File	Edit	View	Search	Terminal	Help
Non-debugging symbols:					
0x00005583fbc6f750	_init				
0x00005583fbc6f780	free@plt				
0x00005583fbc6f790	strncpy@plt				
0x00005583fbc6f7a0	puts@plt				
0x00005583fbc6f7b0	getpid@plt				
0x00005583fbc6f7c0	__stack_chk_fail@plt				
0x00005583fbc6f7d0	printf@plt				
0x00005583fbc6f7e0	snprintf@plt				
0x00005583fbc6f7f0	strncat@plt				
---Type <return> to continue, or q <return> to quit---					
0x00005583fbc6f800	malloc@plt				
0x00005583fbc6f810	IO getc@plt				

La dirección resultante, como podemos comprobar con gdb, corresponde a malloc en una tabla en la página de texto.

File	Edit	View	Search	Terminal	Help
5583fbc6f7e0	ff25d217	20006806	000000e9	80ffffff	
5583fbc6f7f0	ff25ca17	20006807	000000e9	70ffffff	
5583fbc6f800	ff25c217	20006808	000000e9	60ffffff	
5583fbc6f810	ff25ba17	20006809	000000e9	50ffffff	
5583fbc6f820	ff25d217	20006690	00000000	00000000	

File	Edit	View	Search	Terminal	Help
0000000000000800 <malloc@plt>:					
800:	ff 25 c2 17 20 00	jmpq	*0x2017c2(%rip)	# 201fc8 <malloc@GLIBC_2.2.5>	
806:	68 08 00 00 00	pushq	\$0x8		
80b:	e9 60 ff ff ff	jmpq	770 <.plt>		

La primera instrucción es un salto indirecto. Primero se hace un salto relativo de 0x2017c2 al rip de 0x5583fbc6f806.

#### Hexadecimal Calculation—Add, Subtract, Multiply, or Divide

Result

Hex value:

5583fbc6f806 + 2017c2 = **5583FBE70FC8**



Esto nos lleva a una dirección en la sección de datos de sólo lectura, la cual contiene una dirección absoluta.

File	Edit	View	Search	Terminal	Help
5583fbe70fa0	e0c87fa9	397f0000	80bc84a9	397f0000	
5583fbe70fb0	80be77a9	397f0000	50bf77a9	397f0000	
5583fbe70fc0	200e7da9	397f0000	70e07aa9	397f0000	
5583fbe70fd0	90ed79a9	397f0000	00000000	00000000	
5583fbe70fe0	b08a73a9	397f0000	00000000	00000000	

La dirección 0x7f39a97ae070 se encuentra en una biblioteca de texto, en donde se encuentra malloc.

Btexto	7f39a9717-7f39a98fe	1.9 MB	487	r-x	/lib/x86_64-linux-gnu/libc-2.27.so
--------	---------------------	--------	-----	-----	------------------------------------