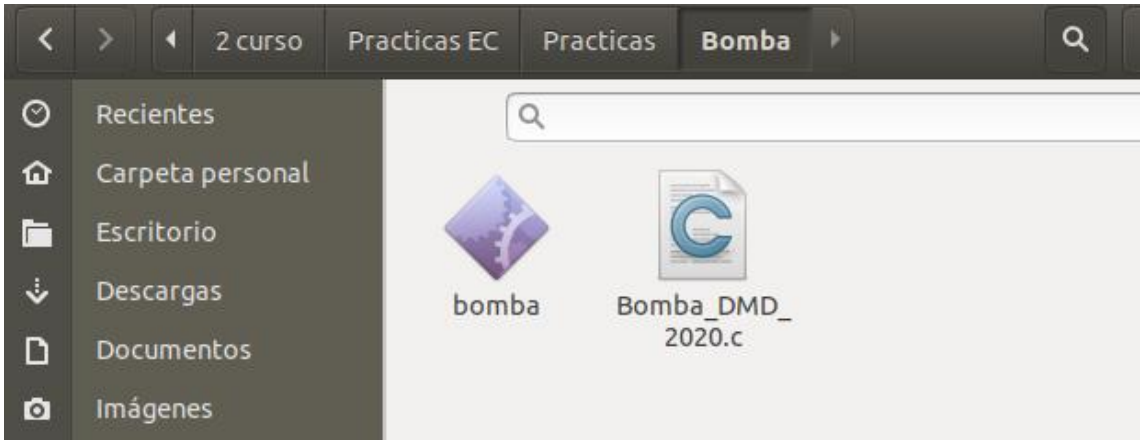
Bomba Digital

David Martínez Díaz

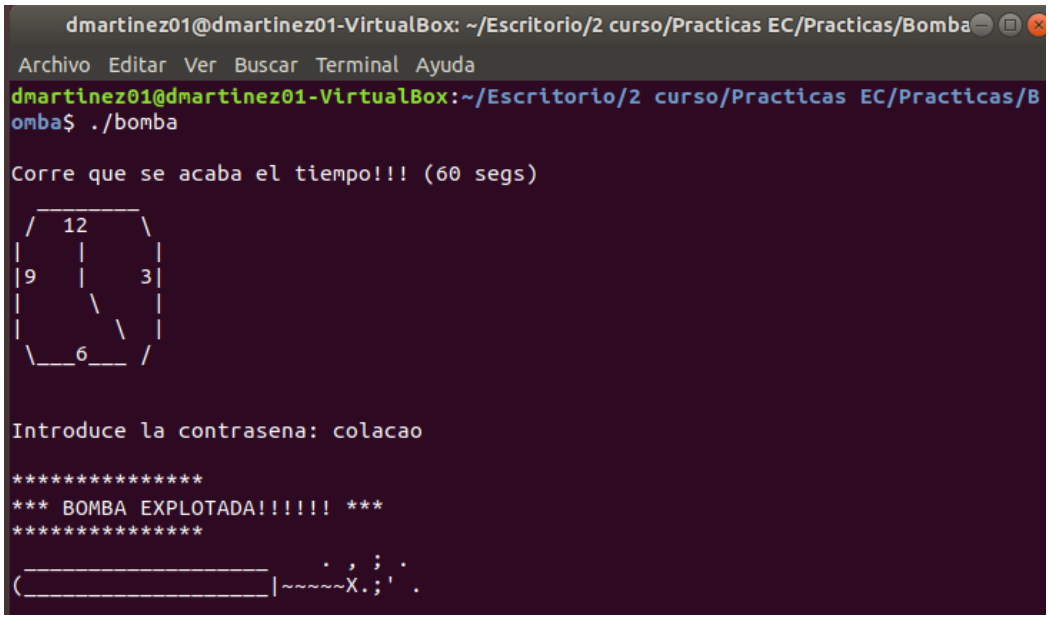
1.- En primer lugar, debemos obtener la bomba correspondiente, en mi caso, utilizare la bomba que yo he creado (“Bomba_DMD_2020.c”), al cual debemos aplicarle el siguiente comando:

gcc -Og Bomba_DMD_2020.c -o bomba -no-pie -fno-guess-branch-probability

Pudiendo utilizar distintos tipos de optimizaciones, entre las que podemos destacar desde la –Og hasta el –O2. Creándose su correspondiente ejecutable.



2. En segundo lugar es ejecutarlo, donde vemos que tenemos que averiguar su contraseña y pin correspondiente.

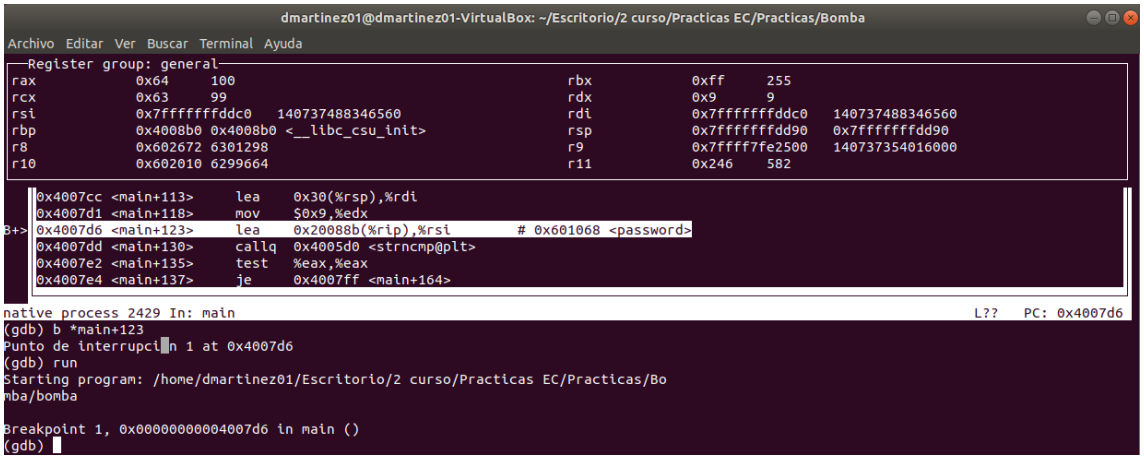


3. Para ello debemos introducirnos en dicho ejecutable a través del debug e ir mirando los registros para comprobar y obtener las contraseñas.

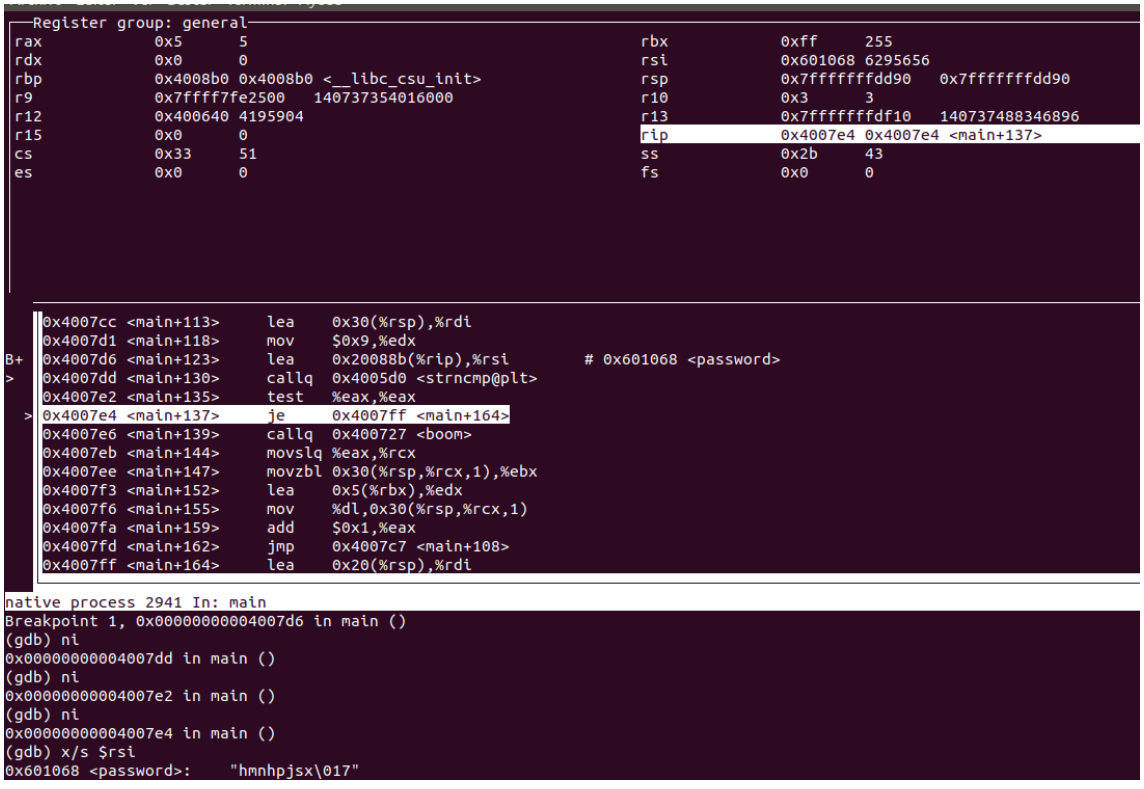
→ gdb bomba

→layout regs

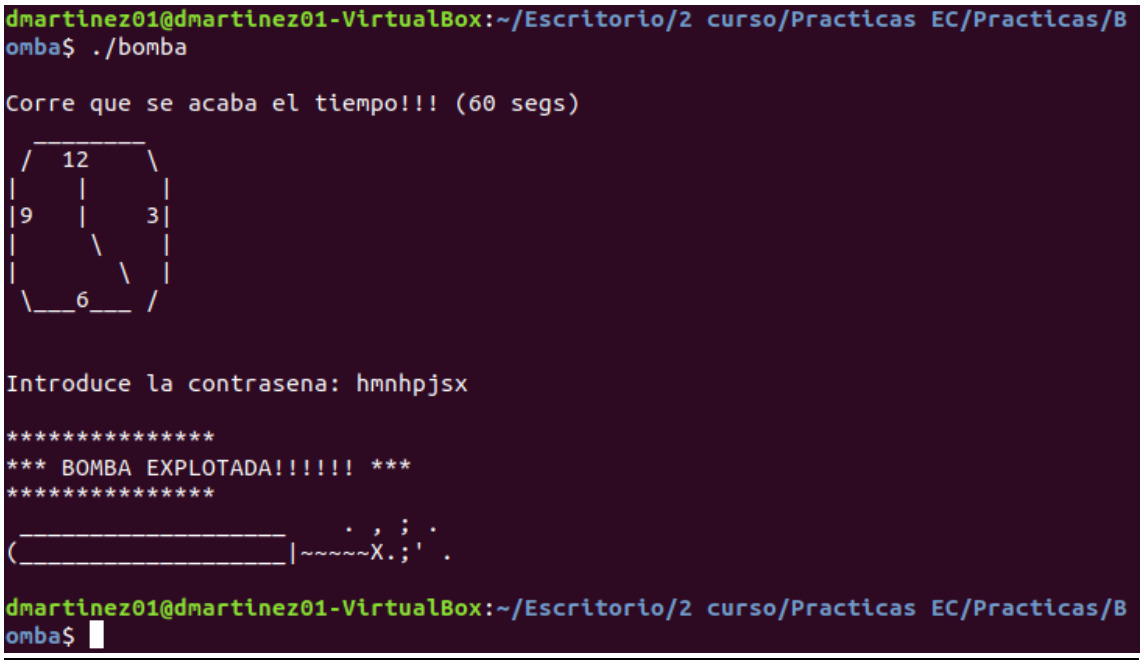
Una vez estamos debugeando, hay que buscar el lugar exacto donde se comparan las contraseñas para poder continuar con el ejercicio, donde si nos damos cuenta, esta se realiza en el string compare, en la línea (*main+130).



Por tanto, para sacar su contraseña debo ver los valores que hay en el registro %rsi, viendo así su contraseña correspondiente:

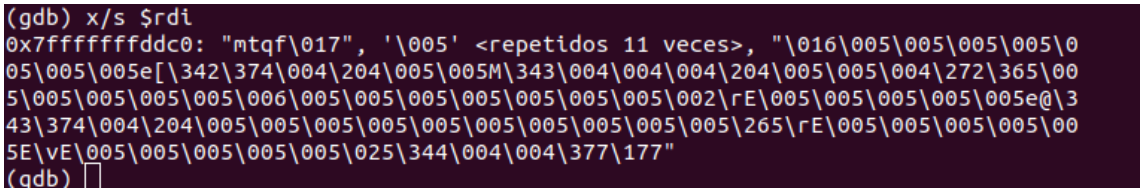


Sin embargo, nos encontramos con el primer problema, si introducimos la contraseña nos explotara la bomba:



Esto quiere decir que la contraseña esta encriptada, y no vamos a poder poner directamente el resultado de %rsi, para ello podemos comprobar con nuestra contraseña anterior y mirar en el registro %rdi para ver dicha comparación y calcular esa razón correspondiente para saber cuánto hay que sumarlo o restarle a esta.

Por ello si introducimos la contraseña “hola” y nos introducimos en el registro %rdi obtenemos lo siguiente:



Donde nuestra nueva contraseña encriptada seria “mtqf”¸ para saber la razón de encriptación simplemente basta saber que valor llega desde la primera letra de nuestra contraseña hasta la primera letra de la contraseña encriptada:

“hola” ---> “mtqf”;

“h” ---> “m”

Para sacarlo nos vamos a la tabla ASCII y vemos sus correspondientes valores:

“h” = 104 // “m” = 109

Con esto llegamos a la conclusión de que la razón es 5, y simplemente para saber su contraseña hay que restarle 5 a cada letra.

Consiguiendo así la contraseña encriptada:

“hmnhpjsx” ---> “chickens”;

Vamos a comprobarlo:

```
dmartinez01@dmartinez01-VirtualBox:~/Escritorio/2 curso/Practicas EC/Practicas/B
omba$ ./bomba

Corre que se acaba el tiempo!!! (60 segs)
  _________
 /  12     \
|     |     |
|9    |    3|
|      \     |
|       \    |
 \___6___ /


Introduce la contrasena: chickens

Introduce el pin:
```

Vemos que la contraseña era correcta pero se nos presenta otra problema, ahora nos pide un pin, veamos entonces de nuevo en el gdb, donde se encuentra este.

Como podemos ver se realiza otro string compare el cual vamos a analizar para saber si ahí se realiza la comparación entre pines. Analizando las sentencias llegamos hasta el main+272, donde se realiza una comparación entre los registros con la orden “jne”:

```
─Register group: general─
rax            0x1      1                                      rbx            0x1      1
rdx            0x7ffff7dcf8d0   140737351842000                rsi            0x1      1
rbp            0x4008b0 0x4008b0 <__libc_csu_init>             rsp            0x7fffffffdd90   0x7fffffffdd90
r9             0x0      0                                      r10            0x7ffff7b80c40   140737349422144
r12            0x400640 4195904                                r13            0x7fffffffdf10   140737488346896
r15            0x0      0                                      rip            0x40086b 0x40086b <main+272>
cs             0x33     51                                     ss             0x2b     43
es             0x0      0                                      fs             0x0      0

   0x400853 <main+248>     jne    0x400866 <main+267>
   0x400855 <main+250>     lea    0x2a6(%rip),%rdi        # 0x400b02
   0x40085c <main+257>     mov    $0x0,%eax
   0x400861 <main+262>     callq  0x400620 <__isoc99_scanf@plt>
   0x400866 <main+267>     cmp    $0x1,%ebx
   0x400869 <main+270>     jne    0x400823 <main+200>
B+> 0x40086b <main+272>     mov    0x2007ef(%rip),%eax        # 0x601060 <passcode>
   0x400871 <main+278>     cmp    %eax,0xc(%rsp)
   0x400875 <main+282>     je     0x40087c <main+289>
   0x400877 <main+284>     callq  0x400727 <boom>
   0x40087c <main+289>     lea    0x10(%rsp),%rdi
   0x400881 <main+294>     mov    $0x0,%esi
   0x400886 <main+299>     callq  0x4005f0 <gettimeofday@plt>
   0x40088b <main+304>     mov    0x10(%rsp),%rax
   0x400890 <main+309>     sub    0x20(%rsp),%rax
```

Donde si miramos bien las sentencias y hacemos un par de next instructions vemos como sacan un dato de la pila en la dirección de (%rip+0x2007ef), y lo mete en el registro %eax, si lo mostramos vemos que:

```
(gdb) print $eax
$1 = 2544
(gdb)
```

Vamos a comprobar si este pin es el correcto:

```
Corre que se acaba el tiempo!!! (60 segs)
  _________
 /  12     \
|     |     |
|9    |    3|
|      \     |
|       \    |
 \___6___ /


Introduce la contrasena: chickens

Introduce el pin: 2544

---- NOS SALVAMOS!!!! ----
            __
           / _|
  ___   __ _| |_ ___
 / __| / _` |  _/ _ \
 __  | (_| | || __/
 |___/ \__,_|_| \___|

dmartinez01@dmartinez01-VirtualBox:~/Escritorio/2 curso/Practicas EC/Practicas/Bomba$
```

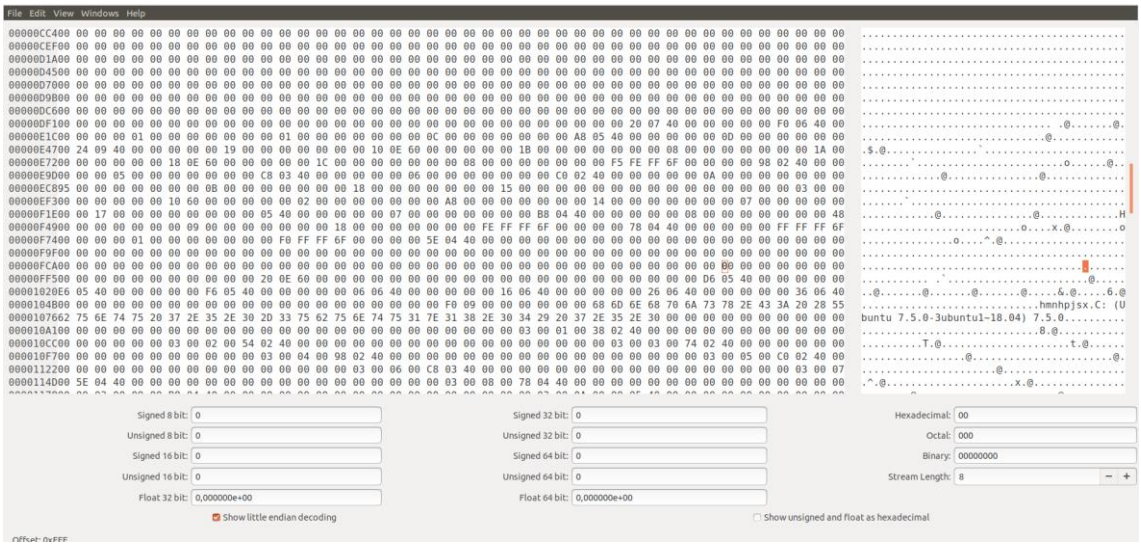
Como hemos podido comprobar el pin que habíamos obtenido era correcto y no se encontraba encriptado por lo que hemos podido desactivar la bomba. Por lo que esta sería la manera de desactivar mi bomba.

- **Modificar la contraseña de la bomba:**

Para modificar la contraseña vamos a utilizar el comando ghex que permite ver el código de manera hexadecimal:

“ghex Bomba_DMD_2020”

Donde nos aparecerá lo siguiente:



Una vez estamos dentro, tenemos que buscar dicha contraseña, que en mi caso se encuentra encriptada:

```
..........  .  ......................@.....
..@.......@.......@.......@.....&.@.....6.@
.............................hmnhpjsx.C: (U
buntu 7.5.0-3ubuntu1~18.04) 7.5.0..........
.............................8.@...........
```

Una vez que la hemos sacado vemos que su Offset es 0x106F, simplemente si la modificamos debería cambiar (teniendo en cuenta la razón de encriptación).

Por ejemplo, si yo cambio la contraseña de esta manera:

“chickens” → “hmnhpjsx”.

Si yo cambiase la “x” del final por una “p”, aplicándole la razón de 5 en este caso:

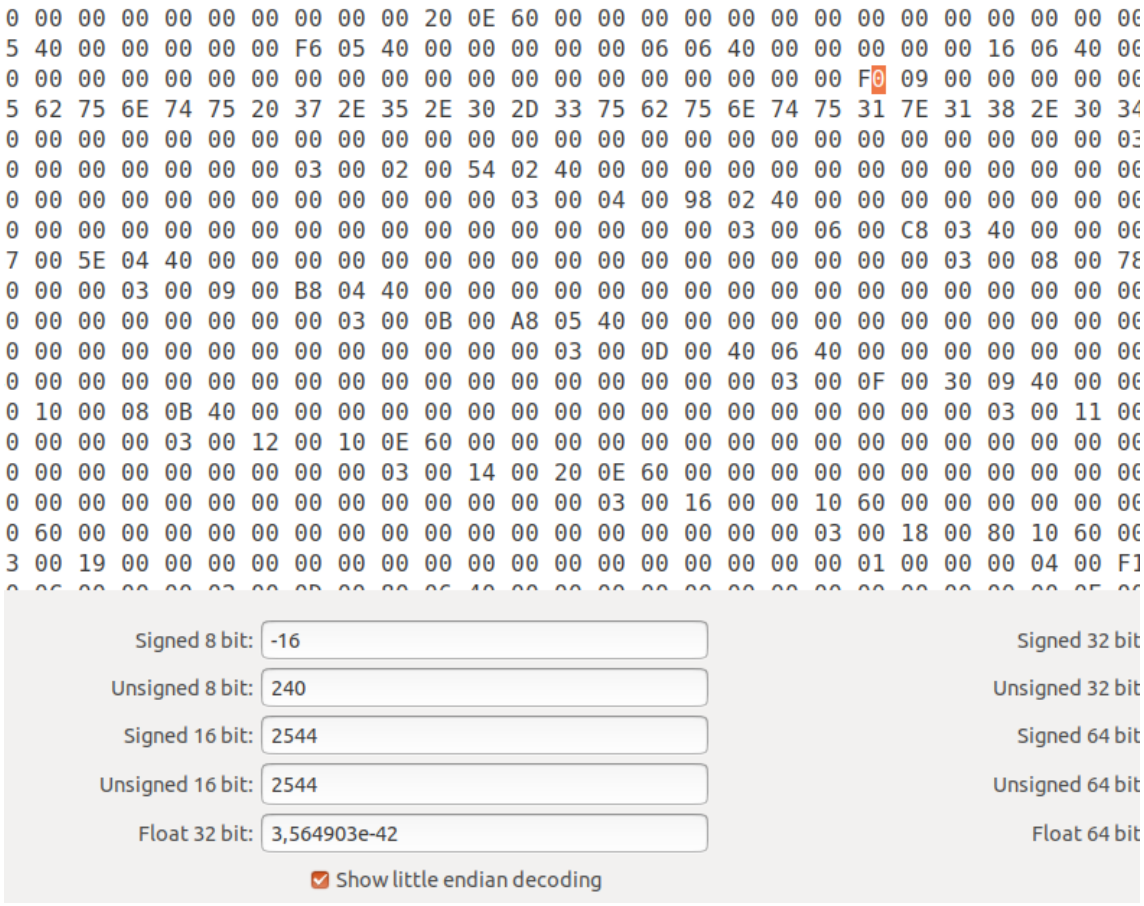
Se quedaría el código de tal manera que:

“chickenk” → “hmnhpjsp”.

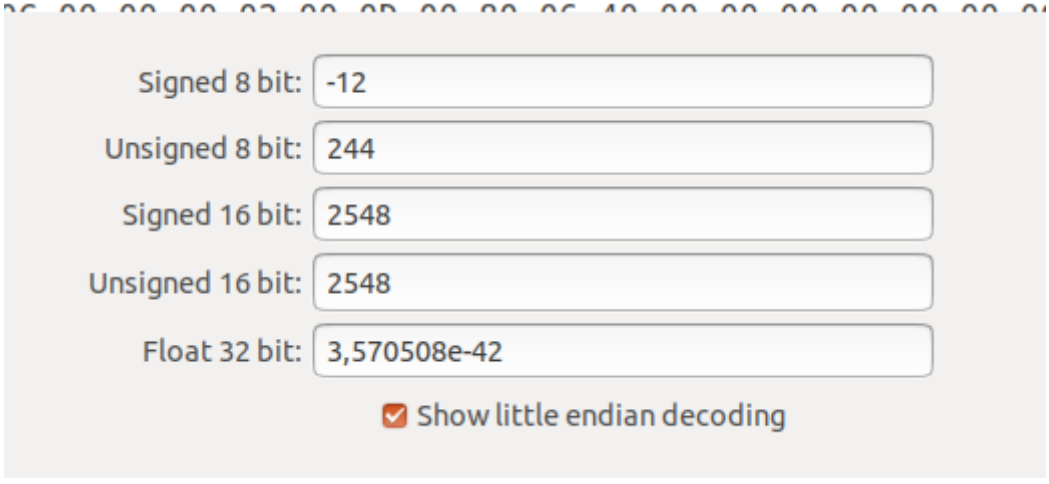
```
dmartinez01@dmartinez01-VirtualBox:~/Escritorio/Practica 4$ ghex bomba
dmartinez01@dmartinez01-VirtualBox:~/Escritorio/Practica 4$ ./bomba

Corre que se acaba el tiempo!!! (60 segs)
  ________
 /  12    \
|    |     |
|9   |    3|
|     \     |
|       \   |
 \___6___ /


Introduce la contrasena: chickenk

Introduce el pin:
```

Vemos que si me acepta la contraseña y la hemos podido modificar con éxito.

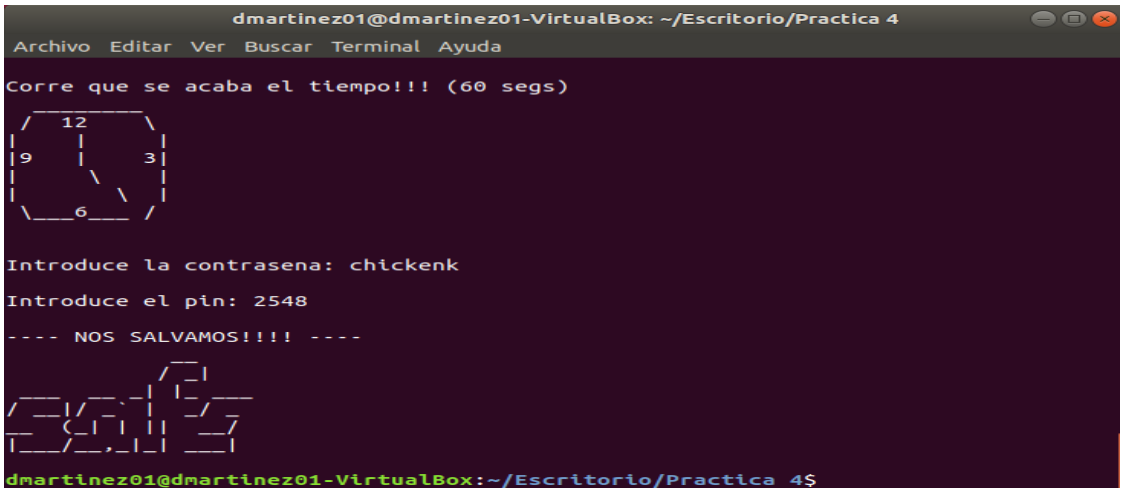
Para modificar el pin, vamos a emplear otra vez el comando ghex,



Una vez encontramos nuestro pin, probamos a cambiarlo poniendo F4 por ejemplo, quedando:



Por ultimo vamos a comprobar si funciona:



Y vemos que se ha modificado con éxito.