En esta bomba se aplicará el mismo procedimiento que en la anterior. Sacaremos el pin y contraseña "alteradas" y haremos ingeniería inversa de la función que los codifique.

1. Obtener el password. En la función c_compp encontraremos la línea
```
0x40078f <c_compp+52>   lea    0x2008d2(%rip),%rsi        # 0x601068 <password>
```
Mirando en 0x601068 veremos la clave "alterada".
```
(gdb) p (char*) 0x601068
$2 = 0x601068 <password> "pxuflhodjr\n"
```

2. Obtener el pin En la función c_compc encontraremos la linea
```
0x4007bc <c_compc>      mov    0x20089e(%rip),%eax        # 0x601060 <passcode>
```
Mirando en 0x601060 veremos el pin "alterado"
```
(gdb) p(int)passcode
$3 = 896
```

3. Función de codificación de PIN y contraseña.
Las funciones que se encargan de encriptar los códigos que pasamos en los parámetros están, respectivamente, en *c_compp* y *c_compc*

3.1 Codificación de contraseña
La función c_comp tiene el siguiente código ASM:
```
0x40075b <c_compp>      sub    $0x8,%rsp
0x40075f <c_compp+4>    mov    %rdi,%r8
0x400762 <c_compp+7>    mov    $0x0,%esi
0x400767 <c_compp+12>   movslq %esi,%rdx
0x40076a <c_compp+15>   mov    $0xffffffffffffffff,%rcx
0x400771 <c_compp+22>   mov    $0x0,%eax
0x400776 <c_compp+27>   mov    %r8,%rdi
0x400779 <c_compp+30>   repnz scas %es:(%rdi),%al
0x40077b <c_compp+32>   mov    %rcx,%rax
0x40077e <c_compp+35>   not    %rax
0x400781 <c_compp+38>   sub    $0x2,%rax
0x400785 <c_compp+42>   cmp    %rax,%rdx
0x400788 <c_compp+45>   jb     0x4007a7 <c_compp+76>
0x40078a <c_compp+47>   mov    $0xc,%edx
0x40078f <c_compp+52>   lea    0x2008d2(%rip),%rsi        # 0x601068 <password>
0x400796 <c_compp+59>   mov    %r8,%rdi
0x400799 <c_compp+62>   callq  0x4005d0 <strncmp@plt>
0x40079e <c_compp+67>   test   %eax,%eax
0x4007a0 <c_compp+69>   je     0x4007b7 <c_compp+92>
0x4007a2 <c_compp+71>   callq  0x400727 <boom>
0x4007a7 <c_compp+76>   add    %r8,%rdx
0x4007aa <c_compp+79>   movzbl (%rdx),%eax
0x4007ad <c_compp+82>   add    $0x3,%eax
0x4007b0 <c_compp+85>   mov    %al,(%rdx)
0x4007b2 <c_compp+87>   add    $0x1,%esi
0x4007b5 <c_compp+90>   jmp    0x400767 <c_compp+12>
0x4007b7 <c_compp+92>   add    $0x8,%rsp
0x4007bb <c_compp+96>   retq
```

Aplicando ingeniería inversa, la función suma 3 a cada letra de la cadena a evaluar, antes de compararla con la original.

Dicho esto, si la clave “alterada” es "pxuflhodjr\n", realmente es *“murcielago\n”*

3.2 Codificación de PIN

```
0x4007bc <c_compc>      mov    0x20089e(%rip),%eax        # 0x601060 <passcode>
0x4007c2 <c_compc+6>    lea    -0x650(,%rax,4),%eax
0x4007c9 <c_compc+13>   cmp    %edi,%eax
0x4007cb <c_compc+15>   je     0x4007d6 <c_compc+26>
0x4007cd <c_compc+17>   sub    $0x8,%rsp
0x4007d1 <c_compc+21>   callq  0x400727 <boom>
0x4007d6 <c_compc+26>   repz retq
```

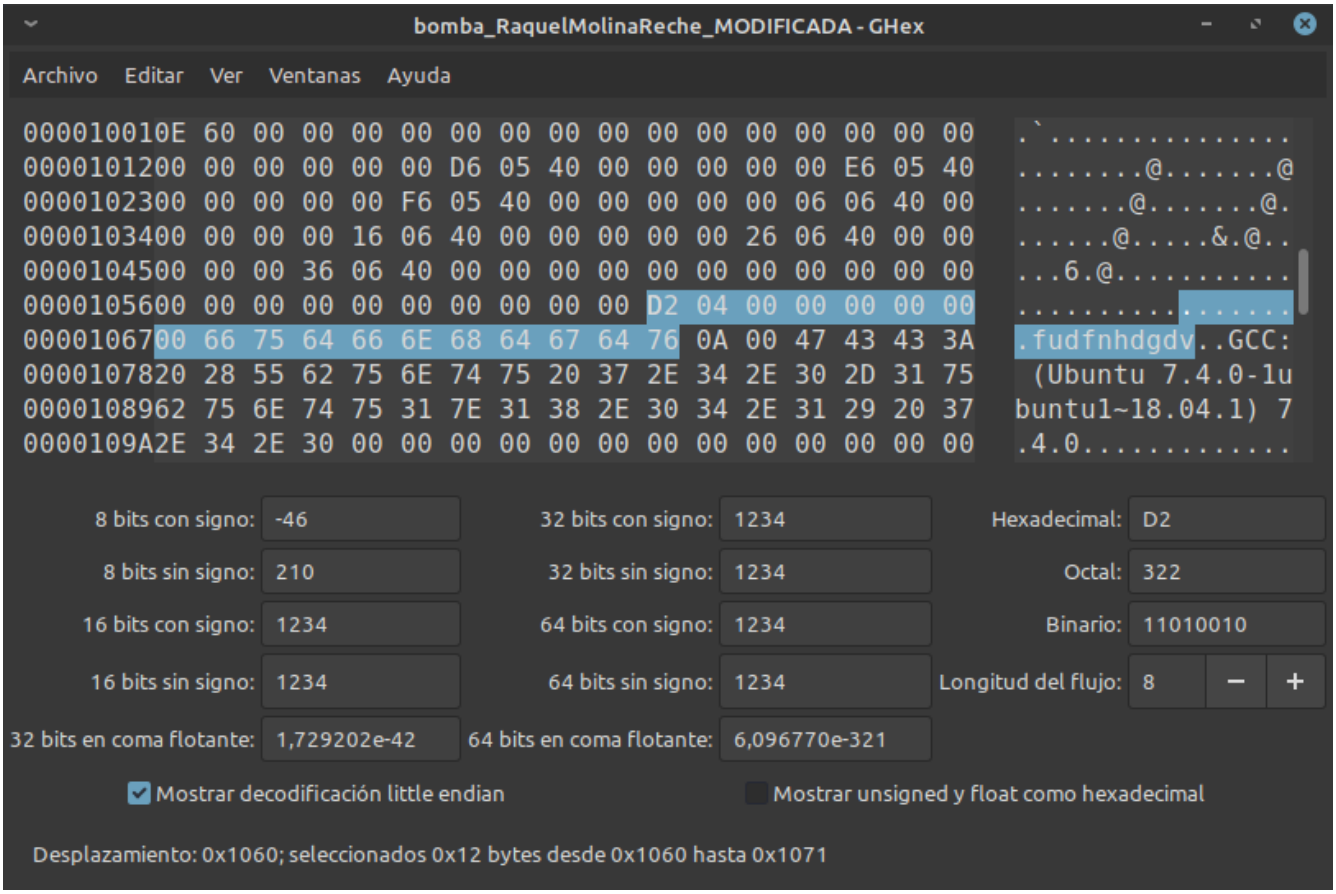
Aplicando ingeniería inversa, descubrimos esta linea:
`0x4007c2 <c_compc+6>    lea    -0x650(,%rax,4),%eax`
La cual hace que la cadena a comparar se multiplique por 4 y se le sume 0x650 = 1616. Hecho esto, si el código “alterado” es 896, el código 896*4+1616 = 1968.

Hecho esto, las claves son:

Contraseña: murcielago
PIN: 1968

La alteración de las claves se da con respecto a la codificación.
Entonces, si quiero usar de clave *“crackeadas\n”*, sumando 3 sería *“fudfnhdgdv\n”*.
La “semilla” es 896 = 0x380, en little endian, 80 03.
Cambiamos la “semilla” a 1234 = 0x4D2, en little endian, D2 04.

Así pues, 1234 * 4 - 1616 = 3320.



Las claves de la bomba modificada son:

Contraseña mod: crackeadas
PIN mod: 3320