

ESTRUCTURA DE COMPUTADORES -

2º GRADO INGENIERÍA INFORMÁTICA

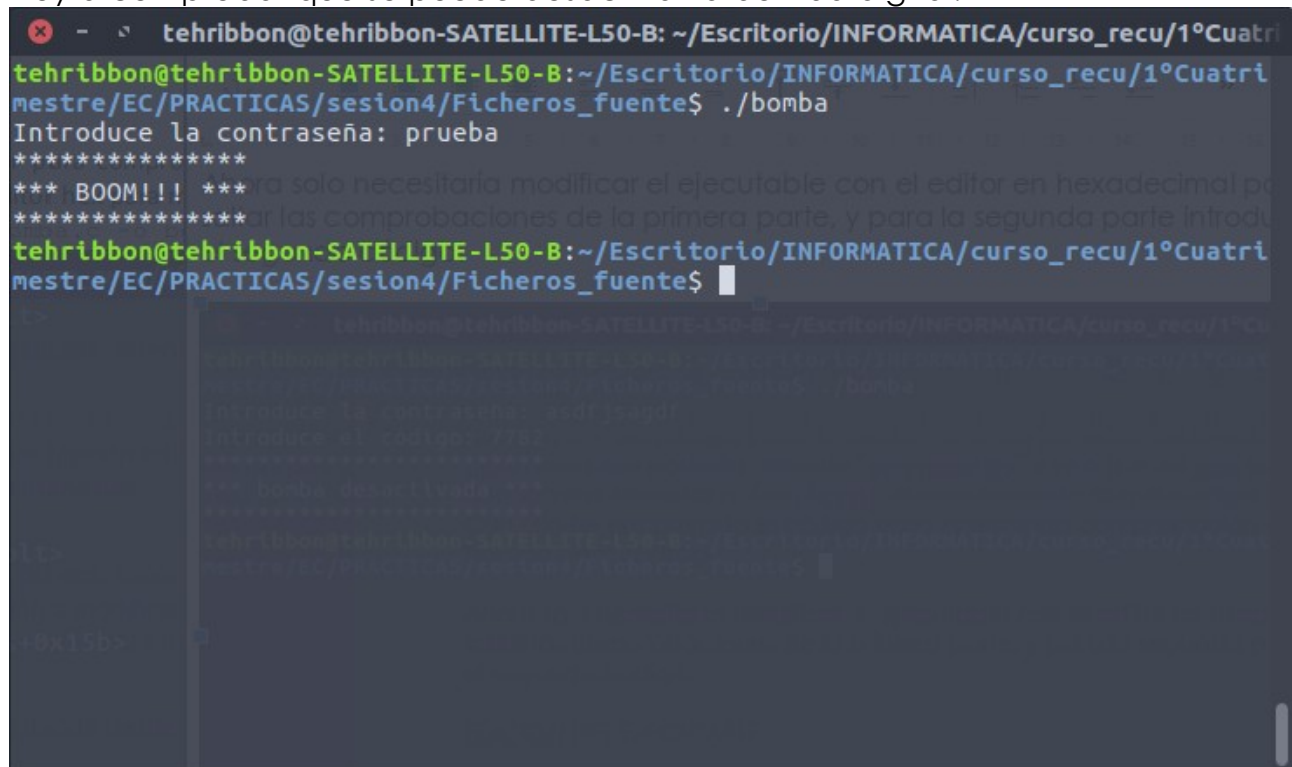
PRÁCTICA 4.- BOMBA DIGITAL – DESENSAMBLADORES

Mario Antonio López Ruiz - 45109755Q

CÓMO DESACTIVAR LA BOMBA

1.INICIO

Voy a comprobar que se puede desactivar la bomba digital:



```
tehribbon@tehribbon-SATELLITE-L50-B: ~/Escritorio/INFORMATICA/curso_recu/1ºCuatri
tehribbon@tehribbon-SATELLITE-L50-B:~/Escritorio/INFORMATICA/curso_recu/1ºCuatri
mestre/EC/PRACTICAS/sesion4/Ficheros_fuente$ ./bomba
Introduce la contraseña: prueba
*****
*** BOOM!!! ***
*****
*** para solo necesitaría modificar el ejecutable con el editor en hexadecimal pa
*****
*** las comprobaciones de la primera parte, y para la segunda parte introdu
tehribbon@tehribbon-SATELLITE-L50-B:~/Escritorio/INFORMATICA/curso_recu/1ºCuatri
mestre/EC/PRACTICAS/sesion4/Ficheros_fuente$
```

Lo he realizado utilizando el depurador gdb directamente desde terminal, en vez de utilizar ddd, ya que funciona de forma menos eficiente y me resulta mas rápido/sencillo.

Abro el archivo con la orden "gdb bomba", y en otro terminal concurrente procedo a realizar el desensamblado del archivo con objdump para ir haciendo el seguimiento.

2.EVITAR COMPROBACIONES

```
tehribbon@tehribbon-SATELLITE-L50-B: ~/Escritorio/INFORMATICA/curso_recu/1°Cuatr
Type "apropos word" to search for commands related to "word"...
Leyendo símbolos desde bomba...(no se encontraron símbolos de depuración)hecho.
(gdb) disas main
Dump of assembler code for function main:
0x08048846 <+0>:    lea     0x4(%esp),%ecx
0x0804884a <+4>:    and     $0xffffffff0,%esp
0x0804884d <+7>:    pushl   -0x4(%ecx)
0x08048850 <+10>:   push    %ebp
0x08048851 <+11>:   mov     %esp,%ebp
0x08048853 <+13>:   push    %ecx
0x08048854 <+14>:   sub     $0x84,%esp
0x0804885a <+20>:   mov     %gs:0x14,%eax
0x08048860 <+26>:   mov     %eax,-0xc(%ebp)
0x08048863 <+29>:   xor     %eax,%eax
0x08048865 <+31>:   sub     $0x8,%esp
0x08048868 <+34>:   push    $0x0
0x0804886a <+36>:   lea     -0x80(%ebp),%eax
0x0804886d <+39>:   push    %eax
0x0804886e <+40>:   call    0x80484a0 <gettimeofday@plt>
0x08048873 <+45>:   add     $0x10,%esp
0x08048876 <+48>:   sub     $0xc,%esp
0x08048879 <+51>:   push    $0x8048a74
0x0804887e <+56>:   call    0x8048480 <printf@plt>
0x08048883 <+61>:   add     $0x10,%esp
0x08048886 <+64>:   mov     0x804a060,%eax
0x0804888b <+69>:   sub     $0x4,%esp
--Type <return> to continue, or q <return> to quit--
```

Desde gdb observo que hay en el main con la orden "disas main".

Coloco un breakpoint en el main con "break main" y comienzo a ejecutar con nexti para ver cuando explota la bomba.

Cuando llega a la posición 0x080488d2 veo que llama a la función CmpContrs, por lo tanto voy a comprobar si es ahí donde explota la bomba.

Coloco otro breakpoint con "break CmpContrs" y vuelvo a ejecutar hasta llegar a ese punto.

Al ejecutar esa función con nexti se puede observar que se está produciendo una serie de bucles, por lo tanto ejecutar con nexti no es viable(vamos a tener que ejecutar esa orden muchas veces).

Mirando el código en ensamblador, se observa que en 0x08048801 es donde se llama a la función boom().

```
tehribbon@tehribbon-SATELLITE-L50-B: ~/Escritorio/INFORMATICA/curso_recu/1°Cuatr
80487ce:    8b 45 ec          mov     -0x14(%ebp),%eax
80487d1:    0f b6 55 db      movzbl -0x25(%ebp),%edx
80487d5:    88 14 08          mov     %dl,(%eax,%ecx,1)
80487d8:    83 ec 0c          sub     $0xc,%esp
80487db:    68 3c a0 04 08    push    $0x804a03c
80487e0:    e8 fb fc ff ff    call    80484e0 <strlen@plt>
80487e5:    83 c4 10          add     $0x10,%esp
80487e8:    89 c2             mov     %eax,%edx
80487ea:    8b 45 ec          mov     -0x14(%ebp),%eax
80487ed:    83 ec 04          sub     $0x4,%esp
80487f0:    52               push    %edx
80487f1:    50               push    %eax
80487f2:    ff 75 d4          pushl   -0x2c(%ebp)
80487f5:    e8 16 fd ff ff    call    8048510 <strncmp@plt>
80487fa:    83 c4 10          add     $0x10,%esp
80487fd:    85 c0             test    %eax,%eax
80487ff:    74 05             je      8048806 <CmpContrs+0x15b>
8048801:    e8 25 fe ff ff    call    804862b <boom>
8048806:    89 dc             mov     %ebx,%esp
8048808:    90               nop
8048809:    8b 45 f4          mov     -0xc(%ebp),%eax
804880c:    65 33 05 14 00 00 xor     %gs:0x14,%eax
8048813:    74 05             je      804881a <CmpContrs+0x16f>
8048815:    e8 96 fc ff ff    call    80484b0 <__stack_chk_fail@plt>
```

Pongo un breakpoint en ese punto para ver que ocurre con “break *0x08048801” y ejecuto hasta llegar a esa línea. Puedo borrar los breakpoints anteriores con “delete 1” y “delete 2”.

Llego a ese punto y compruebo que, efectivamente ahí es donde la bomba explota. Leyendo la ordena anterior vemos que realiza un test de eax, con eax y llama a boom si %eax es igual a %eax.

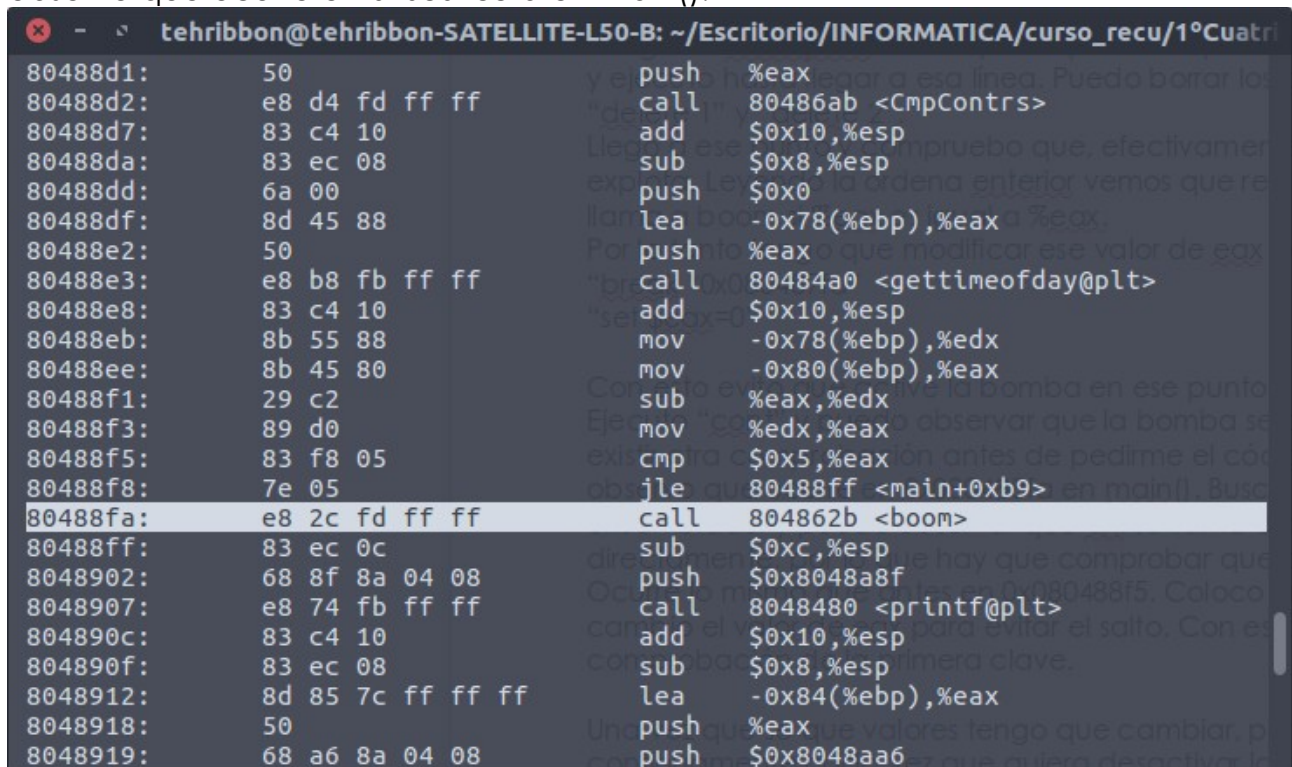
Por lo tanto tengo que modificar ese valor de eax para evitar el salto con:

“break *0x080487fa”

“set \$eax=0”

Con esto evito que active la bomba en ese punto.

Ejecuto “cont” y puedo observar que la bomba se activa, por lo que tiene que existir otra comprobación antes de pedirme el código. Lo compruebo con nexti y observo que ocurre en 0x080488fa en main().



```
tehribbon@tehribbon-SATELLITE-L50-B: ~/Escritorio/INFORMATICA/curso_recu/1ºCuatri
80488d1: 50          push    %eax
80488d2: e8 d4 fd ff ff  call   80486ab <CmpContrs>
80488d7: 83 c4 10     add     $0x10,%esp
80488da: 83 ec 08     sub     $0x8,%esp
80488dd: 6a 00       push    $0x0
80488df: 8d 45 88     lea     -0x78(%ebp),%eax
80488e2: 50          push    %eax
80488e3: e8 b8 fb ff ff  call   80484a0 <gettimeofday@plt>
80488e8: 83 c4 10     add     $0x10,%esp
80488eb: 8b 55 88     mov     -0x78(%ebp),%edx
80488ee: 8b 45 80     mov     -0x80(%ebp),%eax
80488f1: 29 c2       sub     %eax,%edx
80488f3: 89 d0       mov     %edx,%eax
80488f5: 83 f8 05     cmp     $0x5,%eax
80488f8: 7e 05       jle     80488ff <main+0xb9>
80488fa: e8 2c fd ff ff  call   804862b <boom>
80488ff: 83 ec 0c     sub     $0xc,%esp
8048902: 68 8f 8a 04 08  push   $0x8048a8f
8048907: e8 74 fb ff ff  call   8048480 <printf@plt>
804890c: 83 c4 10     add     $0x10,%esp
804890f: 83 ec 08     sub     $0x8,%esp
8048912: 8d 85 7c ff ff ff  lea     -0x84(%ebp),%eax
8048918: 50          push    %eax
8048919: 68 a6 8a 04 08  push   $0x8048aa6
```

Busco esa línea en el código ensamblador y puedo observar que ahí se llama a la función boom() directamente, por lo que hay que comprobar que comparación se realiza antes.

Ocorre lo mismo que antes en 0x080488f5. Coloco un breakpoint en ese punto y cambio el valor de eax (set \$eax=0) para que se produzca siempre el salto y no se llame a boom. Con esto he conseguido saltarme la comprobación de la primera clave.

Una vez que se que valores tengo que cambiar, para no tener que hacerlo continuamente cada vez que quiera desactivar la bomba, se puede usar el editor en hexadecimal para evitar esos saltos, transformando los saltos del tipo JE a JMP de la forma:

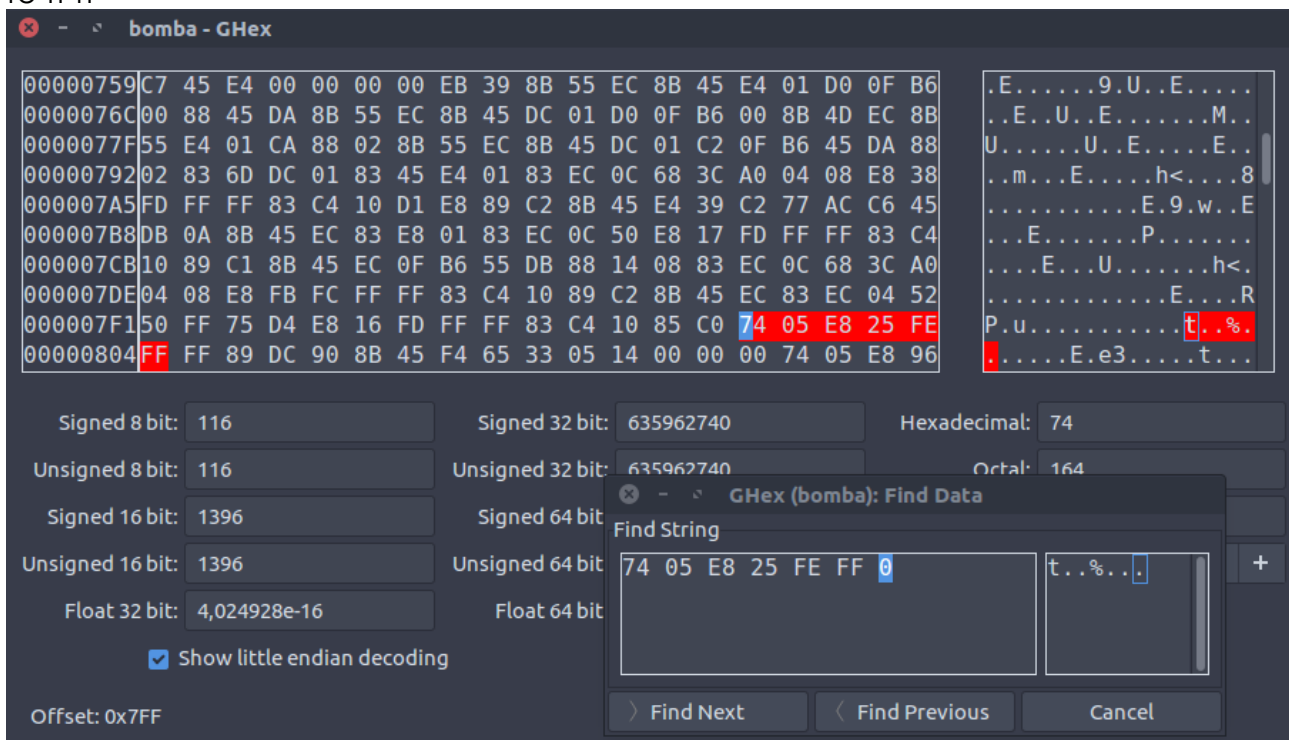
-Busco la dirección en la que se produce el salto en ensamblador


```

tehribbon@tehribbon-SATELLITE-L50-B: ~/Escritorio/INFORMATICA/curso_recu/1ºCuatri
80487d1: 0f b6 55 db      movzbl -0x25(%ebp),%edx
80487d5: 88 14 08          mov     %dl,(%eax,%ecx,1)
80487d8: 83 ec 0c          sub     $0xc,%esp
80487db: 68 3c a0 04 08    push   $0x804a03c
80487e0: e8 fb fc ff ff    call   80484e0 <strlen@plt>
80487e5: 83 c4 10          add     $0x10,%esp
80487e8: 89 c2             mov     %eax,%edx
80487ea: 8b 45 ec          mov     -0x14(%ebp),%eax
80487ed: 83 ec 04          sub     $0x4,%esp
80487f0: 52               push   %edx
80487f1: 50               push   %eax
80487f2: ff 75 d4          pushl  -0x2c(%ebp)
80487f5: e8 16 fd ff ff    call   8048510 <strncmp@plt>
80487fa: 83 c4 10          add     $0x10,%esp
80487fd: 85 c0             test    %eax,%eax
80487ff: 74 05             je      8048806 <CmpContrs+0x15b>
8048801: e8 25 fe ff ff    call   804862b <boom>
8048806: 89 dc             mov     %ebx,%esp
8048808: 90               nop
8048809: 8b 45 f4          mov     -0xc(%ebp),%eax
804880c: 65 33 05 14 00 00 00 xor     %gs:0x14,%eax
8048813: 74 05             je      804881a <CmpContrs+0x16f>
8048815: e8 96 fc ff ff    call   80484b0 <__stack_chk_fail@plt>
804881a: 8b 5d fc          mov     -0x4(%ebp),%ebx

```

-Abro el archivo con el editor hexadecimal ghex, y busco la cadena 74 05 e8 25 fe ff ff



Y substituyo 74→eb para modificar el tipo de salto.

-Realizo lo mismo con los otros saltos que encontré anteriormente (74 05 e8 96 fc ff ff)

3.OBTENCIÓN DEL CÓDIGO

Siguiendo por donde lo hemos dejado antes, ejecuto nexti hasta ver en que parte punto se activa la bomba.

Se observa que en 0x08048930 entra en la funcion <error>, vamos a ver que ocurre ahí.

Busco en ensamblador en que posición se encuentra la función y pongo un punto de ruptura con "break error".

Comienza la función y vamos a hacer un seguimiento de que ocurre en \$eax, ya que observando el código vemos que la bomba se activa si lo que hay en %eax y en -0xc(%ebp) no son iguales.

Por ello, como en eax vamos a tener el código que he introducido, compruebo que hay en -0xc(%ebp) y eso lo puedo ver en la instrucción 0x804882d. Avanzo con nexti hasta esa posición, ejecuto "info registers" y veo que en eax está el valor que se va a trasladar a -0xc(%ebp), el cual coincide con el código (7782). Con ello he encontrado el código para la segunda comprobación y tengo la bomba desactivada.

```
(gdb) nexti
0x0804882d in error ()
(gdb) info registers
eax             0x1e66      7782
ecx             0x1         1
edx             0xf7fad87c
ebx             0x0         0
esp             0xfffffdb0   0xfffffdb0
ebp             0xffffcdc8   0xffffcdc8
esi             0xf7fac000   -134561792
edi             0xf7fac000   -134561792
eip             0x804882d     0x804882d <error+14>
eflags          0x207       [ CF PF IF ]
cs              0x23         35
ss              0x2b         43
ds              0x2b         43
es              0x2b         43
fs              0x0         0
gs              0x63         99
(gdb)
```

Ahora podría iniciar el ejecutable sin que se active la bomba.

```
tehribbon@tehribbon-SATELLITE-L50-B: ~/Escritorio/INFORMATICA/curso_recu/1ºCuatri
mestre/EC/PRACTICAS/sesion4/Ficheros_fuente$ ./bomba
Introduce la contraseña: asdfjsagdf
Introduce el código: 7782
*****
*** bomba desactivada ***
*****
tehribbon@tehribbon-SATELLITE-L50-B: ~/Escritorio/INFORMATICA/curso_recu/1ºCuatri
mestre/EC/PRACTICAS/sesion4/Ficheros_fuente$
```