

# Memoria sobre la segunda fase de HackingForce

---

*Grupo 4 – PL4, FCR GIITIN01 (20-21)*

*UO283319 – Juan Francisco Mier Montoto*

*UO282574 - Miguel del Riego Lázaro*

## Índice

- [Fase 2 - Salvando al mundo](#)
  - [Nombre del subgrupo](#)
  - [Desactivando etapas](#)
    - [Bomba 1](#)
    - [Bomba 2](#)
    - [Bomba 3](#)
  - [Modificación del código](#)
- [Reparto de trabajo](#)

## Fase 2 – Salvando al mundo

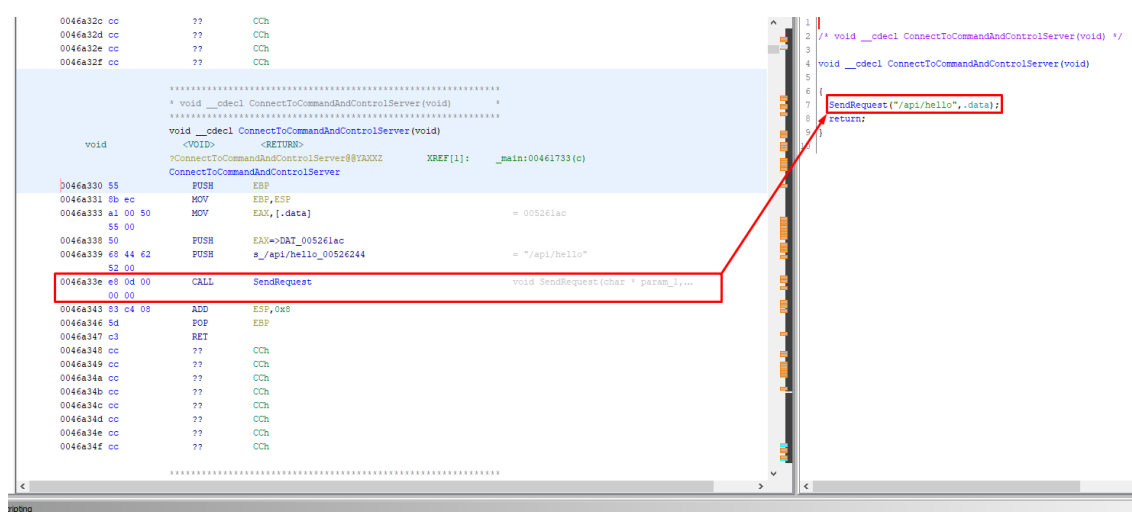
Tras aprender completar el entrenamiento en Assembly y C++, “alguien” ha decidido utilizar sus conocimientos para poner a todo el mundo a prueba. Lo primero de todo será descubrir de quién se trata.

### Nombre del subgrupo

Para obtener el nombre del subgrupo, hay que analizar los paquetes que envía y recibe la aplicación principal al ser ejecutada, más concretamente durante el procedimiento “ConnectToCommandAndControlServer()”.

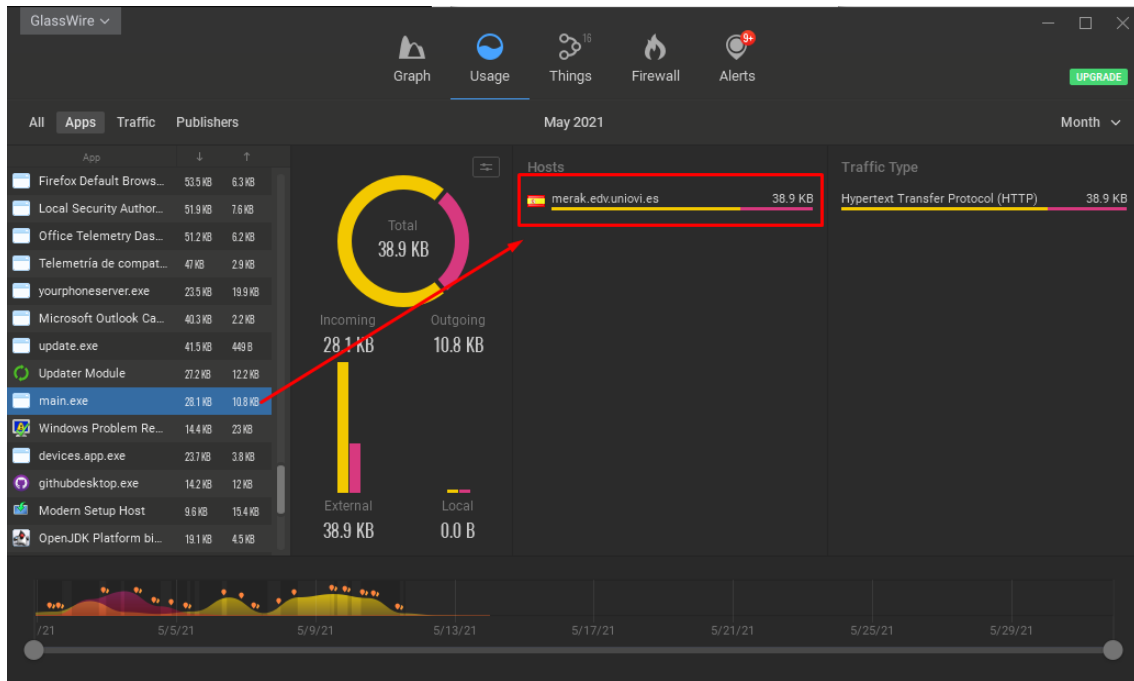
```
6
7     using std::cout;
8     using std::cin;
9     using std::endl;
10
11     int main()
12     {
13         ConnectToCommandAndControlServer();
14
15         Stage1();
16
17         cout << "Stage 1 disabled" << endl;
18
19         Stage2();
20
21         cout << "Stage 2 disabled" << endl;
22
23         Stage3();
24
25         cout << "Stage 3 disabled" << endl;
26
27         BombDisabled();
28         cout << "Wow, you've just saved the Earth!" << endl;
29         return 1;
```

Analizando el ejecutable con Ghidra, podemos ver lo que hace realmente este procedimiento.

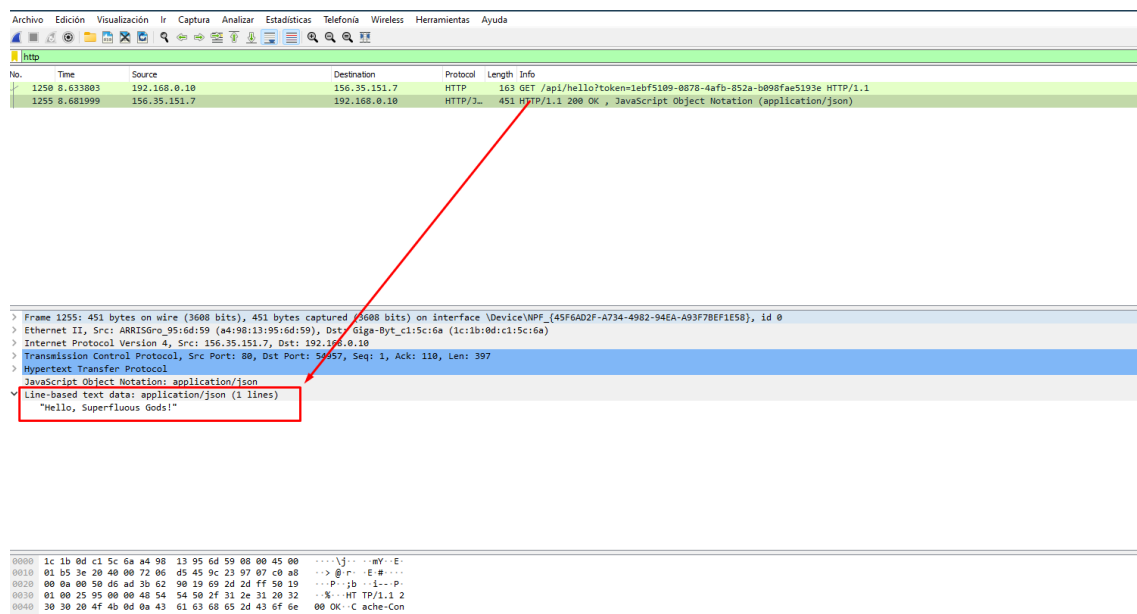


Al parecer, está mandando una señal muy sencilla al servidor mediante “SendRequest()”, que es el procedimiento encargado también de enviar el número de la bomba y la contraseña al desactivar una bomba.

Al ejecutar el programa, podemos observar claramente esta petición, que está dirigida a un servidor de uniovi:



Para observar este paquete, abrimos Wireshark antes de la ejecución y filtramos por “http”:



Como podemos observar, a la petición GET de SendRequest() se devuelve un 200 OK que incluye un json de una sola línea que contiene muy convenientemente el nombre del subgrupo, en este caso “Superfluous Gods”.

## Desactivando etapas

Esta es, sin duda, la parte más difícil del programa: desactivar las bombas. Para hacerlo, ya habíamos visto que podríamos mandar un GET manualmente para desactivarlas como se hace en el código, como por ejemplo en la bomba 1:

```
Defuse(1, "bcf1884a-cfdd-4520-8dffa-56d3447ecde3");
```

O al llegar al final del archivo, al desactivar todas las bombas:

```
SendRequest("/api/disabled", .data);
```

Pero lo que se nos pide es buscar en el código para introducir los códigos que desactiven las bombas. El método general para todas bombas será poner un punto de interrupción en la fase que queramos examinar, pasar a desensamblado e introducirnos (F11) en el procedimiento para poder observarlo y examinarlo.

### Bomba 1

Accediendo al método principal, vemos que la clave de esta etapa se encuentra en una comparación de cadenas "strcmp", en la dirección de memoria señalada en pantalla.

```
00465522 push     3E8h
00465527 lea      eax,[ebp-3ECh]
0046552D push     eax
0046552E mov     ecx,offset std::cin (0556BB0h)
00465533 call    std::basic_istream<char,std::char_traits<char> >::get
00465538 push     525A14h
0046553D lea      ecx,[ebp-3ECh]
00465543 push     ecx
00465544 call    strcmp (04E34C0h)
00465549 add     esp,8
0046554C test    eax,eax
0046554E je     Stage1+4Ch (046555Ch)
00465550 push     1
00465552 call    Explode (0465450h)
00465557 add     esp,4
0046555A jmp     Stage1+5Bh (046556Bh)
0046555C push     525A20h
00465561 push     1
```

46 %

Al ir a esa dirección de memoria, vemos que se compara la cadena introducida con el penúltimo valor introducido en la pila.

```
--- minkernel\crt\ucrt\src\appcrt\string\i386\strcmp.asm ----
004E34C0 mov     edx,dword ptr [esp+4]
004E34C4 mov     ecx,dword ptr [esp+8]
004E34C8 test    edx,3
004E34CE jne     dodwords+40h (04E3510h)
004E34D0 mov     eax,dword ptr [edx]
004E34D2 cmp     al,byte ptr [ecx]
004E34D4 jne     dodwords+38h (04E3508h)
004E34D6 test    al,al
004E34D8 je     dodwords+30h (04E3500h)
004E34DA cmp     ah,byte ptr [ecx+1]
004E34DD jne     dodwords+38h (04E3508h)
```

0x465544 4609348

Memoria 1

Dirección: 0x00525A14

0x00525A14 55 2e 18 3f 2e 62 57 65 00 00 00 62 63 66 31 38 38 34 61 2d 63 66 64 64 2d 34 35 32 30 2d 38 64 66 66 2d 35 36 64 33 34 34 37 65 63 64 65 33 00 00 00 63 38 37 38 04 01 57 30 2d 32 35 U.8?.bwe

0x00525A53 63 38 37 34 32 33 66 2d 39 66 63 35 2d 39 33 36 63 37 34 64 35 31 38 65 39 00 00 00 00 65 39 32 33 65 63 32 33 2d 37 31 66 39 2d 34 66 64 39 2d 61 39 64 34 2d 39 64 61 33 34 38 36 62 38 39 c8-423f-9fcs-936c

0x00525A92 43 66 00 00 00 66 61 6c 73 65 00 00 74 72 75 65 00 00 43 00 3a 00 5c 00 50 00 72 00 6f 00 67 00 72 00 61 00 6d 00 20 00 46 00 69 00 6c 00 65 00 73 00 20 00 28 00 78 00 38 00 36 cf...false...tru

0x00525AD1 00 29 00 5c 00 4d 00 69 00 63 00 72 00 6f 00 66 00 74 00 28 00 55 00 69 00 73 00 75 00 61 00 6c 00 20 00 53 00 74 00 75 00 64 00 69 00 6f 00 5c 00 32 00 30 00 31 00 37 00 5c 00 ).\N.i.c.r.o.s.t

0x00525B10 43 00 6f 00 6d 00 6d 00 75 00 6e 00 69 00 74 00 79 00 5c 00 56 00 43 00 5c 00 54 00 6f 00 6f 00 6c 00 73 00 5c 00 4d 00 53 00 56 00 43 00 5c 00 31 00 34 00 2e 00 31 00 32 00 2e 00 32 00 35 C.o.m.m.u.n.i.t.y

0x00525B4F 00 38 00 5c 00 37 00 5c 00 69 00 6e 00 63 00 6c 00 75 00 64 00 65 00 5c 00 78 00 73 00 74 00 72 00 69 00 6e 00 67 00 20 00 73 00 75 00 62 00 73 00 .8.2.7.\i.n.c.i.i

Desensamblado de X main.cpp

Dirección: Stage1(void)

Opciones de visualización

```

00465519 mov     dword ptr [ebp-4],3E8h
00465520 push    0
00465522 push    3E8h
00465527 lea     eax,[ebp-3ECh]
0046552D push    eax
0046552E mov     ecx,offset std::cin (0556BB0h)
00465533 call   std::basic_istream<char,std::char_traits<char> >::getline (0469C60h)
00465538 push    525A14h
0046553D lea     ecx,[ebp-3ECh]
00465543 push    ecx
00465544 call   strcmp (04E34C0h)
00465549 add     esp,8
0046554C test    eax,eax
0046554E je     Stage1+4Ch (046555Ch)
00465550 push    1
00465552 call   Explode (0465450h)
00465557 add     esp,4
0046555A jmp     Stage1+5Bh (046556Bh)

```

Podemos observar claramente que la cadena localizada en la posición “525A14h” es “U.8?.bwe”, nuestra contraseña para la primera bomba.

```

ffset std::cin (0556BB0h)
basic_istream<char, std::char_traits<char> >::getline
4h
ebp-3ECh
p (04E34C0h)
ax
1+4Ch (046555Ch)
de (046556Bh)

```

U.8?.bwe  
Stage 1 disabled

Lo introducimos en la consola y vemos que, efectivamente, hemos conseguido la contraseña correcta y desactivado la bomba de manera satisfactoria.

## Bomba 2

La segunda bomba es algo más complicado de resolver. Lo primero de todo, observamos que hay un bucle, que ocurre tres veces:

```

00465577 mov     dword ptr [ebp-0Ch],3
0046557E mov     dword ptr [ebp-4],0
00465585 jmp     Stage2+20h (0465590h)
00465587 mov     eax,dword ptr [ebp-4]
0046558A add     eax,1
0046558D mov     dword ptr [ebp-4],eax
00465590 cmp     dword ptr [ebp-4],3
00465594 jge     Stage2+3Ah (04655AAh)
00465596 mov     ecx,dword ptr [ebp-4]
00465599 lea     edx,[ebp+ecx*4-18h]
0046559D push    edx
0046559E mov     ecx,offset std::cin (0556BB0h)
004655A3 call   std::basic_istream<char,std::char_traits<char> >::ope
004655A8 jmp     Stage2+17h (0465587h)
004655AA mov     dword ptr [ebp-8],1
004655B1 lea     ebx,[ebp-18h]
004655B4 mov     eax,dword ptr [ebx+8]
004655B7 add     eax,dword ptr [ebx]

```

En este bucle, se leen tres números enteros de consola y se almacenan en un vector.  
Posteriormente, después de terminar de leer, se mueve la posición inicial del vector a "ebx":

```
004655B1 lea ebx,[ebp-18h]
```

Después, se mueven el primer y el tercer número del vector de la entrada a "eax":

```
004655B4 mov eax,dword ptr [ebx+8]
004655B7 add eax,dword ptr [ebx]
```

Por último se comprueba si ese número es igual o no a -6 (0xFFFFFFFFh). Si lo es, se salta a la desactivación. En la siguiente foto, se destaca en rojo el camino que sigue si eax es igual a 6, y en negro si no lo es.

```
004655B9 cmp eax,0FFFFFFFFh
004655BC jne X Stage2+55h (04655C5h)
004655BE mov dword ptr [ebp-8],0
004655C5 cmp dword ptr [ebp-8],0
004655C9 je V Stage2+67h (04655D7h)
004655CB push 2
004655CD call Explode (0465450h) X
004655D2 add esp,4
004655D5 jmp Stage2+76h (04655E6h)
004655D7 push 525A48h
004655DC push 2
004655DE call Defuse (04654C0h) V
```

The diagram illustrates the execution flow of the assembly code. A red path starts at the 'jne' instruction (004655BC), goes down to the 'je' instruction (004655C9), then continues through the 'push' (004655CB), 'call' (004655CD), 'add' (004655D2), 'jmp' (004655D5), 'push' (004655D7), 'push' (004655DC), and finally 'call' (004655DE) instructions, ending with a red checkmark. A black path starts at the 'jne' instruction (004655BC), goes right to 'Stage2+55h', then down to 'Stage2+67h', then right to 'Explode (0465450h)', and finally down to 'Stage2+76h'. Red 'X' marks are placed over the 'jne' and 'call Explode' instructions, while black checkmarks are placed over the 'je' and 'call Defuse' instructions.

Con todo esto, se concluye que una entrada válida para desactivar esta bomba es aquella en la que el primer y último número (de tres) deben sumar -6. El valor del segundo número es irrelevante.

```
.basic_string

The screenshot shows a debugger window with a memory dump. The address range is from 004655B1 to 004655DE. The dump shows the following data: 'U.8?.bwe', 'Stage 1 disabled', 'dword ptr [ebp-8], -3', 'Stage 2 disabled', '0FFFFFFFFh', 'e2+55h (04655C5h)', 'd ptr [ebp-8], 0', 'd ptr [ebp-8], 0', 'e2+67h (04655D7h)', 'd ptr [ebp-8], 0', 'e2+76h (04655E6h)'. The window title is 'G:\OneDrive - Universidad de Oviedo\Univ\YTZ\FCR\HackingForce\secondPhase\main.exe'.


```

Lo introducimos en la consola y vemos que, efectivamente, hemos conseguido la contraseña correcta y desactivado la bomba de manera satisfactoria.

### Bomba 3

En la tercera y última bomba, se trabaja con máscaras y desplazamiento de bits, al igual que durante la fase de entrenamiento. Para resumir el proceso del procedimiento, existen dos condiciones que deben de cumplir los DOS números introducidos para que la bomba se desactive:

- El bit 29 del primero de ellos debe de ser 0.
- El bit 9 del primero y el 17 de segundo deben de ser diferentes.

La explicación de estos resultados es un tanto extensa y es mejor empezar por abajo, es decir, decidiendo cuándo se activa o desactiva la bomba:

Si no se cumple alguna de las dos condiciones, obviamente la bomba “estalla”. En el código, las dos condiciones siguen una cadena lógica: primero se evalúa una y luego la otra:

```
00465635 sar      ecx,1Dh
00465638 mov      dword ptr [ebp-14h],ecx
0046563B mov      edx,dword ptr [ebp-0Ch]
0046563E cmp      edx,dword ptr [ebp-10h]
00465641 je       Stage3+59h (0465649h)
00465643 cmp      dword ptr [ebp-14h],1
00465647 jne     Stage3+65h (0465655h) ►
00465649 push     3
0046564B call    Explode (0465450h)
00465650 add      esp,4
00465653 jmp     Stage3+74h (0465664h)
00465655 push     525A70h
0046565A push     3
0046565C call    Defuse (04654C0h)
```

En la fotografía, **el primer jump** NO se debe cumplir, mientras que **el segundo** Sí. Así, el programa llega a la instrucción “00465655”, con lo que se termina desactivando la bomba.

Para que ocurra esto, se ha de cumplir lo siguiente:

$$\begin{array}{lcl} \text{ebp}-0\text{ch} & \neq & \text{ebp}-10\text{h} \\ \text{ebp}-14\text{h} & \neq & 1 \end{array}$$

Para resolver el valor de estas direcciones de memoria, solo hay que subir un poco en el código:



```

0046560F  mov     edx,dword ptr [ebp-8]
00465612  and     edx,20000h
00465618  sar     edx,11h
0046561B  mov     dword ptr [ebp-0Ch],edx
0046561E  mov     eax,dword ptr [ebp-4]
00465621  and     eax,200h
00465626  sar     eax,9
00465629  mov     dword ptr [ebp-10h],eax
0046562C  mov     ecx,dword ptr [ebp-4]
0046562F  and     ecx,20000000h
00465635  sar     ecx,1Dh
00465638  mov     dword ptr [ebp-14h],ecx

```

Gracias a estas instrucciones, podemos transformar lo que teníamos antes a lo siguiente:

*edx*  $\neq$  *eax*  
*ecx*  $\neq$  1

Ahora, solo tenemos que buscar dónde obtienen los registros sus valores, algo bastante sencillo:

0046560F	mov	edx,dword ptr [ebp-8]
00465612	and	edx,20000h
00465618	sar	edx,11h
0046561B	mov	dword ptr [ebp-0Ch],edx
0046561E	mov	eax,dword ptr [ebp-4]
00465621	and	eax,200h
00465626	sar	eax,9
00465629	mov	dword ptr [ebp-10h],eax
0046562C	mov	ecx,dword ptr [ebp-4]
0046562F	and	ecx,20000000h
00465635	sar	ecx,1Dh
00465638	mov	dword ptr [ebp-14h],ecx

De la anterior captura, se obtiene que:

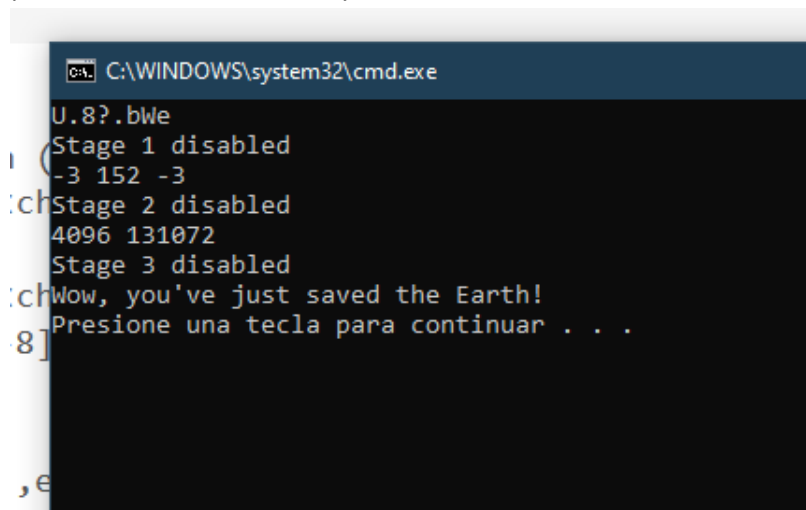
- Resaltado en azul, *edx* es la máscara que se le aplica al segundo número en su bit 17, el cual se mueve posteriormente 17 posiciones a la derecha quedando en un solo dígito, ya sea 1 o 0.

- Resaltado en **rojo**, eax es la máscara que se le aplica al primer número en su bit 9, que de igual manera que antes se mueve hasta dejar un solo dígito.
- Resaltado en **amarillo**, ecx, es la cadena más grande, que se le aplica al primer número de nuevo y mueve su bit 29 (0x1Dh) y lo deja en un solo número.

Ahora, podemos traducir los registros que teníamos antes a algo más legible:

bit n° 17 del 2º n°m ≠ bit n°9 del 1º n°m.  
bit n° 23 del 1º n°m ≠ 1

A partir de esto, podemos fabricar números a medida de manera bastante sencilla, para probar nuestros resultados y desactivar la bomba final.



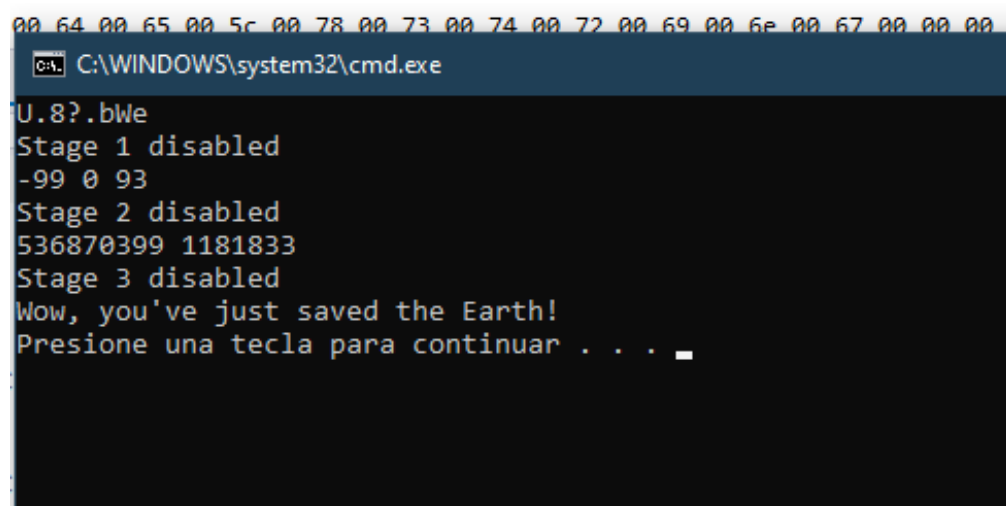
```

C:\WINDOWS\system32\cmd.exe
U.8?.bWe
Stage 1 disabled
-3 152 -3
Stage 2 disabled
4096 131072
Stage 3 disabled
Wow, you've just saved the Earth!
Presione una tecla para continuar . . .

```

En este resultado, hemos utilizado el 4096 como primer número. Obviamente, su bit 23 no es uno, y su bit 9 tampoco lo es. En cambio, el bit 17 del segundo número sí que es uno, con lo que se cumplen ambas condiciones y la bomba se desactiva.

También podemos probar otros resultados que sigan cumpliendo ambas condiciones, como por ejemplo:



```

00 64 00 65 00 5c 00 78 00 73 00 74 00 72 00 69 00 6e 00 67 00 00 00
C:\WINDOWS\system32\cmd.exe
U.8?.bWe
Stage 1 disabled
-99 0 93
Stage 2 disabled
536870399 1181833
Stage 3 disabled
Wow, you've just saved the Earth!
Presione una tecla para continuar . . .

```

## Modificación del ejecutable

En esta segunda parte, lo que se pedía era modificar el ejecutable en sí para que, sea cual sea la contraseña introducida, el programa desactive la bomba y así “salvar el mundo”.

Para ello, se utiliza el programa “HxD”, que modifica los valores hexadecimales del código.

Para saber qué valores modificar, primero tenemos que fijarnos en lo que hace el programa y cuándo salta al explotar y reemplazar dichas instrucciones.

Para las dos primeras bombas, el proceso es similar: reemplazar el jump condicional que lleva al defuse cuando la contraseña sea correcta y modificarlo por un jump incondicional que siempre lleve ahí:

```

Dirección: Stage1(void)
▼ Opciones de visualización
00465599 8D 54 8D E8      lea     edx,[ebp+ecx*4-18h]
0046559D 52              push   edx
0046559E B9 B0 6B 55 00   mov     ecx,offset std::cin (0556B80h)
004655A3 E8 C8 0F 00 00   call   std::basic_istream<char,std::char_traits<char> >::operator>> (0466570h)
004655A8 EB DD          jmp     Stage2+17h (0465587h)
004655AA C7 45 F8 01 00 00 mov     dword ptr [ebp-8],1
004655B1 8D 5D E8      lea     ebx,[ebp-18h]
004655B4 8B 43 08      mov     eax,dword ptr [ebx+8]
004655B7 03 03      add     eax,dword ptr [ebx]
004655B9 83 F8 FA      cmp     eax,0FFFFFFFAh
004655BC 75 07      jne     Stage2+55h (04655C5h)
004655BE C7 45 F8 00 00 00 mov     dword ptr [ebp-8],0
004655C5 83 7D F8 00   cmp     dword ptr [ebp-8],0
004655C9 74 0C      je      Stage2+67h (04655D7h)
004655CB 6A 02      push   2
004655CD E8 7E FE FF FF   call   Explode (0465450h)
004655D2 83 C4 04      add     esp,4
004655D5 EB 0F      jmp     Stage2+76h (04655E6h)
004655D7 68 48 5A 52 00 push   525A48h

```

Observamos los valores en hexadecimal de las instrucciones cercanas y buscamos la instrucción en HxD:

00004880	FF	FF	FF	52	8B	45	FC	50	E8	C3	4E	00	00	83	C4	08	ÿÿÿ<EufEÄN...fÄ.
00004890	68	20	21	46	00	68	CC	59	52	00	68	98	6A	55	00	E8	h f.hYR.h"jUj
000048A0	FC	C3	FF	FF	83	C4	08	8B	C8	E8	02	DE	FF	FF	6A	00	uÄÿÿfÄ.<ÈÈ.bÿÿj.
000048B0	E8	8B	14	05	00	8B	E5	5D	C3	CC	CC	CC	CC	CC	CC	CC	Ü...äJÄiüüüüü
000048C0	55	8B	EC	81	EC	CC	00	00	C7	45	FC	F0	59	52	00	00	E<i.i.i...ÇEüYR
000048D0	8B	45	0C	50	8B	4D	08	51	68	FC	59	52	00	68	C8	00	<E.P<M.QhüYR.hÈ.
000048E0	00	00	8D	95	34	FF	FF	FF	52	E8	02	4E	00	00	83	C4	...4ÿÿÿRÈ.N..fÄ
000048F0	14	8D	85	34	FF	FF	FF	50	8B	4D	FC	51	E8	4F	4E	00	...4ÿÿÿP<MüQèÖN
00004900	00	83	C4	08	8B	E5	5D	C3	CC	CC	CC	CC	CC	CC	CC	CC	fÄ.<JÄiüüüüüüü
00004910	55	8B	EC	81	EC	EC	03	00	00	C7	45	FC	E8	03	00	00	Ü<i.i.i...ÇEüÈ...
00004920	6A	00	68	08	03	00	00	8D	85	14	FC	FF	FF	50	B9	B0	j.hÈ.....üÿÿP<
00004930	6B	55	00	E8	28	47	00	00	68	14	5A	52	00	8D	8D	14	kUÈ.(G.h.ZR...<
00004940	FC	FF	FF	51	E8	77	DF	07	00	83	C4	08	85	C0	EB	0C	üÿÿQèWb...fÄ...ÄÈ.
00004950	6A	01	E8	F9	FE	FF	FF	83	C4	04	EB	0F	68	20	5A	52	j.eüÿÿÿfÄ.e.h ZR
00004960	00	6A	01	E8	58	FF	FF	FF	83	C4	08	8B	E5	5D	C3	CC	.j.èÿÿÿÿfÄ.<JÄi
00004970	55	8B	EC	83	EC	18	53	C7	45	F4	03	00	00	00	C7	45	Ü<i.fYi.ÇEÖ...ÇE
00004980	FC	00	00	00	EB	09	8B	45	FC	83	C0	01	89	45	FC	00	ü...è.<EufÄ.hEü
00004990	83	7D	FC	03	7D	14	8B	4D	FC	8D	54	8D	E8	52	B9	B0	f}U}...<Mu.T.ÈR?
000049A0	6B	55	00	E8	C8	0F	00	00	EB	14	C7	45	F8	01	00	00	kU.EÈ...èYÇE...
000049B0	00	8D	5D	E8	8B	43	08	03	03	83	F8	FA	75	07	C7	45	...jè<C...fouüÇE
000049C0	F8	00	00	00	00	83	7D	F8	00	74	0C	62	02	E8	7E	FE	ø....fjè.Çj.è<b
000049D0	FF	FF	83	C4	04	EB	0F	68	48	52	00	6A	02	E8	ED	DD	ÿÿfÄ.e.hHZR.j.èY
000049E0	FE	FF	FF	83	C4	08	5B	8B	E5	5D	C3	CC	CC	CC	CC	CC	bÿÿfÄ.<JÄiüüüüü
000049F0	55	8B	EC	83	EC	14	8D	45	F8	50	8D	4D	FC	51	B9	B0	Ü<i.f.i...ÈøP.MüQ?
00004A00	6B	55	00	E8	68	0F	00	00	8B	C8	E8	61	0F	00	00	8B	kU.èh...<Èèa...<
00004A10	55	F8	81	E2	00	00	02	00	C1	FA	11	89	55	F4	8B	45	Üø.Ä...Äü.hEÜ<E
00004A20	FC	25	00	02	00	00	C1	F8	09	89	45	F0	8B	4D	FC	81	ü...Äø.hEÜ<Mü.
00004A30	E1	00	00	00	20	C1	F9	1D	89	4D	EC	8B	55	F4	3B	55	ä... Äü.hMü<Üø;
00004A40	F0	74	06	83	7D	EC	01	75	0C	6A	03	E8	00	FE			

Finalmente, reemplazamos el valor por lo que queremos:

```

00004900 00 83 C4 08 8B E5 5D C3 CC CC CC CC CC CC CC .JA.<ajAiiiiiii
00004910 55 8B EC 81 EC EC 03 00 00 C7 45 FC E8 03 00 00 U<ì.ìì...ÇEüè...
00004920 6A 00 68 E8 03 00 00 8D 85 14 FC FF FF 50 B9 B0 j.hè.....üÿÿP¹°
00004930 6B 55 00 E8 28 47 00 00 68 14 5A 52 00 8D 8D 14 kU.è(G..h.ZR....
00004940 FC FF FF 51 E8 77 DF 07 00 83 C4 08 85 C0 EB 0C üÿÿQèwB...fÄ...Äè.
00004950 6A 01 E8 F9 FE FF FF 83 C4 04 EB 0F 68 20 5A 52 j.èüÿÿÿfÄ.è.h ZR
00004960 00 6A 01 E8 58 FF FF FF 83 C4 08 8B E5 5D C3 CC .j.èXÿÿÿfÄ.<âjÄî
00004970 55 8B EC 83 EC 18 53 C7 45 F4 03 00 00 00 C7 45 U<ifi.SÇEó....ÇE
00004980 FC 00 00 00 00 EB 09 8B 45 FC 83 C0 01 89 45 FC ü....è.<EüfÄ.Eü
00004990 83 7D FC 03 7D 14 8B 4D FC 8D 54 8D E8 52 B9 B0 f)ü.}.<Mü.T.èR¹°
000049A0 6B 55 00 E8 C8 0F 00 00 EB DD C7 45 F8 01 00 00 kU.èÈ...ëÝÇEø...
000049B0 00 8D 5D E8 8B 43 08 03 03 83 F8 FA 75 07 C7 45 ..jè<C...føúu.ÇE
000049C0 F8 00 00 00 00 83 7D F8 00 EB 0C 6A 02 E8 7E FE ø....f}ø.È.j.è~p
000049D0 FF FF 83 C4 04 EB 0F 68 10 5A 52 00 6A 02 E8 DD ÿÿfÄ.è.hHZR.j.èÝ
000049E0 FE FF FF 83 C4 08 5B 8B E5 5D C3 CC CC CC CC CC bÿÿfÄ. [<âjÄiiiiî
000049F0 55 8B EC 83 EC 14 8D 45 F8 50 8D 4D FC 51 B9 B0 U<ifi..EøP.MüQ¹°
00004A00 6B 55 00 E8 68 0F 00 00 8B C8 E8 61 0F 00 00 8B kU.èh...<Èèa...<
00004A10 55 F8 81 E2 00 00 02 00 C1 FA 11 89 55 F4 8B 45 Uø.â....Áú.Uø<E
00004A20 FC 25 00 02 00 00 C1 F8 09 89 45 F0 8B 4D FC 81 ü%....Áø.Eø<Mü.
00004A30 E1 00 00 00 20 C1 F9 1D 89 4D EC 8B 55 F4 3B 55 á... Áú.Mi<Uø;U
00004A40 F0 74 06 83 7D EC 01 75 0C 6A 03 E8 00 FE FF FF øt.f}ì.u.j.è.bÿÿ
00004A50 83 C4 04 EB 0F 68 70 5A 52 00 6A 03 E8 5F FE FF fÄ.è.hpZR.j.è bÿ
00004A60 FF 83 C4 08 8B E5 5D C3 CC CC CC CC CC CC CC ÿfÄ.<âjÄiiiiiiiî
00004A70 55 8B EC 8B 45 0C 50 8B 4D 08 E8 F1 44 00 00 5D U<ì<E.P<M.èfD..]
00004A80 C3 CC CC CC CC CC CC CC CC CC CC CC CC CC Äiiiiiiiiiiiiiiiî

```

Igual con la segunda bomba, primero buscamos:

```

Operaciones de ensamblador
346561B 89 55 F4      mov     dword ptr [ebp-0Ch],edx
346561E 8B 45 FC      mov     eax,dword ptr [ebp-4]
3465621 25 00 02 00 00 and     eax,200h
3465626 C1 F8 09      sar     eax,9
3465629 89 45 F0      mov     dword ptr [ebp-10h],eax
346562C 8B 4D FC      mov     ecx,dword ptr [ebp-4]
346562F 81 E1 00 00 00 20 and     ecx,20000000h
3465635 C1 F9 1D      sar     ecx,1Dh
3465638 89 4D EC      mov     dword ptr [ebp-14h],ecx
346563B 8B 55 F4      mov     edx,dword ptr [ebp-0Ch]
346563E 3B 55 F0      cmp     edx,dword ptr [ebp-10h]
3465641 74 06      je      Stage3+59h (0465649h)
3465643 83 7D EC 01    cmp     dword ptr [ebp-14h],1
3465647 75 0C      jne     Stage3+65h (0465655h)
3465649 6A 03      push    3
346564B E8 00 FE FF FF call    Explode (0465450h)
3465650 83 C4 04      add     esp,4
3465653 EB 0F      jmp     Stage3+74h (0465664h)
3465655 68 70 5A 52 00 push    525A70h

```

Luego encontramos:

```

0004870 34 FF FF FF 51 E8 76 4E 00 00 83 C4 10 8D 95 34 4ÿÿÿQèVN..fÄ...4
0004880 FF FF FF 52 8B 45 FC 50 E8 C3 4E 00 00 83 C4 08 ÿÿÿR<EüPeÄN..fÄ.
0004890 68 20 21 46 00 68 CC 59 52 00 68 98 6A 55 00 E8 h !F.hIYR.h~jU.è
00048A0 FC C3 FF FF 83 C4 08 8B C8 E8 02 DE FF FF 6A 00 üÄÿÿfÄ.<Èè.Bÿÿj.
00048B0 E8 8B 14 05 00 8B E5 5D C3 CC CC CC CC CC CC CC è<...<ÄjÄiïiïiïi
00048C0 55 8B EC 81 EC CC 00 00 00 C7 45 FC F0 59 52 00 U<i.iï...ÇEüöYR.
00048D0 8B 45 0C 50 8B 4D 08 51 68 FC 59 52 00 68 C8 00 <E.P<M.QhüYR.hÈ.
00048E0 00 00 8D 95 34 FF FF FF 52 E8 02 4E 00 00 83 C4 ...4ÿÿÿRè.N..fÄ
00048F0 14 8D 85 34 FF FF FF 50 8B 4D FC 51 E8 4F 4E 00 ...4ÿÿÿP<MüQèON.
0004900 00 83 C4 08 8B E5 5D C3 CC CC CC CC CC CC CC CC .fÄ.<ÄjÄiïiïiïiïi
0004910 55 8B EC 81 EC EC 03 00 00 C7 45 FC E8 03 00 00 U<i.iï...ÇEüè...
0004920 6A 00 68 E8 03 00 00 8D 85 14 FC FF FF 50 B9 B0 j.hè.....üÿÿP°
0004930 6B 55 00 E8 28 47 00 00 68 14 5A 52 00 8D 8D 14 kU.è(G..h.ZR....
0004940 FC FF FF 51 E8 77 DF 07 00 83 C4 08 85 C0 EB 0C üÿÿQèwB..fÄ...Äè.
0004950 6A 01 E8 F9 FE FF FF 83 C4 04 EB 0F 68 20 5A 52 j.èüÿÿÿfÄ.è.h ZR
0004960 00 6A 01 E8 58 FF FF FF 83 C4 08 8B E5 5D C3 CC .j.èXÿÿÿfÄ.<ÄjÄi
0004970 55 8B EC 83 EC 18 53 C7 45 F4 03 00 00 00 C7 45 U<i.fi..SQEö....ÇE
0004980 FC 00 00 00 00 EB 09 8B 45 FC 83 C0 01 89 45 FC ü....è.<EüfÄ.wEü
0004990 83 7D FC 03 7D 14 8B 4D FC 8D 54 8D E8 52 B9 B0 f)ü.}<Mü.T.èR°
00049A0 6B 55 00 E8 C8 0F 00 00 EB DD C7 45 F8 01 00 00 kU.èÈ...èYÇEø...
00049B0 00 8D 5D E8 8B 43 08 03 03 83 F8 FA 75 07 C7 45 ..jè<C...föüu.ÇE
00049C0 F8 00 00 00 00 83 7D F8 00 EB 0C 6A 02 E8 7E FE ø....f)ø.è.j.è~p
00049D0 FF FF 83 C4 04 EB 0F 68 48 5A 52 00 6A 02 E8 DD ÿÿfÄ.è.hH2R.j.èY
00049E0 FE FF FF 83 C4 08 5B 8B E5 5D C3 CC CC CC CC CC CC pÿÿfÄ.[<ÄjÄiïiïiïi
00049F0 55 8B EC 83 EC 14 8D 45 F8 50 8D 4D FC 51 B9 B0 U<i.fi..EøP.MüQ°
0004A00 6B 55 00 E8 68 0F 00 00 8B C8 E8 61 0F 00 00 8B kU.èh...<Èèa...<
0004A10 55 F8 81 E2 00 00 02 00 C1 FA 11 89 55 F4 8B 45 Uø.ä....Äü.wUö<E
0004A20 FC 25 00 02 00 00 C1 F8 09 89 45 F0 8B 4D FC 81 ü%....Äø.wEö<Mü.
0004A30 E1 00 00 00 20 C1 F9 1D 89 4D EC 8B 55 F4 3B 55 ä... Äü.wMü<Uö;U
0004A40 F0 74 06 83 7D EC 01 75 0C 6A 03 E8 00 FE FF FF ø[f]i.u.j.è.pÿÿ
0004A50 83 74 04 EB 0F 68 70 5A 52 00 6A 03 E8 5F FE FF fÄ.è.hp2R.j.è_pÿ
0004A60 FF 83 C4 08 8B E5 5D C3 CC CC CC CC CC CC CC CC ÿfÄ.<ÄjÄiïiïiïiïi
0004A70 55 8B EC 8B 45 0C 50 8B 4D 08 E8 F1 44 00 00 5D U<i<E.P<M.èÄD..j
0004A80 C3 CC CC CC CC CC CC CC CC CC CC CC CC CC CC Äiïiïiïiïiïiïiïi
0004A90 55 8B EC 51 8B 45 0C 50 8B 4D 08 51 E8 CF FF FF U<iQ<E.P<M.Qèÿÿÿ

```

Y, por último, reemplazamos:

```

00004980 FC 00 00 00 00 EB 09 8B 45 FC 83 C0 01 89 45 FC
00004990 83 7D FC 03 7D 14 8B 4D FC 8D 54 8D E8 52 B9 B0
000049A0 6B 55 00 E8 C8 0F 00 00 EB DD C7 45 F8 01 00 00
000049B0 00 8D 5D E8 8B 43 08 03 03 83 F8 FA 75 07 C7 45
000049C0 F8 00 00 00 00 83 7D F8 00 EB 0C 6A 02 E8 7E FE
000049D0 FF FF 83 C4 04 EB 0F 68 48 5A 52 00 6A 02 E8 DD
000049E0 FE FF FF 83 C4 08 5B 8B E5 5D C3 CC CC CC CC CC
000049F0 55 8B EC 83 EC 14 8D 45 F8 50 8D 4D FC 51 B9 B0
00004A00 6B 55 00 E8 68 0F 00 00 8B C8 E8 61 0F 00 00 8B
00004A10 55 F8 81 E2 00 00 02 00 C1 FA 11 89 55 F4 8B 45
00004A20 FC 25 00 02 00 00 C1 F8 09 89 45 F0 8B 4D FC 81
00004A30 E1 00 00 00 20 C1 F9 1D 89 4D EC 8B 55 F4 3B 55
00004A40 F0 EB 12 83 7D EC 01 75 0C 6A 03 E8 00 FE FF FF
00004A50 83 C4 04 EB 0F 68 70 5A 52 00 6A 03 E8 5F FE FF
00004A60 FF 83 C4 08 8B E5 5D C3 CC CC CC CC CC CC CC CC
00004A70 55 8B EC 8B 45 0C 50 8B 4D 08 E8 F1 44 00 00 5D
00004A80 C3 CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC
00004A90 55 8B EC 51 8B 45 0C 50 8B 4D 08 51 E8 CF FF FF
00004AA0 FF 83 C4 08 0F B6 D0 85 D2 75 09 C7 45 FC 01 00

```

Para la tercera bomba, simplemente hay que cambiar un salto condicional a otro salto incondicional, esta vez variando la cantidad de instrucciones que se saltan para que redirija correctamente a Defuse():



```

0046561B 89 55 F4      mov     dword ptr [ebp-0Ch],edx
0046561E 8B 45 FC      mov     eax,dword ptr [ebp-4]
00465621 25 00 02 00 00 and     eax,200h
00465626 C1 F8 09      sar     eax,9
00465629 89 45 F0      mov     dword ptr [ebp-10h],eax
0046562C 8B 4D FC      mov     ecx,dword ptr [ebp-4]
0046562F 81 E1 00 00 00 20 and     ecx,20000000h
00465635 C1 F9 1D      sar     ecx,1Dh
00465638 89 4D EC      mov     dword ptr [ebp-14h],ecx
0046563B 8B 55 F4      mov     edx,dword ptr [ebp-0Ch]
0046563E 3B 55 F0      cmp     edx,dword ptr [ebp-10h]
00465641 74 06        je      Stage3+59h (0465649h)
00465643 83 7D EC 01   cmp     dword ptr [ebp-14h],1
00465647 75 0C        jne     Stage3+65h (0465655h)
00465649 6A 03        push    3
0046564B E8 00 FE FF FF call    Explode (0465450h)
00465650 83 C4 04      add     esp,4
00465653 EB 0F        jmp     Stage3+74h (0465664h)
00465655 68 70 5A 52 00 push    525A70h

```

100 %

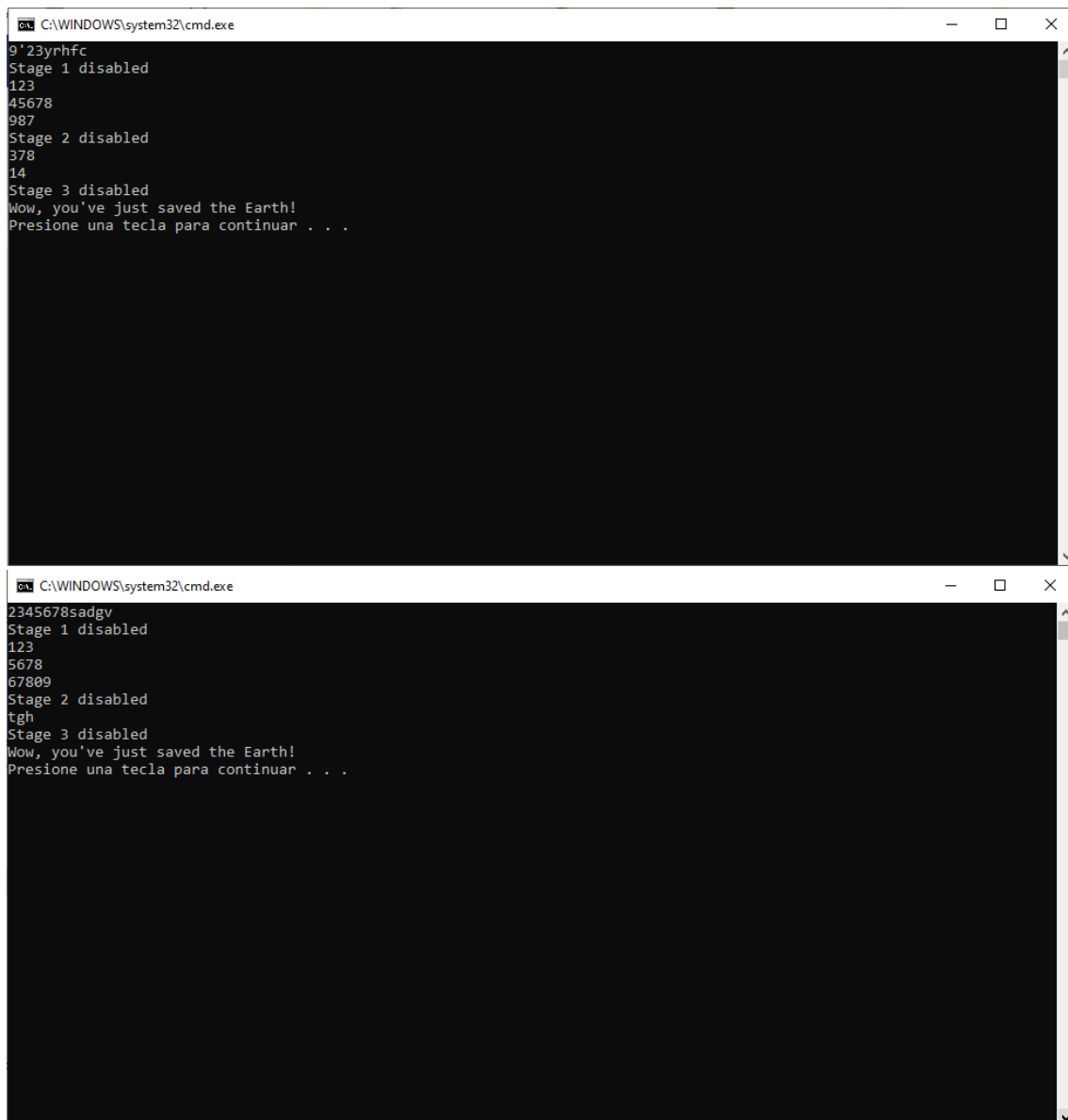
Inspección 1

```

00004960 00 6A 01 E8 58 FF FF FF 83 C4 08 8B E5 5D C3 CC .j.èXÿÿÿfÄ.<â)Äi
00004970 55 8B EC 83 EC 18 53 C7 45 F4 03 00 00 00 C7 45 U<ifi.SÇEô....ÇE
00004980 FC 00 00 00 00 EB 09 8B 45 FC 83 C0 01 89 45 FC ü....ë.<EüfÄ.%Eü
00004990 83 7D FC 03 7D 14 8B 4D FC 8D 54 8D E8 52 B9 B0 f}ü.).<Mü.T.èR¹°
000049A0 6B 55 00 E8 C8 0F 00 00 EB DD C7 45 F8 01 00 00 kU.èÈ...èÝÇEø...
000049B0 00 8D 5D E8 8B 43 08 03 03 83 F8 FA 75 07 C7 45 ..]è<C...föüü.ÇE
000049C0 F8 00 00 00 00 83 7D F8 00 EB 0C 6A 02 E8 7E FE ø....f}ø.è.j.è~p
000049D0 FF FF 83 C4 04 EB 0F 68 48 5A 52 00 6A 02 E8 DD ÿÿfÄ.è.hHZR.j.èÝ
000049E0 FE FF FF 83 C4 08 5B 8B E5 5D C3 CC CC CC CC CC CC pÿÿfÄ.<â)Äiiiiiii
000049F0 55 8B EC 83 EC 14 8D 45 F8 50 8D 4D FC 51 B9 B0 U<ifi..EøP.MüQ¹°
00004A00 6B 55 00 E8 68 0F 00 00 8B C8 E8 61 0F 00 00 8B kU.èh...<Èèa...<
00004A10 55 F8 81 E2 00 00 02 00 C1 FA 11 89 55 F4 8B 45 Uø.á....Áü.%Uô<E
00004A20 FC 25 00 02 00 00 C1 F8 09 89 45 F0 8B 4D FC 81 ü%....Áø.%Eø<Mü.
00004A30 E1 00 00 00 20 C1 F9 1D 89 4D EC 8B 55 F4 3B 55 á... Áü.%Mi<Uô;U
00004A40 F0 EB 12 83 7D EC 01 75 0C 6A 03 E8 00 FE FF FF ø[f]i.u.j.è.pÿÿ
00004A50 83 C4 04 EB 0F 68 70 5A 52 00 6A 03 E8 5F FE FF fÄ.è.hpZR.j.è_pÿ
00004A60 FF 83 C4 08 8B E5 5D C3 CC CC CC CC CC CC CC CC CC pÿfÄ.<â)Äiiiiiii
00004A70 55 8B EC 8B 45 0C 50 8B 4D 08 E8 F1 44 00 00 5D U<i<E.P<M.èñD..]
00004A80 C3 CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC Äiiiiiiiiiiiiiiii
00004A90 55 8B EC 51 8B 45 0C 50 8B 4D 08 51 E8 CF FF FF U<IQ<E.P<M.Qèÿÿÿ
00004AA0 FF 83 C4 08 0F B6 D0 85 D2 75 09 C7 45 FC 01 00 ÿfÄ...ÿD..Ou.ÇEü...

```

Y ahora, ejecutamos el ejecutable modificado introduciendo contraseñas inválidas para comprobar que funciona sin importar la entrada:



```
C:\WINDOWS\system32\cmd.exe
9'23yrhfc
Stage 1 disabled
123
45678
987
Stage 2 disabled
378
14
Stage 3 disabled
Wow, you've just saved the Earth!
Presione una tecla para continuar . . .

C:\WINDOWS\system32\cmd.exe
2345678sadgv
Stage 1 disabled
123
5678
67809
Stage 2 disabled
tgh
Stage 3 disabled
Wow, you've just saved the Earth!
Presione una tecla para continuar . . .
```

Y con esto, hemos salvado el mundo y hemos desactivado toda posible ofensa contra la humanidad.

## Reparto del trabajo

El trabajo se ha repartido de esta manera:

- De manera conjunta, se han resuelto las tres bombas con sus tres correspondientes contraseñas.
- Miguel del Riego: Modificación del ejecutable mediante HxD
- Juan Mier: Memoria y Wireshark.

En total, se han dedicado alrededor de 8 horas por persona al proyecto.