Resolución bomba de Claudia Salado

Alumno: Juan Sánchez Rodríguez

Para resolver esta práctica he usado la herramienta gdb.

Primero hacemos un "break main" y usamos "run" para que el programa avance. Después usamos "disas" para ver todo el programa, nos saldría lo siguiente:

```
=> 0x000000000040075b <+0>: push   %rbx
   0x000000000040075c <+1>:   sub    $0xa0,%rsp
   0x0000000000400763 <+8>:   mov    %fs:0x28,%rax
   0x000000000040076c <+17>: mov    %rax,0x98(%rsp)
   0x0000000000400774 <+25>: xor    %eax,%eax
   0x0000000000400776 <+27>: lea    0x10(%rsp),%rdi
   0x000000000040077b <+32>: mov    $0x0,%esi
   0x0000000000400780 <+37>: callq  0x4005f0 <gettimeofday@plt>
   0x0000000000400785 <+42>: lea    0x2bc(%rip),%rsi        # 0x400a48
   0x000000000040078c <+49>: mov    $0x1,%edi
   0x0000000000400791 <+54>: mov    $0x0,%eax
   0x0000000000400796 <+59>: callq  0x400610 <__printf_chk@plt>
   0x000000000040079b <+64>: lea    0x30(%rsp),%rdi
   0x00000000004007a0 <+69>: mov    0x2008c9(%rip),%rdx        # 0x601070
<stdin@@GLIBC_2.2.5>
   0x00000000004007a7 <+76>: mov    $0x64,%esi
   0x00000000004007ac <+81>: callq  0x400600 <fgets@plt>
   0x00000000004007b1 <+86>: test   %rax,%rax
   0x00000000004007b4 <+89>: je     0x400785 <main+42>
   0x00000000004007b6 <+91>: lea    0x30(%rsp),%rdi
   0x00000000004007bb <+96>: mov    $0xc,%edx
   0x00000000004007c0 <+101>:           lea    0x200899(%rip),%rsi        # 0x601060 <password>
   0x00000000004007c7 <+108>:           callq  0x4005d0 <strncmp@plt>
```

```
   0x00000000004007cc <+113>:     mov    %eax,%ebx
   0x00000000004007ce <+115>:     test   %eax,%eax
   0x00000000004007d0 <+117>:     je     0x4007da <main+127>
   0x00000000004007d2 <+119>:     callq  0x400727 <boom>
   0x00000000004007d7 <+124>:     add    $0x1,%ebx
   0x00000000004007da <+127>:     cmp    $0x9,%ebx
   0x00000000004007dd <+130>:     jle    0x4007d7 <main+124>
   0x00000000004007df <+132>:     lea    0x20(%rsp),%rdi
   0x00000000004007e4 <+137>:     mov    $0x0,%esi
   0x00000000004007e9 <+142>:     callq  0x4005f0 <gettimeofday@plt>
   0x00000000004007ee <+147>:     mov    0x20(%rsp),%rax
   0x00000000004007f3 <+152>:     sub    0x10(%rsp),%rax
   0x00000000004007f8 <+157>:     cmp    $0x5,%rax
   0x00000000004007fc <+161>:     jg     0x400806 <main+171>
   0x00000000004007fe <+163>:     mov    %ebx,0x200878(%rip)        # 0x60107c
<passcode>
   0x0000000000400804 <+169>:     jmp    0x400810 <main+181>
   0x0000000000400806 <+171>:     callq  0x400727 <boom>
   0x000000000040080b <+176>:     cmp    $0x1,%ebx
   0x000000000040080e <+179>:     je     0x400855 <main+250>
   0x0000000000400810 <+181>:     lea    0x24f(%rip),%rsi        # 0x400a66
---Type <return> to continue, or q <return> to quit---
   0x0000000000400817 <+188>:     mov    $0x1,%edi
   0x000000000040081c <+193>:     mov    $0x0,%eax
   0x0000000000400821 <+198>:     callq  0x400610 <__printf_chk@plt>
   0x0000000000400826 <+203>:     lea    0xc(%rsp),%rsi
   0x000000000040082b <+208>:     lea    0x248(%rip),%rdi        # 0x400a7a
   0x0000000000400832 <+215>:     mov    $0x0,%eax
   0x0000000000400837 <+220>:     callq  0x400620 <__isoc99_scanf@plt>
   0x000000000040083c <+225>:     mov    %eax,%ebx
   0x000000000040083e <+227>:     test   %eax,%eax
```

```
   0x0000000000400840 <+229>:           jne    0x40080b <main+176>
   0x0000000000400842 <+231>:           lea    0x234(%rip),%rdi       # 0x400a7d
   0x0000000000400849 <+238>:           mov    $0x0,%eax
   0x000000000040084e <+243>:           callq  0x400620 <__isoc99_scanf@plt>
   0x0000000000400853 <+248>:           jmp    0x40080b <main+176>
   0x0000000000400855 <+250>:           mov    0x200821(%rip),%eax       # 0x60107c
<passcode>
   0x000000000040085b <+256>:           cmp    %eax,0xc(%rsp)
   0x000000000040085f <+260>:           je     0x400866 <main+267>
   0x0000000000400861 <+262>:           callq  0x400727 <boom>
   0x0000000000400866 <+267>:           lea    0x10(%rsp),%rdi
   0x000000000040086b <+272>:           mov    $0x0,%esi
   0x0000000000400870 <+277>:           callq  0x4005f0 <gettimeofday@plt>
   0x0000000000400875 <+282>:           mov    0x10(%rsp),%rax
---Type <return> to continue, or q <return> to quit---
   0x000000000040087a <+287>:           sub    0x20(%rsp),%rax
   0x000000000040087f <+292>:           cmp    $0x5,%rax
   0x0000000000400883 <+296>:           jle    0x40088a <main+303>
   0x0000000000400885 <+298>:           callq  0x400727 <boom>
   0x000000000040088a <+303>:           callq  0x400741 <defused>
```

Nos fijamos en la siguiente parte:

```
   0x00000000004007ac <+81>: callq  0x400600 <fgets@plt>
   0x00000000004007b1 <+86>: test   %rax,%rax
   0x00000000004007b4 <+89>: je     0x400785 <main+42>
   0x00000000004007b6 <+91>: lea    0x30(%rsp),%rdi
   0x00000000004007bb <+96>: mov    $0xc,%edx
   0x00000000004007c0 <+101>:           lea    0x200899(%rip),%rsi       # 0x601060 <password>
   0x00000000004007c7 <+108>:           callq  0x4005d0 <strncmp@plt>
```

Podemos ver que ahí es cuando se introduce la contraseña, que está almacenada en "0x601060". Usamos "print (char*) 0x601060" para ver la contraseña almacenada y nos responde con "$1 = 0x601060 <password> "diezletras\n"". Si intentamos probar con esta contraseña podemos ver que efectivamente es correcta.

Para conseguir el pin borramos el break que hemos puesto antes usando "d 1" y nos fijamos en el código:

```
  0x000000000040084e <+243>:    callq  0x400620 <__isoc99_scanf@plt>
  0x0000000000400853 <+248>:    jmp    0x40080b <main+176>
  0x0000000000400855 <+250>:    mov    0x200821(%rip),%eax        # 0x60107c
<passcode>
  0x000000000040085b <+256>:    cmp    %eax,0xc(%rsp)
  0x000000000040085f <+260>:    je     0x400866 <main+267>
  0x0000000000400861 <+262>:    callq  0x400727 <boom>
```

Y creamos un breakpoint en "0x0000000000400861 <+262>: callq 0x400727 <boom>" usando "br *main+262" y volvemos a usar "run" introducimos la contraseña obtenida anteriormente y probamos con cualquier pin, si nos fijamos en el código:

```
  0x0000000000400855 <+250>:    mov    0x200821(%rip),%eax        # 0x60107c
<passcode>
  0x000000000040085b <+256>:    cmp    %eax,0xc(%rsp)
```

Podemos ver que el pin se encuentra almacenado en "0x60107c". Usamos "x/d 0x60107c" y nos responde con "0x60107c <passcode>: 10". Si intentamos probar con el pin "10" vemos que la bomba ha sido desactivada.