# Estructura de Computadores: Práctica 3 (Bomba)

**Patricia Córdoba Hidalgo**

## Cómo desactivar mi bomba (PCH_bomb):

### Cómo desactivarla sin saber la contraseña con gdb:

Primero, abrimos 2 terminales en el directorio donde tengamos la bomba. En una de ellas se ejecuta la bomba con `gdb` (cuyo ejecutable se habrá creado con la orden `gcc -m32 PCH_bomb.c -o PCH_bomb`. En la otra, usamos la orden `objdump -d PCH_bomb` para desensamblar el ejecutable y poder ver las instrucciones que el programa realiza.

En la terminal del `objdump` se busca dónde empieza el , en mi caso, en la dirección `0x0804868b`. Una vez en el main, buscamos la instrucción `call 80484f0 <strncmp@plt>`, que se encarga de comparar el string que le pasamos como contraseña con la verdadera contraseña del programa. Un poco más abajo debe estar la instrucción `test %eax,%eax`, en la dirección `0x080487a1`, clave para que no explote la bomba, ya que si no da 0, se llama a la función `boom()` y la bomba explota. Es por esto que debemos poner un `breakpoint` en la dirección de dicha instrucción, en la terminal con el `gdb`. El siguiente `breakpoint` se pone en la instrucción `cmp $0x3c,%eax`, en la dirección `0x080487da`, la cual comprueba si el tiempo invertido en escribir la contraseña es superior a 60 segundos, que es el tiempo máximo autorizado. En `%eax` guarda el tiempo invertido, obtenido con la instrucción `call 8048480 <gettimeofday@plt>`. Tras esto, localizamos la instrucción `cmp -0x8c(%ebp),%eax`, en la dirección `0x08048853`, que compara el valor del código intruducido con el válido, y ponemos otro`breakpoint` en esta instrucción. EL último `breakpoint` se coloca en la dirección `0x080487a1`, donde se encuentra la instrucción `cmp $0x3c,%eax`, la cual vuelve a comprobar si el tiempo invertido en escribir el código es superior a 60 segundos.

Una vez introducidos los `breakpoints`, lazamos el programa en el `gdb` con la orden `run`. El programa nos pide la contraseña. Podemos intruducir lo que queramos, ya que el objetivo de éste apartado es desactivar la bomba de todas maneras.

Llegamos al primer `breakpoint`. Una vez aquí hacemos `info reg`, donde vemos que el valor del registro `%eax` es -1 (en mi caso). Con la orden `set $eax=0` cambiamos el valor del registro para poder evitar que se llame a la función `boom()` al haber introducido una contraseña errónea. Continuamos la ejecución del programa con la orden `cont`, que nos llevará directos al segundo `breakpoint`. Una vez aquí, volvemos a cambiar el valor del registro `%eax`, con `set $eax=0`, para evitar que explote la bomba por consumir más tiempo que el estipulado. Seguimos con la ejecución del programa, donde nos pide el código, como antes, no hay necesidad de introducir el correcto. Tras esto, llegamos al tercer `breakpoint`. Con la orden `info reg` podemos observar que en `%eax` se guarda el valor que habíamos introducido. Como vemos en la terminal del código desensamblado, el valor del código correcto debe

estar en `-0x8c(%ebp)`. Usamos la orden `p*(int*)($ebp-140)`, donde 140 se corresponde con `0x8c`. Tras hacer esto, vemos el verdadero código, `4444`. Una vez que lo sabemos, usamos `set $eax=4444` para cambiar el valor que le pasamos por el correcto.

AL llegar al último `breakpoint`, volvemos a cambiar el valor del registro `%eax`, con `set $eax=0`, para evitar que explote la bomba por consumir más tiempo que el estipulado.

Así conseguimos desactivar la bomba.

**Cómo averiguar la contraseña:**

Para obtener la contraseña, volvemos a abrir las dos terminales con en el apartado anterior. Ahora, buscamos en la terminal del `objdump -d PCH_bomb` la intrucción `call 80484f0 <strncmp@plt>`, que, como señalamos previamente, se encarga de comparar el string que le pasamos como contraseña con la verdadera contraseña del programa.

Podemos ver que los argumentos de esa función son `%eax` y `0x804a0a5`. En el registro `%eax` se guarda la contraseña que nosotros introdujimos por pantalla, mientras que `0x804a0a5` es la contraseña válida. Podemos ver cual es con un volcado de memoria: `x/s 0x804a0a5`. Tras ejecutar esto vemos que la contraseña correcta es `t...t...\n`.

Ya hemos obtenido la contraseña de la bomba. Se mostró como averiguar el código, `4444`, en el apartado anterior.

## Cómo desactivar la bomba MaxiBomb:

Buscamos la instrucción `call 80484e0 <strncmp@plt>` para encontrar, un poco más arriba los argumentos de ésta. Es en la dirección `0x080486a5` donde ponemos el primer `breakpoint`.

El segundo `breakpoint` irá en la dirección `0x08048719`, donde está la instrucción `cmp %eax,%edx`.

A continuación, lanzamos el programa con la orden `run`. Introducimos una contraseña cualquiera y el programa sigue hasta llegar al `breakpoint`. Una vez parado, miramos el contenido del registro `%edx` con la orden `x/s $edx`, y visualizamos la contraseña correcta del programa: `0x804a030 <password>: "holaquetal\n"`. Ya hemos obtenido la contraseña, ahora queda sacar el código.

El código lo visualizamos con el segundo `breakpoint`. Una vez llegado a él, volvemos a visualizar el contenido de los registros, con `info reg`, y observamos que en `%edx` se encuentra el código que introdujimos, y es en `%eax` donde se encuentra el código válido, `1998`.

Con esto ya hemos descifrado tanto la contraseña, que es `holaquetal`, como el código, `1998`.