Estructura de Computadores: Práctica 4

Bomba Digital - Pablo Olivares Martínez

Bomba incial: contraseña = BREAKINGBAD y código = 1969

Bomba modificada: contraseña = AAAA y código = 0

En esta práctica nos centraremos en desactivar la bomba digital mediante el uso de depurador GDB y herramientas hexadecimales. Para ello procedamos a la construcción de la bomba. En mi caso, he decidido poner de **contraseña "BREAKINGBAD"** y de **código de acceso "1969"**. Para complicar un poco la extracción de la clave, he creado **dos funciones: passwordCheck() y passcodeCheck()**. Ambas, lo que hacen es, a partir de las variables almacenadas, que son **password = ";K>:DBG@;:="** y **passcode = 7**, transformarlos para obtener nuestra contraseña y nuestro código. El método utilizado ha sido, para la contraseña restarle el passcode a su correspondiente valor ASCII y para el passcode, hacer una serie de operaciones de suma, resta y producto de valores. Dicho esto, he aquí el código en C:

```c
#include <stdio.h>      // para printf()
#include <stdlib.h>     // para exit()
#include <string.h>     // para strncmp()/strlen()
#include <sys/time.h>   // para gettimeofday(), struct timeval

// La contrasenia es BREAKINGBAD, que es ascii -7 es:
char password[] = ";K>:DBG@;:=";
int passcode = 7;


void fail(){
    printf("\n");
    printf("***************\n");
    printf("*** BOOM!!! ***\n");
    printf("***************\n");
    printf("\n");
    exit(-1);
}

void success(){
    printf("\n");
    printf("*************************\n");
    printf("*** bomba desactivada ***\n");
    printf("*************************\n");
    printf("\n");
    exit(0);
}

//Comprobación de la clave
void numberCheck(int number){

    int aux = passcode;
    aux = passcode*13783 - 16*5907;

    if(number != aux)
```

```
        fail();
}

//Combrobacion de la contraseña
void passwordCheck(char *p){

    int c = 0;
    char aux[strlen(password)];

    while(c < strlen(password)){
        aux[c] = p[c] - passcode;
        c++;
    }

    if(strncmp(aux, password, strlen(password)))
        fail();
}

int main(){

#define SIZE 100
    char pass[SIZE];
    int code;

#define TLIM 60
    struct timeval tv1, tv2;    // gettimeofday() secs-usecs

    gettimeofday(&tv1, NULL);

    printf("Introduce la contraseña: ");
    fgets(pass, SIZE, stdin);
    passwordCheck(pass);

    gettimeofday(&tv2, NULL);
    if (tv2.tv_sec - tv1.tv_sec > TLIM)
        fail();

    printf("Introduce el código: ");
    scanf("%i", &code);
    numberCheck(code);

    gettimeofday(&tv1, NULL);
    if (tv1.tv_sec - tv2.tv_sec > TLIM)
        fail();

    success();
}
```

Compilamos la bomba:

```
gcc -Og bomba.c -o bomba -no-pie -fno-guess-branch-probability
```

Ahora, tratemos de averiguar la contraseña y el código. Para ello, comenzaremos usando la herramienta **ltrace** de linux, la cual nos va a permitir ver los valores que compara la función **strcmp**:

```
pablo@laptop:/$ ltrace -i -S ./bomba
```

Introducimos por ejemplo "aaaaaaaa" para ver que sucede:

```
strncmp("ZZZZZZZZ\003\371\371\032\377\177", ";K>:DBG@;:=", 11)  = 31
```

Por tanto, razonando vemos que la función passwordCheck convierte mi valor "aaaaaaaa" en "ZZZZZZZZ". De esta forma nos damos cuenta que la función resta 7 al valor ASCII de cada caracter. Ahora, transformamos ";K>:DBG@;:=" y descubrimos que la contraseña es "BREAKINGBAD".

Una vez hecho esto, accedamos a gdb con la siguiente configuración para hallar el código:

```
0x401190 <deregister_tm_clones>         mov    $0x404088,%eax
                                      |
|   0x401195 <deregister_tm_clones+5>       cmp    $0x404088,%rax
                                        |
|   0x40119b <deregister_tm_clones+11>      je     0x4011b0
<deregister_tm_clones+32>                                       |
|   0x40119d <deregister_tm_clones+13>      mov    $0x0,%eax
                                        |
|   0x4011a2 <deregister_tm_clones+18>      test   %rax,%rax
                                        |
|   0x4011a5 <deregister_tm_clones+21>      je     0x4011b0
<deregister_tm_clones+32>                                       |
|   0x4011a7 <deregister_tm_clones+23>      mov    $0x404088,%edi
                                        |
|   0x4011ac <deregister_tm_clones+28>      jmpq   *%rax
                                        |
|   0x4011ae <deregister_tm_clones+30>      xchg   %ax,%ax
                                        |
|   0x4011b0 <deregister_tm_clones+32>      retq
                                        |
|   0x4011b1 <deregister_tm_clones+33>      data16 nopw %cs:0x0(%rax,%rax,1)
                                        |
|   0x4011bc <deregister_tm_clones+44>      nopl   0x0(%rax)
                                        |
|   0x4011c0 <register_tm_clones>           mov    $0x404088,%esi
                                      |
|   0x4011c5 <register_tm_clones+5>         sub    $0x404088,%rsi
                                      |
|   0x4011cc <register_tm_clones+12>        mov    %rsi,%rax
                                        |
|   0x4011cf <register_tm_clones+15>        shr    $0x3f,%rsi
                                      |
|   0x4011d3 <register_tm_clones+19>        sar    $0x3,%rax
                                        |
|   0x4011d7 <register_tm_clones+23>        add    %rax,%rsi
                                        |
|   0x4011da <register_tm_clones+26>        sar    %rsi
                                      |
|   0x4011dd <register_tm_clones+29>        je     0x4011f0
<register_tm_clones+48>                                         |
|   0x4011df <register_tm_clones+31>        mov    $0x0,%eax
                                        |
```

```
|   0x4011e4 <register_tm_clones+36>        test   %rax,%rax
                                           |
|   0x4011e7 <register_tm_clones+39>        je     0x4011f0
<register_tm_clones+48>                                    |
|   0x4011e9 <register_tm_clones+41>        mov    $0x404088,%edi
                                          |
|   0x4011ee <register_tm_clones+46>        jmpq   *%rax
                                           |
|   0x4011f0 <register_tm_clones+48>        retq
                                           |
|   0x4011f1 <register_tm_clones+49>        data16 nopw %cs:0x0(%rax,%rax,1)
                                           |
|   0x4011fc <register_tm_clones+60>        nopl   0x0(%rax)
                                           |
|   0x401200 <__do_global_dtors_aux>        endbr64
                                          |
|   0x401204 <__do_global_dtors_aux+4>      cmpb   $0x0,0x2e8d(%rip)        #
0x404098 <completed.8060>                  |
|   0x40120b <__do_global_dtors_aux+11>     jne    0x401220
<__do_global_dtors_aux+32>                                 |
|   0x40120d <__do_global_dtors_aux+13>     push   %rbp
                                          |
|   0x40120e <__do_global_dtors_aux+14>     mov    %rsp,%rbp
                                           |
|   0x401211 <__do_global_dtors_aux+17>     callq  0x401190
<deregister_tm_clones>                                     |
|   0x401216 <__do_global_dtors_aux+22>     movb   $0x1,0x2e7b(%rip)        #
0x404098 <completed.8060>                  |
|   0x40121d <__do_global_dtors_aux+29>     pop    %rbp
                                          |
|   0x40121e <__do_global_dtors_aux+30>     retq
                                           |
|   0x40121f <__do_global_dtors_aux+31>     nop
                                          |
|   0x401220 <__do_global_dtors_aux+32>     retq
                                           |
|   0x401221 <__do_global_dtors_aux+33>     data16 nopw %cs:0x0(%rax,%rax,1)
                                           |
|   0x40122c <__do_global_dtors_aux+44>     nopl   0x0(%rax)
                                           |
|   0x401230 <frame_dummy>                  endbr64
                                          |
|   0x401234 <frame_dummy+4>                jmp    0x4011c0
<register_tm_clones>                                       |
|   0x401236 <fail>                         endbr64
                                          |
|   0x40123a <fail+4>                       push   %rax
                                          |
|   0x40123b <fail+5>                       pop    %rax
                                          |
|   0x40123c <fail+6>                       sub    $0x8,%rsp
                                           |
|   0x401240 <fail+10>                      mov    $0xa,%edi
                                           |
|   0x401245 <fail+15>                      callq  0x4010c0 <putchar@plt>
                                          |
|   0x40124a <fail+20>                      lea    0xdcd(%rip),%rdi        #
0x40201e                                   |
```

```
|   0x401251 <fail+27>                    callq  0x4010e0 <puts@plt>
                                                                    |
|   0x401256 <fail+32>                    lea    0xda7(%rip),%rdi        #
0x402004                                                             |
|   0x40125d <fail+39>                    callq  0x4010e0 <puts@plt>
                                                                    |
|   0x401262 <fail+44>                    lea    0xdb5(%rip),%rdi        #
0x40201e                                                             |
|   0x401269 <fail+51>                    callq  0x4010e0 <puts@plt>
                                                                    |
|   0x40126e <fail+56>                    mov    $0xa,%edi
                                                                    |
|   0x401273 <fail+61>                    callq  0x4010c0 <putchar@plt>
                                                                   |
|   0x401278 <fail+66>                    mov    $0xffffffff,%edi
                                                                   |
|   0x40127d <fail+71>                    callq  0x401140 <exit@plt>
                                                                    |
|   0x401282 <success>                    endbr64
                                                                   |
|   0x401286 <success+4>                  push   %rax
                                                                   |
|   0x401287 <success+5>                  pop    %rax
                                                                   |
|   0x401288 <success+6>                  sub    $0x8,%rsp
                                                                    |
|   0x40128c <success+10>           mov    $0xa,%edi
                                                                    |
|   0x401291 <success+15>           callq  0x4010c0 <putchar@plt>
                                                                   |
|   0x401296 <success+20>           lea    0xd77(%rip),%rdi        # 0x402014
                                                                   |
|   0x40129d <success+27>           callq  0x4010e0 <puts@plt>
                                                                    |
|   0x4012a2 <success+32>           lea    0xd85(%rip),%rdi        # 0x40202e
                                                                   |
|   0x4012a9 <success+39>           callq  0x4010e0 <puts@plt>
                                                                    |
|   0x4012ae <success+44>           lea    0xd5f(%rip),%rdi        # 0x402014
                                                                   |
|   0x4012b5 <success+51>           callq  0x4010e0 <puts@plt>
                                                                    |
|   0x4012ba <success+56>           mov    $0xa,%edi
                                                                    |
|   0x4012bf <success+61>           callq  0x4010c0 <putchar@plt>
                                                                   |
|   0x4012c4 <success+66>           mov    $0x0,%edi
                                                                    |
|   0x4012c9 <success+71>           callq  0x401140 <exit@plt>
                                                                    |
|   0x4012ce <numberCheck>          endbr64
                                                                   |
|   0x4012d2 <numberCheck+4>        imul   $0x35d7,0x2d94(%rip),%eax        #
0x404070 <passcode>                                                  |
|   0x4012dc <numberCheck+14>       sub    $0x17130,%eax
                                                                    |
|   0x4012e1 <numberCheck+19>       cmp    %edi,%eax
                                                                    |
```

```
|   0x4012e3 <numberCheck+21>       je     0x4012f3 <numberCheck+37>
|                                          |
|   0x4012e5 <numberCheck+23>       sub    $0x8,%rsp
|                                          |
|   0x4012e9 <numberCheck+27>       mov    $0x0,%eax
|                                          |
|   0x4012ee <numberCheck+32>       callq  0x401236 <fail>
|                                          |
|   0x4012f3 <numberCheck+37>       retq
|                                          |
|   0x4012f4 <passwordCheck>        endbr64
|                                         |
|   0x4012f8 <passwordCheck+4>      push   %rbp
|                                         |
|   0x4012f9 <passwordCheck+5>      mov    %rsp,%rbp
|                                          |
|   0x4012fc <passwordCheck+8>      sub    $0x10,%rsp
|                                         |
|   0x401300 <passwordCheck+12>     mov    %rdi,%r8
|                                         |
|   0x401303 <passwordCheck+15>     mov    %fs:0x28,%rax
|                                          |
|   0x40130c <passwordCheck+24>     mov    %rax,-0x8(%rbp)
|                                          |
|   0x401310 <passwordCheck+28>     xor    %eax,%eax
|                                          |
|   0x401312 <passwordCheck+30>     lea    0x2d5f(%rip),%rdi        # 0x404078
<password>                                 |
|   0x401319 <passwordCheck+37>     mov    $0xffffffffffffffff,%rcx
|                                         |
|   0x401320 <passwordCheck+44>     repnz scas %es:(%rdi),%al
|                                         |
|   0x401322 <passwordCheck+46>     not    %rcx
|                                         |
|   0x401325 <passwordCheck+49>     add    $0xe,%rcx
|                                          |
|   0x401329 <passwordCheck+53>     mov    %rcx,%rax
|                                          |
|   0x40132c <passwordCheck+56>     and    $0xfffffffffffffff0,%rax
|                                         |
|   0x401330 <passwordCheck+60>     and    $0xfffffffffffff000,%rcx
|                                         |
|   0x401337 <passwordCheck+67>     mov    %rsp,%rdx
|                                          |
|   0x40133a <passwordCheck+70>     sub    %rcx,%rdx
|                                          |
|   0x40133d <passwordCheck+73>     cmp    %rdx,%rsp
|                                          |
|   0x401340 <passwordCheck+76>     je     0x401354 <passwordCheck+96>
|                                          |
|   0x401342 <passwordCheck+78>     sub    $0x1000,%rsp
|                                         |
|   0x401349 <passwordCheck+85>     orq    $0x0,0xff8(%rsp)
|                                         |
|   0x401352 <passwordCheck+94>     jmp    0x40133d <passwordCheck+73>
|                                          |
|   0x401354 <passwordCheck+96>     mov    %rax,%rdx
|                                          |
```

```
|   0x401357 <passwordCheck+99>     and    $0xfff,%edx
                                          |
|   0x40135d <passwordCheck+105>    sub    %rdx,%rsp
                                          |
|   0x401360 <passwordCheck+108>    test   %rdx,%rdx
                                          |
|   0x401363 <passwordCheck+111>    je     0x40136b <passwordCheck+119>
                                         |
|   0x401365 <passwordCheck+113>    orq    $0x0,-0x8(%rsp,%rdx,1)
                                         |
|   0x40136b <passwordCheck+119>    mov    %rsp,%r9
                                         |
|   0x40136e <passwordCheck+122>    mov    $0x0,%esi
                                          |
|   0x401373 <passwordCheck+127>    movslq %esi,%r10
                                          |
|   0x401376 <passwordCheck+130>    lea    0x2cfb(%rip),%rdi        # 0x404078
<password>                                |
|   0x40137d <passwordCheck+137>    mov    $0xffffffffffffffff,%rcx
                                         |
|   0x401384 <passwordCheck+144>    mov    $0x0,%eax
                                          |
|   0x401389 <passwordCheck+149>    repnz scas %es:(%rdi),%al
                                         |
|   0x40138b <passwordCheck+151>    not    %rcx
                                         |
|   0x40138e <passwordCheck+154>    lea    -0x1(%rcx),%rdx
                                          |
|   0x401392 <passwordCheck+158>    cmp    %rdx,%r10
                                          |
|   0x401395 <passwordCheck+161>    jb     0x4013b4 <passwordCheck+192>
                                         |
|   0x401397 <passwordCheck+163>    lea    0x2cda(%rip),%rsi        # 0x404078
<password>                                |
|   0x40139e <passwordCheck+170>    mov    %r9,%rdi
                                         |
|   0x4013a1 <passwordCheck+173>    callq  0x4010d0 <strncmp@plt>
                                        |
|   0x4013a6 <passwordCheck+178>    test   %eax,%eax
                                          |
|   0x4013a8 <passwordCheck+180>    je     0x4013cb <passwordCheck+215>
                                         |
|   0x4013aa <passwordCheck+182>    mov    $0x0,%eax
                                          |
|   0x4013af <passwordCheck+187>    callq  0x401236 <fail>
                                          |
|   0x4013b4 <passwordCheck+192>    movzbl (%r8,%r10,1),%eax
                                          |
|   0x4013b9 <passwordCheck+197>    sub    0x2cb1(%rip),%al         # 0x404070
<passcode>                                 |
|   0x4013bf <passwordCheck+203>    movslq %esi,%rdx
                                          |
|   0x4013c2 <passwordCheck+206>    mov    %al,(%r9,%rdx,1)
                                         |
|   0x4013c6 <passwordCheck+210>    add    $0x1,%esi
                                          |
|   0x4013c9 <passwordCheck+213>    jmp    0x401373 <passwordCheck+127>
                                         |
```

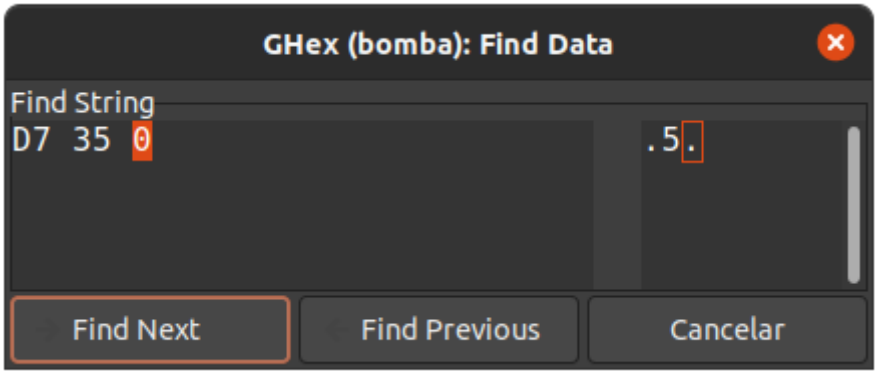
```
|   0x4013cb <passwordCheck+215>    mov    -0x8(%rbp),%rax
                                           |
|   0x4013cf <passwordCheck+219>    xor    %fs:0x28,%rax
                                           |
|   0x4013d8 <passwordCheck+228>    je     0x4013df <passwordCheck+235>
                                          |
|   0x4013da <passwordCheck+230>    callq  0x4010f0 <__stack_chk_fail@plt>
                                           |
|   0x4013df <passwordCheck+235>    leaveq
                                           |
|   0x4013e0 <passwordCheck+236>    retq
                                           |
|   0x4013e1 <main>                 endbr64
                                          |
|   0x4013e5 <main+4>               push   %rbx
                                          |
|   0x4013e6 <main+5>               sub    $0xa0,%rsp
                                          |
|   0x4013ed <main+12>              mov    %fs:0x28,%rax
                                           |
|   0x4013f6 <main+21>              mov    %rax,0x98(%rsp)
                                           |
|   0x4013fe <main+29>              xor    %eax,%eax
                                           |
|   0x401400 <main+31>              lea    0x10(%rsp),%rdi
                                           |
|   0x401405 <main+36>              mov    $0x0,%esi
                                           |
|   0x40140a <main+41>              callq  0x401100 <gettimeofday@plt>
                                           |
|   0x40140f <main+46>              lea    0xc32(%rip),%rsi        # 0x402048
                                          |
|   0x401416 <main+53>              mov    $0x1,%edi
                                           |
|   0x40141b <main+58>              mov    $0x0,%eax
                                           |
|   0x401420 <main+63>              callq  0x401120 <__printf_chk@plt>
                                           |
|   0x401425 <main+68>              lea    0x30(%rsp),%rbx
                                           |
|   0x40142a <main+73>              mov    0x2c5f(%rip),%rdx        # 0x404090
<stdin@@GLIBC_2.2.5>                       |
|   0x401431 <main+80>              mov    $0x64,%esi
                                          |
|   0x401436 <main+85>              mov    %rbx,%rdi
                                           |
|   0x401439 <main+88>              callq  0x401110 <fgets@plt>
                                          |
|   0x40143e <main+93>              mov    %rbx,%rdi
                                           |
|   0x401441 <main+96>              callq  0x4012f4 <passwordCheck>
                                          |
|   0x401446 <main+101>     lea    0x20(%rsp),%rdi
                                           |
|   0x40144b <main+106>     mov    $0x0,%esi
                                           |
|   0x401450 <main+111>     callq  0x401100 <gettimeofday@plt>
                                           |
```

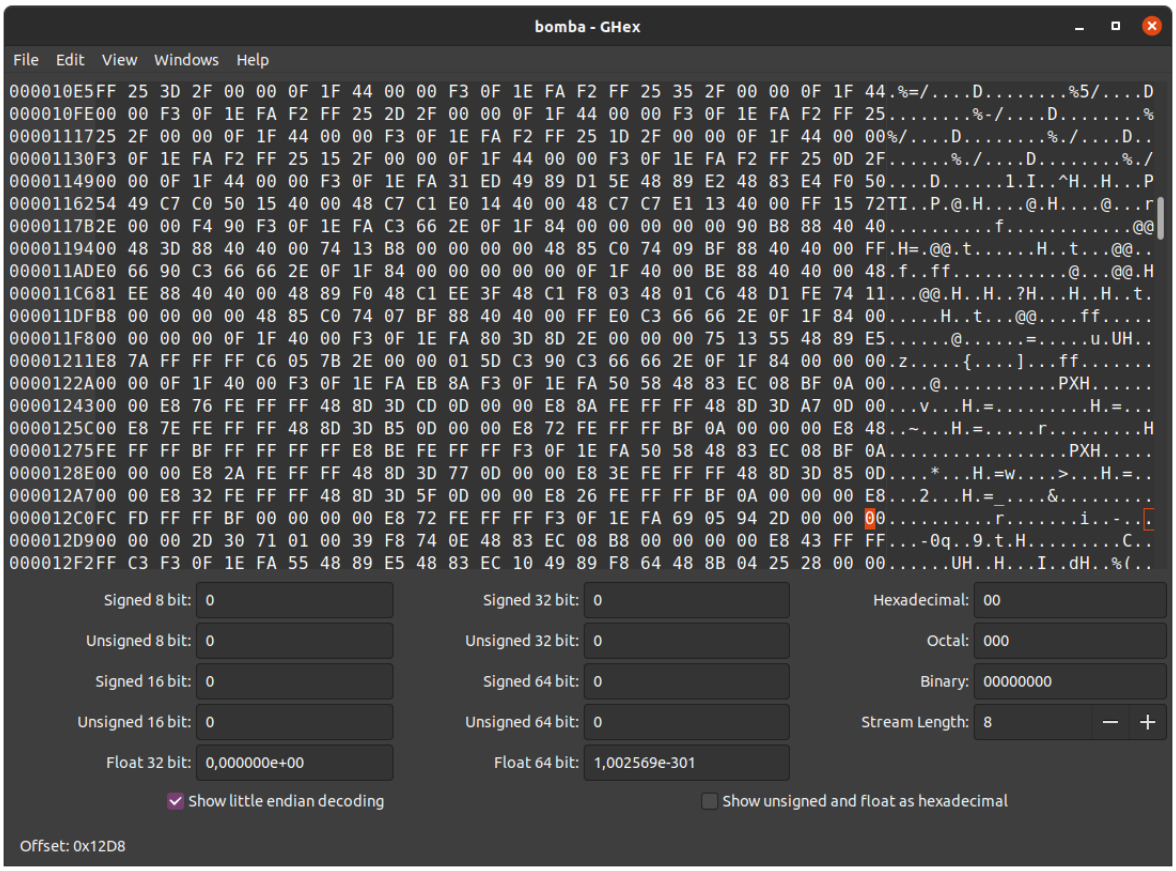
```
   0x401455 <main+116>     mov    0x20(%rsp),%rax
   0x40145a <main+121>     sub    0x10(%rsp),%rax
   0x40145f <main+126>     cmp    $0x3c,%rax
   0x401463 <main+130>     jle    0x40146f <main+142>
   0x401465 <main+132>     mov    $0x0,%eax
   0x40146a <main+137>     callq  0x401236 <fail>
   0x40146f <main+142>     lea    0xbed(%rip),%rsi        # 0x402063
   0x401476 <main+149>     mov    $0x1,%edi
   0x40147b <main+154>     mov    $0x0,%eax
   0x401480 <main+159>     callq  0x401120 <__printf_chk@plt>
   0x401485 <main+164>     lea    0xc(%rsp),%rsi
   0x40148a <main+169>     lea    0xbe9(%rip),%rdi        # 0x40207a
   0x401491 <main+176>     mov    $0x0,%eax
   0x401496 <main+181>     callq  0x401130 <__isoc99_scanf@plt>
   0x40149b <main+186>     mov    0xc(%rsp),%edi
   0x40149f <main+190>     callq  0x4012ce <numberCheck>
   0x4014a4 <main+195>     lea    0x10(%rsp),%rdi
   0x4014a9 <main+200>     mov    $0x0,%esi
   0x4014ae <main+205>     callq  0x401100 <gettimeofday@plt>
   0x4014b3 <main+210>     mov    0x10(%rsp),%rax
   0x4014b8 <main+215>     sub    0x20(%rsp),%rax
   0x4014bd <main+220>     cmp    $0x3c,%rax
   0x4014c1 <main+224>     jle    0x4014cd <main+236>
   0x4014c3 <main+226>     mov    $0x0,%eax
   0x4014c8 <main+231>     callq  0x401236 <fail>
   0x4014cd <main+236>     mov    $0x0,%eax
   0x4014d2 <main+241>     callq  0x401282 <success>
   0x4014d7                nopw   0x0(%rax,%rax,1)
```

Investigando un poco por la función numberCheck(), nos damos cuenta de lo siguiente:

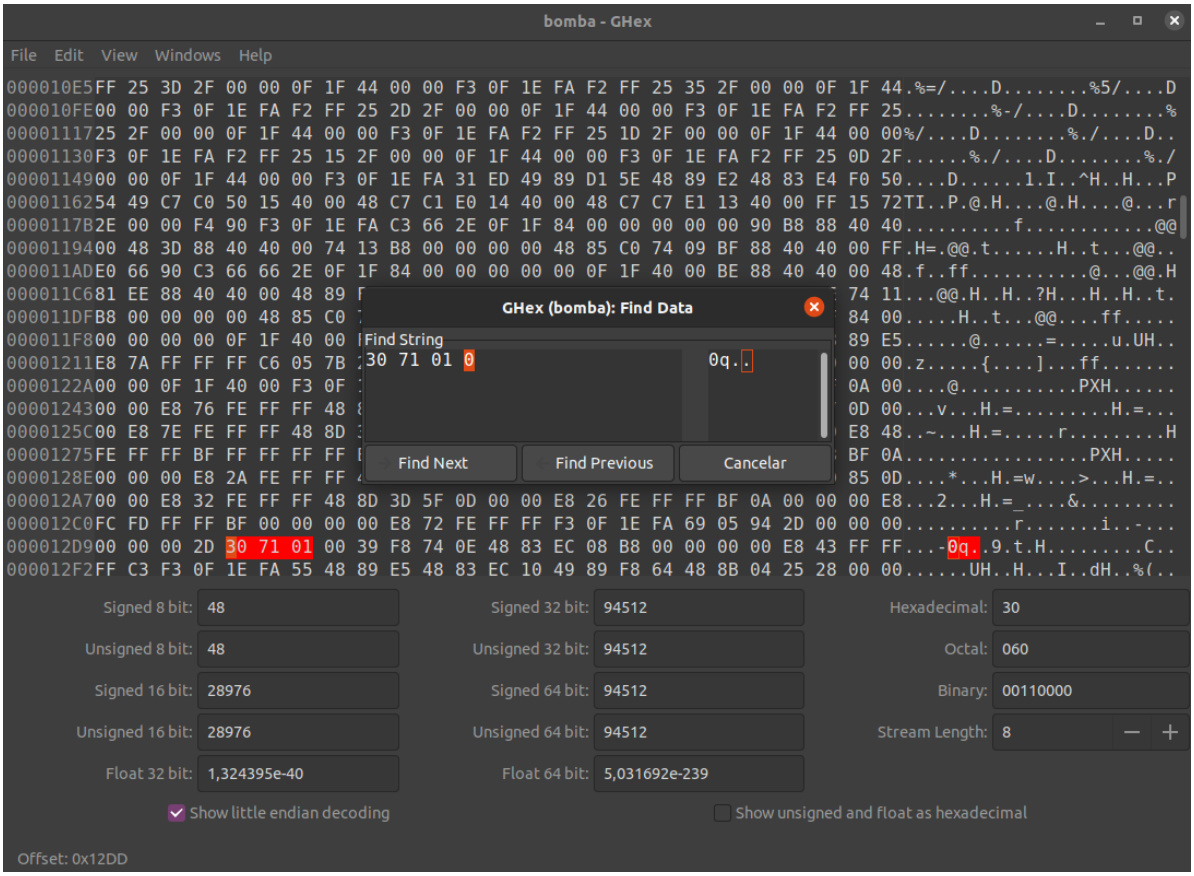
```
0x4012d2 <numberCheck+4>        imul   $0x35d7,0x2d94(%rip),%eax        #
0x404070 <passcode>
0x4012dc <numberCheck+14>       sub    $0x17130,%eax
0x4012e1 <numberCheck+19>       cmp    %edi,%eax
0x4012e3 <numberCheck+21>       je     0x4012f3 <numberCheck+37>
```

Como podemos ver, la función hace la siguiente operación: codigo = 13783 * passcode - 94512. Durante la depuración, nos damos cuenta en la comparación que el valor del registro rax = 1969. Por tanto, ya tenemos que la **contraseña = "BREAKINGBAD"**, la variable **password = ";K>:DBG@;:="**, el **código = 1969** y despejando de la operación de arriba sabemos que **passcode = 7**. Ahora que sabemos todo esto, modifiquemos sus valores. Para ello usaremos el editor hexadecimal **GHex**. Primero modificaremos el código por tanto buscamos el passcode y lo modificamos en hexadecimal. Para ello, pondremos a 0 el valor 13783 con el fin de anular ese producto y quedarnos sólamenrte con el número que le suma, tambíen modificado. 13783 es D735 en hexadecimal y lo pondremos a 0, de esta manera anulamos el valor del passcode sea el que sea:

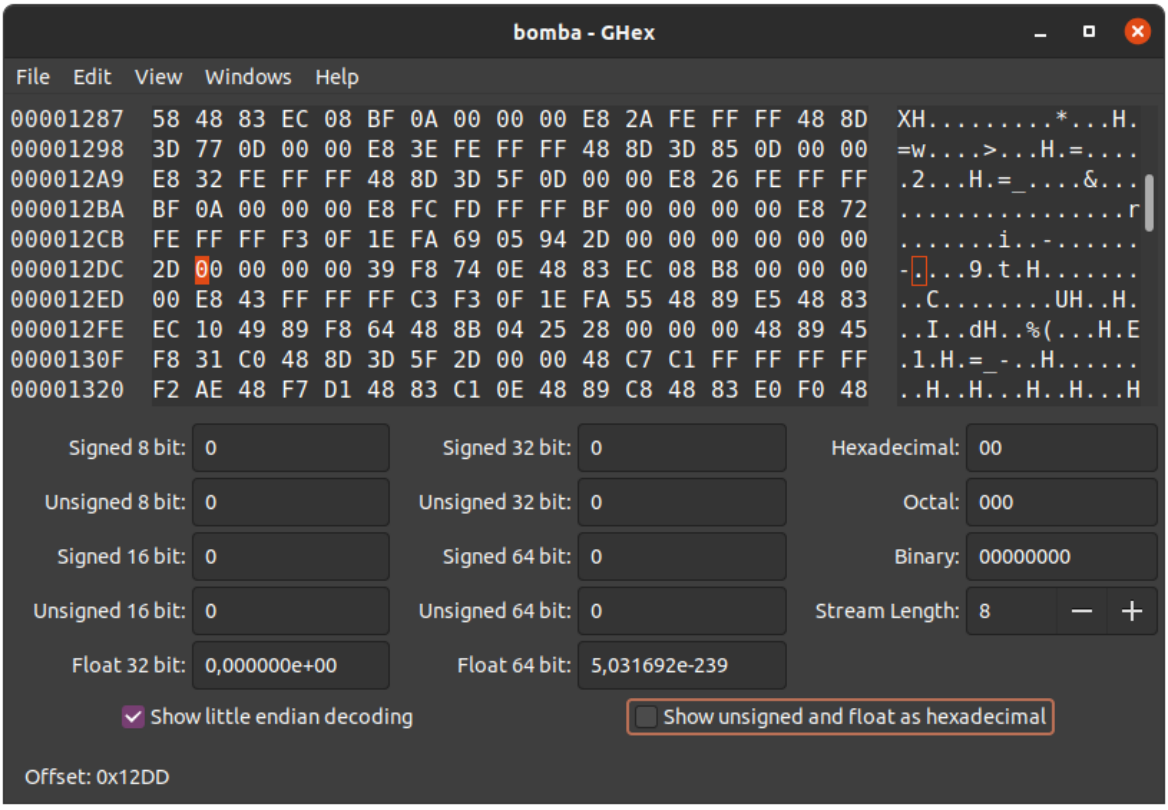




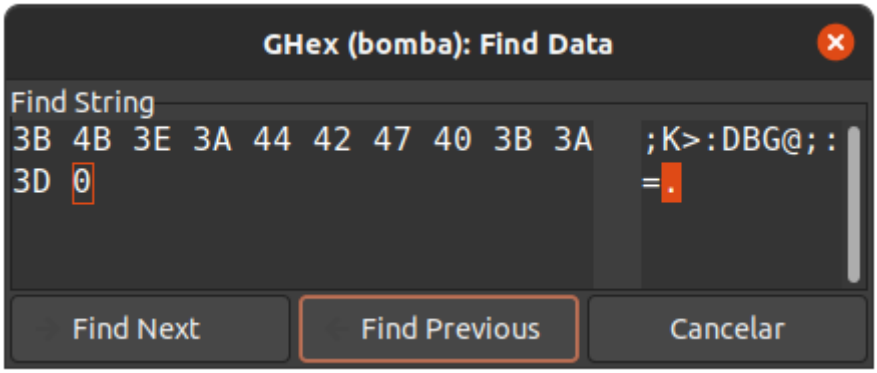
Una vez modificado quedaría así. Ahora busquemos 94512:

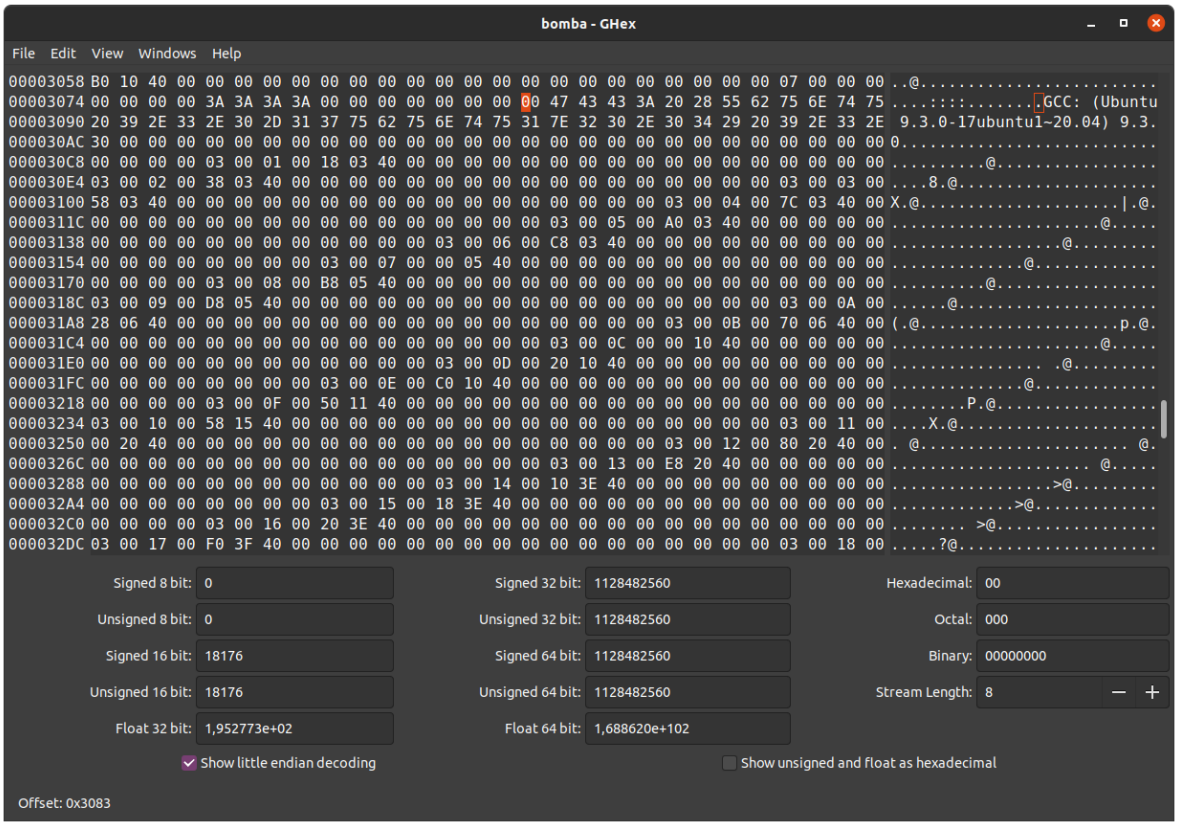


Que lo modificaremos a 0:



Finalmente cambiaremos la contraseña por AAAA:





Y de esta forma tendríamos la bomba formada, resuelta y modificada.