作业2

拆弹

学习网站:

> csapp-bomblab 详解 - 知乎 (zhihu.com)
>
> 手把手教你拆解 CSAPP 的 炸弹实验室 BombLab - 知乎 (zhihu.com)

```C
git clone git@github.com:cs351/mp1-learner0904.git
```

mp1-learner0904

```Plain Text
cd mp1-learner0904
./getbomb.sh
```

炸弹 41

实时榜单

> https://moss.cs.iit.edu/cs351/bomblab-scoreboard.html

拆弹

```Shell
cd mp1-learner0904
cd bomb41
```

一开始就在explode_bomb 和 下一个 phase 处设置断点

b explode_bomb

b phase_1

b phase_2

b phase_3

b phase_4

b phase_5

b phase_6

```shell
# 使用gdb运行bomb文件
gdb bomb

# 在炸弹爆炸函数处设置断点，即使当我们错误输入时也能阻止炸弹爆炸
b explode_bomb

# 从 bomb.c 中可以知道，phase_1(input) 处理输入字符串，故我们在此函数入口
b phase_1

# 运行程序
r

# 此时进行输入拆弹字符串（由于不知道，所以随机输入，反正有断点）
# 触发断点
# 显示反汇编窗口
layout asm


# 观察窗口中的“callq”调用的哪一个函数，在函数入口处设置断点
#

# 查看寄存器值
info registers

```

disas strings_not_equal

Border relations with Canada have never been better.

1 2 4 8 16 32

I was trying to give Tina Fey more material.

0 1 3 6 10 15

0 986

24 2

4 1 5 6 3 2 / 2 3 6 5 1 4

```
0000000000401058 <func4>:
  401058:       85 ff                   test   %edi,%edi
  40105a:       7e 2b                   jle    401087 <func4+0x2f>
  40105c:       89 f0                   mov    %esi,%eax
  40105e:       83 ff 01                cmp    $0x1,%edi
  401061:       74 2e                   je     401091 <func4+0x39>
  401063:       41 54                   push   %r12
  401065:       55                      push   %rbp
  401066:       53                      push   %rbx
  401067:       89 f5                   mov    %esi,%ebp
  401069:       89 fb                   mov    %edi,%ebx
  40106b:       8d 7f ff                lea    -0x1(%rdi),%edi
  40106e:       e8 e5 ff ff ff          callq  401058 <func4>
  401073:       44 8d 64 05 00          lea    0x0(%rbp,%rax,1),%r12d
  401078:       8d 7b fe                lea    -0x2(%rbx),%edi
  40107b:       89 ee                   mov    %ebp,%esi
  40107d:       e8 d6 ff ff ff          callq  401058 <func4>
  401082:       44 01 e0                add    %r12d,%eax
  401085:       eb 06                   jmp    40108d <func4+0x35>
  401087:       b8 00 00 00 00          mov    $0x0,%eax
  40108c:       c3                      retq
  40108d:       5b                      pop    %rbx
  40108e:       5d                      pop    %rbp
  40108f:       41 5c                   pop    %r12
  401091:       f3 c3                   repz retq

0000000000401093 <phase_4>:
  401093:       48 83 ec 18             sub    $0x18,%rsp
  401097:       64 48 8b 04 25 28 00    mov    %fs:0x28,%rax
  40109e:       00 00
  4010a0:       48 89 44 24 08          mov    %rax,0x8(%rsp)
  4010a5:       31 c0                   xor    %eax,%eax
  4010a7:       48 89 e1                mov    %rsp,%rcx
  4010aa:       48 8d 54 24 04          lea    0x4(%rsp),%rdx
  4010af:       be 3d 29 40 00          mov    $0x40293d,%esi
  4010b4:       e8 87 fb ff ff          callq  400c40 <__isoc99_sscanf@plt>
  4010b9:       83 f8 02                cmp    $0x2,%eax
  4010bc:       75 0b                   jne    4010c9 <phase_4+0x36>
  4010be:       8b 04 24                mov    (%rsp),%eax
  4010c1:       83 e8 02                sub    $0x2,%eax
  4010c4:       83 f8 02                cmp    $0x2,%eax
  4010c7:       76 05                   jbe    4010ce <phase_4+0x3b>
  4010c9:       e8 b8 05 00 00          callq  401686 <explode_bomb>
  4010ce:       8b 34 24                mov    (%rsp),%esi
  4010d1:       bf 05 00 00 00          mov    $0x5,%edi
  4010d6:       e8 7d ff ff ff          callq  401058 <func4>
  4010db:       3b 44 24 04             cmp    0x4(%rsp),%eax
  4010df:       74 05                   je     4010e6 <phase_4+0x53>
  4010e1:       e8 a0 05 00 00          callq  401686 <explode_bomb>
  4010e6:       48 8b 44 24 08          mov    0x8(%rsp),%rax
  4010eb:       64 48 33 04 25 28 00    xor    %fs:0x28,%rax
  4010f2:       00 00
  4010f4:       74 05                   je     4010fb <phase_4+0x68>
  4010f6:       e8 95 fa ff ff          callq  400b90 <__stack_chk_fail@plt>
```

```
  4010fb:       48 83 c4 18             add    $0x18,%rsp
  4010ff:       c3                      retq
```

Plain Text

```
__int64 __fastcall func4(int a1, __int64 a2)
{
  __int64 result; // rax
  int v3; // r12d

  if ( a1 <= 0 )
    return 0LL;
  result = (unsigned int)a2;
  if ( a1 != 1 )
  {
    v3 = a2 + func4((unsigned int)(a1 - 1), a2);
    return v3 + (unsigned int)func4((unsigned int)(a1 - 2), (unsigned int)a2);
  }
  return result;
}

unsigned __int64 __fastcall phase_4(__int64 a1)
{
  unsigned int v2; // [rsp+0h] [rbp-18h] BYREF
  int v3; // [rsp+4h] [rbp-14h] BYREF
  unsigned __int64 v4; // [rsp+8h] [rbp-10h]

  v4 = __readfsqword(0x28u);
  if ( (unsigned int)__isoc99_sscanf(a1, "%d %d", &v3, &v2) != 2 || v2 - 2 > 2 )
    explode_bomb();
  if ( (unsigned int)func4(5LL, v2) != v3 )
    explode_bomb();
  return __readfsqword(0x28u) ^ v4;
}
```

```
0000000000401100 <phase_5>:
  401100: 53                    push   %rbx
  401101: 48 83 ec 10           sub    $0x10,%rsp
  401105: 48 89 fb              mov    %rdi,%rbx
  401108: 64 48 8b 04 25 28 00  mov    %fs:0x28,%rax
  40110f: 00 00
  401111: 48 89 44 24 08        mov    %rax,0x8(%rsp)
  401116: 31 c0                 xor    %eax,%eax
  401118: e8 77 02 00 00        callq  401394 <string_length>
  40111d: 83 f8 06              cmp    $0x6,%eax
  401120: 74 05                 je     401127 <phase_5+0x27>
  401122: e8 5f 05 00 00        callq  401686 <explode_bomb>

  401127: b8 00 00 00 00        mov    $0x0,%eax
  40112c: 0f b6 14 03           movzbl (%rbx,%rax,1),%edx
  401130: 83 e2 0f              and    $0xf,%edx
  401133: 0f b6 92 e0 26 40 00  movzbl 0x4026e0(%rdx),%edx  ---"maduiersnfotvbylS
o you think you can stop the bomb with ctrl-c, do you?"
  40113a: 88 54 04 01           mov    %dl,0x1(%rsp,%rax,1)
  40113e: 48 83 c0 01           add    $0x1,%rax
  401142: 48 83 f8 06           cmp    $0x6,%rax
  401146: 75 e4                 jne    40112c <phase_5+0x2c>
  401148: c6 44 24 07 00        movb   $0x0,0x7(%rsp)
  40114d: be 96 26 40 00        mov    $0x402696,%esi   ----"flames"
  401152: 48 8d 7c 24 01        lea    0x1(%rsp),%rdi
  401157: e8 56 02 00 00        callq  4013b2 <strings_not_equal>
  40115c: 85 c0                 test   %eax,%eax
  40115e: 74 05                 je     401165 <phase_5+0x65>
  401160: e8 21 05 00 00        callq  401686 <explode_bomb>
  401165: 48 8b 44 24 08        mov    0x8(%rsp),%rax
  40116a: 64 48 33 04 25 28 00  xor    %fs:0x28,%rax
  401171: 00 00
  401173: 74 05                 je     40117a <phase_5+0x7a>
  401175: e8 16 fa ff ff        callq  400b90 <__stack_chk_fail@plt>
  40117a: 48 83 c4 10           add    $0x10,%rsp
  40117e: 5b                    pop    %rbx
  40117f: c3                    retq
```

maduiersnfotvbylSo you think you can stop the bomb with ctrl-c, do you?

转换为 flames

f -- 9

l -- 15

a -- 1

m -- 0

e -- 5

s -- 7

Plain Text

```
ascii
i 69
o 6f
a 61
p 70
e 65
g 67
```

Plain Text

```
0000000000401180 <phase_6>:
  401180: 41 55                 push   %r13
  401182: 41 54                 push   %r12
  401184: 55                    push   %rbp
  401185: 53                    push   %rbx
  401186: 48 83 ec 68           sub    $0x68,%rsp
  40118a: 64 48 8b 04 25 28 00  mov    %fs:0x28,%rax
  401191: 00 00
  401193: 48 89 44 24 58        mov    %rax,0x58(%rsp)
  401198: 31 c0                 xor    %eax,%eax
  40119a: 48 89 e6              mov    %rsp,%rsi
  40119d: e8 1a 05 00 00        callq  4016bc <read_six_numbers>
  4011a2: 49 89 e4              mov    %rsp,%r12
  4011a5: 41 bd 00 00 00 00     mov    $0x0,%r13d
  4011ab: 4c 89 e5              mov    %r12,%rbp
  4011ae: 41 8b 04 24           mov    (%r12),%eax
  4011b2: 83 e8 01              sub    $0x1,%eax
  4011b5: 83 f8 05              cmp    $0x5,%eax
  4011b8: 76 05                 jbe    4011bf <phase_6+0x3f>
  4011ba: e8 c7 04 00 00        callq  401686 <explode_bomb>
  4011bf: 41 83 c5 01           add    $0x1,%r13d
  4011c3: 41 83 fd 06           cmp    $0x6,%r13d
  4011c7: 74 3d                 je     401206 <phase_6+0x86>
  4011c9: 44 89 eb              mov    %r13d,%ebx
  4011cc: 48 63 c3              movslq %ebx,%rax
  4011cf: 8b 04 84              mov    (%rsp,%rax,4),%eax
  4011d2: 39 45 00              cmp    %eax,0x0(%rbp)
  4011d5: 75 05                 jne    4011dc <phase_6+0x5c>
  4011d7: e8 aa 04 00 00        callq  401686 <explode_bomb>
  4011dc: 83 c3 01              add    $0x1,%ebx
  4011df: 83 fb 05              cmp    $0x5,%ebx
  4011e2: 7e e8                 jle    4011cc <phase_6+0x4c>
  4011e4: 49 83 c4 04           add    $0x4,%r12
  4011e8: eb c1                 jmp    4011ab <phase_6+0x2b>
  4011ea: 48 8b 52 08           mov    0x8(%rdx),%rdx
  4011ee: 83 c0 01              add    $0x1,%eax
  4011f1: 39 c8                 cmp    %ecx,%eax
  4011f3: 75 f5                 jne    4011ea <phase_6+0x6a>
  4011f5: 48 89 54 74 20        mov    %rdx,0x20(%rsp,%rsi,2)
  4011fa: 48 83 c6 04           add    $0x4,%rsi
  4011fe: 48 83 fe 18           cmp    $0x18,%rsi
  401202: 75 07                 jne    40120b <phase_6+0x8b>
  401204: eb 19                 jmp    40121f <phase_6+0x9f>
  401206: be 00 00 00 00        mov    $0x0,%esi
  40120b: 8b 0c 34              mov    (%rsp,%rsi,1),%ecx
  40120e: b8 01 00 00 00        mov    $0x1,%eax
  401213: ba f0 42 60 00        mov    $0x6042f0,%edx
  401218: 83 f9 01              cmp    $0x1,%ecx
  40121b: 7f cd                 jg     4011ea <phase_6+0x6a>
  40121d: eb d6                 jmp    4011f5 <phase_6+0x75>
  40121f: 48 8b 5c 24 20        mov    0x20(%rsp),%rbx
  401224: 48 8d 44 24 20        lea    0x20(%rsp),%rax
  401229: 48 8d 74 24 48        lea    0x48(%rsp),%rsi
  40122e: 48 89 d9              mov    %rbx,%rcx
```

```
401231: 48 8b 50 08           mov    0x8(%rax),%rdx
401235: 48 89 51 08           mov    %rdx,0x8(%rcx)
401239: 48 83 c0 08           add    $0x8,%rax
40123d: 48 89 d1              mov    %rdx,%rcx
401240: 48 39 f0              cmp    %rsi,%rax
401243: 75 ec                 jne    401231 <phase_6+0xb1>
401245: 48 c7 42 08 00 00 00  movq   $0x0,0x8(%rdx)
40124c: 00
40124d: bd 05 00 00 00        mov    $0x5,%ebp
401252: 48 8b 43 08           mov    0x8(%rbx),%rax
401256: 8b 00                 mov    (%rax),%eax
401258: 39 03                 cmp    %eax,(%rbx)  -- 保证链表是递增（不递减）的
40125a: 7e 05                 jle    401261 <phase_6+0xe1>
40125c: e8 25 04 00 00        callq  401686 <explode_bomb>
401261: 48 8b 5b 08           mov    0x8(%rbx),%rbx
401265: 83 ed 01              sub    $0x1,%ebp
401268: 75 e8                 jne    401252 <phase_6+0xd2>
40126a: 48 8b 44 24 58        mov    0x58(%rsp),%rax
40126f: 64 48 33 04 25 28 00  xor    %fs:0x28,%rax
401276: 00 00
401278: 74 05                 je     40127f <phase_6+0xff>
40127a: e8 11 f9 ff ff        callq  400b90 <__stack_chk_fail@plt>
40127f: 48 83 c4 68           add    $0x68,%rsp
401283: 5b                    pop    %rbx
401284: 5d                    pop    %rbp
401285: 41 5c                 pop    %r12
401287: 41 5d                 pop    %r13
401289: c3                    retq
```

0x6042f0 <node1>: "j"

一个节点 node 的结构如下：

前 8 个字节是 value（逆着存储的，比如下面 node1 的 value 是 006a）

后 8 个是地址 address

```
0x6042f0 <node1>:       0x6a    0x00    0x00    0x00    0x01    0x00    0x00    0
x00
0x6042f8 <node1+8>:     0x00    0x43    0x60    0x00    0x00    0x00    0x00    0
x00
0x604300 <node2>:       0x3e    0x03    0x00    0x00    0x02    0x00    0x00    0
x00
0x604308 <node2+8>:     0x10    0x43    0x60    0x00    0x00    0x00    0x00    0
x00
0x604310 <node3>:       0x42    0x02    0x00    0x00    0x03    0x00    0x00    0
x00
0x604318 <node3+8>:     0x20    0x43    0x60    0x00    0x00    0x00    0x00    0
x00
0x604320 <node4>:       0x9e    0x00    0x00    0x00    0x04    0x00    0x00    0
x00
0x604328 <node4+8>:     0x30    0x43    0x60    0x00    0x00    0x00    0x00    0
x00
0x604330 <node5>:       0x67    0x00    0x00    0x00    0x05    0x00    0x00    0
x00
0x604338 <node5+8>:     0x40    0x43    0x60    0x00    0x00    0x00    0x00    0
x00
0x604340 <node6>:       0x5f    0x03    0x00    0x00    0x06    0x00    0x00    0
x00
0x604348 <node6+8>:     0x00    0x00    0x00    0x00    0x00    0x00    0x00    0
x00
0x604350 <user_password>:       0x71    0x79    0x54    0x36    0x74    0x4b    0
x45    0x6c
0x604358 <user_password+8>:     0x79    0x47    0x49    0x63    0x35    0x59    0
x51    0x4e
0x604360 <user_password+16>:    0x37    0x75    0x48    0x48    0x00    0x79    0
x6c    0x69
0x604368 <userid+3>:    0x33    0x39    0x35    0x00    0x29    0x00    0x00    0
x00
```

Plain Text

```
006a
033e
0242
009e
0067
035f
```

Plain Text

5 1 4 3 2 6

Plain Text

```
unsigned __int64 __fastcall phase_6(__int64 a1)
{
  __int64 v1; // rdx
  __int64 v2; // rcx
  int *v3; // r12
  int v4; // r13d
  int v5; // ebx
  _QWORD *v6; // rdx
  int v7; // eax
  __int64 i; // rsi
  int v9; // ecx
  __int64 v10; // rbx
  __int64 *v11; // rax
  __int64 v12; // rcx
  __int64 v13; // rdx
  int v14; // ebp
  int v16[8]; // [rsp+0h] [rbp-88h] BYREF
  __int64 v17[5]; // [rsp+20h] [rbp-68h] BYREF
  char v18[16]; // [rsp+48h] [rbp-40h] BYREF
  unsigned __int64 v19; // [rsp+58h] [rbp-30h]

  v19 = __readfsqword(0x28u);
  read_six_numbers(a1, v16);
  v3 = v16;
  v4 = 0;
  while ( 1 )
  {
    if ( (unsigned int)(*v3 - 1) > 5 )
      explode_bomb(a1, v16, v1, v2);
    if ( ++v4 == 6 )
      break;
    v5 = v4;
    do
    {
      if ( *v3 == v16[v5] )
        explode_bomb(a1, v16, v1, v2);
      ++v5;
    }
    while ( v5 <= 5 );
    ++v3;
  }
  for ( i = 0LL; i != 6; ++i )
  {
    v9 = v16[i];
    v7 = 1;
    v6 = &node1;
    if ( v9 > 1 )
    {
      do
      {
        v6 = (_QWORD *)v6[1];
        ++v7;
      }
      while ( v7 != v9 );
```

```
    }
    v17[i] = (__int64)v6;
  }
  v10 = v17[0];
  v11 = v17;
  v12 = v17[0];
  do
  {
    v13 = v11[1];
    *(_QWORD *)(v12 + 8) = v13;
    ++v11;
    v12 = v13;
  }
  while ( v11 != (__int64 *)v18 );
  *(_QWORD *)(v13 + 8) = 0LL;
  v14 = 5;
  do
  {
    if ( *(_DWORD *)v10 > **(_DWORD **)(v10 + 8) )
      explode_bomb(a1, v18, v13, v13);
    v10 = *(_QWORD *)(v10 + 8);
    --v14;
  }
  while ( v14 );
  return __readfsqword(0x28u) ^ v19;
}
```