一 phase_0

1. 运行函数，打断点到main
2. 分析bomb汇编代码，phase_0把输入的字符串与0x804a1ef处的字符串相比较，如果相等则拆弹成功。
3. 使用x/s 0x804a1ef得到"Simplicity favors regularity."

二 phase_1

1. 用display /i *pc*打印汇编语句
2. 运行函数，打断点到phase_1
3. x/s 0x804a20d得到"%d%d"知输入2个int数字
4. 80494c0: 83 f8 02 cmp $0x2,%eax作证输入2个
5. 引发bomb的有 80494d9: 39 c2 cmp %eax,%edx
6. 此处eax==edx 不爆炸
7. 2063597568 edx 0x7b000000 2063597568
8. 1102242151 edx 0x41b2e167 1102242151

三 phase_2

1. read_n_numbers
2. 8049b49: 68 d7 a2 04 08 x/s 0x804a2d7 得到%d
3. 8049527: 3d ca 00 00 00 cmp $0xca,%eax 202
4. 202 / 2 + 1 循环6次

四 phase_3

phase_3过程

1. 运行函数，打断点到phase_3
2. 804959b: 83 7d f0 01 cmpl $0x1,-0x10(%ebp) <= 1
3. 80495cb: ff e0 jmp *%eax 第一句决定了跳转 80495cd: 81 45 f4 58 02 00 00 addl $0x258,-0xc(%ebp) 80495d4: 81 45 f4 e1 02 00 00 addl $0x2e1,-0xc(%ebp) 80495db: 81 6d f4 58 02 00 00 subl $0x258,-0xc(%ebp) 80495e2: 81 45 f4 58 02 00 00 addl $0x258,-0xc(%ebp) 80495e9: 81 45 f4 e1 02 00 00 addl $0x2e1,-0xc(%ebp) 80495f0: 81 6d f4 58 02 00 00 subl $0x258,-0xc(%ebp) 80495f7: 81 45 f4 e1 02 00 00 addl $0x2e1,-0xc(%ebp) 80495fe: 81 6d f4 e1 02 00 00 subl $0x2e1,-0xc(%ebp) 8049605: 81 45 f4 58 02 00 00 addl $0x258,-0xc(%ebp)

五 phase_4

phase_4过程

1. 运行函数，打断点到phase_4
2. x/s 0x804a20d得到"%d%d"知输入2个int数字
3. 分析func4汇编代码得到

六 phase_5

x/64ux 0x804c220打出数组硬看13次下标索引，并求和。
%d %d 输入两个数第一个数低四位为3 第二个数为求和 100

```
1 int a[16]={0xa,0x2,0xe,0x7,
2 0x8,0xc,0xf,0xb,
3 0x0,0x4,0x1,0xd,
4 0x3,0x9,0x6,0x5};
```

七 phase_6

phase_6过程

1. 运行函数，打断点到phase_6
2. 由12行callq 401435<read_six_numbers>可知读取6个值
3. 由17，18行知读取的值不能大于6
4. 由27，28行知读取的值各不相同
5. r13寄存器用于控制循环, 当r13寄存器中的值为6时,跳转到地址0x401168处
6. 查询x/24 0x6032f0得

```
1 0x6032f0 <node1>:     189     1       6304512 0
2 0x603300 <node2>:     397     2       6304528 0
3 0x603310 <node3>:     86      3       6304544 0
4 0x603320 <node4>:     954     4       6304560 0
5 0x603330 <node5>:     369     5       6304576 0
6 0x603340 <node6>:     992     6       0       0
```

7. 51到61行将链表组织起来
8. 63到71行以递增形式组成。
   即$node3(86)->node1(189)->node5(392)->node2(397)->node4(954)->node6(992)$
9. 所以密码为3 1 5 2 4 6

八　secret_phase

secret_phase过程

1. 运行函数，打断点到phase_defused
2. 查询x/s 0x402599得"%d %d %s"知输入2个int数字1个字符串
3. 查询x/s 0x4025a2得DrEvil，猜测在phase_defused中调用secret_phase的条件是在12 3后加上DrEvil
4. 查询x/s 0x603110得

   上面表示的是一个二叉排序树，其中n1为根节点，nxy为第x层第y个节点。
5. 分析fun7的汇编代码得

```
int fun7(struct Node *node, int value)
{   //node in %rdi,value in %rsi,return_value in %eax
    //require %eax to be 5
    if(!node)
        return -1;
    int t = node->val;
    if (t > value)
    {
        node = node->left;
        int return_value = 2 * fun7(node, value);
        return return_value;
    }
    else{
        int return_value=0;
        if (value == t)
            return return_value;
        node = node->right;
        int return_value = 0x1 + 2 * fun7(node, value);
        return return_value;
    }
}
```

6. 得到密码为47

九　总结——拆弹密码

```
Simplicity favors regularity.
2063597568 1102242151

0 738
12 3 DrEvil
```

JDOEFG
3 1 5 2 4 6
47