第一题：

b phase\_1

b explode\_bomb

b 0x401342

p \*(char \*)$rbp

$rbx保存输入字符串

$rbp保存固定字符串

第一题答案为

第二题

Set args answer

b phase\_2

si 0x400f0a 表示第一个必须是1

x/6w $rsp 即为输入的6个数字

400f1a 后一个数必须是前者的两倍

答案为 1 2 4 8 16 32 64

第三题

400f60 cmp 0x1, %eax 可知Sscanf函数入参个数至少大于1

第一个参数：\*($rsp + 0x8)

第二个参数：$($rsp + 0xc)

400f6a cmpl 0x7, 0x8(%rsp) 表示第一个参数需小于等于7

400f75jmp可知，从rax\*8+0x402470地址处获得跳转目标, eax值从0到7

打印跳转表结构，

(gdb) x/8g 0x402470

0x402470: 0x0000000000400f7c 0x0000000000400fb9

0x402480: 0x0000000000400f83 0x0000000000400f8a

0x402490: 0x0000000000400f91 0x0000000000400f98

0x4024a0: 0x0000000000400f9f 0x0000000000400fa6

易得8组解：

0, 172

1, 311

2, 707

3, 256

4, 389

5, 206

6, 682

7, 327

C语言伪代码如下：

第四题：

Jne 表示sscanf只能接收刚好两个参数

第一个参数：\*($rsp + 0x8)

第二个参数：$($rsp + 0xc)

第一个参数小于等于14，第二个参数必须为0

写出C递归函数，发现四组解

第五题：

p (char \*)$rdi 即我们输入的四负川

40108b $rbx打印处输入字符串

x /6b $rbx

x/6x 0x40245e目表字符串

0x66 f 0x6c l 0x79 y 0x65 e 0x72 r 0x73 s

(gdb) x/15c 0x4024b0

0x4024b0 <array.3449>: 109 'm' 97 'a' 100 'd' 117 'u' 105 'i' 101 'e' 114 'r' 115 's'

0x4024b8 <array.3449+8>: 110 'n' 102 'f' 111 'o' 116 't' 118 'v' 98 'b' 121 'y'

分析匯編，只需輸入字符串的低4位為0x9, 0xF, 0xE, 0x5, 0x6, 0x7

有非常多組解，這裏都取大寫字母，即IONUVW

ASCLI查詢網站: <http://c.biancheng.net/c/ascii/>

第六題：

第一个数需和后面都不同

b \*0x40114d

(gdb) info registers

rax 0x6 6

rbx 0x7fffffffe548 140737488348488

rcx 0x7fffffffe3d0 140737488348112

rdx 0x0 0

rsi 0x0 0

rdi 0x7fffffffddc0 140737488346560

rbp 0x0 0x0

rsp 0x7fffffffe3e0 0x7fffffffe3e0

r8 0x7ffff7dd5060 140737351864416

r9 0x0 0

r10 0x0 0

r11 0x0 0

r12 0x400c90 4197520

r13 0x7fffffffe3e0 140737488348128

r14 0x0 0

r15 0x0 0

rip 0x40110b 0x40110b <phase\_6+23>

eflags 0x206 [ PF IF ]

cs 0x33 51

ss 0x2b 43

ds 0x0 0

es 0x0 0

fs 0x0 0

gs 0x0 0

链表

0x6032d0 <node1>: 0x000000010000014c 0x00000000006032e0

0x6032e0 <node2>: 0x00000002000000a8 0x00000000006032f0

0x6032f0 <node3>: 0x000000030000039c 0x0000000000603300

(gdb)

0x603300 <node4>: 0x00000004000002b3 0x0000000000603310

0x603310 <node5>: 0x00000005000001dd 0x0000000000603320

0x603320 <node6>: 0x00000006000001bb 0x0000000000000000

**(gdb) x /24d 0x6032d0**

**0x6032d0 <node1>: 332 1 6304480 0**

**0x6032e0 <node2>: 168 2 6304496 0**

**0x6032f0 <node3>: 924 3 6304512 0**

**0x603300 <node4>: 691 4 6304528 0**

**0x603310 <node5>: 477 5 6304544 0**

**0x603320 <node6>: 443 6 0 0**

**Rbx**

**0x6032d0 <node1>: 0x0000014c 0x00000001 0x006032e0 0x00000000**

**0x6032e0 <node2>: 0x000000a8 0x00000002 0x006032f0 0x00000000**

**0x6032f0 <node3>: 0x0000039c 0x00000003 0x00603300 0x00000000**

**0x603300 <node4>: 0x000002b3 0x00000004 0x00603310 0x00000000**

**0x603310 <node5>: 0x000001dd 0x00000005 0x00603320 0x00000000**

**0x603320 <node6>: 0x000001bb 0x00000006 0x00000000 0x00000000**

**隐藏关**

**观察secret\_phase, 只有第6关出现**

**第四关添加特定字符串即可进入隐藏关**

**传递字符串须能用过strtol转换成整数，且大小必须小于等于1001**

**(gdb) x/32g $rdi**

**0x6030f0 <n1>: 0x0000000000000024 0x0000000000603110**

**0x603100 <n1+16>: 0x0000000000603130 0x0000000000000000**

**0x603110 <n21>: 0x0000000000000008 0x0000000000603190**

**0x603120 <n21+16>: 0x0000000000603150 0x0000000000000000**

**0x603130 <n22>: 0x0000000000000032 0x0000000000603170**

**0x603140 <n22+16>: 0x00000000006031b0 0x0000000000000000**

**0x603150 <n32>: 0x0000000000000016 0x0000000000603270**

**0x603160 <n32+16>: 0x0000000000603230 0x0000000000000000**

**0x603170 <n33>: 0x000000000000002d 0x00000000006031d0**

**0x603180 <n33+16>: 0x0000000000603290 0x0000000000000000**

**0x603190 <n31>: 0x0000000000000006 0x00000000006031f0**

**0x6031a0 <n31+16>: 0x0000000000603250 0x0000000000000000**

**0x6031b0 <n34>: 0x000000000000006b 0x0000000000603210**

**0x6031c0 <n34+16>: 0x00000000006032b0 0x0000000000000000**

**0x6031d0 <n45>: 0x0000000000000028 0x0000000000000000**

**0x6031e0 <n45+16>: 0x0000000000000000 0x0000000000000000**