HGAME Week1 WriteUp

RE

Maze

1. 打开 IDA F5 查看 main 函数

2. 从这部分可知 flag 中的%s 为 输入内容

```
50  if ( v5 == (char *)&unk_60243C )
51  {
52    sprintf(&v7, "hgame{%s}", s, v4);
53    puts("You win!");
54    printf("Flag is: ");
55    puts(&v7);
56    exit(0);
57  }
58LABEL_22:
59  puts("You died");
60  exit(0);
61}
```

3. 从这部分可知 输入的字符串仅能是 “wasd” 四个键

```
15  v5 = (char *)&unk_6020C4;
16  while ( (signed int)v4 < SHIDWORD(v4) )
17  {
18    v3 = s[(signed int)v4];
19    if ( v3 == 100 )
20    {
21      v5 += 4;
22    }
23    else if ( v3 > 100 )
24    {
25      if ( v3 == 115 )
26      {
27        v5 += 64;
28      }
29      else
30      {
31        if ( v3 != 119 )
32        {
33LABEL_12:
34          puts("Illegal input!");
35          exit(0);
36        }
37        v5 -= 64;
38      }
39    }
40    else
41    {
42      if ( v3 != 97 )
43        goto LABEL_12;
44      v5 -= 4;
45    }
```

4. 根据不同的输入 会改变指针 v5 所指的地址 且根据所指的地址中所存储的值是否为 0 判断输入的对错

   v3 为 100 v5 进四位 'd'；v3 为 115 v5 进 64 位 's'

   v3 为 119 v5 退 64 位 'w'；v3 为 97 v5 退四位 'a'

5. 所以只要知道令指针从 6020C4 移至 60243C 的输入即可输出正确答案

```
.data:00000000006020C3                  db    0
.data:00000000006020C4 unk_6020C4       db    0
.data:00000000006020C5                  db    1
.data:000000000060243A                  db    1
.data:000000000060243B                  db    0
.data:000000000060243C unk_60243C       db    0
.data:000000000060243D                  db    1
```

*因为不会拉数据就没写脚本，用计算器算的 没有岔路挺快的

bitwise_operation2

1. shift + F12 查看字符串可以看出 flag 应该和 4sy_Re_ 和 asylif3 两个字符串有关

```
C    Just one last step
C    Congratulations! You are already familiar with bitwise operat…
C    Flag is your input.
C    ;*3$\"
C    4sy_Re_
C    asylif3
```

2.可以看出输入格式应该是 hgame{v24 v25} ， v24 v25 为两段字符串再经过函数处理

```
29  v27 = __readfsqword(0x28u);
30  sub_4007E6();
31  v6 = 76;
32  v7 = 60;
33  v8 = -42;
34  v9 = 54;
35  v10 = 80;
36  v11 = -120;
37  v12 = 32;
38  v13 = -52;
39  __isoc99_scanf("%39s", &s);
40  if ( strlen(&s) == 39 && s == 104 && v19 == 103 && v20 == 97 && v21 == 109 && v22 == 101 && v23 == 123 && v26 == 125 )
41  {
42    v14 = 0LL;
43    v15 = 0;
44    v16 = 0LL;
45    v17 = 0;
46    sub_400616((__int64)&v14, (__int64)&v24);
47    sub_400616((__int64)&v16, (__int64)&v25);
```

3. 进入函数 sub_400616 可以看到 输入为 0~f 每两位 十六进制转换为数值 Q，

Q = 第一位 * 16 + 第二位

```
  for ( i = 0; i <= 7; ++i )
  {
    if ( *(_BYTE *)(2 * i + a2) <= 96 || *(_BYTE *)(2 * i + a2) > 102 )
    {
      if ( *(_BYTE *)(2 * i + a2) <= 47 || *(_BYTE *)(2 * i + a2) > 57 )
      {
LABEL_17:
        puts("Illegal input!");
        exit(0);
      }
      *(_BYTE *)(i + a1) = *(_BYTE *)(2 * i + a2) - 48;
    }
    else
    {
      *(_BYTE *)(i + a1) = *(_BYTE *)(2 * i + a2) - 87;
    }
    if ( *(_BYTE *)(2 * i + 1LL + a2) <= 96 || *(_BYTE *)(2 * i + 1LL + a2) > 102 )
    {
      if ( *(_BYTE *)(2 * i + 1LL + a2) <= 47 || *(_BYTE *)(2 * i + 1LL + a2) > 57 )
        goto LABEL_17;
      result = (_BYTE *)(i + a1);
      *result = 16 * *result + *(_BYTE *)(2 * i + 1LL + a2) - 48;
    }
    else
    {
      result = (_BYTE *)(i + a1);
      *result = 16 * *result + *(_BYTE *)(2 * i + 1LL + a2) - 87;
    }
```

4．可以看到两组数值通过位运算加密后 与 4sy_Re_ 和 asylif3 进行比较

```
48   for ( i = 0; i <= 7; ++i )
49   {
50     *((_BYTE *)&v14 + i) = ((*((_BYTE *)&v14 + i) & 0xE0) >> 5) | 8 * *((_BYTE *)&v14 + i);
51     *((_BYTE *)&v14 + i) = *((_BYTE *)&v14 + i) & 0x55 ^ ((*((_BYTE *)&v16 + 7 - i) & 0xAA) >> 1) | *((_BYTE *)&v14 + i) & 0xAA;
52     *((_BYTE *)&v16 + 7 - i) = 2 * (*((_BYTE *)&v14 + i) & 0x55) ^ *((_BYTE *)&v16 + 7 - i) & 0xAA | *((_BYTE *)&v16 + 7 - i) & 0x55;
53     *((_BYTE *)&v14 + i) = *((_BYTE *)&v14 + i) & 0x55 ^ ((*((_BYTE *)&v16 + 7 - i) & 0xAA) >> 1) | *((_BYTE *)&v14 + i) & 0xAA;
54   }
55   for ( j = 0; j <= 7; ++j )
56   {
57     *((_BYTE *)&v14 + j) ^= *(&v6 + j);
58     if ( *((_BYTE *)&v14 + j) != byte_602050[j] )
59     {
60       puts("sry, wrong flag");
61       exit(0);
62     }
63   }
64   for ( k = 0; k <= 7; ++k )
65   {
66     *((_BYTE *)&v16 + k) ^= *((_BYTE *)&v14 + k) ^ *(&v6 + k);
67     if ( *((_BYTE *)&v16 + k) != byte_602060[k] )
68     {
69       puts("Just one last step");
70       exit(0);
71     }
72   }
73   puts("Congratulations! You are already familiar with bitwise operation.");
74   puts("Flag is your input.");
75   exit(0);
76 }
77 puts("Illegal input!");
78 exit(0);
79}
```

6. 即 输入的 32 位的 0~f 的字符 为 v24 v25 为两段字符串

7. 制作脚本算出 32 位的字符串 0f233e63637982d2 66cbf41ecb1b0102

```c
#include <stdio.h>
int main(void)
{
    int kk[8] = {76,60,214,54,80,136,32,204};
    int x1[8] = {0};
    int x2[8] = {0};
    char *string = "e4sy_Re_";
    char *string2 = "Easylif3";
    for(int i=0;i<=7;i++)
    {
        x2[i] = string2[i] ^ (string[i] ^ kk[i]);
        printf("%d ",x2[i]);
    }
    printf("\n");
    for(int i=0;i<=7;i++)
    {
        x1[i] = (string[i] ^ kk[i]);
        printf("%d ",x1[i]);
    }
    printf("\n\n");

    for(int e=0;e<8;e++)
    {
        for(int i=0;i<256;i++)
        {
            if(x1[e] == ((((i & 0x55) ^ ((x2[7-e] & 0xAA)>>1)) | (i & 0xAA))))
            {
                printf("%d ",i);
                x1[e] = i;
                break;
            }
        }
        for(int i=0;i<256;i++)
        {
            if(x2[7-e] == (((((x1[e] & 0x55) * 2) ^ (i & 0xAA)) | (i & 0x55))))
```

```c
                x2[7-e] = i;
                break;
            }
        }
        for(int i=0;i<256;i++)
        {
            if(x1[e] == ((((i & 0x55) ^ ((x2[7-e] & 0xAA ) >> 1)) | (i & 0xAA))))
            {
                printf("%d ",i);
                x1[e] = i;
                break;
            }
        }
        for(int i=0;i<256;i++)
        {
            if(x1[e] == ((((i & 0xE0) >> 5) | ((i * 8) % 256))))
            {
                printf("%d ",i);
                x1[e] = i;
                break;
            }
        }
        printf("\n");
    }

    printf("\n\n");
    for(int i=0;i<8;i++)
    {
        printf("%x%x",x1[i]/16,x1[i]%16);
    }
    printf(" ");
    for(int i=0;i<8;i++)
    {
        printf("%x%x",x2[i]/16,x2[i]%16);
    }
```

PWN

Hard_AAAAA

1.IDA 进入 F5 查看 main 函数 可以看到比较 v5 与 7 位字符串 即可达到目的

```
{
  char s; // [esp+0h] [ebp-ACh]
  char v5; // [esp+7Bh] [ebp-31h]
  unsigned int v6; // [esp+A0h] [ebp-Ch]
  int *v7; // [esp+A4h] [ebp-8h]

  v7 = &argc;
  v6 = __readgsdword(0x14u);
  alarm(8u);
  setbuf(_bss_start, 0);
  memset(&s, 0, 0xA0u);
  puts("Let's 0O0o\\0O0!");
  gets(&s);
  if ( !memcmp("0O0o", &v5, 7u) )
    backdoor();
  return 0;
}
```

2. 可以看到字符串为 “0O0o\0O0”

```
a:080486D0
a:080486E0 a0o0o           db '0O0o',0
a:080486E5 a00             db 'O0',0
```

3. 因为只能输入在 v5 且 由此可知相对于 s 的偏移 为 7B

```
{
  char s; // [esp+0h] [ebp-ACh]
  char v5; // [esp+7Bh] [ebp-31h]
  unsigned int v6; // [esp+A0h] [ebp-Ch]
  int *v7; // [esp+A4h] [ebp-8h]
```

4. Linux 上制作脚本 并运行得到权限后 执行 cat flag 得到 flag

Crypto　　　　ubuntu 炸了 图就没了写写思路吧

InfantRSA

1.在网上找了解决 rsa 的脚本，把数据丢进去直接解出来了

Affine

1.由程序可知是对字符的更改 因为前五位已知为 hgame 所以得到五组样例，

制作脚本可以猜出 A B MOD 的值

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import gmpy2
from secret import A, B, flag
assert flag.startswith('hgame{') and flag.endswith('}')

TABLE = 'zxcvbnmasdfghjklqwertyuiop1234567890QWERTYUIOPASDFGHJKLZXCVBNM'
MOD = len(TABLE)

cipher = ''
for b in flag:
    i = TABLE.find(b)
    if i == -1:
        cipher += b
    else:
        ii = (A*i + B) % MOD
        cipher += TABLE[ii]

print(cipher)
# A8I5z{xr1A_J7ha_vG_TpH410}
```

2.将字符串和数值带入 运行 即得到 flag

Reorder

1.多次尝试发现是对字符的打乱，将每个位置标上不同字符，再重组即可的 flag

```
c0c0@ubuntu: ~
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
c0c0@ubuntu:~$ nc 47.98.192.231 25002
> 12345{abcdefghijklmnopqrstuvwxy}
fg435idehcba{21jvwnmoytuxsrqplk}
> 12345{abcdefghijklmnopqrstuvwxy}
fg435idehcba{21jvwnmoytuxsrqplk}
> 12345{abcdefghijklmnopqrstuvwxy}
fg435idehcba{21jvwnmoytuxsrqplk}
> 12345{abcdefghijklmnopqrstuvwxy}
fg435idehcba{21jvwnmoytuxsrqplk}
> 12345{abcdefghijklmnopqrstuvwxy}
fg435idehcba{21jvwnmoytuxsrqplk}
> 12345{abcdefghijklmnopqrstuvwxy}
fg435idehcba{21jvwnmoytuxsrqplk}
> 12345{abcdefghijklmnopqrstuvwxy}
fg435idehcba{21jvwnmoytuxsrqplk}
> 12345{abcdefghijklmnopqrstuvwxy}
fg435idehcba{21jvwnmoytuxsrqplk}
> 12345{abcdefghijklmnopqrstuvwxy}
fg435idehcba{21jvwnmoytuxsrqplk}
> 12345{abcdefghijklmnopqrstuvwxy}
fg435idehcba{21jvwnmoytuxsrqplk}
Rua!!!
5Imaept+m$Uj{ghL0nePR!Ti!ATum_3}
```

MISC

欢迎参加 HGame

1. 题目给出

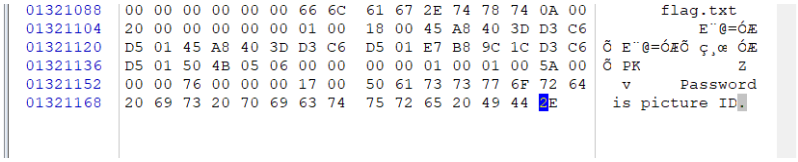
   Li0tIC4uLi0tIC4tLi4gLS4tLiAtLS0tLSAtLSAuIC4uLS0uLSAtIC0tLSAuLi0tLi0gLi4tLS0gLS0tLS0gLi4tLS0gLS0tLS0gLi4tLS4tIC4uLi4gLS0uIC4tIC0tIC4uLi0t

2. Base64 在线解码

3. 可以看出第二层是摩斯密码 直接手算 即可得出 flag

壁纸

1. 用 winhex 查看图片



2. 发现不是 FF D9 的 jpg 的正常结尾，更改图片后缀为 zip 可得压缩包，通过提示在 SauceNAO Image Search 寻找图片 ID 76953815

3. 解压 zip，进行 16 进制到字符串的转换 可得 flag