[goneb5] HGAME 2019 week-1 writeup

RE

Brainfxxker

先将字符串拆分

```
,>++++++++++[<---------->-]<++[+.]
,>+++++++++[<---------->-]<-[+.]
,>+++++++[<------->-]<---[+.]
,>++++++[<------>-]<+++[+.]
,>++++++++[<---------->-]<++[+.]
,>++++++++++[<---------->-]<--[+.]
,>++++++++++[<-------->-]<-----[+.]
,>++++++++++[<---------->-]<+[+.]
,>+++++++++[<-------->-]<---[+.]
```

再结合 oyeye 给的文档不难看出：这其实是需要你输入一个字符，然后通过一个一个循环减后，输出运算后的字符. 通过反推可以得到每一个输入字符串，组合起来就是 flag: hgame{bR4!NfUck}

HelloRe

UE 打开得到 flag:

```
000007a0h: 4C 89 EA 4C 89 F6 44 89 FF 41 FF 14 DC 48 83 C3 ; L嫈L鄤D?A  .踧兜
000007b0h: 01 48 39 EB 75 EA 48 83 C4 08 5B 5D 41 5C 41 5D ; .H9雞闔兡.[]A\A]
000007c0h: 41 5E 41 5F C3 90 66 2E 0F 1F 84 00 00 00 00 00 ; A^A_腺f...?....
000007d0h: F3 C3 00 00 48 83 EC 08 48 83 C4 08 C3 00 00 00 ; 竺..H冹.H兡.?..
000007e0h: 01 00 02 00 50 6C 65 61 73 65 20 69 6E 70 75 74 ; ....Please input
000007f0h: 20 79 6F 75 72 20 6B 65 79 3A 00 68 67 61 6D 65 ;  your key:.hgame
00000800h: 7B 57 65 6C 63 30 6D 33 5F 74 30 5F 52 33 5F 57 ; {Welc0m3_t0_R3_W
00000810h: 6F 72 6C 64 21 7D 00 73 75 63 63 65 73 73 00 66 ; orld!}.success.f
00000820h: 61 69 6C 65 64 2E 2E 00 01 1B 03 3B 34 00 00 00 ; ailed......;4...
00000830h: 05 00 00 00 08 FD FF FF 80 00 00 00 88 FD FF FF ; .....?  €...塱
00000840h: 50 00 00 00 7E FE FF FF A8 00 00 00 38 FF FF FF ; P...~?  ?..8
```

hgame{Welc0m3_t0_R3_World!}

わかります

首先 IDA 看下

```
__int64 __fastcall main(__int64 a1, char **a2, char **a3)
{
  __int64 result; // rax@4
  __int64 v4; // rsi@4
  char input; // [sp+0h] [bp-40h]@1
  __int64 v6; // [sp+38h] [bp-8h]@1

  v6 = *MK_FP(__FS__, 40LL);
  memset(&input, 0, 0x30uLL);
  puts("You are a good Reverser!");
  puts(off_602250);
  puts("wakalimasu.Give me your starlight!");
  fgets(&input, 47, v602260);
  if ( (unsigned __int8)check(&input) )
    puts("you are top star!");
  else
    puts("non-non dayo~");
  result = 0LL;
  v4 = *MK_FP(__FS__, 40LL) ^ v6;
  return result;
}
```

就一个 check，进去看看

```
flag = 1;
str_len = strlen(input);
if ( str_len <= 37 )
{
  ptr1 = malloc1(36);
  ptr2 = malloc1(36);
  for ( i = 0; i < str_len; ++i )
  {
    ptr1[i] = (char)(input[i] >> 4);
    ptr2[i] = input[i] & 0xF;
  }
  v8 = matrix_mul((__int64)ptr1, (__int64)matrix3, 6);
  v9 = matrix_add((__int64)ptr2, (__int64)matrix3, 6);
  for ( j = 0; j <= 35; ++j )
  {
    if ( v8[j] != result1[j] || v9[j] != result2[j] )
      flag = 0;
  }
```

这里将输入的数据拆分放到两个数组里

```
  signed int j; // [sp+18h] [bp-28h]@6
  signed int str_len; // [sp+1Ch] [bp-24h]@1
  _DWORD *ptr1; // [sp+20h] [bp-20h]@3
  _DWORD *ptr2; // [sp+28h] [bp-18h]@3
  _DWORD *v8; // [sp+30h] [bp-10h]@6
  _DWORD *v9; // [sp+38h] [bp-8h]@6

  flag = 1;
  str_len = strlen(input);
  if ( str_len <= 37 )
  {
    ptr1 = malloc1(36);
    ptr2 = malloc1(36);
    for ( i = 0; i < str_len; ++i )
    {
      ptr1[i] = (char)(input[i] >> 4);
      ptr2[i] = input[i] & 0xF;
    }
    v8 = matrix_mul((__int64)ptr1, (__int64)matrix3, 6);
    v9 = matrix_add((__int64)ptr2, (__int64)matrix3, 6);
    for ( j = 0; j <= 35; ++j )
    {
      if ( v8[j] != result1[j] || v9[j] != result2[j] )
        flag = 0;
    }
    free(ptr1);
    free(ptr2);
    free(v8);
    free(v9);
    result = (unsigned __int8)flag;
  }
  else
  {
    result = 0LL;
  }
  return result;
}
```

继续跟进

```
_DWORD *__fastcall matrix_mul(__int64 ptr1, __int64 matrix3, int num_6)
{
  int v4; // [sp+Ch] [bp-34h]@1
  int i; // [sp+2Ch] [bp-14h]@1
  int j; // [sp+30h] [bp-10h]@2
  int k; // [sp+34h] [bp-Ch]@3
  _DWORD *v8; // [sp+38h] [bp-8h]@1

  v4 = num_6;
  v8 = malloc1(num_6);
  for ( i = 0; i < v4; ++i )
  {
    for ( j = 0; j < v4; ++j )
    {
      for ( k = 0; k < v4; ++k )
        v8[v4 * i + j] += *(_DWORD *)(4LL * (v4 * i + k) + ptr1) * *(_DWORD *)(4LL * (v4 * k + j) + matrix3);
    }
  }
  return v8;
}
```

这熟悉的感觉？难道说是线性代数!!!

```
_DWORD *__fastcall matrix_add(__int64 ptr2, __int64 matrix3, int num_6)
{
  int v4; // [sp+Ch] [bp-24h]@1
  int i; // [sp+20h] [bp-10h]@1
  int j; // [sp+24h] [bp-Ch]@2
  _DWORD *v7; // [sp+28h] [bp-8h]@1

  v4 = num_6;
  v7 = malloc1(num_6);
  for ( i = 0; i < v4; ++i )
  {
    for ( j = 0; j < v4; ++j )
      v7[v4 * i + j] = *(_DWORD *)(4LL * (v4 * i + j) + ptr2) + *(_DWORD *)(4LL * (v4 * i + j) + matrix3);
  }
  return v7;
}
```

看这一个十分明显就是矩阵加法，更让我确信了考的就是线性代数的矩阵加法和矩阵乘法

```
for ( j = 0; j <= 35; ++j )
{
  if ( v8[j] != result1[j] || v9[j] != result2[j] )
    flag = 0;
}
free(ptr1);
```

找到运算后的结果矩阵看看

```
.data:0000000000602120 result1         dd 122, 207, 140, 149, 142, 168
.data:0000000000602120                                         ; DATA XREF: ch
.data:0000000000602120                 dd 95, 201, 122, 145, 136, 167
.data:0000000000602120                 dd 112, 192, 127, 137, 134, 147
.data:0000000000602120                 dd 95, 207, 110, 134, 133, 173
.data:0000000000602120                 dd 136, 212, 160, 162, 152, 179
.data:0000000000602120                 dd 121, 193, 2 dup(126), 119, 147
.data:00000000006021B0                 dd 0
.data:00000000006021B4                 dd 0
.data:00000000006021B8                 dd 0
.data:00000000006021BC                 dd 0
.data:00000000006021C0 ; int result2[]
.data:00000000006021C0 result2         dd 16, 2 dup(8), 14, 6, 11, 5 ; DATA XR
.data:00000000006021C0                 dd 23, 5, 10, 12, 23, 14
.data:00000000006021C0                 dd 23, 19, 7, 8, 10, 4
.data:00000000006021C0                 dd 13, 22, 17, 11, 22, 6
.data:00000000006021C0                 dd 14, 2, 11, 18, 9, 5
.data:00000000006021C0                 dd 2 dup(8), 10, 16, 13
```

还有另外一个参与运算的矩阵

```
matrix3          dd 8, 1, 7, 2 dup(1), 0, 4 ; DATA
                                            ; check+DI
                 dd 8, 1, 2, 3, 9, 3
                 dd 8, 2 dup(6), 4, 8, 3, 5
                 dd 7, 2 dup(8), 7, 0, 9, 0
                 dd 2, 3, 4, 2, 3, 2
                 dd 5, 4, 0
                 dd 0
                 dd 0
                 dd 0
```

现在目的很明确了，需要反解处通过变形后的两个矩阵，再通过(ptr1[i])*16 + ptr2[i] 还原 input 的字符。本来想手工硬刚的，但是想到我辣鸡的线代，果断在网上找工具 orz

Ptr1:

	C_1	C_2	C_3	C_4	C_5	C_6
1	6	6	6	6	6	7
2	3	5	7	6	6	6
3	6	5	4	6	7	7
4	3	7	5	6	7	5
5	7	6	7	7	5	7
6	7	6	6	3	6	7

ptr2:

	C_1	C_2	C_3	C_4	C_5	C_6
1	8	7	1	13	5	11
2	1	15	4	8	9	14
3	11	15	13	1	4	2
4	1	8	15	9	3	15
5	6	5	2	9	15	5
6	3	5	6	5	12	13

加一下以 ascii 码形式输出得到 flag:

hgame{1_think_Matr1x_is_very_usef5l}

r & xor

IDA 打开

```
70  puts("Input the flag:");
71  __isoc99_scanf("%s", s);
72  if ( strlen(s) == 35 )
73  {
74    for ( i = 0; i < 35; ++i )
75    {
76      if ( s[i] != (v6[i] ^ *((_BYTE *)&v31 + i)) )
77      {
78        puts("Wrong flag , try again later!");
79        result = 0;
80        goto LABEL_9;
81      }
82    }
83    puts("You are right! Congratulations!!");
84    result = 0;
85  }
86  else
87  {
88    puts("Wrong flag , try again later!");
89    result = 0;
90  }
91LABEL_9:
92  v4 = *MK_FP(__FS__, 40LL) ^ v37;
93  return result;
94}
```

简单的异或判断，通过反推能解出 flag， 看看 v31 的位置

```
v37 = *MK_FP(__FS__, 40LL);
v31 = '0Y{emagh';
v32 = '_3byam_u';
v33 = '1ht_deen';
v34 = '!!!en0_s';
v35 = '}!!';
memset(v6, 0, 0x90uLL);
```

连起来是 hgame{Y0u_mayb3_need_th1s_0ne!!!!!}

```
int v6[6]; // [sp+10h] [bp-130h]@1
int v7; // [sp+28h] [bp-118h]@1
int v8; // [sp+30h] [bp-110h]@1
int v9; // [sp+38h] [bp-108h]@1
int v10; // [sp+3Ch] [bp-104h]@1
int v11; // [sp+40h] [bp-100h]@1
int v12; // [sp+44h] [bp-FCh]@1
int v13; // [sp+48h] [bp-F8h]@1
int v14; // [sp+4Ch] [bp-F4h]@1
int v15; // [sp+50h] [bp-F0h]@1
int v16; // [sp+54h] [bp-ECh]@1
int v17; // [sp+5Ch] [bp-E4h]@1
int v18; // [sp+60h] [bp-E0h]@1
int v19; // [sp+64h] [bp-DCh]@1
int v20; // [sp+68h] [bp-D8h]@1
int v21; // [sp+6Ch] [bp-D4h]@1
int v22; // [sp+70h] [bp-D0h]@1
int v23; // [sp+74h] [bp-CCh]@1
int v24; // [sp+78h] [bp-C8h]@1
int v25; // [sp+80h] [bp-C0h]@1
int v26; // [sp+84h] [bp-BCh]@1
int v27; // [sp+88h] [bp-B8h]@1
int v28; // [sp+8Ch] [bp-B4h]@1
int v29; // [sp+90h] [bp-B0h]@1
int v30; // [sp+94h] [bp-ACh]@1
```

```
45  memset(v6, 0, 0x90uLL);
46  v7 = 1;
47  v8 = 7;
48  v9 = 92;
49  v10 = 18;
50  v11 = 38;
51  v12 = 11;
52  v13 = 93;
53  v14 = 43;
54  v15 = 11;
55  v16 = 23;
56  v17 = 23;
57  v18 = 43;
58  v19 = 69;
59  v20 = 6;
60  v21 = 86;
61  v22 = 44;
62  v23 = 54;
63  v24 = 67;
64  v25 = 66;
65  v26 = 85;
66  v27 = 126;
67  v28 = 72;
68  v29 = 85;
69  v30 = 30;
```

V6 的位置先被初始化为零，然后在对其中某些位置数据进行修改，具体位置就是 v7~v30 相对 v6 的位置

```
#include"stdio.h"
#include"stdlib.h"
int main()
{
    int i;
    char s[] = "hgame{Y0u_mayb3_need_th1s_0ne!!!!!}";
    int key[] = { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01,
                  0x00, 0x07, 0x00, 0x5c, 0x12, 0x26, 0x0B,
                  0x5D, 0x2B, 0x0B, 0x17, 0x00, 0x17, 0x2B,
                  0x45, 0x06, 0x56, 0x2C, 0x36, 0x43, 0x00,
                  0x42, 0x55, 0x7E, 0x48, 0x55, 0x1E, 0x00 };
    for (i = 0; i < 35; i++)
        printf("%c", s[i]^key[i]);
    printf("\n");
    system("pause");
    return 0;
}
```

```
G:\hgame\week1\bin\Project1\Debug\Project1.exe
hgame{X0r_1s_interest1ng_isn't_it?}
请按任意键继续. . .
```

Flag: hgame{X0r_1s_interest1ng_isn't_it?}

Pro 的 Python 教室(一)

```
import base64
import hashlib

enc1 = 'hgame{'
enc2 = 'SGVyZY8xc18zYXN5Xw==' 
enc3 = 'Pyth0n}'

print 'Welcome to Processor\'s Python Classroom!\n'
print 'Here is Problem One.'
print 'There\'re three parts of the flag.'

print '--------------------------------'

print 'Plz input the first part:'
first = raw_input()
if first == enc1:
    pass
else:
    print 'Sorry , You\'re so vegatable!'
    exit()

print 'Plz input the secend part:'
secend = raw_input()
secend = base64.b64encode(secend)
if secend == enc2:
    pass
else:
    print 'Sorry , You\'re so vegatable!'
    exit()

print 'Plz input the third part:'
third = raw_input()
third = base64.b32decode(third)
if third == enc3:
    pass
else:
    print 'Sorry , You\'re so vegatable!'
    exit()

print 'Oh, You got it !'
```

很简单的逻辑，将标红出 base64 解码下拼接得到 flag:
hgame{ Here_1s_3asy_ Pyth0n}

PWN

aaaaaaaaaa

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  signed int v3; // eax@4
  signed int v5; // [sp+Ch] [bp-4h]@1

  setbuf(_bss_start, 0LL);
  signal(14, handle);
  alarm(0xAu);
  puts("Welcome to PWN'world!let us aaaaaaaaaa!!!");
  v5 = 0;
  while ( 1 )
  {
    v3 = v5++;
    if ( v3 > 99 )
      break;
    if ( getchar() != 97 )
      exit(0);
  }
  system("/bin/sh");
  return 0;
}
```

很简单的逻辑，发送 100 个 a 就能拿到 shell 了（之前没接触过 pwn 结果拿到 shell 了还傻 fufu 的瞎搞，问了徐大哥才发现 ls 后就能看到一个 flag 的文件，打开就是）

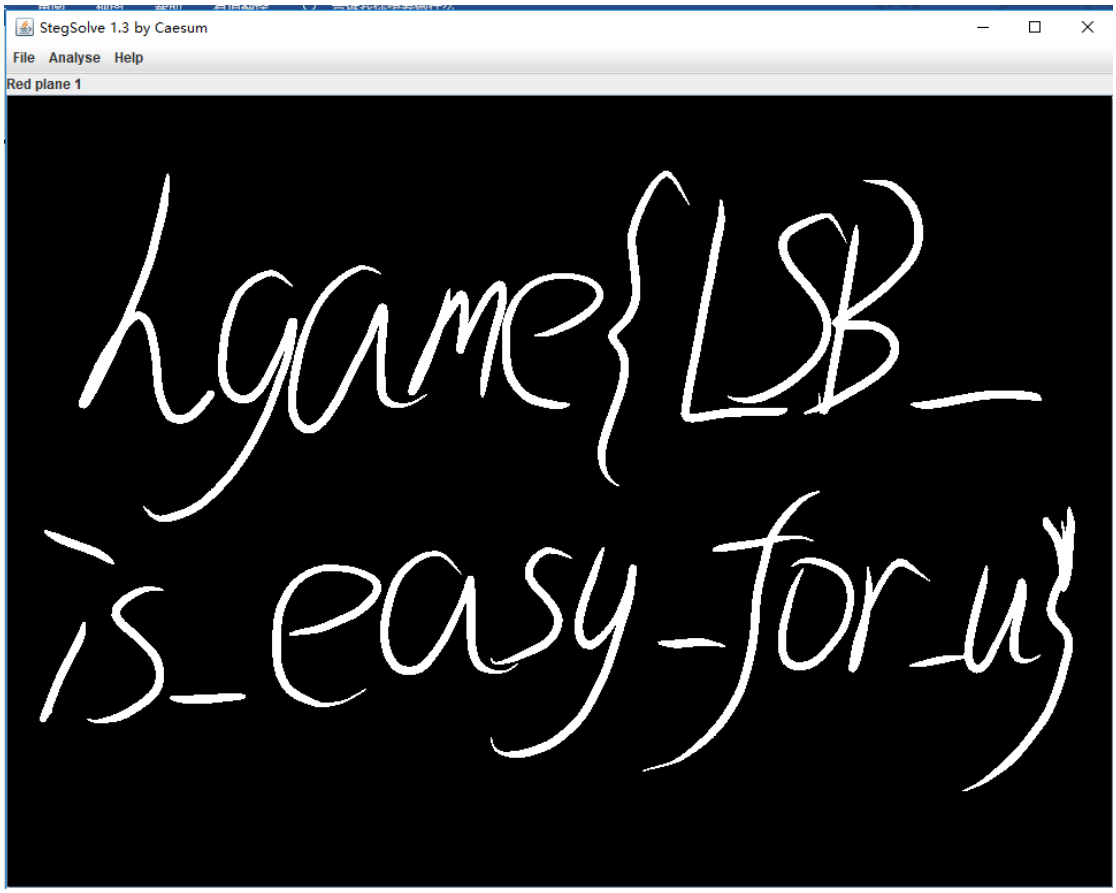
附上脚本

```
gonbe5@ubuntu:/home/gonbe5/Desktop$ python pwn1.py
[+] Opening connection to 118.24.3.214 on port 9999: Done
[*] Switching to interactive mode
$ ls
aaaaaaaaaa
bin
dev
flag
lib
lib64
run.sh
$ cat flag
hgame{Aa4_4aA_4a4aAAA}$
```

hgame{Aa4_4aA_4a4aAAA}

misc

Hidden Image in LSB



丢进 stegsolve 得到 flag
hgame{LSB_is_easy_for_u}

打字机

Google 搜出这个图片 直接翻译 flag
hgame{My_vi0let_tyPewRiter}

Broken Chest

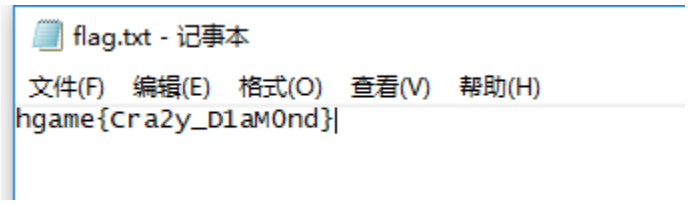
根据资料,先用 UE 打开修复 ZIP 头

```
00000000h: 50 4B 03 04 14 00 09 00 08 00 55 BB 35 4E CE 7C ; PK........U?N蜊
00000010h: B3 B0 22 00 00 00 14 00 00 00 08 00 00 00 66 6C ; 嘲"...........fl
00000020h: 61 67 2E 74 78 74 67 49 3F 48 A0 BE 53 8B 38 E4 ; ag.txtgI?H牼S??
00000030h: 5A 42 49 02 08 5D 55 A6 4A 67 B2 B3 CE B0 6E C1 ; ZBI..]U  g渤伟n?
00000040h: 0B 85 DC EB 4F 91 4D BF 50 4B 07 08 CE 7C B3 B0 ; .呠隣慚縋K..蜊嘲
00000050h: 22 00 00 00 14 00 00 00 50 4B 01 02 1F 00 14 00 ; ".......PK......
00000060h: 09 00 08 00 55 BB 35 4E CE 7C B3 B0 22 00 00 00 ; ....U?N蜊嘲"...
00000070h: 14 00 00 00 08 00 24 00 00 00 00 00 00 00 20 00 ; ......$....... .
00000080h: 00 00 00 00 00 00 66 6C 61 67 2E 74 78 74 0A 00 ; ......flag.txt..
00000090h: 20 00 00 00 00 00 01 00 18 00 3E 2C 76 B6 9D B1 ;  .........>,v棋?
000000a0h: D4 01 3E 2C 76 B6 9D B1 D4 01 1D F1 7E C5 9C B1 ; ?>,v棋痹..駘艤?
000000b0h: D4 01 50 4B 05 06 00 00 00 00 01 00 01 00 5A 00 ; ?PK..........Z.
000000c0h: 00 00 58 00 00 00 10 00 53 30 6D 45 54 68 31 6E ; ..X.....S0mETh1n
000000d0h: 67 5F 55 35 65 66 75 4C                         ; g_U5efuL
```

解压发现需要密码，上 ARCHPR 硬刚
得到一个 flag.txt 发现同样需要密码
刚刚在 UE 里面发现一个很可疑的字符串

```
00000090h: 20 00 00 00 00 00 01 00 18 00 3E 2C 76 B6 9D B1 ;  .........>,v棋?
000000a0h: D4 01 3E 2C 76 B6 9D B1 D4 01 1D F1 7E C5 9C B1 ; ?>,v棋痹..駘艤?
000000b0h: D4 01 50 4B 05 06 00 00 00 00 01 00 01 00 5A 00 ; ?PK..........Z.
000000c0h: 00 00 58 00 00 00 10 00 53 30 6D 45 54 68 31 6E ; ..X.....S0mETh1n
000000d0h: 67 5F 55 35 65 66 75 4C                         ; g_U5efuL
```

输入 S0mETh1ng_U5EfuL 顺利解压



Flag: hgame{Cra2y_D1aM0nd}