Hgame week1 writeup

2019年2月2日 18:11

Re:

0x01 Hello Re

签到题,拉进IDA ,搞定;

```
__int64 __fastcall main(__int64 a1, char **a2, char **a3)
{
  char s[8]; // [rsp+0h] [rbp-30h]
  __int64 v5; // [rsp+8h] [rbp-28h]
  __int64 v6; // [rsp+10h] [rbp-20h]
  __int64 v7; // [rsp+18h] [rbp-18h]
  unsigned __int64 v8; // [rsp+28h] [rbp-8h]

  v8 = __readfsqword(0x28u);
  *(_QWORD *)s = 0LL;
  v5 = 0LL;
  v6 = 0LL;
  v7 = 0LL;
  puts("Please input your key:");
  fgets(s, 32, stdin);
  s[strlen(s) - 1] = 0;
  if ( !strcmp(s, "hgame{Welc0m3_t0_R3_World!}") )
    puts("success");
  else
    puts("failed..");
  return 0LL;
}
```

0x02 Pro的python教室(一)

签到2.0

```
import base64
import hashlib

enc1 = 'hgame{Here_1s_3asy_'
enc2 = 'SGVyZV8xc18zYXN5Xw=='
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 second 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 !'
```

0x03 r&xor

拖进IDA,根据字符限制可以看出flag是35个字符,s[i] == a[i]^b[i]

```
  puts("Input the flag:");
  __isoc99_scanf("%s", s);
  if ( strlen(s) == 35 )
  {
    for ( i = 0; i < 35; ++i )
    {
      if ( s[i] != (v5[i] ^ *((char *)&v30 + i)) )
      {
        puts("Wrong flag , try again later!");
        return 0;
      }
    }
    puts("You are right! Congratulations!!");
    result = 0;
  }
  else
  {
    puts("Wrong flag , try again later!");
    result = 0;
  }
  return result;
}
```

之后写出逆算法

```
#include<stdio.h>

int main()
{
  int a[] = {00,00,00,00,00,00,01,00,07,00,
             92,18,38,11,93,43,11,23,
             00,23,43,69,06,86,44,54,67,
             00,66,85,126,72,85,30,00};
  int b[] = {0x68,0x67,0x61,0x6d,0x65,0x7b,0x59,0x30,
             0x75,0x5f,0x6d,0x61,0x79,0x62,0x33,0x5f,
             0x6e,0x65,0x65,0x64,0x5f,0x74,0x68,0x31,
             0x73,0x5f,0x31,0x6e,0x65,0x21,0x21,0x21,
             0x21,0x21,0x7d};
    /*
    int b[] = {0x30,0x59,0x7b,0x65,0x6d,0x61,0x67,0x68,
               0x5f,0x33,0x62,0x79,0x61,0x6d,0x5f,0x75,
               0x31,0x68,0x74,0x5f,0x64,0x65,0x65,0x6e,
               0x21,0x21,0x21,0x65,0x6e,0x30,0x5f,0x73,
               0x7d,0x21,0x21};
    */
    int i;
    int s[35];
   for(i = 0 ;i < 35 ; i++)
   {
     s[i] = a[i]^b[i];
     printf("%c",s[i]);
   }
  return 0;
}
```

得到flag:hgame{X0r_1s_interest1ng_isn't_it?}

0x04 brainfxxker

第一眼望去,wtf?

大概查了一下,了解了brainfxxk这种语言,其实就不难了

只要做到输入过程中无输出,所输入的即为flag

```
}


int main() {
  bf::Parser parser;
  parser.execute(",>++++++++++[<---------->-]<++[
  system("pause");
}
/*,>++++++++++[<---------->-]<++[+.]
,>+++++++++[<--------->-]<-[+.]
,>+++++++[<------->-]<---[+.]
,>++++++[<------>-]<+++[+.]
,>+++++++++[<---------->-]<++[+.]
,>++++++++++[<---------->-]<--[+.]
,>++++++++++[<-------->-]<-----[+.]
,>++++++++++[<---------->-]<+[+.]
,>+++++++++[<-------->-]<---[+.]
bR4!NfUcK
*/
```

最开始理解成了输出的ascii为0,和正确的flag都差1,py了一下o爷爷,才终于改对了

Flag:hgame{bR4!NfUcK}

0x05 わかります

反汇编 得到程序

可以得到flag的长度为36个字符

并且在p0[]中存储着flag的ascii的高位,P1[]中存着低位

而由函数f3可知 p1[i] 等于 b[i] - a2[i] (a1[],a2[],b[]均为已知)

```
__int64 __fastcall f1(const char *a1)
{
  unsigned __int8 flag; // [rsp+13h] [rbp-2Dh]
  signed int i; // [rsp+14h] [rbp-2Ch]
  signed int j; // [rsp+18h] [rbp-28h]
  signed int n; // [rsp+1Ch] [rbp-24h]
  void * p0; // [rsp+20h] [rbp-20h]
  void * p1; // [rsp+28h] [rbp-18h]
  void * p2; // [rsp+30h] [rbp-10h]
  void * p3; // [rsp+38h] [rbp-8h]

  flag = 1;
  n = strlen(a1);
  if ( n > 37 )
    return 0;
  p0 = set(36);                          // 分配指针空间并初始化

  p1 = set(36);                          //36*36*4
  for ( i = 0; i < n; ++i )
  {
    *(int *)(p0 + 4 * i) = (char)(a1[i] >> 4);//flag每个右移四位,高位补零
    *(int *)(p1 + 4 * i) = a1[i] & 0xF;   //p1中以特殊方式存着flag,0,4
  }
  p2 = (__int64)f2(p0, (__int64)&void, 6);//p2,p3占的大小均为36x4字节
  p3 = (__int64)f3(p1, (__int64)&void, 6);
  for ( j = 0; j <= 35; ++j )
  {
    if ( *(_DWORD *)(4LL * j + p2) != A[j] || *(_DWORD *)(4LL * j + p3) != B[j] )
      flag = 0;
  }
  free((void *)p0);
  free((void *)p1);
  free((void *)p2);
  free((void *)p3);
  return flag;
}
```

```
int* f2(__int64 a1, __int64 a2, int a3)
{
  int n; // [rsp+Ch] [rbp-34h]
  int i; // [rsp+2Ch] [rbp-14h]
  int j; // [rsp+30h] [rbp-10h]
  int k; // [rsp+34h] [rbp-Ch]
  _DWORD *a; // [rsp+38h] [rbp-8h]

  n = a3;
  a = set(a3);
  for ( i = 0; i < n; ++i )
  {
    for ( j = 0; j < n; ++j )
    {
      for ( k = 0; k < n; ++k )
        a[n * i + j] += *(4 * (n * i + k) + a1) * *(4LL * (n * k + j) + a2);
    }
  }
  return a;
}

int* f3(int* a1, int * a2, int a3)
{
  int v4; // [rsp+Ch] [rbp-24h]
  int i; // [rsp+20h] [rbp-10h]
  int j; // [rsp+24h] [rbp-Ch]
  _DWORD *b; // [rsp+28h] [rbp-8h]

  v4 = a3;                                    // 6
  b = set(6);                                 // 6
                                              // 共分配了6*6*4
  for ( i = 0; i < 6; ++i )
  {
    for ( j = 0; j < 6; ++j )
      b[6 * i + j] = *(4 * (6 * i + j) + a1) + *(4 * (6 * i + j) + a2);
  }
  return b;
}
```

由函数f2 可知 p[1] 是由a[]组成的矩阵左乘a2[]的逆矩阵得到

$$\begin{bmatrix} 122 & 207 & 140 & 149 & 142 & 168 \\ 95 & 201 & 122 & 145 & 136 & 167 \\ 112 & 12 & 127 & 137 & 134 & 147 \\ 95 & 207 & 110 & 134 & 133 & 173 \\ 136 & 212 & 160 & 162 & 152 & 179 \\ 121 & 193 & 126 & 126 & 119 & 147 \end{bmatrix} \cdot \begin{bmatrix} \frac{482}{8477} & \frac{1324}{8477} & \frac{-461}{8477} & \frac{-660}{8477} & \frac{-902}{8477} & \frac{192}{1211} \\ \frac{575}{16954} & \frac{-1871}{33908} & \frac{1503}{33908} & \frac{-2243}{33908} & \frac{5129}{33908} & \frac{9}{1211} \\ \frac{1399}{16954} & \frac{-5997}{33908} & \frac{2389}{33908} & \frac{2887}{33908} & \frac{3663}{33908} & \frac{-254}{1211} \\ \frac{-1947}{16954} & \frac{1087}{33908} & \frac{8297}{33908} & \frac{-6381}{33908} & \frac{-7873}{33908} & \frac{376}{1211} \\ \frac{821}{16954} & \frac{395}{33908} & \frac{-11771}{33908} & \frac{9535}{33908} & \frac{5967}{33908} & \frac{-143}{1211} \\ \frac{-468}{8477} & \frac{1685}{16954} & \frac{649}{16954} & \frac{543}{16954} & \frac{-1801}{16954} & \frac{-101}{1211} \end{bmatrix} = \begin{bmatrix} 6 & 6 & 6 & 6 & 6 & 7 \\ 3 & 5 & 7 & 6 & 6 & 6 \\ \frac{-888}{8477} & \frac{126580}{8477} & \frac{-33727}{8477} & \frac{151797}{8477} & \frac{-171466}{8477} & \frac{6857}{1211} \\ 3 & 7 & 5 & 6 & 7 & 5 \\ 7 & 6 & 7 & 7 & 5 & 7 \\ 7 & 6 & 6 & 3 & 6 & 7 \end{bmatrix}$$

虽然可能是a[2]错了一个数导致矩阵的结果错了一行,不过好在知道个位,根据题意,猜的matrix通过XD.(M和m 的ascii码的个位是一样的,试了好久)

Flag: hgame{1_think_Matrix_is_very_usef5l}

```
68/67/61/6d/65/7b/31/5f/74/68/69/6e/
h  g  a  m  e  {  1  _  t  h  i  n
 b/ f/ d/ 1/ 4/ 2/31/78/5f/69/73/5f/
k  _  M  a  t  r  i  x  _  i  s  _
76/65/72/79/5f/75/73/65/66/35/6c/7d/
v  e  r  y  _  u  s  e  f  5  l  }
```

PWN:

0x01: aaaaaaa

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

  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;
}
```

签到题 a就完事了

```
from pwn import *

r = remote('118.24.3.214',9999)
r.sendline('a'*108)
r.interactive()
```

本来想肝一波pwn的,结果,因为眼睛要做手术,手机电脑都被没收了,week1在周日就提前结束,有点可惜

Web:

0x01:very easy web

把"vidar" Url二次编码就ok了

```
<?php
error_reporting(0);
include("flag.php");

if(strpos("vidar",$_GET['id'])!==FALSE)
    die("<p>干巴爹</p>");

$_GET['id'] = urldecode($_GET['id']);
if($_GET['id'] === "vidar")
{
    echo $flag;
}
highlight_file(__FILE__);
?>
```

```
hgame{urlDecode_Is_GoOd} <?php
error_reporting(0);
include("flag.php");

if(strpos("vidar",$_GET['id'])!==FALSE)
    die("<p>干巴爹</p>");

$_GET['id'] = urldecode($_GET['id']);
if($_GET['id'] === "vidar")
{
    echo $flag;
}
highlight_file(__FILE__);
?>
```

Crypto:

0x01:Mix

根据题名,应该是混合了几种密码

先将摩斯翻译得

744b735f6d6f7944716b7b6251663430657d

在根据ascii可得

tKs_moyDqk{bQf40e}

根据flag的结构,应该是栅栏,解得

tsmyq{Q4eK_oDkbf0}

接下来是显而易见的凯撒密码,解得

hgame{E4sY_cRypt0}