RE-国赛2018 task_reverse_01

原创

Alikas 于 2019-04-17 20:15:53 发布 227 收藏

分类专栏： 逆向 文章标签： RE writeup 国赛2018

本文链接：https://blog.csdn.net/q83182034/article/details/89363792

Alikas-0x09

题目：国赛2018 task_reverse_01

备战国赛，做下来发现自己就是个蒟蒻...。

main()

```
v5 = "CISCN{";
v6 = 6LL;
v7 = &buf;
do
{
  if ( !v6 )
    break;
  v3 = *v5 < (unsigned __int8)*v7;
  v4 = *v5++ == *v7++;
  --v6;
}
if ( (!v3 && !v4) != v3 )
    return 0xFFFFFFFFLL;
```

检查前六个字符是否为CISCN{

```
  v11 = strtok(&s, "_");
```

输入的字符用“_”隔开，这是v11 = 第一部分，设为flag1

```
v12 = (const char *)dest;
memcpy(dest, v11, strlen(v11));
```

将flag1传给dest，v12指向dest

```
v13 = 1;
do
  {
    ++v13;
    v14 = strtok(0LL, "_");
    if ( !v14 )
      return 0xFFFFFFFFLL;
    memcpy((void *)v8[2], v14, strlen(v14));
    ++v8;
  }
  while ( v13 != 3 );
```

猜测flag由三部分组成，为CISCN{flag1_flag2_flag3}

v12指向存储flag1的dest;

而在前文有`v8 = &v16;`所以v8[2]即为v18，故v18为flag2

v19为flag3

```
if ( (unsigned int)check1(v12) )
    return 0xFFFFFFFFLL;
if ( (unsigned int)check2(v18) )
    return 0xFFFFFFFFLL;
if ( (unsigned int)check3(v19) )
    return 0xFFFFFFFFLL;
puts("Congratulations!");
```

接下来是三个判断语句，分别验证flag1、flag2、flag3

[为了方便，我改了一下函数名]

check1(flag1）

```
sub_400876(&v10);
sub_4008A0(&v10);
sub_40108B((unsigned int *)&v10, (char *)a1, strlen(a1));
sub_401170(&v10, &v11);
sub_401285((__int64)v13, &v11, 16);
qword_603640 = v11;
qword_603648 = v12;
```

这一段是我最难逆向分析的一段，这里函数调用函数，调用了很多个函数。分析了好久分析不出来。

然后…就去看大佬的writeup了0.0

大佬猜测为md5加密运算，理由如下：

`大佬1：a1传过来的为字符串，经过一堆函数运算之后却变成了md5值，故猜测这堆函数为md5运算`

`大佬2:32位算法，连续四个赋值常数运算，进到函数之中还有右移4位等操作，故猜测md5加密`

略过这一步，后面的问题迎刃而解

```
for ( i = 0LL; ; ++i )
  {
    v3 = strlen(v13) + 1;
    v4 = i < v3 - 1;
    v5 = i == v3 - 1;
    if ( i >= v3 - 1 )
      break;
    v2 = v13[i];
    if ( (unsigned __int8)(v2 - 'A') <= 5u )
      v13[i] = (signed int)i % 10 + v2;
  }
  v6 = v13;
  v7 = "9F925J9341B490FKJ3J4C4ED3G0J1NF2";
  v8 = 32LL;
  do
  {
    if ( !v8 )
      break;
    v4 = *v7 < (unsigned __int8)*v6;
    v5 = *v7++ == *v6++;
    --v8;
  }
  while ( v5 );
  return (unsigned int)-((!v4 && !v5) != v4);
```

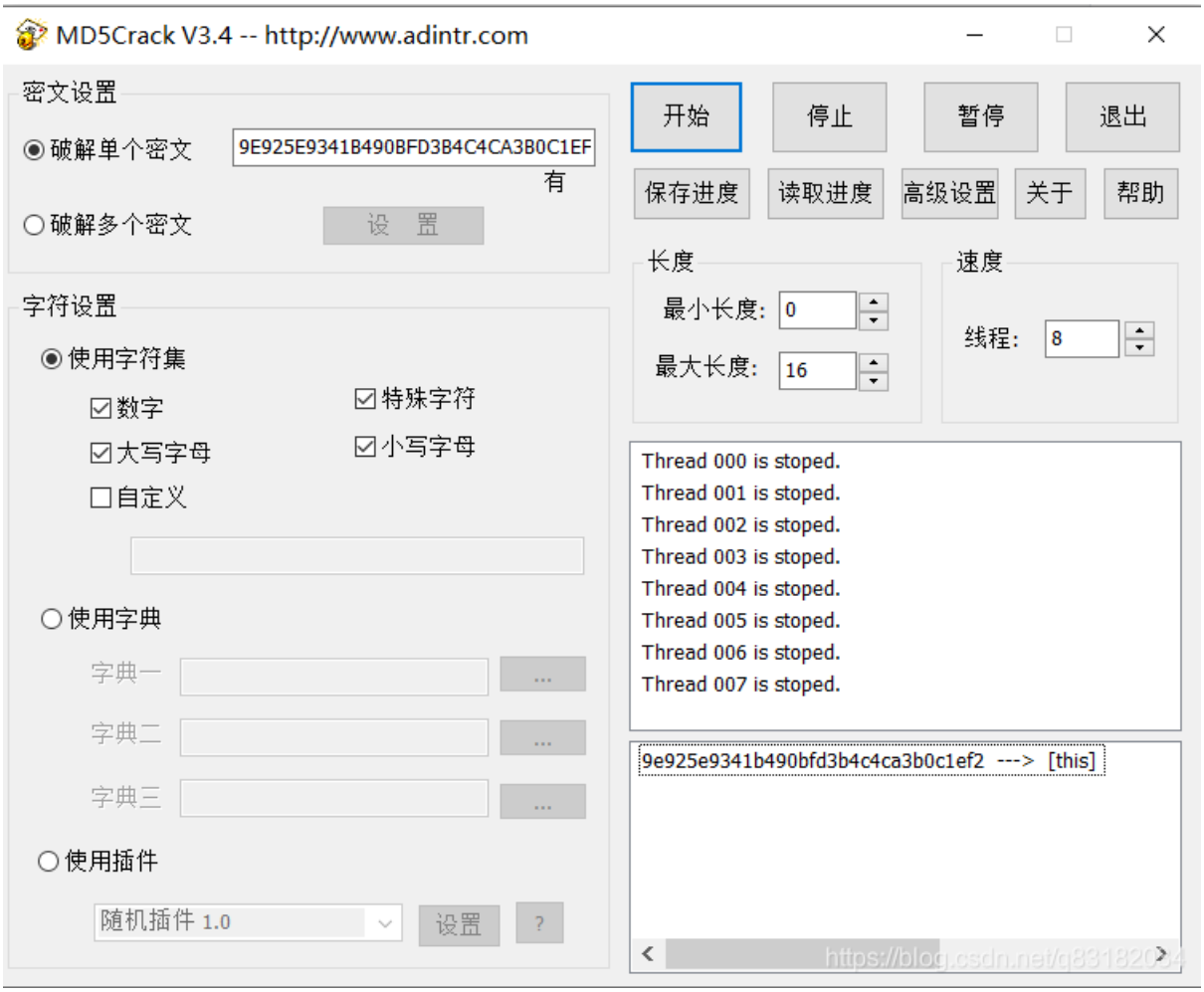
要return 0,猜测v5 = 1，v4 = 0，则do-while循环中，flag1的md5值经过for循环再一步运算后的值应该与

`9F925J9341B490FKJ3J4C4ED3G0J1NF2`相同

则第一部分解密算法为：

```
def check1():
    key = "9F925J9341B490FKJ3J4C4ED3G0J1NF2"
    flag1md5 = ''
    for i in range(len(key)):
        temp = ord(key[i])-i%10
        if temp <= ord('A') + 5 and temp >= ord('A'):
            flag1md5 += chr(temp)
        else:
            flag1md5 += key[i]
    print flag1md5
check1()
```

解出来后是`9E925E9341B490BFD3B4C4CA3B0C1EF2`，扔到md5爆破软件，解出flag1 = this



check2(flag2)

依照check1()，check2()核心算法：

```
v1 = 0LL;
 do
 {
   *((_BYTE *)&v12 + v1) ^= byte_603610[v1];
   ++v1;
 }
 while ( v1 != 16 );
 sub_401285((__int64)v14, &v12, 16);//调试，直接看出这个函数是将16进制数变成字符，例如0x6c->'6','c'
 for ( i = 0LL; ; ++i )
 {
   v4 = strlen(v14) + 1;
   v5 = i < v4 - 1;
   v6 = i == v4 - 1;
   if ( i >= v4 - 1 )
     break;
   v3 = v14[i];
   if ( (unsigned __int8)(v3 - 'A') <= 5u )
     v14[i] = (signed int)i % 10 + v3;
 }
 v7 = v14;
 v8 = "9F925J9341B490FKJ3J4C4ED3G0J1NF2";
 v9 = 32LL;
```

加密多了一块，剩下的与check1()一样,解密代码如下:

```
def check2():
    xor = [0xF2,0x1D,0x98,0xBC,0xCC,0x71,0xA1,0x39,0x1F,0x1F,0x11,0xE2,0x91,0x7C,0xA2,0x70]#数组byte_603610[]
    key = "9E925E9341B490BFD3B4C4CA3B0C1EF2"
    key = key.decode('hex')
    flag2md5 = ''
    for i in range(16):
        flag2md5 += chr(ord(key[i]) ^ xor[i]).encode('hex')
    print flag2md5
check2()
```

解得`6c8fc62f8dc53186ccabd528aa70bc82`，爆破得`f1rs`

check3(flag3)

代码前面一大段与check1()与check2()相同，就key不同，就不赘述了。

而flag3中重点在于这段代码：

```
if ( (!v6 && !v7) == v6 )
  {
    v12 = fopen("flag", "w+");
    if ( v12 )
    {
      v13 = &qword_603630;
      v14 = 0;
      do
      {
        v14 += *(unsigned __int8 *)v13;
        v13 = (__int64 *)((char *)v13 + 1);
      }
      while ( &qword_603640 != v13 );
      v15 = v1[3] ^ (v14 >> 4);
      v16 = v1[4] ^ v14 & 0xF;
      v17 = 0LL;
      do
      {
        if ( v17 & 1 )
          byte_6020E0[v17] ^= v16;
        else
          byte_6020E0[v17] ^= v15;
        ++v17;
      }
      while ( v17 != 5424 );
      fwrite(byte_6020E0, 0x1530uLL, 1uLL, v12);
      fclose(v12);
      result = 0LL;
    }
    else
    {
      result = 0xFFFFFFFFLL;
    }
  }
```

因为v15、v16不知道，但是知道他们是一个“char”类型的数，一个字节，范围在0~255之间，那么就直接爆破文件。256*256次种组合中一定有一次组合是我们需要的数据。脚本如下：

```python
def data():
    f1 = open("hex.txt",'r')  #存放数组byte_6020E0[5424]
    f2 = open("result",'wb')
    for i in range(256):
        for j in range(256):
            for k in range(5424):
                tmp = int(f1.readline(),16)
                if k&1 == 0:
                    f2.write(chr(tmp^i))
                else:
                    f2.write(chr(tmp^j))
            f2.write('\n')
            f1.seek(0) #回到f1文件头

    f2.close()
data()
```

跑了有一会而，文件大小`339 MB`0.0…

打开010Editor，搜索各种文件的文件头0…0…最后发现出题人隐藏的是一张jpg图片。文件头`FF D8 FF E0 00 10 4A 46 49 46 00 01`，从文件头到文件尾`FF D9`截取出来，修改一下文件类型。

得到flag。

最终flag就是`CISCN{this_f1rs_Zer0woRdThyes@T}`

最后，大佬补充说可以直接写成256*256个文件，在ubuntu里面格式是对的，可以显示图标0.0
学到了学到了0.0

总结：做这题大概花了5,6个小时，主要还是基础不牢固，以及…么得脑洞，太菜了…期间各种查资料，各种问大佬，至少，收益匪浅吧！