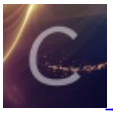
【reversing.kr逆向之旅】Position的writeup

原创

iqiqiya 于 2018-11-15 17:27:33 发布 536 收藏
分类专栏： 我的逆向之路 我的CTF之路 ------reversing.kr 文章标签： 【reversing.kr逆向之旅】Position的wri Position的writeup writeup reversing.kr 逆向


我的逆向之路 同时被 3 个专栏收录
108 篇文章 10 订阅
订阅专栏


我的CTF之路
92 篇文章 5 订阅
订阅专栏


------reversing.kr
11 篇文章 0 订阅
订阅专栏

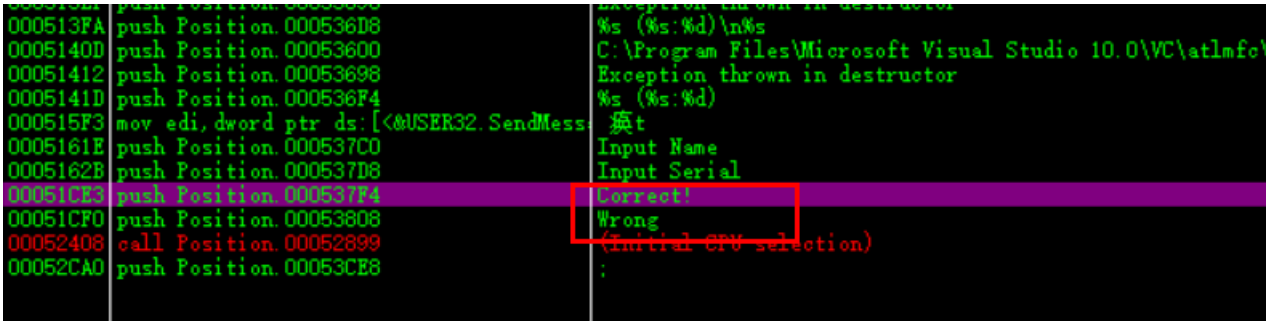
有提示是说flag就是当Serial为76876-77776时的Name　有多解　提示有四位 且最后一位是p

> ReversingKr KeygenMe
>
> Find the Name when the Serial is 76876-77776
> This problem has several answers.
>
> Password is ***p

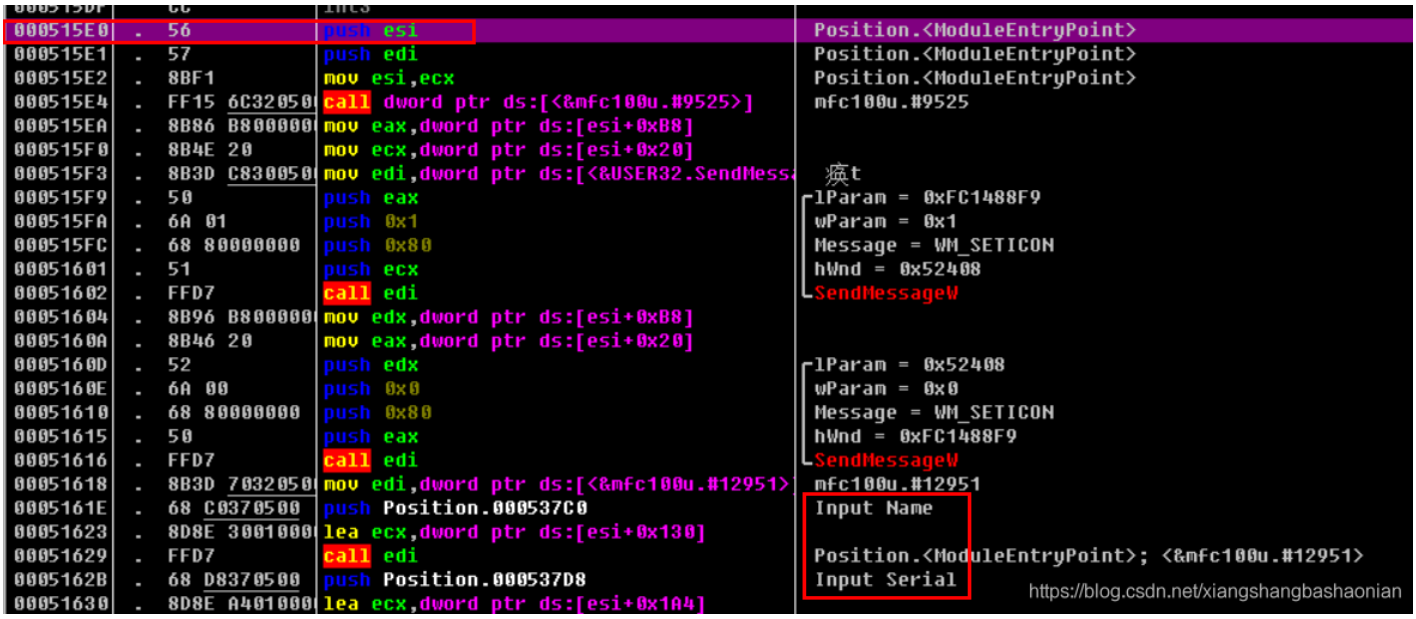
PEiD查不到壳　于是IDA载入

shift+f12找不到什么关键的字符串

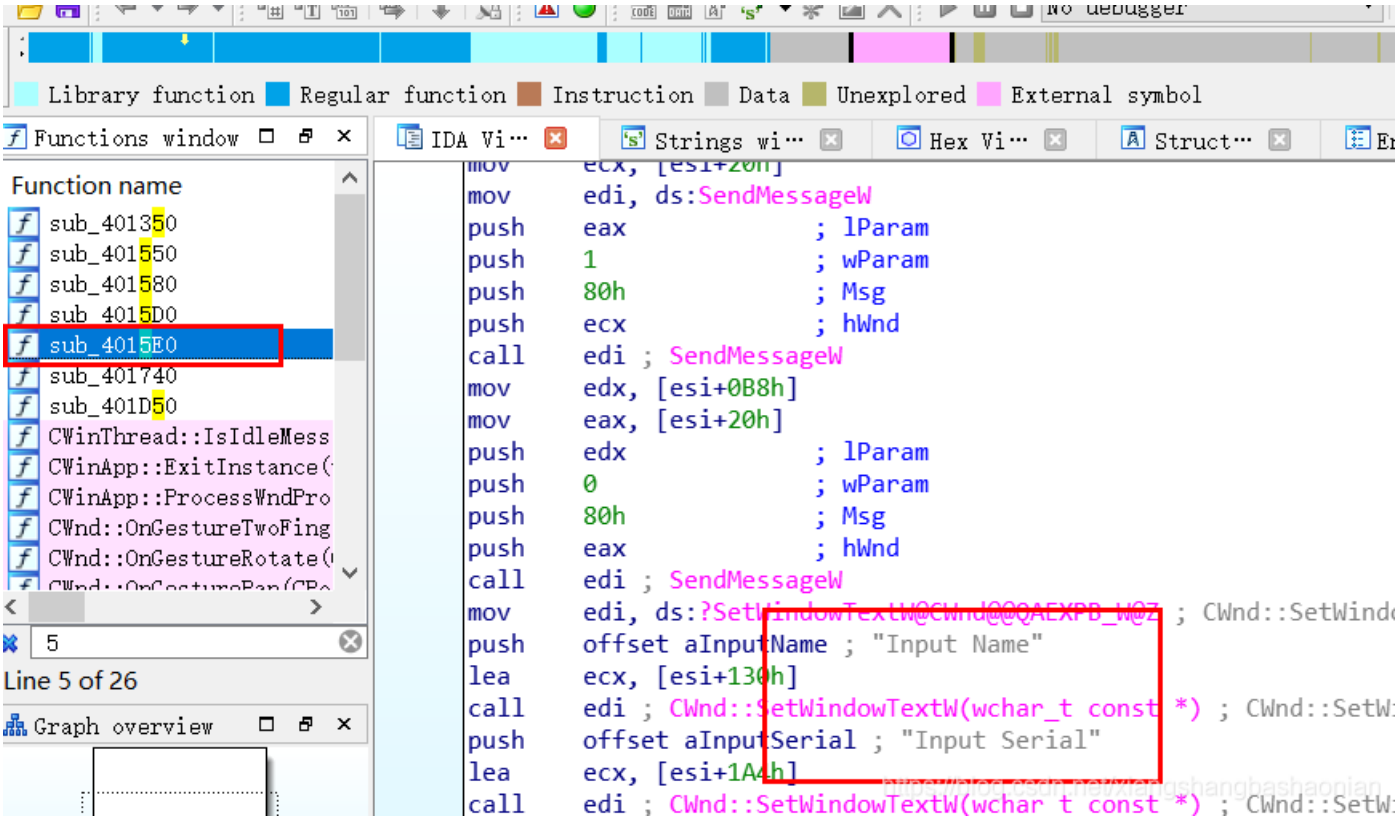
于是用OD载入　发现可以找到关键字符串



双击Input Name　找到函数开始的地址

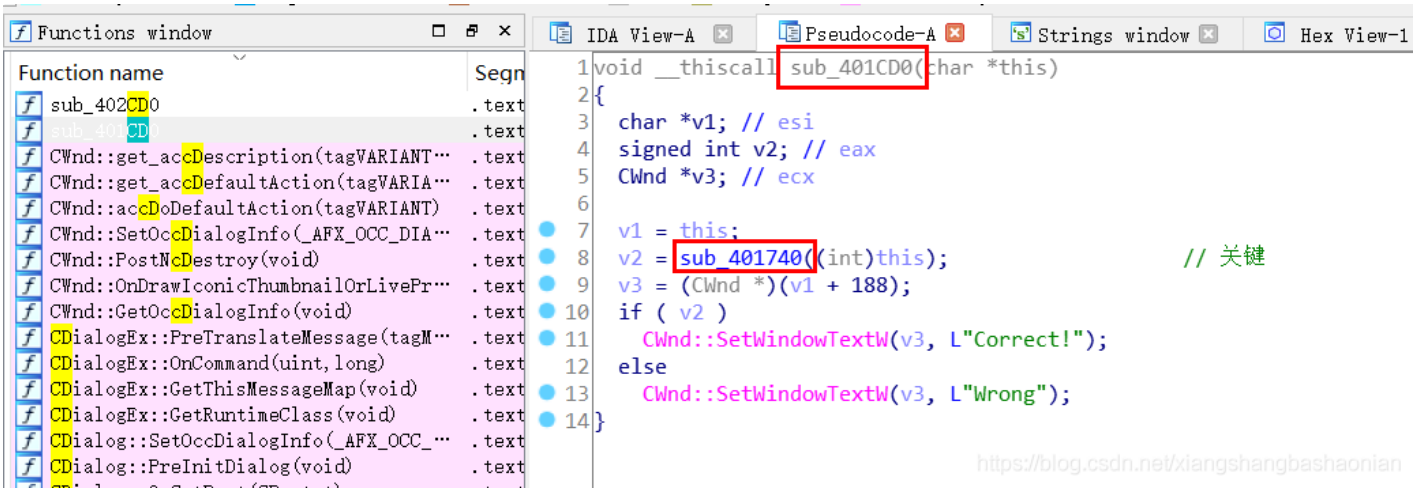


在IDA的函数列表进行过滤



然后F5发现这里没什么用

那就找correct的函数开始地址　F5



发现sub_401740()这个函数处理了我们的输入input 之后将返回值赋给v2 从而判断是否正确

双击进入 发现有API获取输入

```
CWnd::GetWindowTextW(a1 + 304, &v50);
```

一共有两句 v50,v51猜测就是Name,Serial

```
signed int __stdcall sub_401740(int a1)
{
  int v1; // edi
  int v3; // esi
  int v4; // esi
  __int16 v5; // bx
  unsigned __int8 v6; // al
  unsigned __int8 v7; // ST2C_1
  unsigned __int8 v8; // al
  unsigned __int8 v9; // bl
  wchar_t *v10; // eax
  __int16 v11; // di
  wchar_t *v12; // eax
  __int16 v13; // di
  wchar_t *v14; // eax
  __int16 v15; // di
  wchar_t *v16; // eax
  __int16 v17; // di
  wchar_t *v18; // eax
  __int16 v19; // di
  unsigned __int8 v20; // al
  unsigned __int8 v21; // ST2C_1
  unsigned __int8 v22; // al
  unsigned __int8 v23; // bl
  wchar_t *v24; // eax
  __int16 v25; // di
  wchar_t *v26; // eax
  __int16 v27; // di
  wchar_t *v28; // eax
  __int16 v29; // di
  wchar_t *v30; // eax
  __int16 v31; // di
  wchar_t *v32; // eax
  __int16 v33; // si
  unsigned __int8 v34; // [esp+10h] [ebp-28h]
```

```
  unsigned __int8 v35; // [esp+10h] [ebp-28h]
  unsigned __int8 v36; // [esp+11h] [ebp-27h]
  unsigned __int8 v37; // [esp+11h] [ebp-27h]
  unsigned __int8 v38; // [esp+13h] [ebp-25h]
  unsigned __int8 v39; // [esp+13h] [ebp-25h]
  unsigned __int8 v40; // [esp+14h] [ebp-24h]
  unsigned __int8 v41; // [esp+14h] [ebp-24h]
  unsigned __int8 v42; // [esp+19h] [ebp-1Fh]
  unsigned __int8 v43; // [esp+19h] [ebp-1Fh]
  unsigned __int8 v44; // [esp+1Ah] [ebp-1Eh]
  unsigned __int8 v45; // [esp+1Ah] [ebp-1Eh]
  unsigned __int8 v46; // [esp+1Bh] [ebp-1Dh]
  unsigned __int8 v47; // [esp+1Bh] [ebp-1Dh]
  unsigned __int8 v48; // [esp+1Ch] [ebp-1Ch]
  unsigned __int8 v49; // [esp+1Ch] [ebp-1Ch]
  int Name; // [esp+20h] [ebp-18h]
  int Serial; // [esp+24h] [ebp-14h]
  char v52; // [esp+28h] [ebp-10h]
  int v53; // [esp+34h] [ebp-4h]

  ATL::CStringT<wchar_t,StrTraitMFC_DLL<wchar_t,ATL::ChTraitsCRT<wchar_t>>>::CStringT<wchar_t,StrTraitMFC_D
  v1 = 0;
  v53 = 0;
  ATL::CStringT<wchar_t,StrTraitMFC_DLL<wchar_t,ATL::ChTraitsCRT<wchar_t>>>::CStringT<wchar_t,StrTraitMFC_D
  ATL::CStringT<wchar_t,StrTraitMFC_DLL<wchar_t,ATL::ChTraitsCRT<wchar_t>>>::CStringT<wchar_t,StrTraitMFC_D
  LOBYTE(v53) = 2;
  CWnd::GetWindowTextW(a1 + 304, &Name);
  if ( *(Name - 12) == 4 )                      // Name长度等于4
  {
    v3 = 0;
    while ( ATL::CSimpleStringT<wchar_t,1>::GetAt(&Name, v3) >= 'a'// Name都为小写字母
         && ATL::CSimpleStringT<wchar_t,1>::GetAt(&Name, v3) <= 'z' )
    {
      if ( ++v3 >= 4 )
      {
LABEL_7:
        v4 = 0;
        while ( 1 )
        {
          if ( v1 != v4 )
          {
            v5 = ATL::CSimpleStringT<wchar_t,1>::GetAt(&Name, v4);
            if ( ATL::CSimpleStringT<wchar_t,1>::GetAt(&Name, v1) == v5 )// 每个字母都不能相同
              goto LABEL_2;
          }
          if ( ++v4 >= 4 )
          {
            if ( ++v1 < 4 )
              goto LABEL_7;
            CWnd::GetWindowTextW(a1 + 420, &Serial);
            if ( *(Serial - 12) == 11 && ATL::CSimpleStringT<wchar_t,1>::GetAt(&Serial, 5) == '-' )// Seria
            {
              v6 = ATL::CSimpleStringT<wchar_t,1>::GetAt(&Name, 0);// v6 = Name[0]
              v7 = (v6 & 1) + 5;
              v48 = ((v6 >> 4) & 1) + 5;
              v42 = ((v6 >> 1) & 1) + 5;
              v44 = ((v6 >> 2) & 1) + 5;
              v46 = ((v6 >> 3) & 1) + 5;
              v8 = ATL::CSimpleStringT<wchar_t,1>::GetAt(&Name, 1);// v8 = Name[1]
```

```
v34 = (v8 & 1) + 1;
v40 = ((v8 >> 4) & 1) + 1;
v36 = ((v8 >> 1) & 1) + 1;
v9 = ((v8 >> 2) & 1) + 1;
v38 = ((v8 >> 3) & 1) + 1;
v10 = ATL::CSimpleStringT<wchar_t,1>::GetBuffer(&v52);
itow_s(v7 + v9, v10, 0xAu, 10);
v11 = ATL::CSimpleStringT<wchar_t,1>::GetAt(&v52, 0);
if ( ATL::CSimpleStringT<wchar_t,1>::GetAt(&Serial, 0) == v11 )// v11 = Serial[0]
{
  ATL::CSimpleStringT<wchar_t,1>::ReleaseBuffer(&v52, -1);
  v12 = ATL::CSimpleStringT<wchar_t,1>::GetBuffer(&v52);
  itow_s(v46 + v38, v12, 0xAu, 10);
  v13 = ATL::CSimpleStringT<wchar_t,1>::GetAt(&Serial, 1);// v13 = Serial[1]
  if ( v13 == ATL::CSimpleStringT<wchar_t,1>::GetAt(&v52, 0) )
  {
    ATL::CSimpleStringT<wchar_t,1>::ReleaseBuffer(&v52, -1);
    v14 = ATL::CSimpleStringT<wchar_t,1>::GetBuffer(&v52);
    itow_s(v42 + v40, v14, 0xAu, 10);
    v15 = ATL::CSimpleStringT<wchar_t,1>::GetAt(&Serial, 2);// v15 = Serial[2]
    if ( v15 == ATL::CSimpleStringT<wchar_t,1>::GetAt(&v52, 0) )
    {
      ATL::CSimpleStringT<wchar_t,1>::ReleaseBuffer(&v52, -1);
      v16 = ATL::CSimpleStringT<wchar_t,1>::GetBuffer(&v52);
      itow_s(v44 + v34, v16, 0xAu, 10);
      v17 = ATL::CSimpleStringT<wchar_t,1>::GetAt(&Serial, 3);// v17 = Serial[3]
      if ( v17 == ATL::CSimpleStringT<wchar_t,1>::GetAt(&v52, 0) )
      {
        ATL::CSimpleStringT<wchar_t,1>::ReleaseBuffer(&v52, -1);
        v18 = ATL::CSimpleStringT<wchar_t,1>::GetBuffer(&v52);
        itow_s(v48 + v36, v18, 0xAu, 10);
        v19 = ATL::CSimpleStringT<wchar_t,1>::GetAt(&Serial, 4);// v19 = Serial[4]
        if ( v19 == ATL::CSimpleStringT<wchar_t,1>::GetAt(&v52, 0) )
        {
          ATL::CSimpleStringT<wchar_t,1>::ReleaseBuffer(&v52, -1);
          v20 = ATL::CSimpleStringT<wchar_t,1>::GetAt(&Name, 2);// v20 = Name[2]
          v21 = (v20 & 1) + 5;
          v49 = ((v20 >> 4) & 1) + 5;
          v43 = ((v20 >> 1) & 1) + 5;
          v45 = ((v20 >> 2) & 1) + 5;
          v47 = ((v20 >> 3) & 1) + 5;
          v22 = ATL::CSimpleStringT<wchar_t,1>::GetAt(&Name, 3);// v22 = Name[3]
          v35 = (v22 & 1) + 1;
          v41 = ((v22 >> 4) & 1) + 1;
          v37 = ((v22 >> 1) & 1) + 1;
          v23 = ((v22 >> 2) & 1) + 1;
          v39 = ((v22 >> 3) & 1) + 1;
          v24 = ATL::CSimpleStringT<wchar_t,1>::GetBuffer(&v52);
          itow_s(v21 + v23, v24, 0xAu, 10);
          v25 = ATL::CSimpleStringT<wchar_t,1>::GetAt(&Serial, 6);// v25 = Serial[6]
          if ( v25 == ATL::CSimpleStringT<wchar_t,1>::GetAt(&v52, 0) )
          {
            ATL::CSimpleStringT<wchar_t,1>::ReleaseBuffer(&v52, -1);
            v26 = ATL::CSimpleStringT<wchar_t,1>::GetBuffer(&v52);
            itow_s(v47 + v39, v26, 0xAu, 10);
            v27 = ATL::CSimpleStringT<wchar_t,1>::GetAt(&Serial, 7);// v27 = Serial[7]
            if ( v27 == ATL::CSimpleStringT<wchar_t,1>::GetAt(&v52, 0) )
            {
              ATL::CSimpleStringT<wchar_t,1>::ReleaseBuffer(&v52, -1);
              v28 = ATL::CSimpleStringT<wchar_t,1>::GetBuffer(&v52);
```

```
                            itow_s(v43 + v41, v28, 0xAu, 10);
                            v29 = ATL::CSimpleStringT<wchar_t,1>::GetAt(&Serial, 8);// v29 = Serial[8]
                            if ( v29 == ATL::CSimpleStringT<wchar_t,1>::GetAt(&v52, 0) )
                            {
                              ATL::CSimpleStringT<wchar_t,1>::ReleaseBuffer(&v52, -1);
                              v30 = ATL::CSimpleStringT<wchar_t,1>::GetBuffer(&v52);
                              itow_s(v45 + v35, v30, 0xAu, 10);
                              v31 = ATL::CSimpleStringT<wchar_t,1>::GetAt(&Serial, 9);// v31 = Serial[9]
                              if ( v31 == ATL::CSimpleStringT<wchar_t,1>::GetAt(&v52, 0) )
                              {
                                ATL::CSimpleStringT<wchar_t,1>::ReleaseBuffer(&v52, -1);
                                v32 = ATL::CSimpleStringT<wchar_t,1>::GetBuffer(&v52);
                                itow_s(v49 + v37, v32, 0xAu, 10);
                                v33 = ATL::CSimpleStringT<wchar_t,1>::GetAt(&Serial, 10);// v33 = Serial[10
                                if ( v33 == ATL::CSimpleStringT<wchar_t,1>::GetAt(&v52, 0) )
                                {
                                  ATL::CSimpleStringT<wchar_t,1>::ReleaseBuffer(&v52, -1);
                                  ATL::CStringT<wchar_t,StrTraitMFC_DLL<wchar_t,ATL::ChTraitsCRT<wchar_t>>>
                                  ATL::CStringT<wchar_t,StrTraitMFC_DLL<wchar_t,ATL::ChTraitsCRT<wchar_t>>>
                                  ATL::CStringT<wchar_t,StrTraitMFC_DLL<wchar_t,ATL::ChTraitsCRT<wchar_t>>>
                                  return 1;
                                }
                              }
                            }
                          }
                        }
                      }
                    }
                  }
                }
              }
              goto LABEL_2;
            }
          }
        }
      }
    }
  }
LABEL_2:
  ATL::CStringT<wchar_t,StrTraitMFC_DLL<wchar_t,ATL::ChTraitsCRT<wchar_t>>>::~CStringT<wchar_t,StrTraitMFC_
  ATL::CStringT<wchar_t,StrTraitMFC_DLL<wchar_t,ATL::ChTraitsCRT<wchar_t>>>::~CStringT<wchar_t,StrTraitMFC_
  ATL::CStringT<wchar_t,StrTraitMFC_DLL<wchar_t,ATL::ChTraitsCRT<wchar_t>>>::~CStringT<wchar_t,StrTraitMFC_
  return 0;
}
```

```
v6 = Name[0]
v7 = (v6 & 1) + 5
v48 = ((v6 >> 4) & 1) + 5
v42 = ((v6 >> 1) & 1) + 5
v44 = ((v6 >> 2) & 1) + 5
v46 = ((v6 >> 3) & 1) + 5
v8 = Name[1]
v34 = (v8 & 1) + 1
v40 = ((v8 >> 4) & 1) + 1
v36 = ((v8 >> 1) & 1) + 1
v9 = ((v8 >> 2) & 1) + 1
v38 = ((v8 >> 3) & 1) + 1

v7 + v9 = Serial[0]

v46 + v38 = Serial[1]

v42 + v40 = Serial[2]

v44 + v34 = Serial[3]

v48 + v36 = Serial[4]

v20 = Name[2]
v21 = (v20 & 1) + 5
v49 = ((v20 >> 4) & 1) + 5
v43 = ((v20 >> 1) & 1) + 5
v45 = ((v20 >> 2) & 1) + 5
v47 = ((v20 >> 3) & 1) + 5
v22 = Name[3]
v35 = (v22 & 1) + 1
v41 = ((v22 >> 4) & 1) + 1
v37 = ((v22 >> 1) & 1) + 1
v23 = ((v22 >> 2) & 1) + 1
v39 = ((v22 >> 3) & 1) + 1

v21 + v23 = Serial[6]

v47 + v39 = Serial[7]

v43 + v41 = Serial[8]

v45 + v35 = Serial[9]

v49 + v37 = Serial[10]
```

然后写脚本进行爆破即可

先求Name前两位

```
Serial='76876_77776'
for i in range(ord('a'),ord('z')+1):
 for j in range(ord('a'),ord('z')+1):
  v6=i
  v8=j

  v7 = (v6 & 1) + 5
  v48 = ((v6 >> 4) & 1) + 5
  v42 = ((v6 >> 1) & 1) + 5
  v44 = ((v6 >> 2) & 1) + 5
  v46 = ((v6 >> 3) & 1) + 5

  v34 = (v8 & 1) + 1
  v40 = ((v8 >> 4) & 1) + 1
  v36 = ((v8 >> 1) & 1) + 1
  v9 = ((v8 >> 2) & 1) + 1
  v38 = ((v8 >> 3) & 1) + 1

  if v7 + v9 == int(Serial[0]) and v46 + v38 == int(Serial[1]) and v42 + v40 == int(Serial[2]) and v44 + v3
   print chr(i),chr(j)#Name前两位
'''
b u
c q
f t
g p
'''
```

再求后两位

```
Serial='76876_77776'
for i in range(ord('a'),ord('z')+1):
 for j in range(ord('a'),ord('z')+1):
  v20=i
  v22=j

  v21 = (v20 & 1) + 5
  v49 = ((v20 >> 4) & 1) + 5
  v43 = ((v20 >> 1) & 1) + 5
  v45 = ((v20 >> 2) & 1) + 5
  v47 = ((v20 >> 3) & 1) + 5

  v35 = (v22 & 1) + 1
  v41 = ((v22 >> 4) & 1) + 1
  v37 = ((v22 >> 1) & 1) + 1
  v23 = ((v22 >> 2) & 1) + 1
  v39 = ((v22 >> 3) & 1) + 1

  if v21 + v23 == int(Serial[6]) and v47 + v39 == int(Serial[7]) and v43 + v41 == int(Serial[8]) and v45 +
   print chr(i),chr(j)
'''
a y
b m
c i
e x
f l
g h
h u
i q
j e
k a
l t
m p *
n d
'''
```

可以发现最后有p的是 mp

与前两位进行构造可以得到

bump
cqmp
ftmp
gpmp

输入bump 正确

参考链接：

https://veritas501.space/2017/03/04/Reversing.kr%20writeup/