[BUUCTF刷题] Reverse解题方法总结（二）

原创

Y1seco 于 2021-05-12 11:07:03 发布 237 收藏 1

分类专栏： BUUCTF

本文链接：https://blog.csdn.net/qq_45834505/article/details/116128513

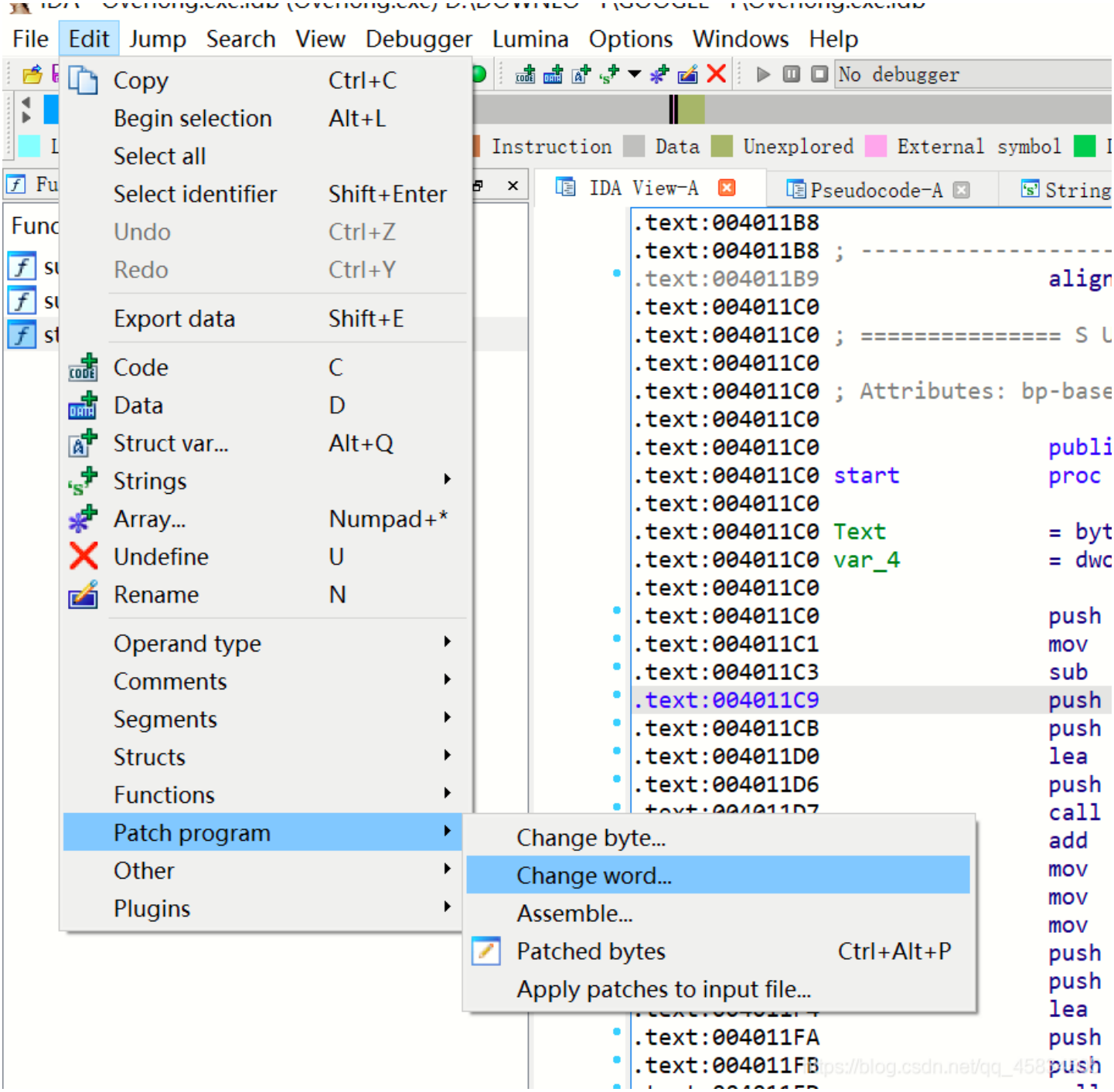
文章目录

知识点补充

```
- 知识点；
python中：
[m : ] 代表列表中的第m+1项到最后一项
[ : n] 代表列表中的第一项到第n项
- 汇编中的test
Test对两个参数(目标，源)执行AND逻辑操作，并根据结果设置标志寄存器，结果本身不会保存。
TEST AX,BX 与 AND AX,BX 命令有相同效果，只是Test指令不改变AX和BX的内容，而AND指令会把结果保存到AX中。
Test命令将两个操作数进行逻辑与运算，并根据运算结果设置相关的标志位。但是，Test命令的两个操作数不会被改变。运算结果在设置过相关标记位后会被丢弃。
test的一个非常普遍的用法是用来测试一方寄存器是否为空：test ecx, ecx
jz somewhere，如果ecx为零，设置ZF零标志为1，jz跳转。
```

[FlareOn6]Overlong

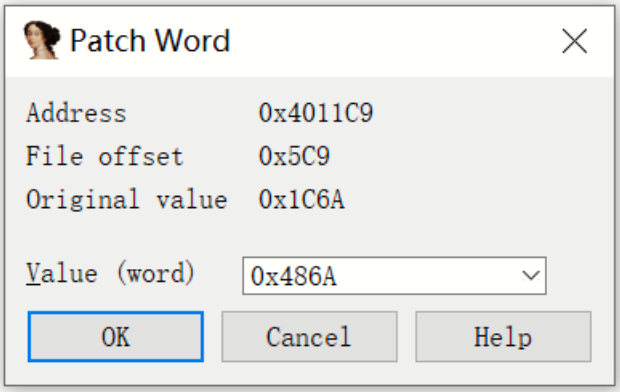
修改IDA汇编指令数据



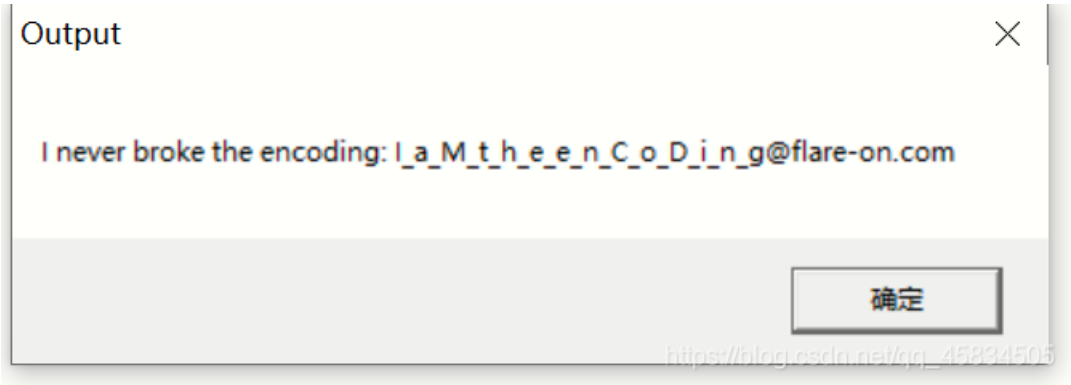
之后点击Apply patches to input file

- 本题提示尽可能输入较多的数据，将0x1C改为0x48，重新运行即可

```
1 int __stdcall start(int a1, int a2, int a3, int a4)
2 {
3   CHAR Text[128]; // [esp+0h] [ebp-84h] BYREF
4   unsigned int v6; // [esp+80h] [ebp-4h]
5
6   v6 = sub_401160(Text, (int)&unk_402008, 72u);
7   Text[v6] = 0;
8   MessageBoxA(0, Text, Caption, 0);
9   return 0;
10 }
```



得到



flag{l_a_M_t_h_e_e_n_C_o_D_i_n_g@flare-on.com}

[FlareOn3]Challenge1

发现是base64加密，并且密码表被修改了，直接写脚本：

```
import base64

encoded_flag = "x2dtJEOmyjacxDemx2eczT5cVS9fVUGvWTuZWjuexjRqy24rV29q"

#ZYXABCDEFGHIJKLMNOPQRSTUVWzyxabcdefghijklmnopqrstuvw0123456789+/
#ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/
_list = list("ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/")

_flag = ""

for i in encoded_flag:
    if (ord(i) <= ord('W') and ord(i) >= ord('A')) or (ord(i) <= ord('w') and ord(i) >= ord('a')):
        _flag += chr(ord(i) + 3)
    elif i == 'X':
        _flag += 'C'
    elif i == 'Y':
        _flag += 'B'
    elif i == 'Z':
        _flag += 'A'
    elif i == 'x':
        _flag += 'c'
    elif i == 'y':
        _flag += 'b'
    elif i == 'z':
        _flag += 'a'
    else:
        _flag += i

print(_flag)

print(base64.b64decode(_flag).decode())
```

也可利用maketrans和tranlate改变映射关系

```
import base64
enc = 'x2dtJEOmyjacxDemx2eczT5cVS9fVUGvWTuZWjuexjRqy24rV29q'
intab = 'ZYXABCDEFGHIJKLMNOPQRSTUVWzyxabcdefghijklmnopqrstuvw0123456789+/'
outtab = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'
transtab = str.maketrans(intab,outtab)

enc = enc.translate(transtab)
print(base64.b64decode(enc).decode())
```

[Zer0pts2020]easy strcmp1

- 知识点：binascii模块的使用

```
binascii模块主要用于二进制数据（byte类型数据）和ASCII的转换

在16进制和字符串的转换过程中，主要用到了以下几个函数：

a2b_hex()：返回16进制的二进制数据表现形式

b2a_hex()：返回二进制数据的16进制表现形式

hexlify()：返回二进制数据的16进制表现形式

unhexlify()：返回16进制的二进制数据表现形式
```

注意：不同进制之间的转换是以整形数据为基础的，如10进制转换为16进制：hex(int(data,10))
应用:

```
a = 'hello, world'
a = a.encode()  #将字符串变成二进制比特流形式
print(a)
b = binascii.b2a_hex(a)  #二进制形式转换成16进制比特流形式
print(b)
b = int(b, 16)  #16进制比特流变成整型数据
print(b)
b = hex(b)[2:]  #16进制的整型数据,[2:]是去除开始的0x
print(b)
```

参考文章

- wp

  分析逻辑

```
IDA View-A | Pseudocode-B | Pseudocode-C | Pseudocode-A
__int64 __fastcall sub_6EA(__int64 a1, __int64 a2)
{
  int i; // [rsp+18h] [rbp-8h]
  int v4; // [rsp+18h] [rbp-8h]
  int j; // [rsp+1Ch] [rbp-4h]

  for ( i = 0; *(_BYTE *)(i + a1); ++i )
    ;
  v4 = (i >> 3) + 1;
  for ( j = 0; j < v4; ++j )
    *(_QWORD *)(8 * j + a1) -= qword_201060[j];
  return qword_201090(a1, a2);
}
```


i>>3即除以8，将a1进行8位一组划分，并与qword_201060数组相减,只需逆向加回来，注意hex存储的小端方式需要反过来

```
.data:0000000000201060 ; _QWORD qword_201060[5]
.data:0000000000201060 qword_201060    dq 0, 410A4335494A0942h, 0B0EF2F50BE619F0h, 4F0A3A064A35282Bh
.data:0000000000201060                                         ; DATA XREF: sub_6EA+62↑o
.data:0000000000201060                 dq 0
.data:0000000000201060 _data           ends
```

主函数：判断输入的a2[1]是否和`zer0pts{********CENSORED********}`相等

```
__int64 __fastcall main(int a1, char **a2, char **a3)
{
  if ( a1 > 1 )
  {
    if ( !strcmp(a2[1], "zer0pts{********CENSORED********}") )
      puts("Correct!");
    else
      puts("Wrong!");
  }
  else
  {
    printf("Usage: %s <FLAG>\n", *a2);
  }
  return 0LL;
}
```

```
m = [0x410A4335494A0942, 0x0B0EF2F50BE619F0, 0x4F0A3A064A35282B]
enc = "********CENSORED********"
import binascii

flag =b""
for i in range(3):
    p = enc[i*8:(i+1)*8] #8位一组
    a = binascii.b2a_hex(p.encode('ascii')[::-1]) #将每一位转换成ascii码并逆序
    b = binascii.a2b_hex(hex(int(a,16)+m[i])[2:])[::-1] #hex()后的数据为16进制，[2:]是去除开头的0x,[::-1]逆序
    flag+= b
print(flag)
```

异常处理机制

参考文章：windows的异常处理机制 , SEH

PS：

水一期（最近有点忙）

未解题： crackMe，[ACTF新生赛2020]Oruga（迷宫问题）