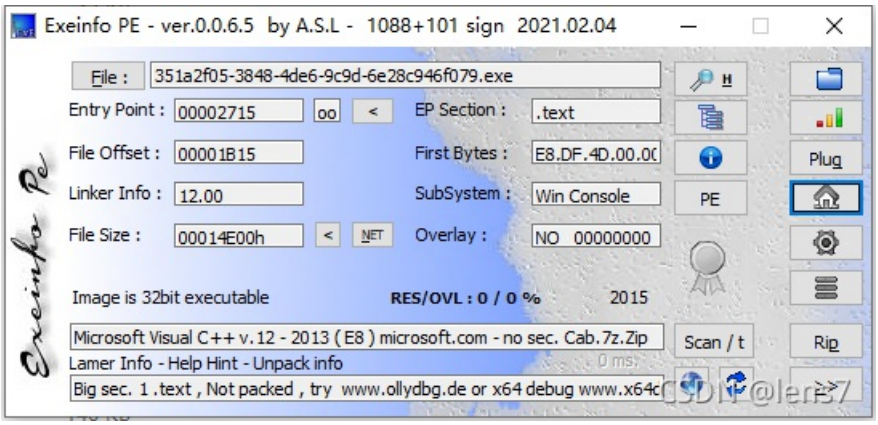
BUUCTF crackMe

原创

1ens 于 2021-11-01 21:58:35 发布 397 收藏
分类专栏： BUU刷题 文章标签： 安全

太菜了 做了好久没做出来，参考其他师傅的解答总结一下

题目背景：



老规矩，Ex PE打开，32位无壳程序，用IDApro打开；主程序如下：

```
int wmain()
{
  FILE *v0; // eax
  FILE *v1; // eax
  char v3; // [esp+3h] [ebp-405h]
  char v4[256]; // [esp+4h] [ebp-404h] BYREF
  char Format[256]; // [esp+104h] [ebp-304h] BYREF
  char v6[256]; // [esp+204h] [ebp-204h] BYREF
  char v7[256]; // [esp+304h] [ebp-104h] BYREF

  printf("Come one! Crack Me~~~\n");
  memset(v7, 0, sizeof(v7));
  memset(v6, 0, sizeof(v6));
  while ( 1 )
  {
    do
    {
      do
      {
        printf("user(6-16 letters or numbers):");
        scanf("%s", v7);
        v0 = (FILE *)sub_4024BE();
        fflush(v0);
      }
      while ( !(unsigned __int8)sub_401000(v7) );
      printf("password(6-16 letters or numbers):");
      scanf("%s", v6);
      v1 = (FILE *)sub_4024BE();
      fflush(v1);
    }
    while ( !(unsigned __int8)sub_401000(v6) );
    sub_401090(v7);
    memset(Format, 0, sizeof(Format));
    memset(v4, 0, sizeof(v4));
    v3 = ((int (__cdecl *)(char *, char *))loc_4011A0)(Format, v4);
    if ( sub_401830((int)v7, v6) )
    {
      if ( v3 )
        break;
    }
    printf(v4);
  }
  printf(Format);
  return 0;
```


主体是一个while循环，如果我们想让while结束，我们就需要使得两个if成立；

我们首先看跟v3有关的loc_4011A0，点进去，发现无法反编译，解决办法IDA无法反编译

编译成功之后：

```
result = 1;
a2[18] = 0;
return result;
```

发现v3的值一直为1；

我们来看看sub_401830函数，主要由两个while构成，我们先来看第一个while

```
while ( v6 < strlen(a2) )
{
  if ( isdigit(a2[v6]) )                    // // digit（数字） 判断输入的这个是数字字符把他转换为数字
  {
    v8 = a2[v6] - '0';
  }
  else if ( isxdigit(a2[v6]) )              // xdigit（十六进制字符） 判断输入的这个是十六进制abcdef
  {
    if ( *((_DWORD *)NtCurrentPeb()->ProcessHeap + 3) != 2 )// 反调试的，具体我也没明白
      a2[v6] = 34;
    v8 = (a2[v6] | 0x20) - 'W';
  }
  else
  {
    v8 = ((a2[v6] | 0x20) - 'a') % 6 + 10;  // 将abcdef转换为对应的十六进制数
  }
  __rdtsc();
  __rdtsc();
  v9 = v8 + 16 * v9;
  if ( !((int)(v6 + 1) % 2) )               // 结合上一步，将v9的值变成分割的两位的值
  {
    v15[v3++ - 1] = v9;                     // v3++ -1 就是一开始v15[-1]也就是v14存储，后面依次加1
    v9 = 0;
  }
  ++v6;
}
```


主要是把输入的转换成数字并且按照每两位分成一组，也就是输入a21c32ba，最后分为0xa2，0x1c，0x32，0xba，接下来看第二个while循环；

```
while ( v5 < 8 )
{
  v10 += a[++v11];
  v12 = a[v11];
  v7 = a[v10];
  a[v10] = v12;
  a[v11] = v7;
  if ( (NtCurrentPeb()->NtGlobalFlag & 0x70) != 0 )
    v12 = v10 + v11;
  v16[v5] = a[(unsigned __int8)(v7 + v12)] ^ v15[v4 - 1];
  if ( (unsigned __int8)*(_DWORD *)&NtCurrentPeb()->BeingDebugged )
  {
    v10 = -83;
    v11 = 43;
  }
  sub_401710(v16, a1, v5++);
  v4 = v5;
  if ( v5 >= (unsigned int)(&v15[strlen(&v14)] - v15) )
    v4 = 0;
}
v13 = 0;
sub_401470(v16, &v13);
return v13 == 0xAB94;
}
```


我们开始从后向前看， 最后需要输出v13==0xAB94，我们先点击进入sub_401470函数查看，

一堆的if判断，把判断条件转换为字符查看，这些字符就是大概他想要得到的字符，不是的话就执行错误的操作，最后进行验证，需要注意是这里

```
  if ( a2[5] != 'f' )
    *a3 |= 0x21u;
  else
    *a3 |= 0x2DCu;
}
if ( a2[5] != 's' )
{
  v5 = (char)a3;
  *a3 ^= 0x1ADu;
}
```

我们可以尝试运算一遍，应该是s，最后得到 a2的值为dbappsec 。最后我们就要求的是a数组的值了（源程序是byte_416050数组，被我改了）

```
v16[v5] = a[(unsigned __int8)(v7 + v12)] ^ v15[v4 - 1];
```


怎么求呢？IDA调试，有反调试没成功，试试OD，

```
loc_401B00:
movzx   ecx, [ebp+var_209]
movzx   edx, [ebp+var_20E]
add     ecx, edx
mov     [ebp+var_209], cl
movzx   eax, [ebp+var_209]
mov     cl, a[eax]
mov     [ebp+var_209], cl
mov     edx, [ebp+var_228]
movzx   eax, [ebp+edx+var_204]
movzx   ecx, [ebp+var_209]
xor     eax, ecx
mov     edx, [ebp+var_21C]
mov     [ebp+edx+var_104], al
mov     eax, large fs:30h
inc     eax
inc     eax
mov     eax, [eax]
and     eax, 0FFh
test    eax, eax
jz      short loc_401B6E
```

先在ida中找到xor异或，查看他的地址（建议先图装查看，找起来方便，再查看地址）

```
.text:00401B37                  movzx   ecx, [ebp+var_209]
.text:00401B3E                  xor     eax, ecx
```


我们现在就要看ecx的值了（XOR destination, source为什么不是eax，怪我汇编没学好，就是xor异或和实际异或的顺序相反的，我是这么记忆的）在**1B3E处下一个硬件断点，然后输入用户名，随便输入一个密码，循环8次记录下ecx的值。

> 0x2a, 0xd7, 0x92, 0xe9, 0x53, 0xe2, 0xc4, 0xcd

最后就是写脚本解密了。

```
str="dbappsec"
flag=""
a=[0x2a, 0xd7, 0x92, 0xe9, 0x53, 0xe2, 0xc4, 0xcd]
for i in range(8):
    flag+=hex(a[i]^ord(str[i]))
print(flag.replace('0x',''),end='')
```

最后输出为4eb5f3992391a1ae，md5加密之后的答案为

flag{d2be2981b84f2a905669995873d6a36c}