逆向-攻防世界-key

原创

S1lenc3 于 2019-12-13 21:29:54 发布 75 收藏

最近搞了个破解专用虚拟机，找不到特别合适的，就凑活用吧。

这个题目好几天了，有好多事，都没做，今天来看一看。

不知道为啥好几个虚拟机都打不开这个文件，报错，还好有一个WIN7刚好打开，不知道什么原因。

打开文件提示:? W? h? a? th? a? p? p? e? n?

IDA搜索字符串，我们还看到了一个路径，说实话作为小白根本看不懂IDA伪代码，看了大佬的writeup知道是比较文件中的字符串，ok接下来就自己操作了，确实看不懂伪代码，直接下断点调试。

调试到如图：

```
.text:002B11A0 loc_2B11A0:                             ; CODE XREF: s
.text:002B11A0 mov     al, byte ptr [ebp+esi+var_24]
.text:002B11A4 lea     ecx, [ebp+var_6C]
.text:002B11A7 xor     al, byte ptr [ebp+esi+Memory]
.text:002B11AB add     al, 16h
.text:002B11AD mov     [ebp+var_128], al
.text:002B11B3 push    dword ptr [ebp+var_128]         ; char
.text:002B11B9 push    1                               ; Size
.text:002B11BB call    sub_2B21E0
.text:002B11C0 inc     esi
.text:002B11C1 cmp     esi, 12h
.text:002B11C4 jl      short loc_2B11A0
```

应该是一次循环加密字符串。

对应这个函数：

```
do                                                  // 通过计算给v35赋值
{
  sub_2B21E0(&v34, 1u, (*((_BYTE *)Memory + v0) ^ *((_BYTE *)&v43 + v0)) + 22);
  ++v0;
}
while ( v0 < 18 );
```

后边还有一次加密：

```
79  do                                                  // 重新判断并给memory赋值
80  {
81    v4 = &v34;
82    if ( v2 >= 0x10 )
83      v4 = v3;
84    sub_2B21E0(Memory, 1u, *((_BYTE *)v4 + v1++) + 9);
85  }
86  while ( v1 < 18 );
```

继续调试，然后就终止了。应该是判断文件不存在，直接输出了？W？h？a？th？a？p？p？e？n？。我们手动添加这个文件。随便输入一串字符，然后运行程序，输入：wrong key。

有进展了，我们下断点继续调试，

```
v13 = sub_12620C0(&v37, v11, v38, (int)v12, v41);
v14 = std::cout;
if ( v13 )
{
  v22 = "=W=r=o=n=g=K=e=y=";
}
else
{
  v15 = sub_1262A00(std::cout, "|------------------
  std::basic_ostre
```

```
int __fastcall(int a1, const char
```

判断v13是否存在，那我们下端在v13赋值的时候，进行调试，在调试过程中，能看到我们文件中的字符串和另一串字符进行比较，我们拷贝下来，提交正确。

```
debug013:0037A348 db  69h ; i
debug013:0037A349 db  64h ; d
debug013:0037A34A db  67h ; g
debug013:0037A34B db  5Fh ; _
debug013:0037A34C db  63h ; c
debug013:0037A34D db  6Eh ; n
debug013:0037A34E db  69h ; i
debug013:0037A34F db  7Eh ; ~
debug013:0037A350 db  62h ; b
debug013:0037A351 db  6Ah ; j
debug013:0037A352 db  62h ; b
debug013:0037A353 db  66h ; f
debug013:0037A354 db  69h ; i
debug013:0037A355 db  7Ch ; |
debug013:0037A356 db  67h ; g
debug013:0037A357 db  73h ; s
debug013:0037A358 db  78h ; x
debug013:0037A359 db  62h ; b
debug013:0037A35A db    0
```