BUUCTF Crackme

原创

chan3301 于 2019-07-29 20:56:56 发布 1778 收藏

分类专栏： 逆向题目练习 文章标签： 逆向

版权声明：本文为博主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。

本文链接：https://blog.csdn.net/sjt670994562/article/details/97679309

版权

逆向题目练习 专栏收录该内容

16 篇文章 1 订阅

订阅专栏

这道题麻烦了我几天，因为一个反调试的点没看到，所以一直没做出来，虽然原理还不懂，但可以先记下来，以后学多了再说（这道题用户名已知）

ida分析，是一个建立了一个表然后与用户名进行错杂的变化，最后再用表里的值与密码进行一些运算得到结果，由此我们倒着看

```
  v14 = 0;
  dbappsec(ebx0, pass_chanhe, &v14);
  return v14 == 0xAB94;
}
```

最后一个函数是实现的目的，密码变成dbappsec

```
67      v12 = 43;
68    }
69    yihuo_end((int)pass_chanhe, (const char *)user, a3++);
70    v5 = a3;
71    if ( a3 >= &p_chn1 + strlen(&p_chn1) + 1 - &v16 )
72      v5 = 0;
73  }
```

```
11  GetStartupInfoW(&StartupInfo);
12  v6 = strlen(uesr);
13  if ( StartupInfo.dwX
14    || StartupInfo.dwY
15    || StartupInfo.dwXCountChars
16    || StartupInfo.dwYCountChars
17    || StartupInfo.dwFillAttribute
18    || StartupInfo.dwXSize
19    || !StartupInfo.dwYSize )
20  {
21    if ( a3 <= v6 )
```

https://blog.csdn.net/sjt670994562

倒数第二个有点小坑，就是用户名与变化后的密码运算，这里是检测是否有调试，有的话就异或，没有的话就是和，所以会有两个结果，我们可以一个一个算

```
if ( *(_DWORD *)(__readfsdword(0x30u) + 104) & 0x70 )
  v13 = v11 + v12;
```

```
if ( *(_DWORD *)(__readfsdword(0x30u) + 2) & 0xFF )
{
```

这一个就是最恶心的地方了，上下两个if都是反调试，把他们的jz jnz变成jmp就行，然后buffet_list表里面才是我们想要的值，这边我们可以在动调的时候吧表的8个值给确定出来，因为用户名已知，一定是不变的

```
while ( v7 < strlen(pass) )
{
  if ( isdigit(pass[v7]) )
  {
    v9 = pass[v7] - 48;
  }
  else if ( isxdigit(pass[v7]) )
  {
    if ( *(_DWORD *)(*(_DWORD *)(__readfsdword(0x30u) + 24) + 12) != 2 )
      pass[v7] = '"';
    v9 = (pass[v7] | 0x20) - 87;
  }
  else
  {
    v9 = ((pass[v7] | 0x20) - 97) % 6 + 10;
  }
  v10 = v9 + 16 * v10;
  if ( !((signed int)(v7 + 1) % 2) )
  {
    *(&p_chn1 + v4++) = v10;
    ebx0 = v4;
    v10 = 0;
  }
  ++v7;
}
```

https://blog.csdn.net/sjt670994562

这一块吧，就是对密码进行了一次变换，我觉得没啥用，不用分析emmmmm，因为动调的时候输入什么经过这段变化。他没有变化就是把输入的16个数变成了8个而已，所以做到这一步，问题就解决了，最后交flag的时候，用的是调试的那个密码，emmmmmmmmm，一般不应该是反调试吗？咱也不敢问，反正这道题逻辑挺清晰的，就是太菜，没看出来，最后挂出py代码

```
a=("dbappsec")
c=[]
b=("welcomebeijing")
d=[0x2A, 0XD7, 0X92, 0XE9, 0X53, 0XE2, 0XC4, 0XCD]
for i in range(0,8):
    c.append(ord(a[i])^ord(b[i]))
print(c)
for i in range(0,8):
    d[i]=hex(d[i]^ord(a[i])).replace("0x",'')
    print(d[i],end='')
```

https://blog.csdn.net/sjt670994562

emmmmmmmmmmmmm，不能太急，一步一步慢慢进步就好