BUU_re_[ACTF新生赛2020]rome

原创

A-Juan 于 2020-08-15 23:13:33 发布 638 收藏

分类专栏： 逆向

逆向 专栏收录该内容

8 篇文章 0 订阅

订阅专栏

拖进IDA，查看main函数

```
int func()
{
  int result; // eax
  int v1; // [esp+14h] [ebp-44h]
  int v2; // [esp+18h] [ebp-40h]
  int v3; // [esp+1Ch] [ebp-3Ch]
  int v4; // [esp+20h] [ebp-38h]
  unsigned __int8 v5; // [esp+24h] [ebp-34h]
  unsigned __int8 v6; // [esp+25h] [ebp-33h]
  unsigned __int8 v7; // [esp+26h] [ebp-32h]
  unsigned __int8 v8; // [esp+27h] [ebp-31h]
  unsigned __int8 v9; // [esp+28h] [ebp-30h]
  int v10; // [esp+29h] [ebp-2Fh]
  int v11; // [esp+2Dh] [ebp-2Bh]
  int v12; // [esp+31h] [ebp-27h]
  int v13; // [esp+35h] [ebp-23h]
  unsigned __int8 v14; // [esp+39h] [ebp-1Fh]
  char v15; // [esp+3Bh] [ebp-1Dh]
  char v16; // [esp+3Ch] [ebp-1Ch]
  char v17; // [esp+3Dh] [ebp-1Bh]
  char v18; // [esp+3Eh] [ebp-1Ah]
  char v19; // [esp+3Fh] [ebp-19h]
  char v20; // [esp+40h] [ebp-18h]
  char v21; // [esp+41h] [ebp-17h]
  char v22; // [esp+42h] [ebp-16h]
  char v23; // [esp+43h] [ebp-15h]
  char v24; // [esp+44h] [ebp-14h]
  char v25; // [esp+45h] [ebp-13h]
  char v26; // [esp+46h] [ebp-12h]
  char v27; // [esp+47h] [ebp-11h]
  char v28; // [esp+48h] [ebp-10h]
  char v29; // [esp+49h] [ebp-Fh]
  char v30; // [esp+4Ah] [ebp-Eh]
  char v31; // [esp+4Bh] [ebp-Dh]
  int i; // [esp+4Ch] [ebp-Ch]

  v15 = 'Q';
  v16 = 's';
  v17 = 'w';
```

```
v18 = '3';
v19 = 's';
v20 = 'j';
v21 = '_';
v22 = 'l';
v23 = 'z';
v24 = '4';
v25 = '_';
v26 = 'U';
v27 = 'j';
v28 = 'w';
v29 = '@';
v30 = 'l';
v31 = 0;
printf("Please input:");
scanf("%s", &v5);
result = v5;
if ( v5 == 'A' )
{
  result = v6;
  if ( v6 == 'C' )
  {
    result = v7;
    if ( v7 == 'T' )
    {
      result = v8;
      if ( v8 == 'F' )
      {
        result = v9;
        if ( v9 == '{' )
        {
          result = v14;
          if ( v14 == '}' )
          {
            v1 = v10;
            v2 = v11;
            v3 = v12;
            v4 = v13;
            for ( i = 0; i <= 15; ++i )
            {
              if ( *((_BYTE *)&v1 + i) > 64 && *((_BYTE *)&v1 + i) <= 90 )//如果是大写字母
                *((_BYTE *)&v1 + i) = (*((char *)&v1 + i) - 51) % 26 + 65;
              if ( *((_BYTE *)&v1 + i) > 96 && *((_BYTE *)&v1 + i) <= 122 )//如果是小写字母
                *((_BYTE *)&v1 + i) = (*((char *)&v1 + i) - 79) % 26 + 97;
            }
            for ( i = 0; i <= 15; ++i )
            {
              result = (unsigned __int8)*(&v15 + i);
              if ( *((_BYTE *)&v1 + i) != (_BYTE)result )
                return result;
            }
            result = printf("You are correct!");
          }
        }
      }
    }
  }
}
return result;
```

```
}
```

v15到v30所显示的值是经过下面的算法加密后的值，所以我们要逆回去。

破解脚本：

```
x = [81,115,119,51,115,106,95,108,122,52,95,85,106,119,64,108]
flag = ''
for k in range(0,16):
    for i in range(0,127):  //ASCII到目前为止共定义了128个字符，挨个儿试
        z = i
        if i > 64 and i <= 90:
            i = (i-51)%26 + 65
        if i > 96 and i <= 122:
            i = (i-79)%26 + 97
        if(i == x[k]):
            flag += chr(z)

print(flag)
```