攻防世界 Reverse高手进阶区 2分题 dmd-50

原创

思源湖的鱼 于 2020-12-06 17:13:38 发布 102 收藏

分类专栏： ctf 文章标签： ctf 攻防世界 reverse

本文链接：https://blog.csdn.net/weixin_44604541/article/details/110746335

ctf 专栏收录该内容

200 篇文章 23 订阅

订阅专栏

前言

继续ctf的旅程

攻防世界Reverse高手进阶区的2分题

本篇是dmd-50的writeup

发现攻防世界的题目分数是动态的

就仅以做题时的分数为准了

解题过程

PE查壳

扔进IDA

得到伪码

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
```

```
__int64 v3; // rax
__int64 v4; // rax
__int64 v5; // rax
__int64 v6; // rax
__int64 v7; // rax
__int64 v8; // rax
__int64 v9; // rax
__int64 v10; // rax
__int64 v11; // rax
__int64 v12; // rax
__int64 v13; // rax
__int64 v14; // rax
__int64 v15; // rax
__int64 v16; // rax
__int64 v17; // rax
__int64 v18; // rax
__int64 v19; // rax
__int64 v20; // rax
__int64 v21; // rax
int result; // eax
__int64 v23; // rax
__int64 v24; // rax
__int64 v25; // rax
__int64 v26; // rax
__int64 v27; // rax
__int64 v28; // rax
__int64 v29; // rax
__int64 v30; // rax
__int64 v31; // rax
__int64 v32; // rax
__int64 v33; // rax
__int64 v34; // rax
__int64 v35; // rax
__int64 v36; // rax
__int64 v37; // rax
char v38; // [rsp+Fh] [rbp-71h]
char v39; // [rsp+10h] [rbp-70h]
char v40; // [rsp+20h] [rbp-60h]
_BYTE *v41; // [rsp+28h] [rbp-58h]
char v42; // [rsp+30h] [rbp-50h]
unsigned __int64 v43; // [rsp+68h] [rbp-18h]

v43 = __readfsqword(0x28u);
std::operator<<<std::char_traits<char>>(&std::cout, "Enter the valid key!\n", envp);
std::operator>><char,std::char_traits<char>>(&edata, &v42);
std::allocator<char>::allocator(&v38);
std::string::string(&v39, &v42, &v38);
md5(&v40, &v39);
v41 = (_BYTE *)std::string::c_str((std::string *)&v40);
std::string::~string((std::string *)&v40);
std::string::~string((std::string *)&v39);
std::allocator<char>::~allocator(&v38);
if ( *v41 != 55
  || v41[1] != 56
  || v41[2] != 48
  || v41[3] != 52
  || v41[4] != 51
  || v41[5] != 56
  || v41[6] != 100
```

```
  || v41[7] != 53
  || v41[8] != 98
  || v41[9] != 54
  || v41[10] != 101
  || v41[11] != 50
  || v41[12] != 57
  || v41[13] != 100
  || v41[14] != 98
  || v41[15] != 48
  || v41[16] != 56
  || v41[17] != 57
  || v41[18] != 56
  || v41[19] != 98
  || v41[20] != 99
  || v41[21] != 52
  || v41[22] != 102
  || v41[23] != 48
  || v41[24] != 50
  || v41[25] != 50
  || v41[26] != 53
  || v41[27] != 57
  || v41[28] != 51
  || v41[29] != 53
  || v41[30] != 99
  || v41[31] != 48 )
{
  v23 = std::operator<<<std::char_traits<char>>(&std::cout, 73LL);
  v24 = std::operator<<<std::char_traits<char>>(v23, 110LL);
  v25 = std::operator<<<std::char_traits<char>>(v24, 118LL);
  v26 = std::operator<<<std::char_traits<char>>(v25, 97LL);
  v27 = std::operator<<<std::char_traits<char>>(v26, 108LL);
  v28 = std::operator<<<std::char_traits<char>>(v27, 105LL);
  v29 = std::operator<<<std::char_traits<char>>(v28, 100LL);
  v30 = std::operator<<<std::char_traits<char>>(v29, 32LL);
  v31 = std::operator<<<std::char_traits<char>>(v30, 75LL);
  v32 = std::operator<<<std::char_traits<char>>(v31, 101LL);
  v33 = std::operator<<<std::char_traits<char>>(v32, 121LL);
  v34 = std::operator<<<std::char_traits<char>>(v33, 33LL);
  v35 = std::operator<<<std::char_traits<char>>(v34, 32LL);
  v36 = std::operator<<<std::char_traits<char>>(v35, 58LL);
  v37 = std::operator<<<std::char_traits<char>>(v36, 40LL);
  std::ostream::operator<<(v37, &std::endl<char,std::char_traits<char>>);
  result = 0;
}
else
{
  v3 = std::operator<<<std::char_traits<char>>(&std::cout, 84LL);
  v4 = std::operator<<<std::char_traits<char>>(v3, 104LL);
  v5 = std::operator<<<std::char_traits<char>>(v4, 101LL);
  v6 = std::operator<<<std::char_traits<char>>(v5, 32LL);
  v7 = std::operator<<<std::char_traits<char>>(v6, 107LL);
  v8 = std::operator<<<std::char_traits<char>>(v7, 101LL);
  v9 = std::operator<<<std::char_traits<char>>(v8, 121LL);
  v10 = std::operator<<<std::char_traits<char>>(v9, 32LL);
  v11 = std::operator<<<std::char_traits<char>>(v10, 105LL);
  v12 = std::operator<<<std::char_traits<char>>(v11, 115LL);
  v13 = std::operator<<<std::char_traits<char>>(v12, 32LL);
  v14 = std::operator<<<std::char_traits<char>>(v13, 118LL);
  v15 = std::operator<<<std::char_traits<char>>(v14, 97LL);
  v16 = std::operator<<<std::char_traits<char>>(v15, 108LL);
```

```
    v17 = std::operator<<<std::char_traits<char>>(v16, 105LL);
    v18 = std::operator<<<std::char_traits<char>>(v17, 100LL);
    v19 = std::operator<<<std::char_traits<char>>(v18, 32LL);
    v20 = std::operator<<<std::char_traits<char>>(v19, 58LL);
    v21 = std::operator<<<std::char_traits<char>>(v20, 41LL);
    std::ostream::operator<<(v21, &std::endl<char,std::char_traits<char>>);
    result = 0;
  }
  return result;
}
```

是个md5加密
然后判断是否为`780438d5b6e29db0898bc4f0225935c0`

那到https://cmd5.com/解密

发现是个二次加密
但程序里只有一次

所以对`grape`进行一次md5加密即可

得到flag

结语

简单题