Bugku旧平台reverse writeup

原创

a370793934 于 2019-11-27 17:11:27 发布 660 收藏
分类专栏： WriteUp 文章标签： Bugku reverse writeup ctf

入门逆向

直接用用ida查看_main函数，r转换成字符

flag{Re_1s_S0_C0OL}

Easy_vb

用ida直接打开，alt+t快捷键搜索{

ctrl+t继续搜索能够找到

MCTF{_N3t_Rev_1s_E4ay_}

改为

flag{_N3t_Rev_1s_E4ay_}

Easy_Re

OD载入直接右键选择中文搜索引擎-->智能s搜索　就可以看到字符串flag

或者

用ida打开，代码审计

输入的v9与&xmmword_413E34处的值比较，双击进去，按r转换为字符串

两处拼在一起为}FTCTUD0tem0c1eW{FTCTUD，这个是倒序，python反转下

```
s="}FTCTUD0tem0c1eW{FTCTUD"
print s[::-1]
```

得到

DUTCTF{We1c0met0DUTCTF}

游戏过关

用ida打开程序

1.Shift+F12查看下字符串。

2.然后双击过去。

3.再按Cirt+X交叉引用显示调用位置

然后F5看下伪代码，关键代码：

```
 for ( i = 0; i < 56; ++i )
  {
    v2[i] ^= v2[i + 68];
    v2[i] ^= 0x13u;
  }
```

打印出done!!! the flag is

然后有两个数组按位异或再和0x13异或生成flag

写python脚本：

```
#coding:utf-8
v2 = [0]*125
v2[68] = 18;
v2[69] = 64;
v2[70] = 98;
v2[71] = 5;
v2[72] = 2;
v2[73] = 4;
v2[74] = 6;
v2[75] = 3;
v2[76] = 6;
v2[77] = 48;
v2[78] = 49;
v2[79] = 65;
v2[80] = 32;
v2[81] = 12;
v2[82] = 48;
```

```
v2[83] = 65;
v2[84] = 31;
v2[85] = 78;
v2[86] = 62;
v2[87] = 32;
v2[88] = 49;
v2[89] = 32;
v2[90] = 1;
v2[91] = 57;
v2[92] = 96;
v2[93] = 3;
v2[94] = 21;
v2[95] = 9;
v2[96] = 4;
v2[97] = 62;
v2[98] = 3;
v2[99] = 5;
v2[100] = 4;
v2[101] = 1;
v2[102] = 2;
v2[103] = 3;
v2[104] = 44;
v2[105] = 65;
v2[106] = 78;
v2[107] = 32;
v2[108] = 16;
v2[109] = 97;
v2[110] = 54;
v2[111] = 16;
v2[112] = 44;
v2[113] = 52;
```

```
v2[114] = 32;
v2[115] = 64;
v2[116] = 89;
v2[117] = 45;
v2[118] = 32;
v2[119] = 65;
v2[120] = 15;
v2[121] = 34;
v2[122] = 18;
v2[123] = 16;
v2[124] = 0;
v2[0] = 123;
v2[1] = 32;
v2[2] = 18;
v2[3] = 98;
v2[4] = 119;
v2[5] = 108;
v2[6] = 65;
v2[7] = 41;
v2[8] = 124;
v2[9] = 80;
v2[10] = 125;
v2[11] = 38;
v2[12] = 124;
v2[13] = 111;
v2[14] = 74;
v2[15] = 49;
v2[16] = 83;
v2[17] = 108;
v2[18] = 94;
v2[19] = 108;
```

```
v2[20] = 84;
v2[21] = 6;
v2[22] = 96;
v2[23] = 83;
v2[24] = 44;
v2[25] = 121;
v2[26] = 104;
v2[27] = 110;
v2[28] = 32;
v2[29] = 95;
v2[30] = 117;
v2[31] = 101;
v2[32] = 99;
v2[33] = 123;
v2[34] = 127;
v2[35] = 119;
v2[36] = 96;
v2[37] = 48;
v2[38] = 107;
v2[39] = 71;
v2[40] = 92;
v2[41] = 29;
v2[42] = 81;
v2[43] = 107;
v2[44] = 90;
v2[45] = 85;
v2[46] = 64;
v2[47] = 12;
v2[48] = 43;
v2[49] = 76;
v2[50] = 86;
```

```
v2[51] = 13;
v2[52] = 114;
v2[53] = 1;
v2[54] = 117;
v2[55] = 126;
v2[56] = 0;

flag = ""
for i in range(56):
    v2[i] ^= v2[i + 68];
    v2[i] ^= 0x13;
    flag += chr(v2[i])
print flag

#不改成列表，原变量运算方法
# v = locals()
# flag = ""
# for i in range(2,58):
#     v["v%d"%i] ^= v["v%d"%(i+57)]
#     v["v%d"%i] ^= 0x13
#     flag += chr(v["v%d"%i])
# print flag
```

zsctf{T9is_tOpic_1s_v5ry_int7resting_b6t_others_are_n0t}

Timer(阿里CTF)

Android逆向先跳过

逆向入门

下载后发现不是pe文件，右键txt打开，看到...开头的，为图像文件

直接复制到浏览器地址栏打开，是二维码，扫描得到

bugku{inde_9882ihsd8-0}

love

用ida打开定位到关键函数

打F5查看伪代码

可以看到有两步加密，第一步是先sub_4110BE(&Str, v0, &v11);用这个函数加密。然后再去循环加密

```
for ( j = 0; j < v8; ++j )
    Dest[j] += j;
```

然后把加密后的字符串与str2相比较。str2的值为e3niflH9b_C@n@dH，先把循环逆向了。

```
#coding=utf-8
s ='e3niflH9b_C@n@dH'
flag =''
for i in range(len(s)):
    flag += chr(ord(s[i])- i)
print flag
```

得到e2lfbDB2ZV95b3V9

然后看sub_4110BE()函数。一串长算法，发现首先将输入的flag每3位变成4位。然后有64位密码表。其实就是个base64加密（记下来,base64加密算法的特征）。

也就是将刚刚得到的值base64解密就是flag。

flag{i_l0ve_you}

LoopAndLoop(阿里CTF)

Android逆向先跳过

easy-100(LCTF)

Android逆向先跳过

SafeBox(NJCTF)

Android逆向先跳过

不好用的ce

od载入，只能搜索字符串

搜索到DeZmqMUhRcP8NgJgzLPdXa

base58解密得flag：

flag{c1icktimes}

Mountain climbing

待续