XCTF reverse新手练习区_1-10

原创

H4ppyD0g 于 2019-08-17 15:40:13 发布 932 收藏 2

本文链接：https://blog.csdn.net/weixin_42172261/article/details/99689073

1 re1

IDA打开，F5进行反汇编成C伪代码，审计代码，发现v5变量的赋值，双击后跳转到汇编，转成字符串爆出flag。

IDA简单使用
shift + f12 字符串窗口
f5 反汇编

---

2 game

shift+f12搜索flag字符串，反汇编，找到flag生成过程。

```
s1 = [123, 32, 18, 98, 119, 108, 65, 41, 124, 80, 125, 38, 124, 111, 74, 49, 83, 108, 94, 108, 84, 6, 96, 83, 44
, 121,104, 110, 32, 95, 117, 101, 99,123,127,119,96,48,107,71,92,29,81,107,90,85,64,12,43,76,86,13,114,1,117,126
,0]
s2 = [18, 64, 98, 5, 2, 4, 6, 3, 6, 48, 49, 65, 32, 12, 48, 65, 31, 78, 62, 32, 49, 32, 1, 57,96, 3, 21, 9, 4, 6
2, 3, 5, 4, 1, 2, 3, 44, 65, 78, 32, 16, 97, 54, 16, 44, 52, 32,64, 89, 45, 32, 65, 15, 34, 18, 16, 0]
flag = ""
for i in range(56):
 t = s1[i] ^ s2[i] ^ 0x13
 flag += chr(t)
print(flag)
```

---

3 Hello, CTF

搜索wrong字符串，转到反汇编代码，把那一串16进制字符串转成字符就可以了。

---

4 open-source

atoi (表示 ascii to integer)把字符串转换成整型数
itoa()将整数value 转换成字符串存入string 指向的内存空间

根据if判断计算出每个变量的值就可以了。

---

5 simple-unpack

用exeinfope查壳，有upx壳， 用upx命令 **upx -d 文件名** 去壳后再用IDA打开，main函数里面有flag。

---

6 logmein

LL是长整型，转成16进制后再转成字符串，又因为字符串是小端顺序，所以需要反转，然后按照反汇编的代码就能求出flag。**16进制转字符是第二次遇到了，可能在逆向里比较常用，记住了。**
为什么是小端序呢，因为一开始用exeinfope查看文件信息是elf，说明是linux的，而linux是小端顺序。反正我猜应该是这样。

```
>>> hex(28537194573619560)
'0x65626d61726168'
>>> v7 = "harambe"
>>> v8 = ":\"AL_RT^L*.?+6/46"
>>> v6 = 7
>>> flag = ""
>>> for i in range(len(v8)):
 t = chr(ord(v7[i%v6]) ^ ord(v8[i]))
 flag += t

>>> flag
'RC3-2016-XORISGUD'
```

---

7 insanity

难的不会，简单的嫌弱智…
IDA打开，定位主函数，发现可疑的str字符串，双击跳转，找到flag。。。

---

8 no-strings-attached

gdb ./lab 启动调试
b decrypt 在decrypt处设置断点
r 运行
n 下一步
`x/200wx $eax`
x:就是用来查看内存中数值的，后面的200代表查看多少个，wx代表是以word字节查看看，$eax代表的eax寄存器中的值。
在这里我们看到0x00000000，这就证明这个字符串结束了，因为，在C中，代表字符串结尾的就是"\0",那么前面的就是经过decrypt函数生成的flag.

```
>>> strings = [57, 52, 52, 55, 123, 121, 111, 117, 95, 97, 114, 101, 95, 97,110, 95, 105, 110, 116, 101, 114, 11
0, 97, 116, 105, 111, 110,97, 108, 95, 109, 121, 115, 116, 101, 114, 121, 125]
>>> flag = ""
>>> for i in range(len(strings)):
 s = chr(strings[i])
 flag += s

>>> flag
'9447{you_are_an_international_mystery}'
```

具体解题思路见攻防世界writeup。

---

9 python-trade

在线python反编译https://tool.lu/pyc/

这里需要注意的是，python3经过b64decode是bytes类型，直接进行运算就可以了

```
import base64
buf = base64.b64decode('XlNkVmtUI1MgXWBZXCFeKY+AaXNt')
flag = ''
for i in buf:
    i -= 16
    i ^= 32
    flag += chr(i)
print(flag)
```

---

10 getit

elf文件 64位
IDA打开，定位main函数。
在写入文件前flag就已经写好了，所以可以用两种方法，调试还看不大懂，这里用脚本自己构建flag。

```
>>> s = 'c61b68366edeb7bdce3c6820314b7498'
>>> t = 'SharifCTF{????????????????????????????????}'
>>> tmp = []
>>> for i in t:
 tmp.append(i)

>>> index = 0
>>> while index<len(s):
 if index & 1:
  v3=1
 else:
  v3=-1
 tmp[index+10]=str(chr(ord(s[index])+v3))
 index+=1

>>> flag = ''
>>> for i in tmp:
 flag +=i

>>> flag
'SharifCTF{b70c59275fcfa8aebf2d5911223c6589}'
```
