攻防世界Reverse题目writeup

原创

hacker_Ben 于 2020-02-28 22:42:31 发布 587 收藏 1

分类专栏： 逆向题目

本文链接：https://blog.csdn.net/hacker_Ben/article/details/104565480

Reverser题目

小白最近做了一些Reverse题目，整理整理

Python-trade

下载文件，发现是pyc文件

用pyc在线反编译网站进行反编译得到py文件

```
import base64

def encode(message):
    s = ''
    for i in message:
        x = ord(i) ^ 32
        x = x + 16
        s += chr(x)

    return base64.b64encode(s)

correct = 'XlNkVmtUI1MgXWBZXCFeKY+AaXNt'
flag = ''
print 'Input flag:'
flag = raw_input()
if encode(flag) == correct:
    print 'correct'
else:
    print 'wrong'
```

- 编码函数主要实现的是对flag的每个字符都先与32异或，然后加16，最后用Base64编码
- 我们可以很容易写出解码函数

```
import base64

correct = 'XlNkVmtUI1MgXWBZXCFeKY+AaXNt'
s=''
correct_d=base64.b64decode(correct)
for i in correct_d:
 x=ord(i)-16
 x=x^32
 s+=chr(x)
print s
```

运行程序就可得到flag

Hello,CTF

- 将程序拖入ida,将main函数进行F5，查看伪代码

```
qmemcpy(&v13, a437261636b4d65, 0x23u);
while ( 1 )
{
  memset(&v10, 0, 0x20u);
  v11 = 0;
  v12 = 0;
  sub_40134B(aPleaseInputYou, v6);
  scanf(aS, v9);
  if ( strlen(v9) > 0x11 )
    break;
  v3 = 0;
  do
  {
    v4 = v9[v3];
    if ( !v4 )
      break;
    sprintf(&v8, asc_408044, v4);
    strcat(&v10, &v8);
    ++v3;
```


- 首先，可以看到先是将字符串复制到v13的位置，
- 然后，后面对输入进行了判断，输入的字符串不能大于17
- 接着，将字符串以十六进制输出，然后，再将得到的十六进制字符添加到v10

```
  if ( !strcmp(&v10, &v13) )
    sub_40134B(aSuccess, v7);
  else
    sub_40134B(aWrong, v7);
}
```

将v10与v13进行比较，两者相同则输出Success
所以v10=v13
Flag=v13
将v13转为字符串就得到flag

simple-unpack

下载文件，用winhex打开发现是elf文件
根据提示是加壳的文件
使用命令脱壳

```
root@kali:~/Desktop# upx -d q1
                       Ultimate Packer for eXecutables
                          Copyright (C) 1996 - 2017
UPX 3.94        Markus Oberhumer, Laszlo Molnar & John Reiser   May 12th 2017

        File size         Ratio      Format      Name
   --------------------   ------   -----------   -----------
    912808 <-    352624   38.63%   linux/amd64   q1

Unpacked 1 file.
root@kali:~/Desktop#
```


用ida打开脱壳之后的程序，直接发现flag

```
.text:00000000004009C3                xor     eax, eax
.text:00000000004009C5                lea     rax, [rbp+var_70]
.text:00000000004009C9                mov     rsi, rax
.text:00000000004009CC                mov     edi, offset a96s ; "%96s"
.text:00000000004009D1                mov     eax, 0
.text:00000000004009D6                call    __isoc99_scanf
.text:00000000004009DB                lea     rax, [rbp+var_70]
.text:00000000004009DF                mov     esi, offset flag ; "flag{Upx_1s_n0t_a_d3liv3r_c0mp4ny}"
.text:00000000004009E4                mov     rdi, rax
.text:00000000004009E7                call    sub_400360
.text:00000000004009EC                test    eax, eax
```

logmein

在ida中打开程序，将main函数f5查看伪代码 sub_4007c0()函数为失败

```
v3 = strlen(s);
if ( v3 < strlen(v8) )
  sub_4007C0();
for ( i = 0; i < strlen(s); ++i )
{
  if ( i >= strlen(v8) )
    sub_4007C0();
  if ( s[i] != (char)(*((_BYTE *)&v7 + i % v6) ^ v8[i]) )
    sub_4007C0();
}
sub_4007F0();
```

根据代码知道输入的password长度应该大于18，即v3>strlen(s)

```
  sub_4007C0();
if ( s[i] != (char)(*((_BYTE *)&v7 + i % v6) ^ v8[i]) )
```

根据这个知道s应该是v7与v8的异或
对于v7,LL是长长整型，v7要转换为16进制然后在转换为字符串，而且字符是小端序，所以把得到的字符翻转然后和v8的每一位进行异或

- 编写代码得到flag

```
v8=":\"AL_RT^L*.?+6/46"
v6=7
v7='harambe'
s=''
for i  in range(len(v8)):
 s+=chr(ord(v7[i%v6])^ord(v8[i]))
print s
```

Open-source

下载文件发现是c语言

```
    if (first != 0xcafe) {
        printf("you are wrong, sorry.\n");
        exit(2);
    }

    unsigned int second = atoi(argv[2]);
    if (second % 5 == 3 || second % 17 != 8) {
        printf("ha, you won't get it!\n");
        exit(3);
    }

    if (strcmp("h4cky0u", argv[3])) {
        printf("so close, dude!\n");
        exit(4);
    }

    printf("Brr wrrr grr\n");

    unsigned int hash = first * 31337 + (second % 17) * 11 + strlen(argv[3]) - 1615810207;
```

发现关键hash计算涉及到first，second，和argv[3]
根据下述判断语句，知道first=0xcafe

```
  if (first != 0xcafe) {
      printf("you are wrong, sorry.\n");
      exit(2);
  }
```

根据下述判断语句，知道second % 5 ！ = 3 || second % 17 = 8

```
  if (second % 5 == 3 || second % 17 != 8) {
      printf("ha, you won't get it!\n");
      exit(3);
  }
```

而argv[3]是用了strcmp语句来比较，所以argv[3]=“h4cky0u”

```
  unsigned int hash = first * 31337 + (second % 17) * 11 + strlen(argv[3]) - 1615810207;
```

根据上述关键条件，写出求hash的代码

```
first= 0xcafe
third="h4cky0u"
hash = first * 31337 + 8 * 11 + len(third) - 1615810207;
print hex(hash)
```

Getit

下载文件用winhex分析，发现是elf文件
用ida打开，将F5查看伪代码

```
v10 = *MK_FP(__FS__, 40LL);
LODWORD(v6) = 0;
while ( (signed int)v6 < strlen(s) )
{
  if ( v6 & 1 )
    v3 = 1;
  else
    v3 = -1;
  *(&t + (signed int)v6 + 10) = s[(signed __int64)(signed int)v6] + v3;
  LODWORD(v6) = v6 + 1;
}
strcpy(filename, "/tmp/flag.txt");
stream = fopen(filename, "w");
fprintf(stream, "%s\n", u, v6);
for ( i = 0; i < strlen(&t); ++i )
{
  fseek(stream, p[i], 0);
  fputc(*(&t + p[i]), stream);
  fseek(stream, 0LL, 0);
  fprintf(stream, "%s\n", u);
}
```


MK_FP是一个宏。功能是做段基址加上偏移地址的运算，也就是取实际地址。
简单分析程序知道while循环计算的应该就是flag，查看t的值

```
; char t
t               db 'S'                  ; DATA XREF: main+65↑w
                                        ; main+C9↑o ...
aHarifctf?????? db 'harifCTF{??????????????????????????????????}',0
                align 20h
```

说明t存放的应该就是flag
我们用python代码编写循环过程，输出得到flag

```
s='c61b68366edeb7bdce3c6820314b7498'
v6=0
v3=0
t='SharifCTF{??????????????????????????????????}'
t=list(t)  //在python中,数字，字符串和元组都是不可变对象。列表是可变对象
while v6<len(s):
 if(v6&1):
  v3=1
 else:
  v3=-1
 t[v6+10]=chr(ord(s[v6])+v3)
 v6 +=1
t="".join(t)
print t
```