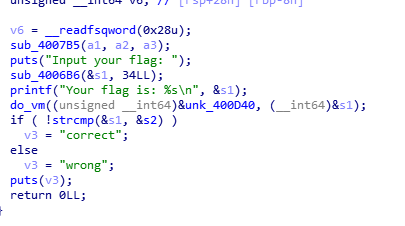
# 题目分析

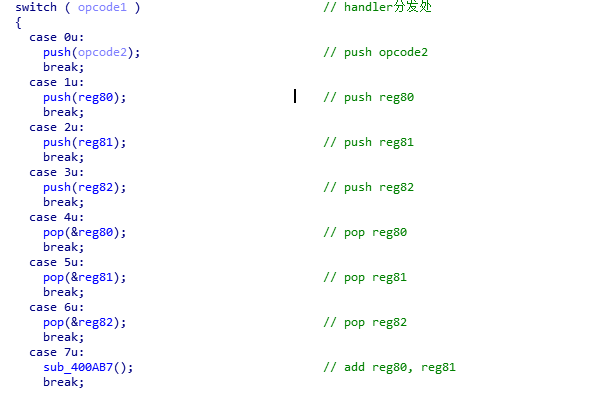
程序用到了vmp，拖进IDA进行反编译。



可以看到do\_vm是对输入进行了处理，unk\_400D40即是opcodes。

我们来看do\_vm函数。





第一个switch处是判断是否又两个操作数，第二个switch处handler分发出。

我们分析出每个handler的功能。

Case 0: push opcode2

Case 1: push reg80

Case 2: push reg81

Case 3: push reg82

Case 4: pop reg80

Case 5: pop reg81

Case 6: pop reg82

Case 7: add reg80, reg81

Case 8: add reg80, opcode2

Case 9: add reg81, opcode2

Case A: add reg82, opcode2

Case B: sub reg80, reg81

Case C: sub reg80, opcode2

Case D: sub reg81, opcode2

Case E: sub reg82, opcode2

Case F: xor reg80, reg81

Case 0x10: cmp reg80, reg81

Case 0x11: call opcde2

Case 0x12: ret

Case 0x13: jmp opcode2

Case 0x14: je opcode2

Case 0x15: push input[reg82]

Case 0x16: pop input[reg82]

Case 0x17:exit

start  
11 2D  
0 22  
05  
10  
14 09  
17  
  
addr9  
0 32  
05  
03  
11 16  
06  
00 16  
5  
11 16  
17  
  
fun16  
e 1  
15  
4  
f  
1  
16  
2  
0 0  
4  
3  
5  
10  
14 2B  
5  
9 3  
13 16  
  
addr2B:  
5  
12  
  
fun2D:  
15  
04  
10  
14 36  
A 1  
13 2d  
  
addr36:  
03   
04  
12

然后我们手动翻译一下opcodes。

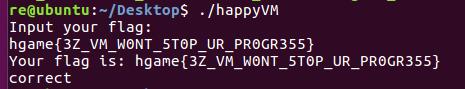
start:  
call 2D //2D函数返回输入flag的长度  
push 22  
pop reg81  
cmp reg80 reg81 //判断flag长度是否为0x22  
je 9  
exit  
  
  
addr9:  
push 32  
pop reg81   
push reg82  
call 16 //对输入进行第一次异或运算，传入的参数为0x32  
pop reg82  
push 16  
pop reg81  
call 16 //对输入进行第二次异或运算，传入的参数为0x16  
exit  
  
  
fun 16:  
sub reg82, 1 //reg82最初的值为0x22，然后一直递减  
push input[reg82]  
pop reg80  
xor reg80, reg81  
push reg80  
pop input[reg82]  
push reg81  
push 0  
pop reg80   
push reg82  
pop reg81  
cmp reg80, reg81 //当reg82值递减到0的时候，跳出函数  
je 2B  
pop reg81  
add reg81, 3  
jmp fun16  
  
addr2B:  
pop reg81  
ret  
  
  
fun 2D: //函数返回输入flag的长度  
push input[reg82]  
pop reg80  
cmp reg80, reg81  
je 36  
add reg82, 1  
jmp 2D  
  
addr 36:  
push reg82  
pop reg80  
ret

可以得到do\_vm的逻辑是首先判断输入的长度是否为0x22，然后两次调用异或函数对输入进行处理。

# 脚本

arr = [132, 131, 157, 145, 129, 151, 215, 190, 67, 114, 97, 115, 115, 12, 106, 112, 115, 17, 72, 44, 52, 51, 49, 54, 35, 52, 62, 92, 35, 78, 23, 17, 25, 89]  
flag = ""  
  
reg81 = 0x16  
for i in range(len(arr))[::-1]:  
 arr[i] = arr[i] ^ reg81  
 reg81 += 3  
  
reg81 = 0x32  
for i in range(len(arr))[::-1]:  
 flag += chr(arr[i] ^ reg81)  
 reg81 += 3  
  
print flag[::-1]





## Flag：hgame{3Z\_VM\_W0NT\_5T0P\_UR\_PR0GR355}