学 bei 上了　writeup

miaES：

对称加密，题目说 key 泄露了也没事，所以直接不用 key，勇敢把密文输进去用 iv 再加密一次，遂出。

```
# key = urandom(16)
# iv = urandom(16)

# ciphertext = encrypt_flag(iv, FLAG)
# print(f'ciphertext = {ciphertext}')
# print(f'iv = {iv}')
# print(f'key = {key}')

ciphertext = b'\xc6*\xe0^\xeb\xd7}\xcf\x99?IT;j{\xf6\x08\xc3J\xad\x0b6\xaa\x82\xb5[]X>m\xcf\xbc7V\xc1(s\xa2>\xf15\xa5\x91kg\xa4IT\xa4I\xc5*7B\x
iv = b"wL\xc58C\x9d3\x7f\xa85\x19\x89\x9b\x8d`'"
key = b'9\x83\x13i\xdbA\xf0\x88\xa9b$^\x7f\x1b\xd0\xb8'

flag=encrypt_flag(iv,ciphertext)
print(flag)
```

```
问题 2  输出  调试控制台  终端  端口

PS C:\Users\lenovo\Desktop\PY> & C:/Users/lenovo/AppData/Local/Programs/Python/Python312/python.exe c:/Users/lenovo/Downloads/task.py
b'NOCTF{0h_mi4_Nev3r_7h0ugHt_h3r_Ae3_w1Ll_8e_DeCryPt3d_c@ngr3tu1at1on3}\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0b\x10\xa0V\x89\xd3\xe7\x81f\x9f\x0b\xc6\x1
PS C:\Users\lenovo\Desktop\PY>
```

Imitate:

流程超级长的题，第一天没有勇气算 n，第二天等了一会就出了（血亏 w）

```
# Ca =
b'eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.TWF5YjM.NGI5N2IwYjUyY2Y0ZTlkMTJl
ZTFkNjhkYTE5MTRlZTUwY2UxOGRjMWViNGZkZDE5YWZiNDIzMGY3OWE2ZmI5YzQwNTI3ZGM
1OGQ0OTIxZmI5ZWI3Zjc1ZGY2ZjBhZGI2MWU1YWQ1MWM4MjA4M2Y5M2IzZWZlZDVjZTM2YW
RjNDQ'
# Cb =
b'eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.Q3J5cDc.MWJjNTI4NDIzNGU2MjgxZmY4
ZDY4YmUwZTNmMWM3ZmQ0OTNmNDAwMDMzNTkyMjRmNjcwOWVmZTM0MjM2OGM0ZDc5ZDBmNTY
wYzAzZGQ4ZDU4ZDRhNjk2NzU1MDU3NjVlZDBjNTgwOWJkMjViZWZlNTlmODY3NTZkYzI0Nm
JmZg'

#我们从 Ca，Cb 中先取出中间的 PAYLOAD 部分，（base64 加密补上"="）
aa=b'TWF5YjM=='
bb=b'Q3J5cDc=='
#解密出 a,b,即 encode()函数中的 data
ad=b64decode(aa)
```

```
bd=b64decode(bb)

# print(ad)
# print(bd)
#原谅我做的时候随缘命名了
#然后用 data,计算出相应的 to_sig
asg=bytes_to_long(ad)
bsg=bytes_to_long(bd)

# print(asg)
# print(bsg)

#取出 Ca,Cb 的 SIGNATURE 部分，并解码
aSG=b'NGI5N2IwYjUyY2Y0ZTlkMTJlZTFkNjhkYTE5MTRlZTUwY2UxOGRjMWViNGZkZDE5Y
WZiNDIzMGY3OWE2ZmI5YzQwNTI3ZGM1OGQ0OTIxZmI5ZWI3Zjc1ZGY2ZjBhZGI2MWU1YWQ1
MWM4MjA4M2Y5M2IzZWZlZDVjZTM2YWRjNDQ=='
bSG=b'MWJjNTI4NDIzNGU2MjgxZmY4ZDY4YmUwZTNmMWM3ZmQ0OTNmNDAwMDMzNTkyMjRmN
jcwOWVmZTM0MjM2OGM0ZDc5ZDBmNTYwYzAzZGQ4ZDU4ZDRhNjk2NzU1MDU3NjVlZDBjNTgw
OWJkMjViZWZlNTlmODY3NTZkYzI0NmJmZg=='
aSG=b64decode(aSG)
bSG=b64decode(bSG)
# print(aSG)
# print(bSG)

#得到前一步 16 进制的 A,B
A=0x4b97b0b52cf4e9d12ee1d68da1914ee50ce18dc1eb4fdd19afb4230f79a6fb9c405
27dc58d4921fb9eb7f75df6f0adb61e5ad51c82083f93b3efed5ce36adc44
B=0x1bc5284234e6281ff8d68be0e3f1c7fd493f40003359224f6709efe342368c4d79d
0f560c03dd8d58d4a69675505765ed0c5809bd25befe59f86756dc246bff

# #通过同余方程组的计算，得到 n（跑了 5 分钟）
e=0x10001
x=pow(A,e)-asg
y=pow(B,e)-bsg
n=GCD(x,y)
# print(n)

n=629338248806181027691662773716388020784274308117803959241186868043682
373883528841879700216240894262894587136358796173127223712439392240252342
7454375365893
#然后用 gift 算出 d
gift =
623854769787005012140895681851956496592749343630597442119311658347814354
1484922844697874279636322776003055456640230725303200462321219636562097
```

```
asg^d===A mod n
bsg^d===B mod n
d*e===1 mod n

A^e===asg mod n
B^e===bsg mod n

d*n*e=gift
```

```
47754305160219588607477451932744562460172642629992714110316010115501972
43951058824991129679927663692540286012424727260428137908706271075996269
46313265014769270777400
Mod =
13978353049249998936680618619097020170704578461795551099467066826436512
56137805341462076046622349727812054157923157946948041258067879625505043
58642185835486015743979918457538944006966377376366330483194448343751231
56746705108588309050308147673813308032577597213521636016082142455984080
07029846267146134034103 41
d=gift*inverse(n,Mod)*inverse(e,Mod)%Mod
# print(d)


#用 n,e,d 算 p,q
p = 1
q = 1
while p==1 and q==1:
    k = d * e - 1
    g = random.randint ( 0 , n )
    while p==1 and q==1 and k % 2 == 0:
        k =k//2
        y = pow(g,k,n)
        if y!=1 and GCD(y-1,n)>1:
            p = GCD(y-1,n)
            q = n//p

# # print(p)
# # print(q)

p=10368066877173060924563012397453584615112811731178429299449406626415 8419199927
q=60699671043959864930192653458667491948182260386734436978995113729505 181882659

#p,q 都试一下，然后就出了
C =
34549321084535799161313994305025115882306195453188814309913499010389026 11605453734080149058230878714433909067049429244397457303301903246046075 36938266367079808006301848264470108499531968168253360898869342451175503 1
s=C*inverse(q,Mod)%Mod
flag=long_to_bytes(s)
print(flag)
```

完结撒花！

Segments:

用 objconv 把 segments.obj 转成汇编的 segments.asm 文件，然后用 VScode 打开，找，找到就完了

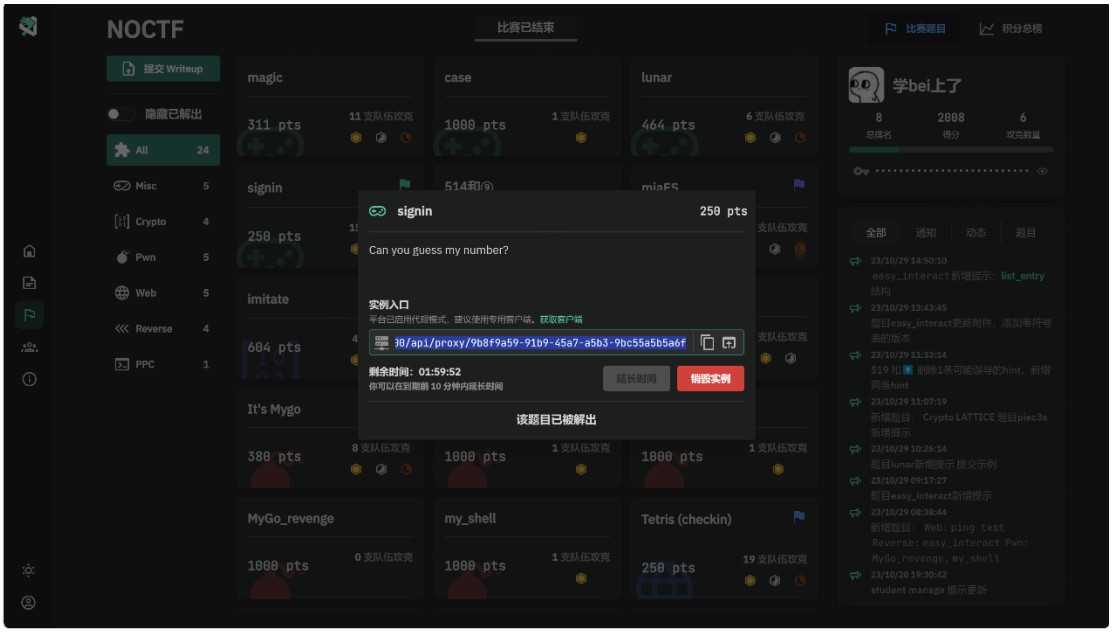
```
SECTION .flag{3 align=1 noexecute                       ; section number
4, data

_VAL1:                                                  ; byte
        db 01H                                          ; 0000 _ .


SECTION .F6BC00 align=1 noexecute                       ; section number
5, data

_VAL2:                                                  ; byte
        db 02H                                          ; 0000 _ .


SECTION .6-BA9F align=1 noexecute                       ; section number
6, data

_VAL3:                                                  ; byte
        db 03H                                          ; 0000 _ .


SECTION .-DCE6- align=1 noexecute                       ; section number
7, data

_VAL4:                                                  ; byte
        db 04H                                          ; 0000 _ .
```

```
SECTION .388A-0 align=1 noexecute                       ; section number
8, data

_VAL5:                                                  ; byte
        db 05H                                          ; 0000 _ .


SECTION .E33861 align=1 noexecute                       ; section number
9, data

_VAL6:                                                  ; byte
        db 06H                                          ; 0000 _ .


SECTION .3E029} align=1 noexecute                       ; section number
10, data

_VAL7:                                                  ; byte
        db 07H                                          ; 0000 _ .
```

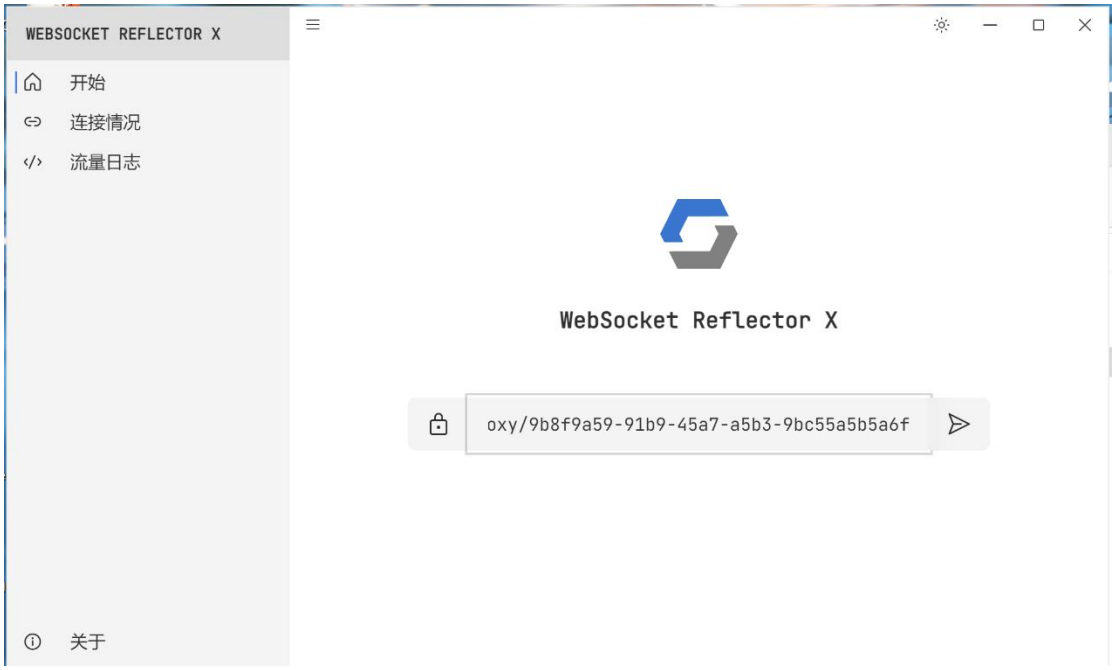
把这七个段拼起来就好了。
flag{3F6BC006-BA9F-DCE6-388A-0E33861}

SignIn：

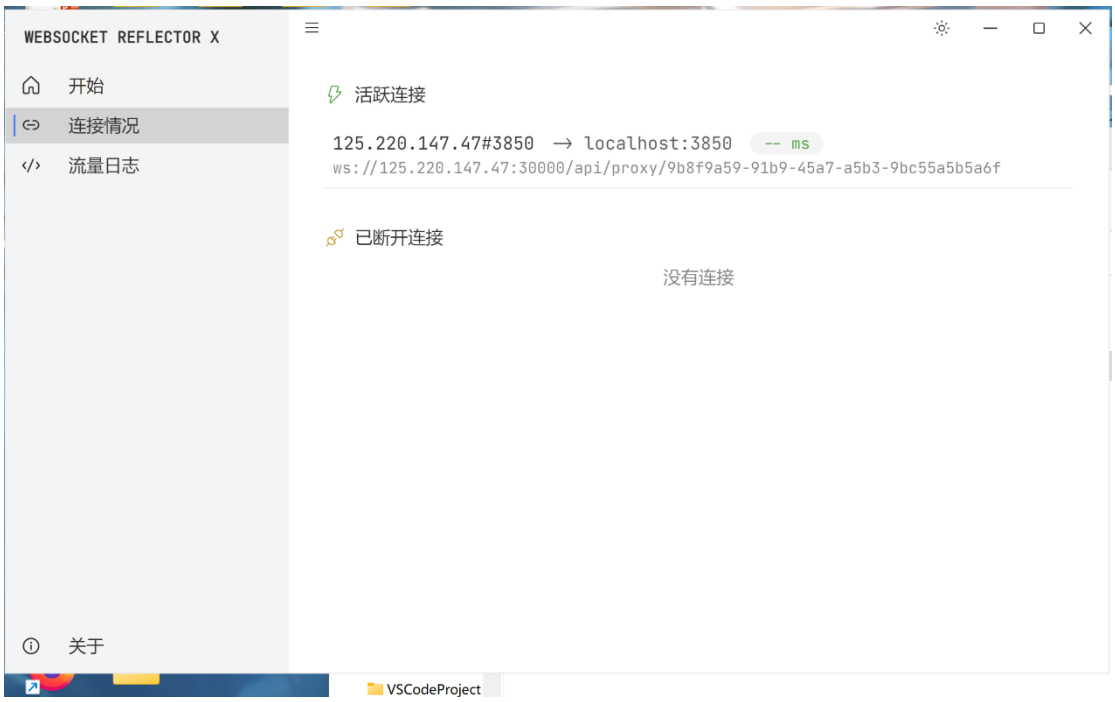
1. 打开容器实例，获取实例入口，复制



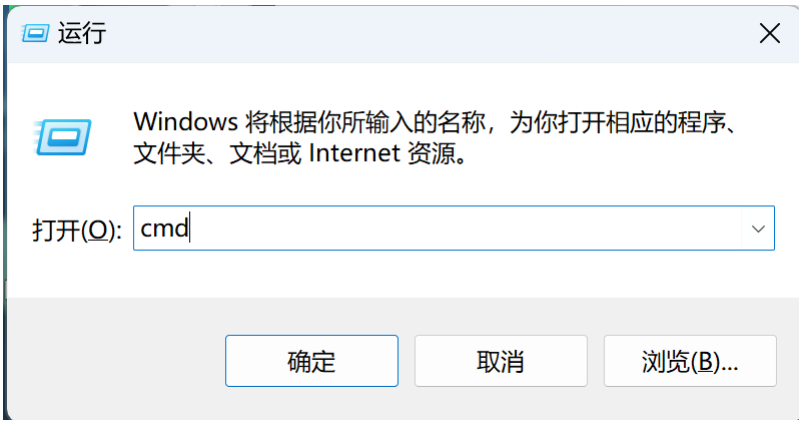
2. 打开 WSRX，输入刚才的入口



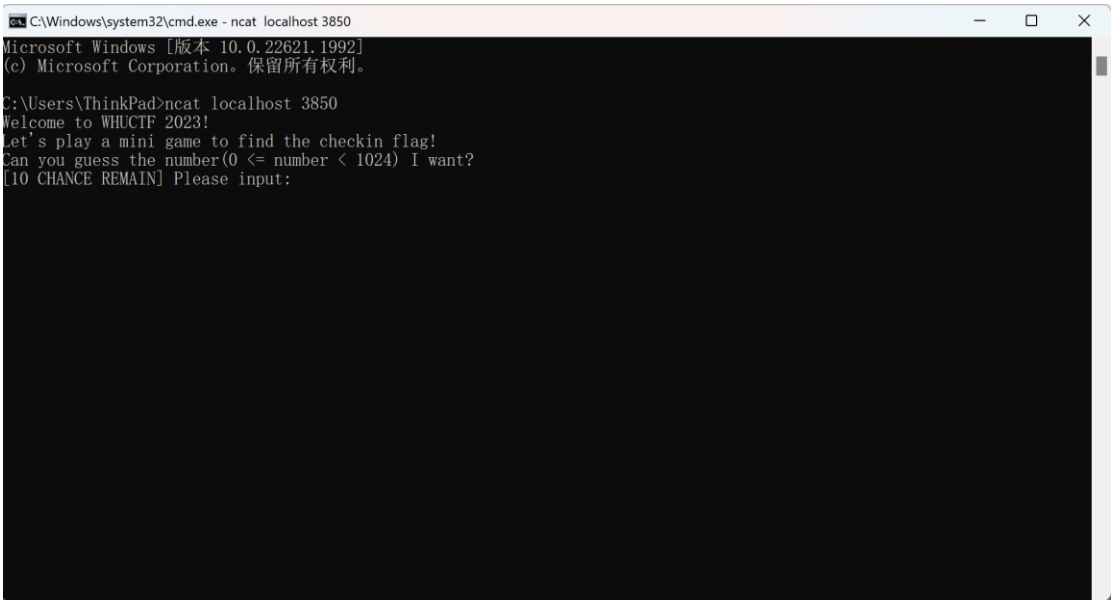
3.获取 localhost，使用 nc 进行访问



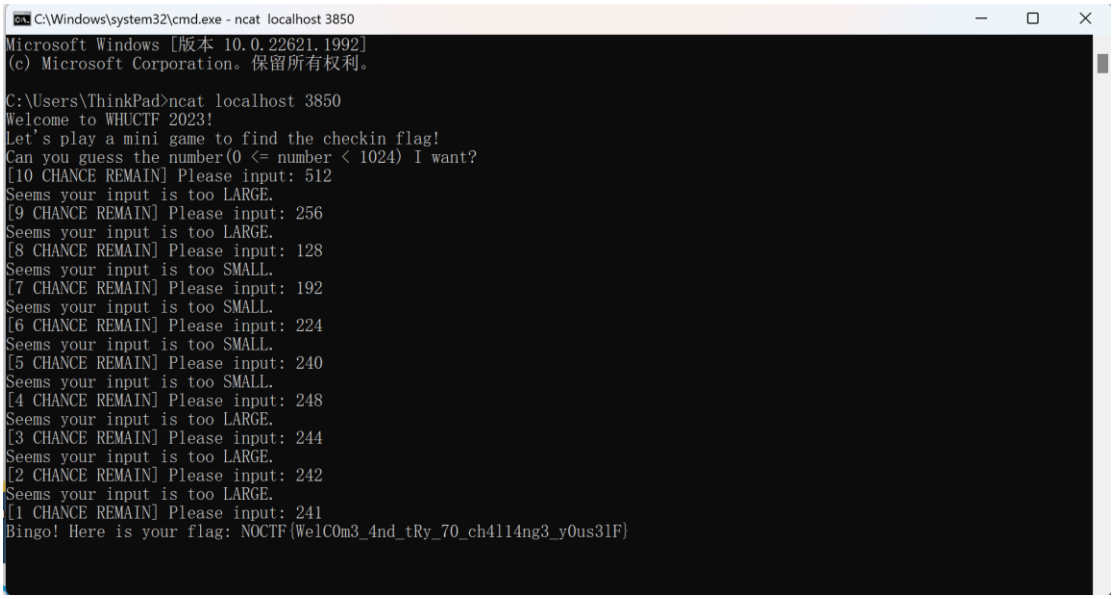
4.在桌面上输入 win + r；输入 cmd



5.输入命令行 ncat localhost 刚才的地址（3850）



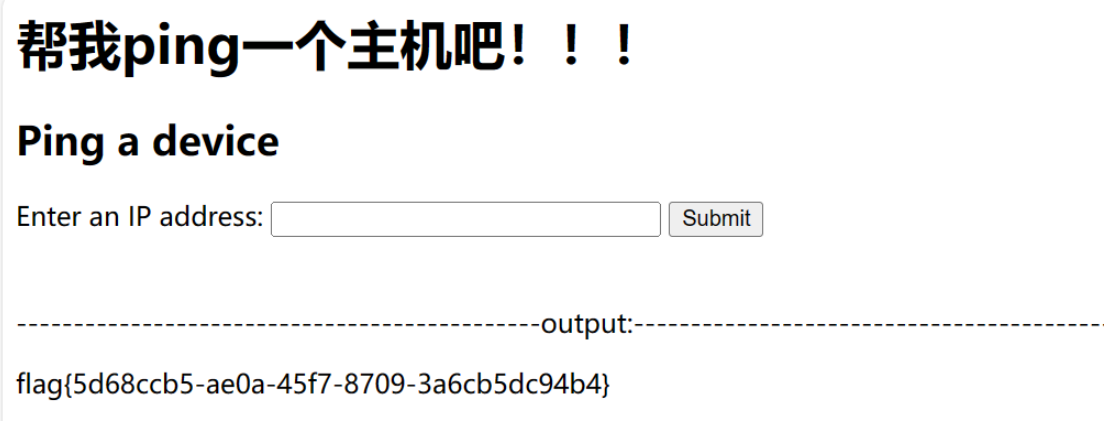
6.进入游戏界面，在 10 次使用机会下利用二分法找到该数字



7.得到 flag

ping test:

打开网页，一开始一直被抓住，（可能因为他把分号给过滤了），所以就使用 127.0.0.300||然后再输命令。l$*s 可以看到 index.php，c$*at 打开可以看到过滤了哪些。然后 ls /打开根目录，看到 flag，c$*at /f$*lag 就完了。



ctf web tetris(checkin):

思路：

打开实例找到网址在网页中打开进入俄罗斯方块

点击 F12 查看网页源代码

在 js 中找到被注释的 flag