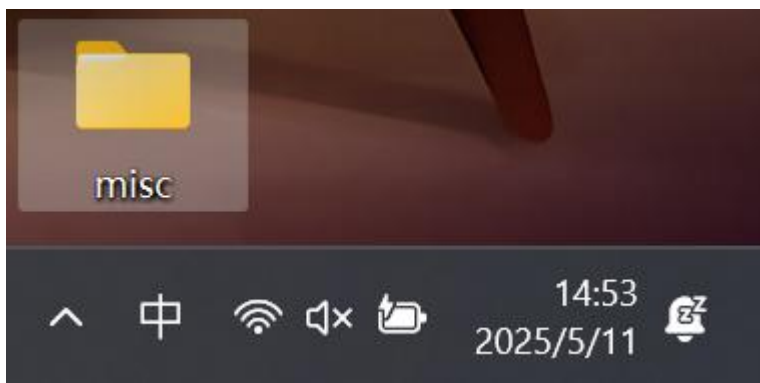


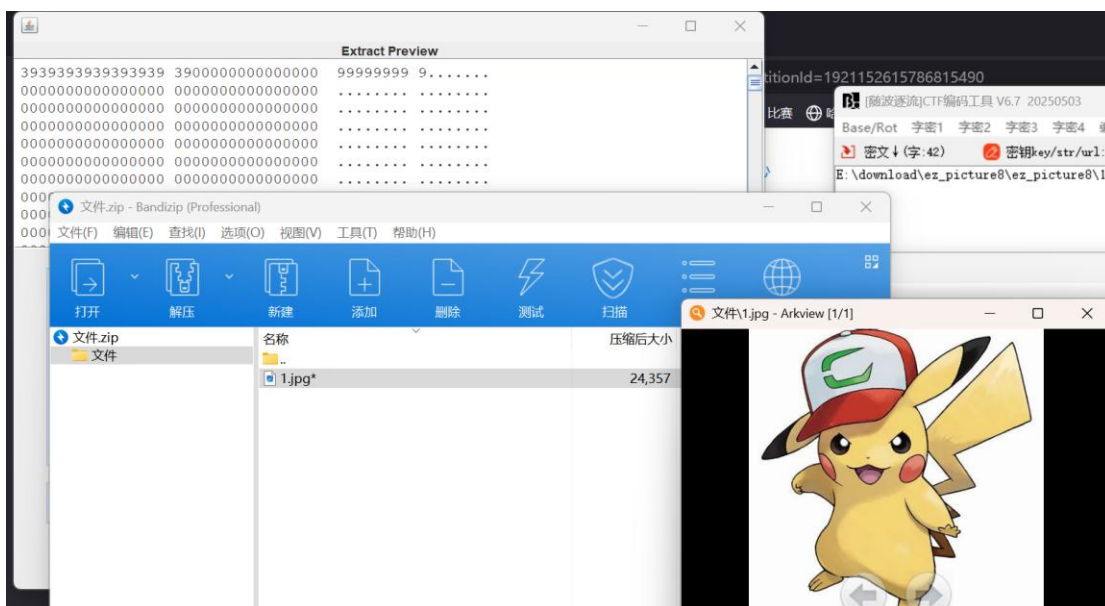
WP 模板

7、ez_picture



应该是 14:43 左右下载的

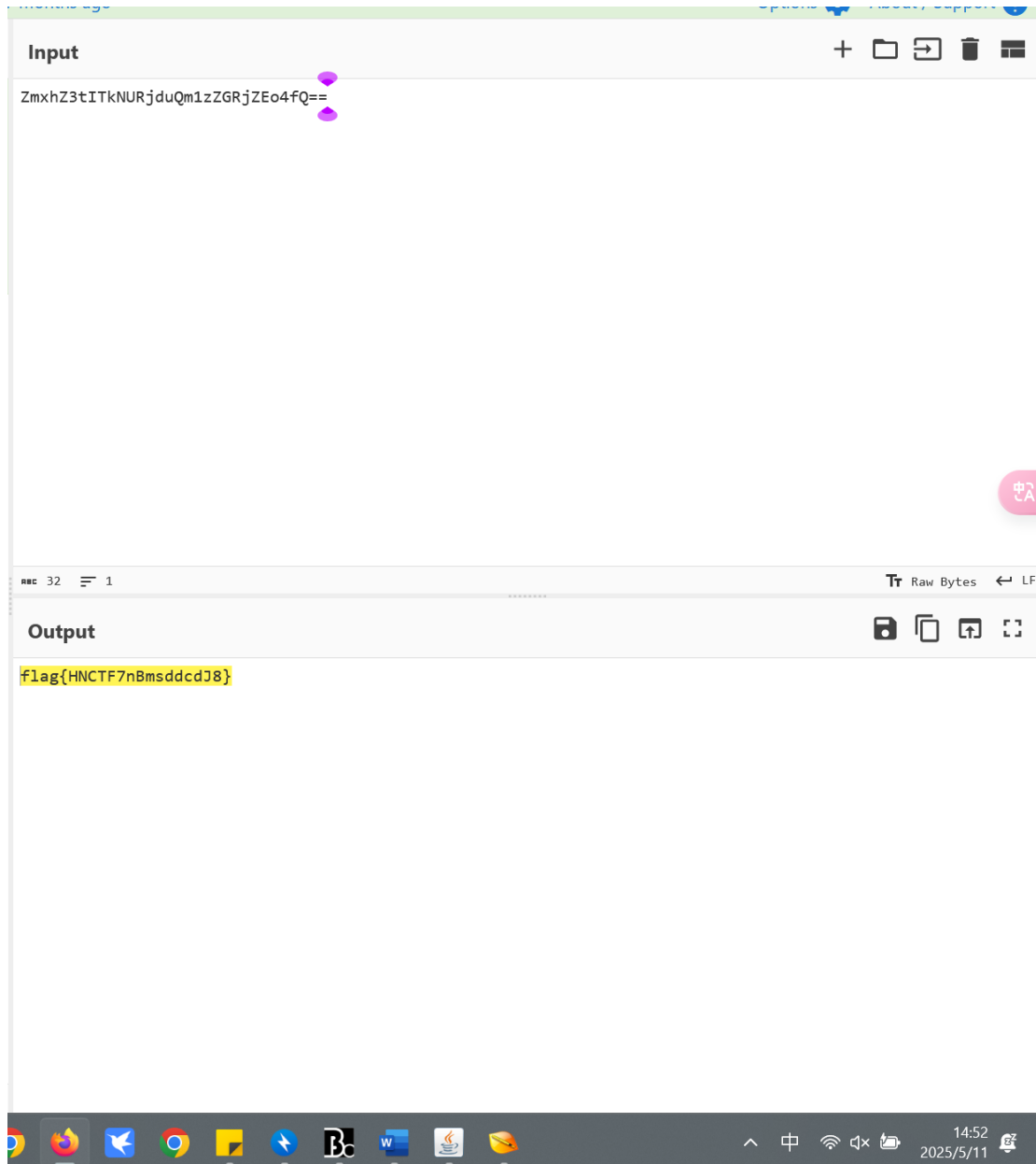
打开压缩包文件，发现 15.jpg 存在 LSB 隐写获得密码 999999999，得到图片 1.jpg



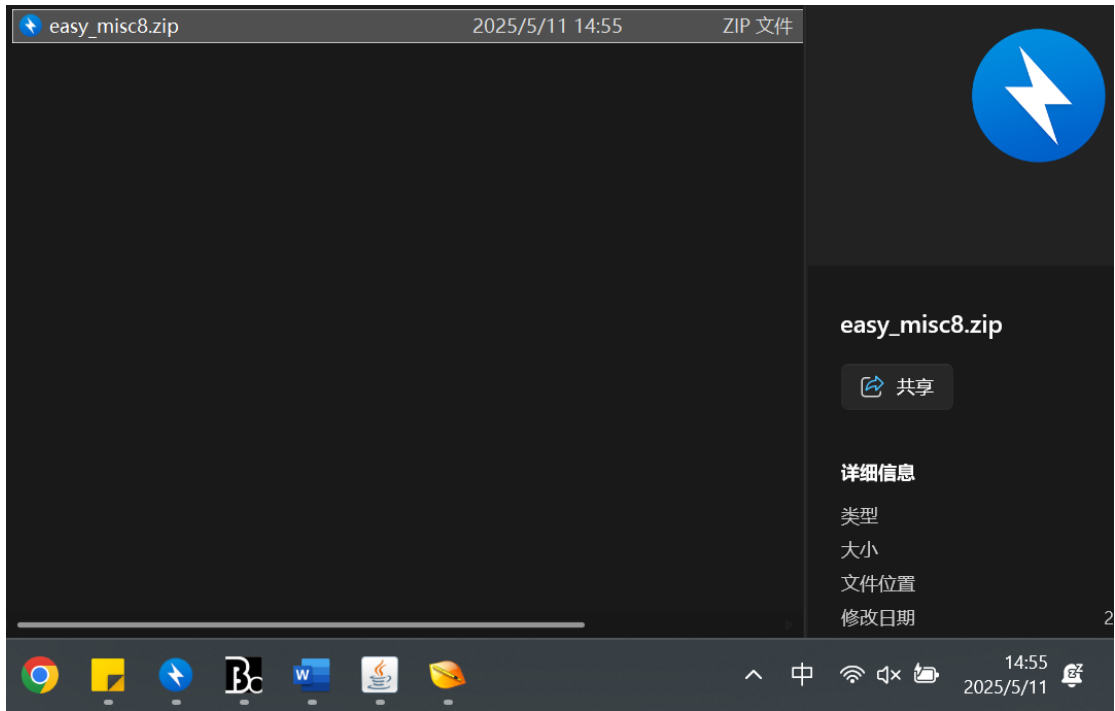
发现存在一个可疑的 base64 编码

```
Image Tag 0x4001: 0
EXIF Padding:   
Image Padding:   
Image XPSubject: ZmxhZ3tITkNURjduQmlzZGRjZEo4fQ==
Image ExifOffset: 342
Image Tag 0x4000: 0
```

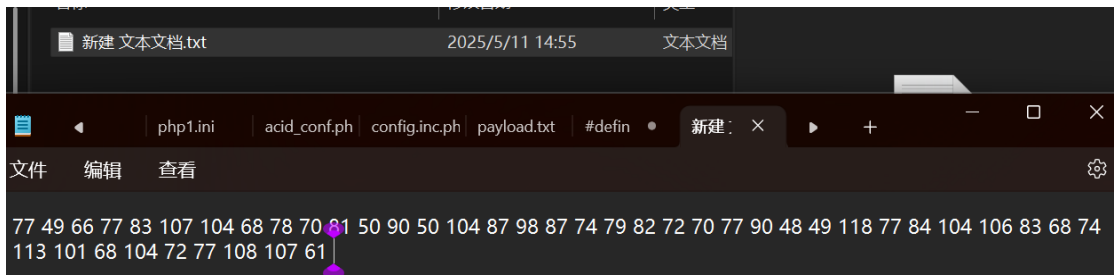
然后利用 CyberChef，base64 解密然后解出 flag



8、easy_misc



打开发现是一串数字，范围大小怀疑是 ASCII 码，



转 acii 码

```
1 str1 = '774966778310710468787081509050104879887747982'
2 result = ''
3
4 # 十六进制转十进制后转ascii码
5 # for i in range(0, len(str1), 2):
6 #     result.append(chr(int(str1[i:i + 2], 16)))
7
8 # 转Ascii码
9 while len(str1):
10     if int(str1[:3]) < 127:
11         result += chr(int(str1[:3]))
12         str1 = str1[3:]
13     else:
14         result += chr(int(str1[:2]))
15         str1 = str1[2:]
16
17 # 以空格分割
18 # print(" ".join(result))
19
20 print(result)
21
```

运行: 编码转换 ×

E:\reverse\python\venv\Scripts\python.exe E:\reverse\python\misc\编码\编码转换.py
M1BMSkhDNFQ2Z2hWbWJORHFMZ01vMTJhSDJqeDhHMLk=
进程已结束,退出代码0

得到 M1BMSkhDNFQ2Z2hWbWJORHFMZ01vMTJhSDJqeDhHMLk=, 利用 CyberChef 解码

From Base64

Alphabet
A-Za-z0-9+/= Remove non-alphabet chars

Strict mode

From Base58

Alphabet
123456789ABCDEFGH ... Remove non-alphabet chars

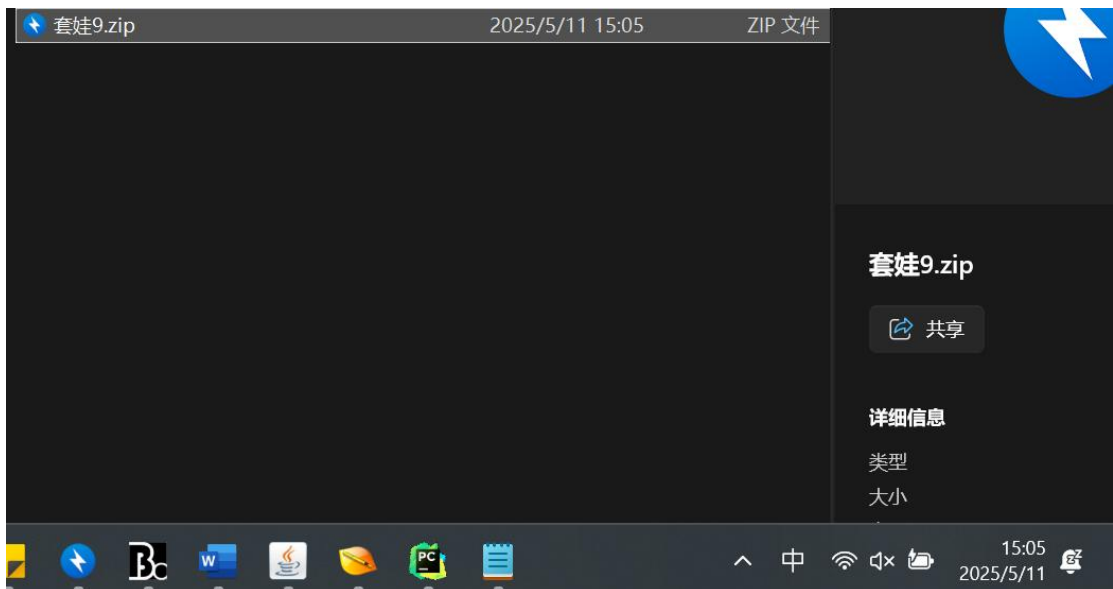
Output

synt{UAPGSuiliu359Fw0N}

得到 synt{UAPGSuiliu359Fw0N}, 但提交不对, 之前答案是 flag{}格式的, 进行移位 (其实就是 rot13), 得到答案



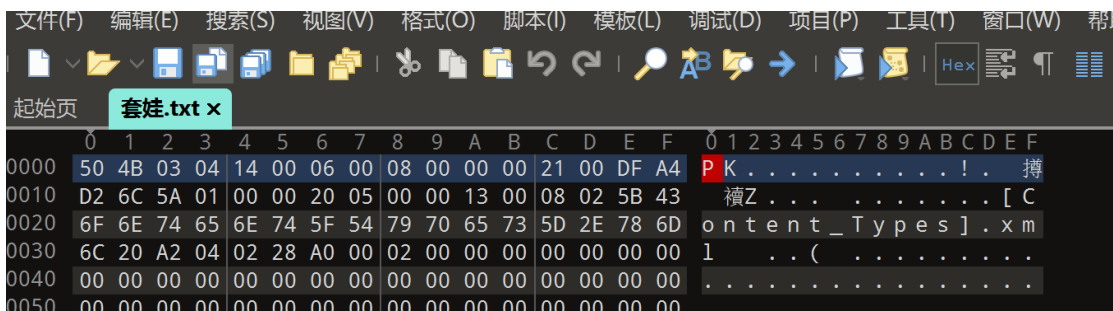
9、套娃



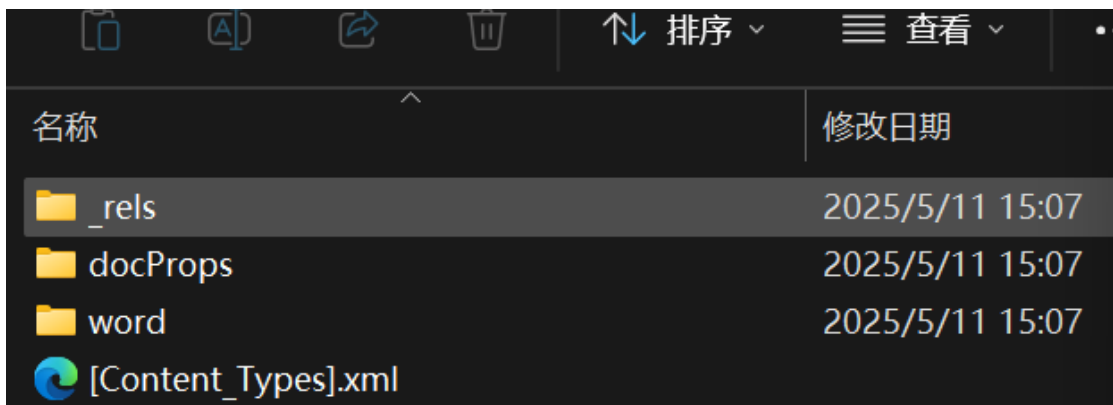
解压压缩包，发现是一个表格，无法打开，提示损坏
 所有 excel 文件本质上都是一个压缩包，我们更改后缀为 zip 打开



发现有一个套娃.txt，用 010 打开发现是压缩包头（504B），更改后缀后再次解压



得到正常 excel 压缩包



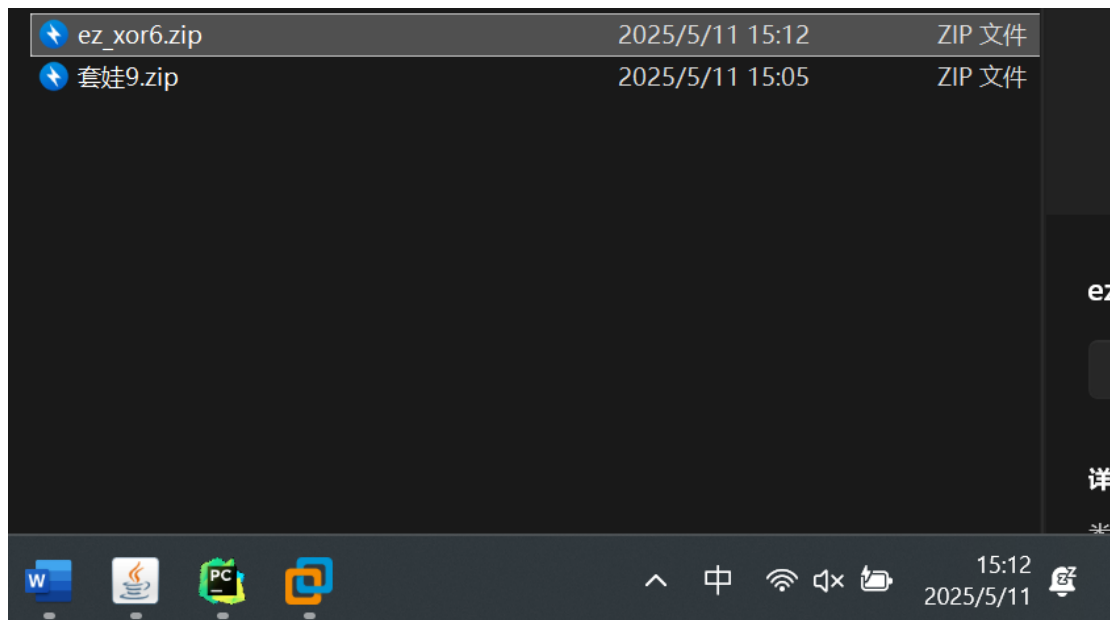
发现 document.xml 中有东西



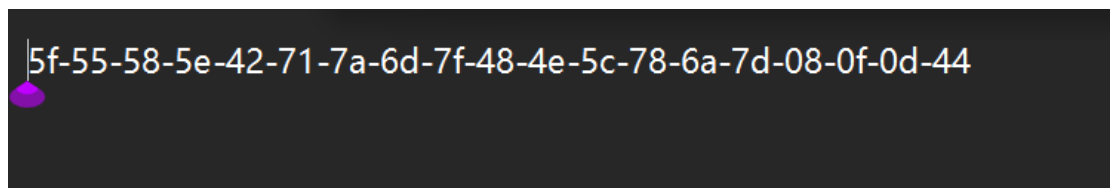
得到答案



10、ez_xor



打开压缩包得到一串十六进制

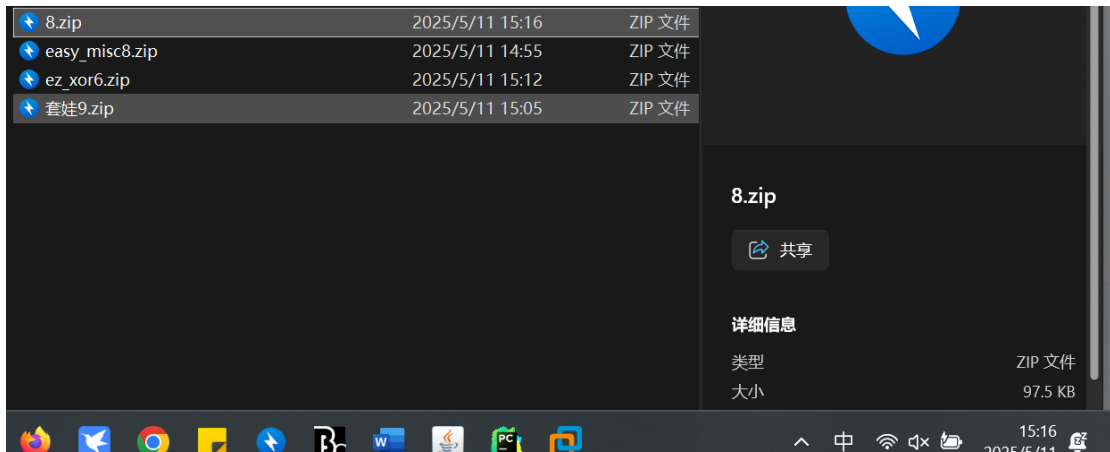


根据题目信息 xor，进行暴力异或

得到答案

题目答案:

11、光隙中的寄生密钥



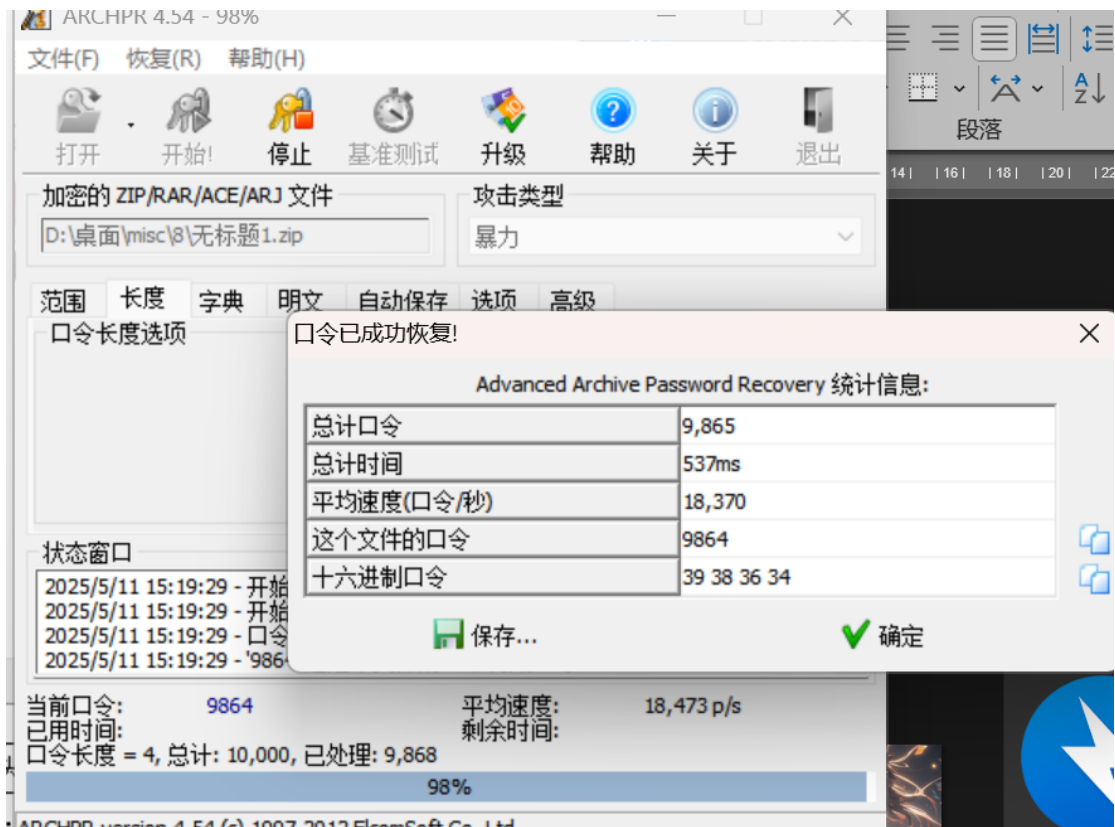
打开压缩包发现有张图片，用 010 打开发现 jpg 尾部之后存在添加数据



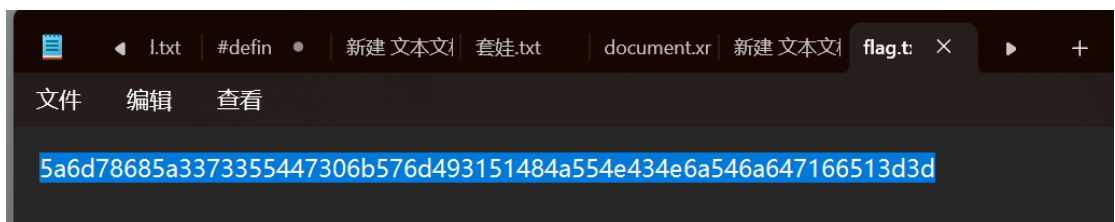
提取出来，504B 开头为压缩包，得到 flag.txt，需要密码

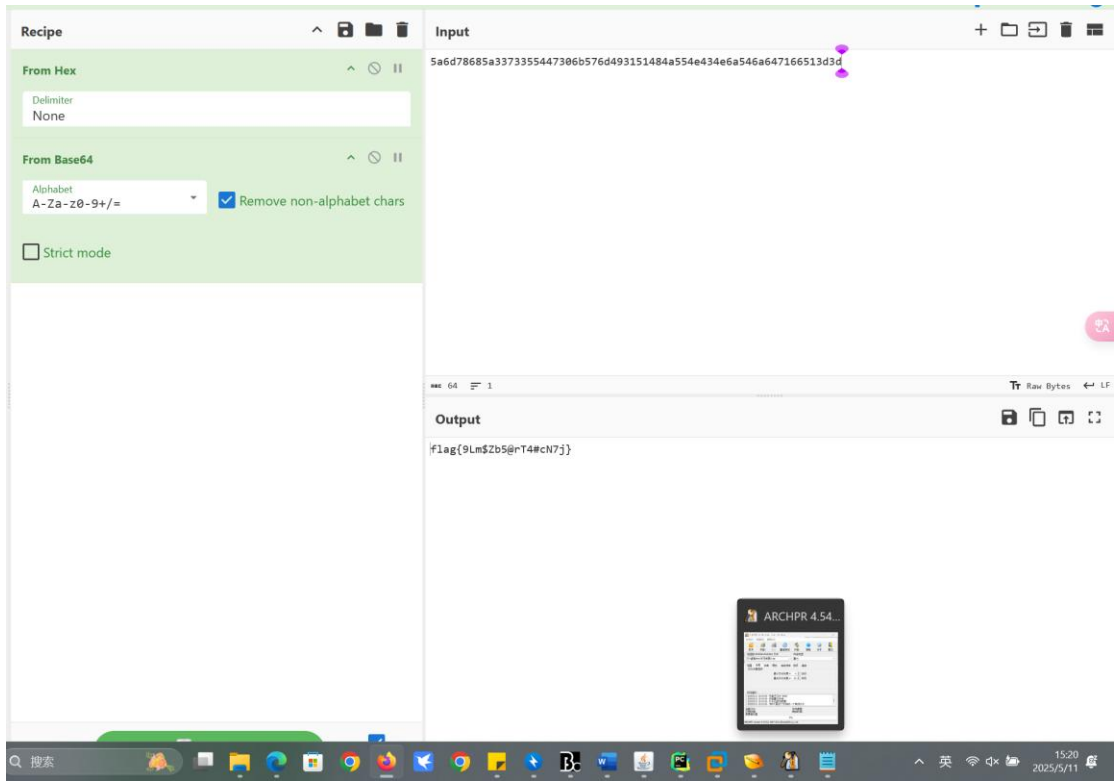


没有其他信息了，进行暴力破解

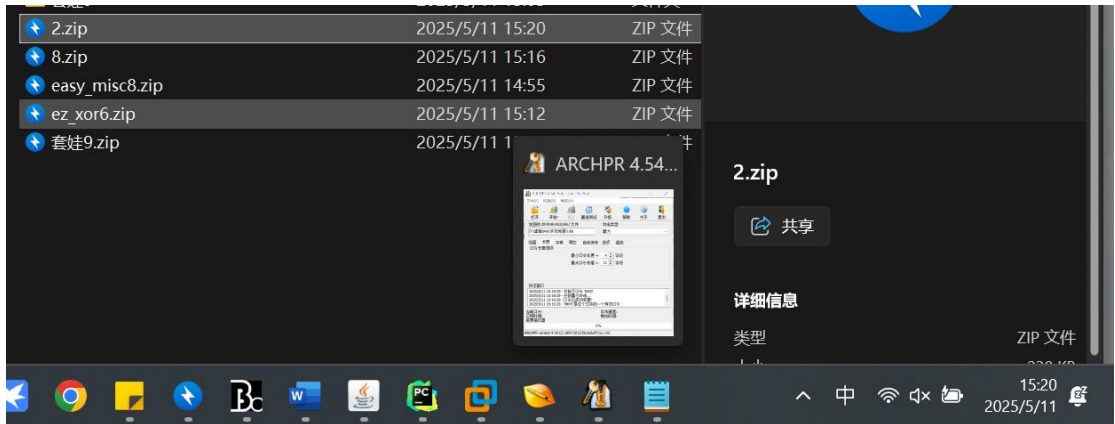


得到压缩包密码 9864，得到一串 16 进制

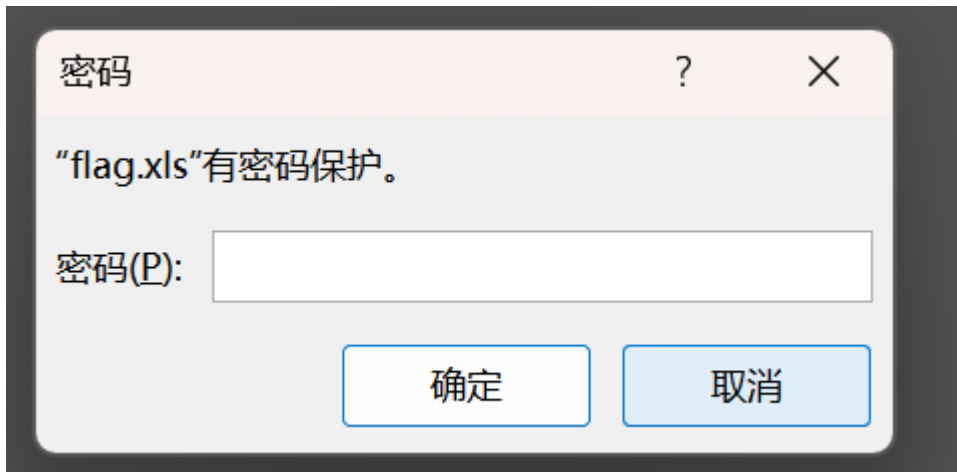




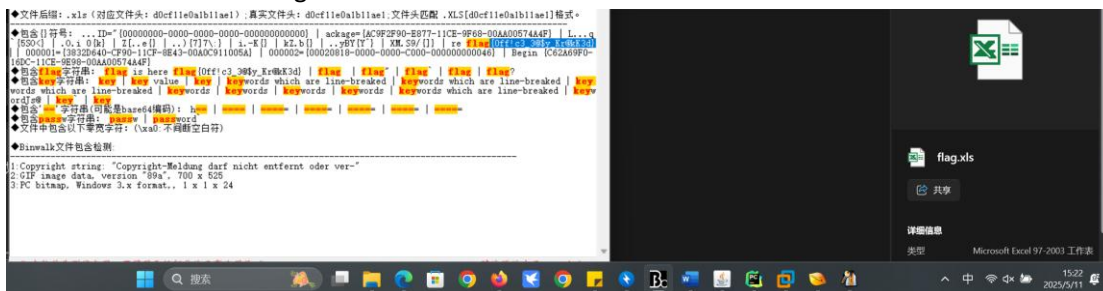
12、被折叠的显影图纸



打开压缩包发现是一个表格，存在密码保护



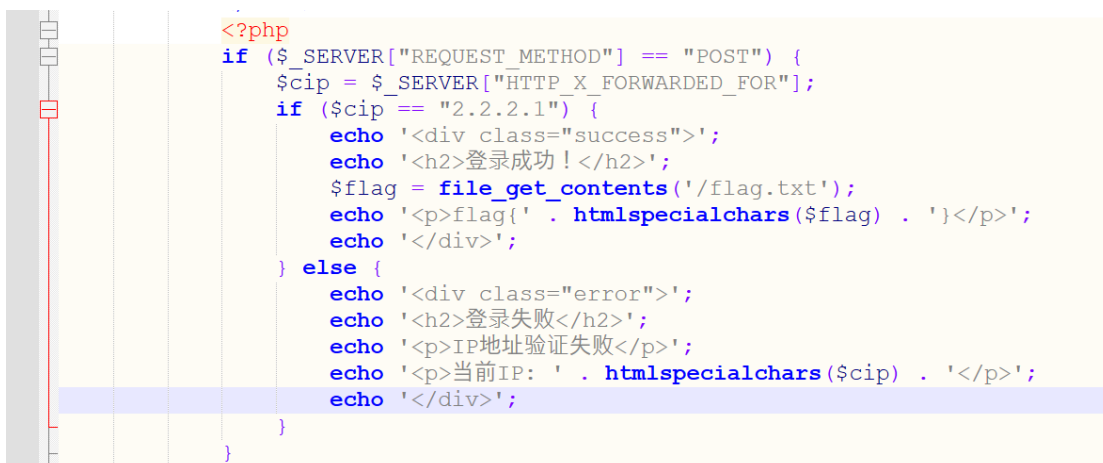
这个一把梭就能得到 flag



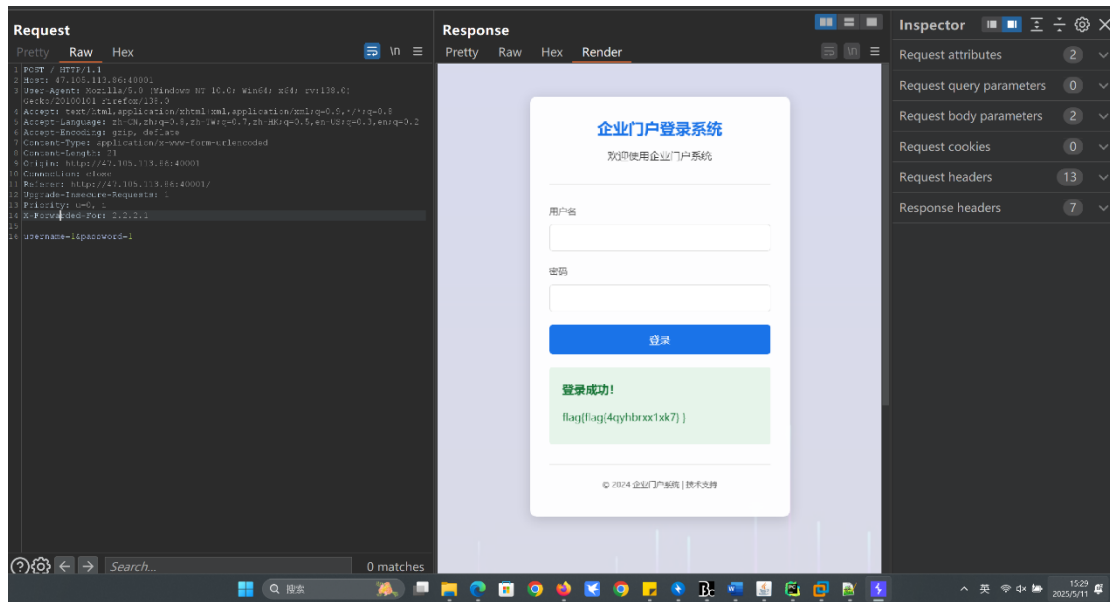
17、YWB_Web_xff



查看源码发现存在 IP 地址验证绕过漏洞，只需要指定 HTTP_X_FORWARDED_FOR 头信息就可以得到 flag.txt 的内容



抓包构造 X-Forwarded-For: 2.2.2.1



得到 flag

18、YWB_Web_命令执行过滤绕过

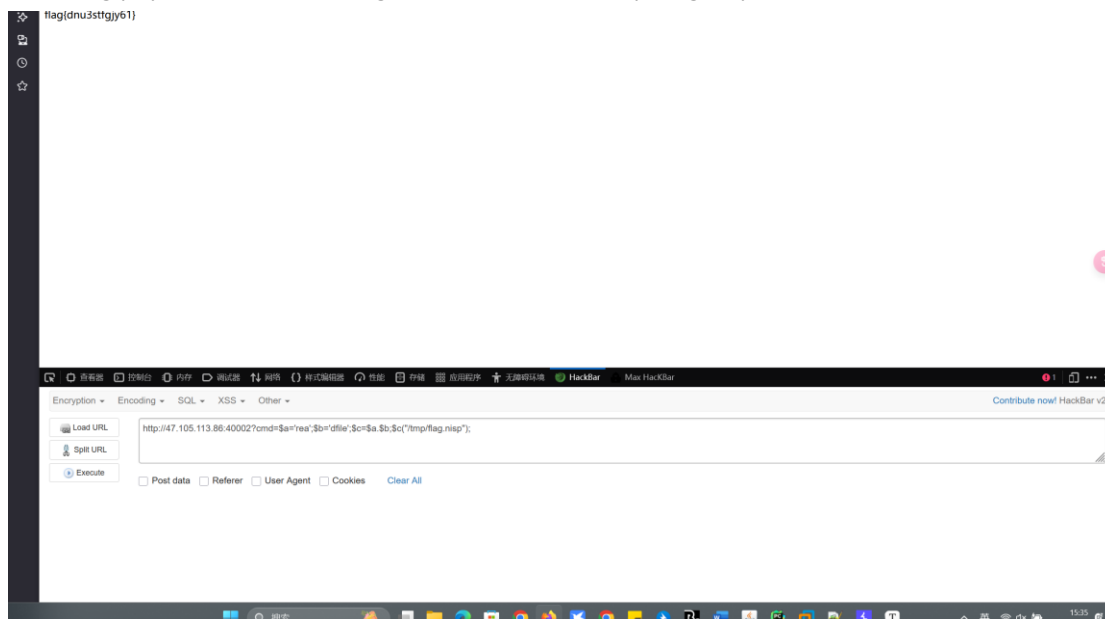
打开发现存在 php 代码，存在命令执行漏洞

```
<?php
# flag in flag.php
include("flag.php");
if(isset($_GET['cmd'])){
    $cmd = $_GET['cmd'];
    if(!preg_match("/system|exec|highlight|show_source|include|passthru|echo|print_r|cat|head|tail|more|less/i", $cmd)){
        if(preg_match("/flag/i", $cmd)){
            eval($cmd);
        } else {
            die("HACK!!!");
        }
    } else {
        die("HACK!!!");
    }
} else {
    highlight_file(__FILE__);
}
?>
```

大概规则是构造命令，命令必须包括 flag 字符串但是不能汉语黑名单中的单词
利用 PHP 的 highlight_file 函数的别名 show_source，并通过字符串拼接绕过黑名单

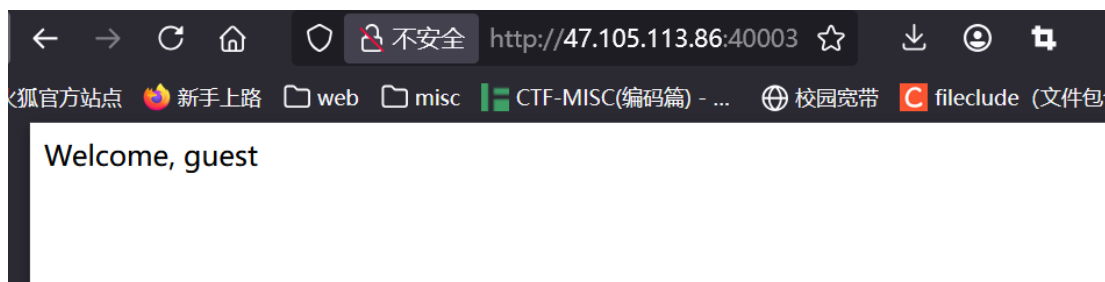


得到 flag.php 的内容，得到 flag 可能存在的路径/tmp/flag.nisp

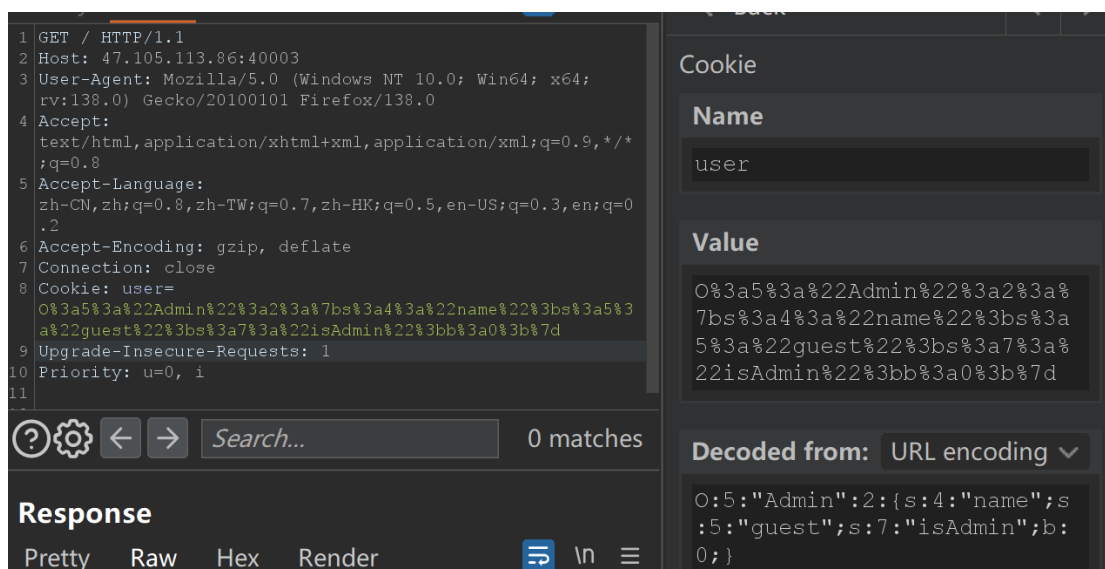


19、YWB_Web_未授权访问

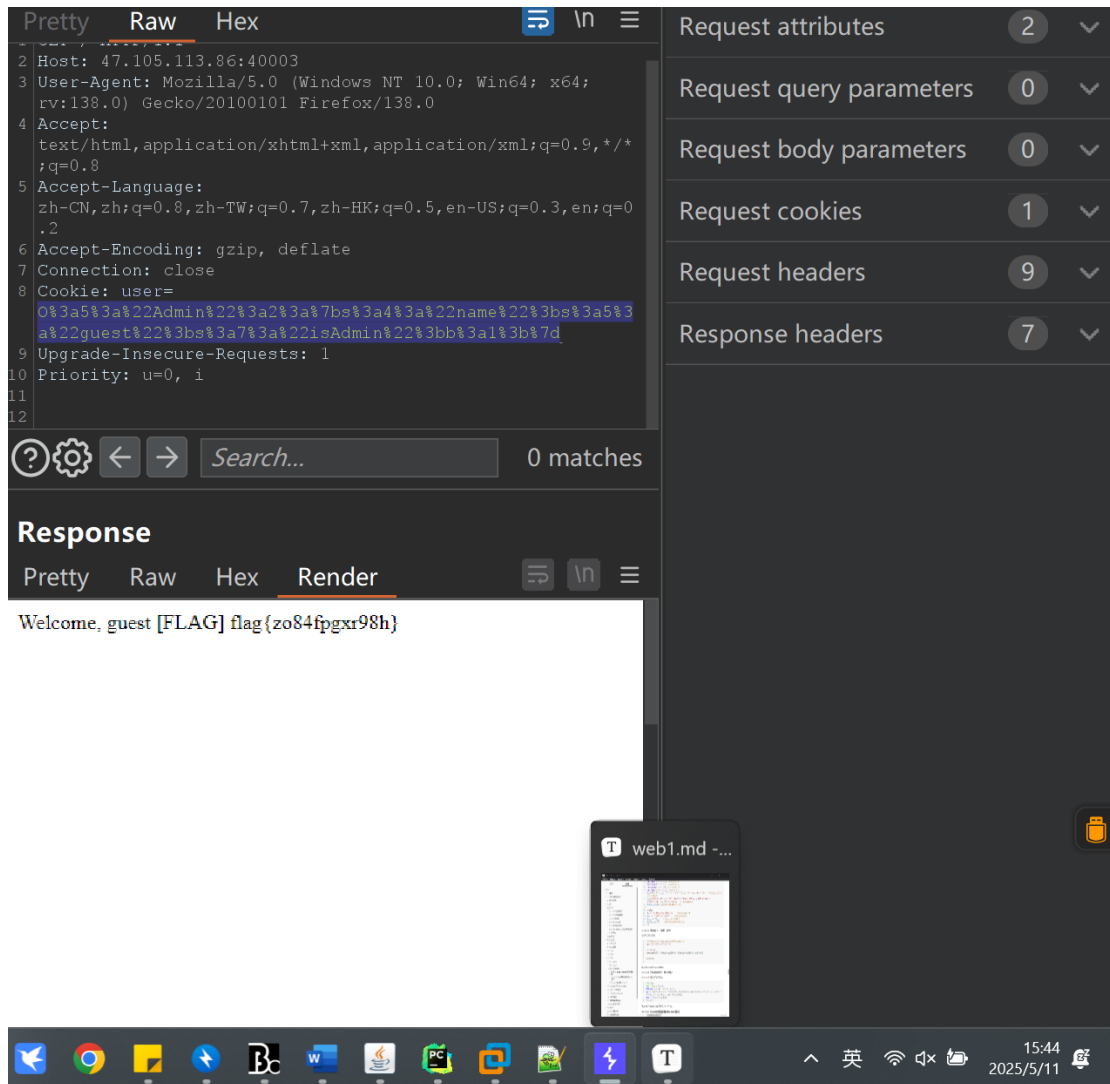
访问网站出现



发现没什么信息，进行抓包，发现存在 cookie，很可疑，进行解密

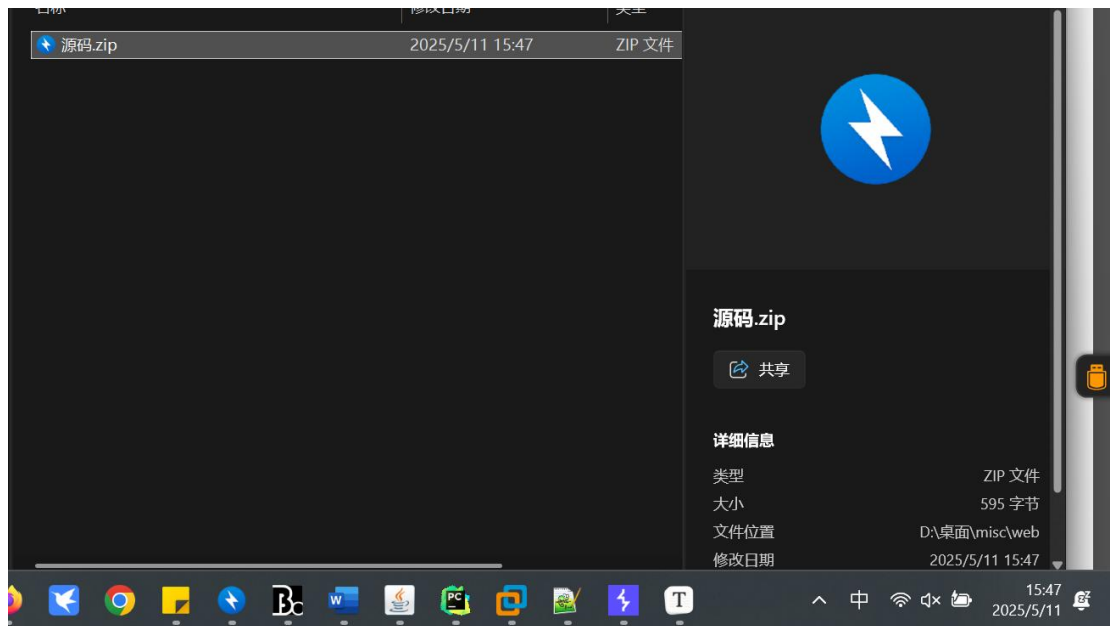


看到有一个 isAdmin 的判定，b 为 0，我们修改为 1 试试



得到 flag

20、YWB_Web_反序列化



21、easyweb

It works! <?php

```
if(isset($_POST['cmd'])) {
    @exec($_POST['cmd'], $res, $rc);
    //echo $rc;
} else {
    echo "It works!";
}

show_source(__FILE__);
?>
```

命令注入漏洞，代码将命令执行结果存储在\$res数组中，但未直接输出结果，我们将输出定向到指定文件里即可

2、cry_rsa

写出脚本就行

The screenshot shows a Windows file explorer window with a list of ZIP files:

File Name	Modified	Type
cry_rsa4.zip	2025/5/11 16:27	ZIP 文件
easy_misc8.zip	2025/5/11 14:55	ZIP 文件
ez_xor6.zip	2025/5/11 15:12	ZIP 文件
套娃9.zip	2025/5/11 15:05	ZIP 文件

The terminal window shows the following Python code and output:

```
18
19 # pq因数分解
20 p = 473398607161
21 q = 4511491
22
23 n = p*q
24 e = 19
25 # c = 287677588809406627799
26
27 on = (p-1)*(q-1)
28 d = gmpy2.invert(e, on)
29 d=d+4
30 print(d)
31 # m = pow(c, d, n)
32 # print(flag.to_bytes(m))
```

The output of the script is:

```
运行: rsa-common x
E:\reverse\python\venv\Scr
2023326077889096383
```

The terminal also shows a text file with the following content:

在一次RSA密钥对生成中,假设 $p=473398607161$, $q=4511491$, $e=19$ 求解出 d ,然后把 d 的值加4为flag值。flag格式为flag{*****}

1、baby_rsa

The screenshot shows a Windows file explorer window with a list of ZIP files:

File Name	Modified	Type
baby_rsa3.zip	2025/5/11 16:33	ZIP 文件
cry_rsa4.zip	2025/5/11 16:27	ZIP 文件
easy_misc8.zip	2025/5/11 14:55	ZIP 文件
ez_xor6.zip	2025/5/11 15:12	ZIP 文件
套娃9.zip	2025/5/11 15:05	ZIP 文件

The terminal window shows the following text:

命令注入漏洞,代码将命令执行结果存储在\$res数组中,但未直接输出结果,我们将输出定向到指定文件里即可

显示隐藏的图标

是一个 py 文件

写出脚本

```
from Crypto.Util.number import getPrime, isPrime, getRandomNBitInteger, bytes_to_long,  
long_to_bytes
```

```
import gmpy2
```

```
import sympy
```

```
q =
```

```
110428348144013242234907008083355974834266917027228724749730385104087025249352  
345946164980361082178532313669767485270254326404723948153912910688118140621712  
922649644396733499972695482991866293857864311557686710317462165131360819813493  
524457615383204504505224030129953230866877990529769205769592709254542472051
```

```
p =
```

```
110428348144013242234907008083355974834266917027228724749730385104087025249352  
345946164980361082178532313669767485270254326404723948153912910688118140621712  
922649644396733499972695482991866293857864311557686710317462165131360819813493  
524457615383204504505224030129953230866877990529769205769592709254542470619
```

```
N =
```

```
121944200738153928809890316115452968541452416753201303148213948434369473733310  
809117871767372029406768096745431388070247394544320890967945320167972464413257  
298565286640713229684288040980699971964903822861263893311790549719276553209782  
989797942453790003366357954902420275196692177844333670215782473401546477628004  
021403210226592723830875444761788020259517680154239721820454054664484315576252  
010123322397749629027500739003839933001461933004851172173197943566527295021001  
676684390079250047691180701053246643791416238162568959339592113811141727785352  
96409639317535751005960540737044457986793503218555306862743329296169569
```

```
e = 65537
```

```
c =
```

```
450481133311187720953900166551639156703810999288427108953730222630439543434311  
257440462606085496281837856085206762125392733072524498486919850555672250905809  
866008305471514667076768712058704928886106320261750726287127981921123123319807  
057453884516162980693254183220704111278633644197508735187353735020346964219899  
921986358104092750515211005131301107311572450256726152418186588387451755584816  
302624020185620762623785966560725574079040403909844445215821690775237507805461  
580261306622976634371431755047207922469479855288675910366834927068284391630765  
2213810947814618810706997339302734827571635179684652559512873381672063
```

```
on = (p-1)*(q-1)
```

```
d = gmpy2.invert(e, on)
```

```
m = pow(c, d, N)
```

```
m = long_to_bytes(m)
```

```
print(m)
```

```
运行: rsa-common x
E:\reverse\python\venv\Scripts\python.exe E:\reverse\python\crypto\rsa\rsa-common.py
b'flag{5c9c885c361541e0b261f58b61db8cec}'

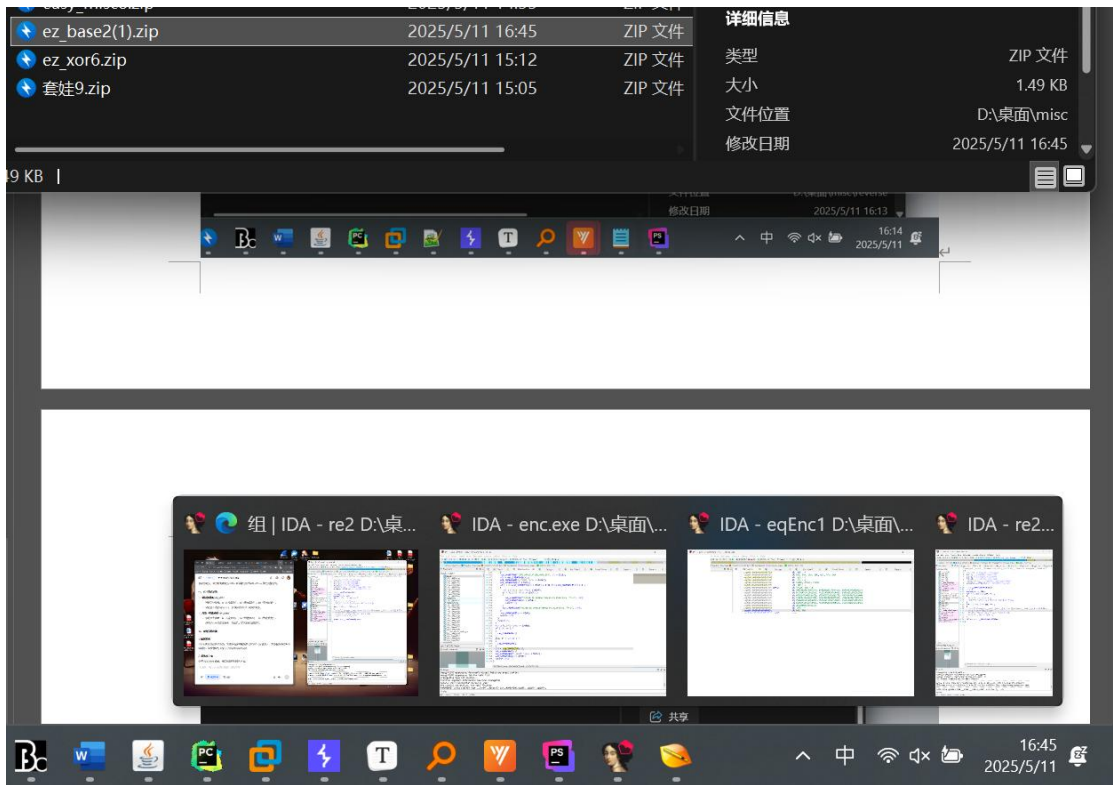
进程已结束,退出代码0
```

将 3 换成 4

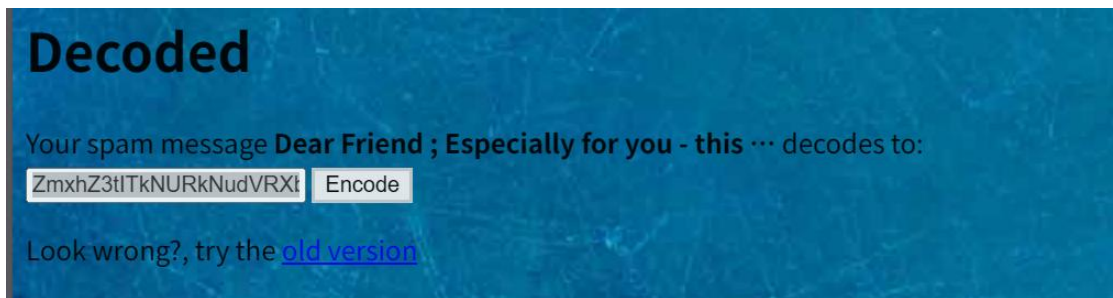
```
文件 编辑 查看

flag{5c9c885c461541e0b261f58b61db8cec}
```

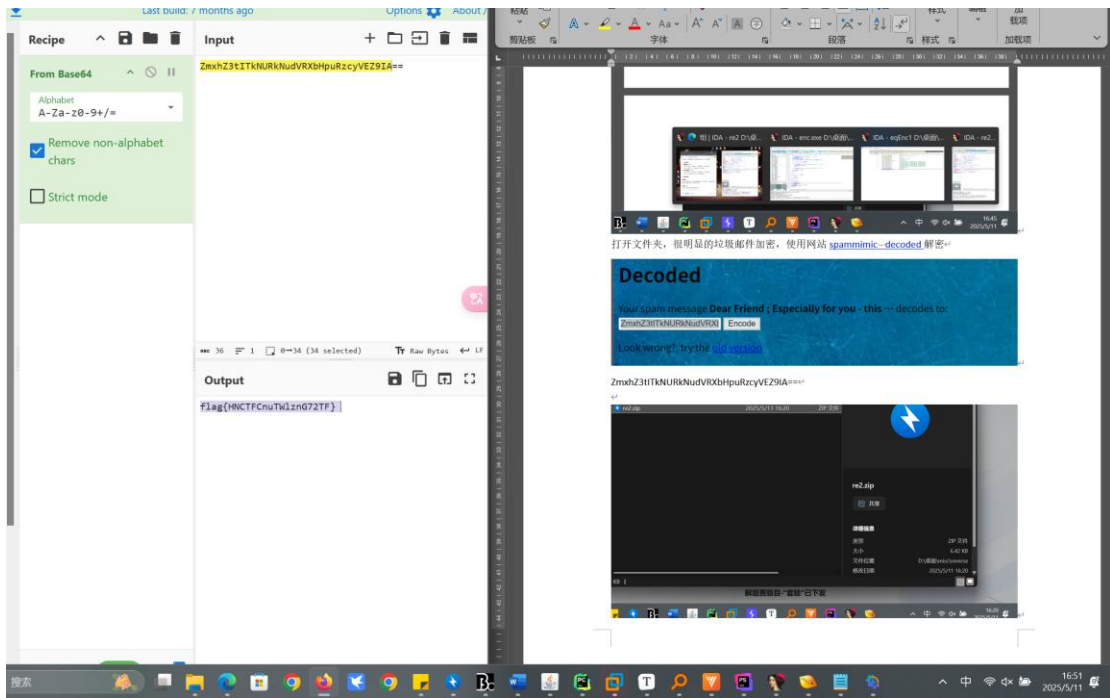
3、ez_base



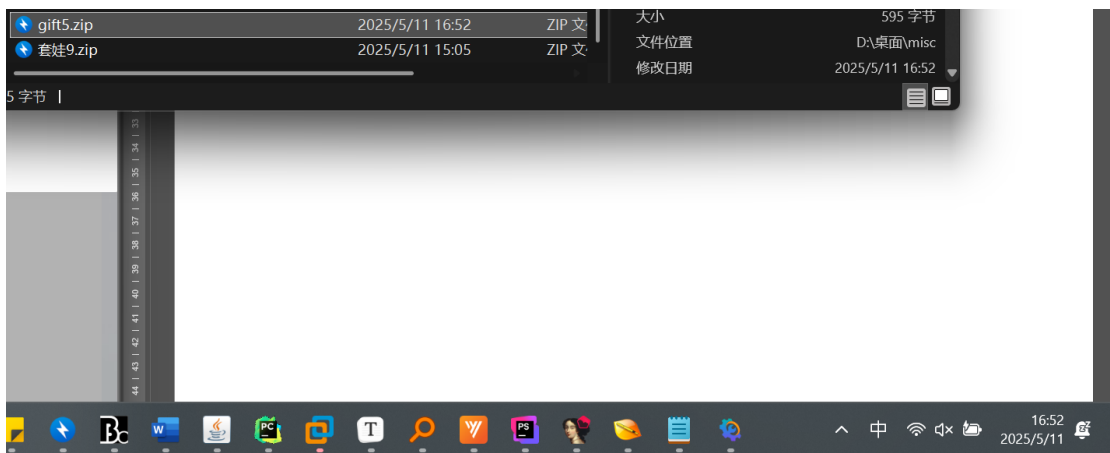
打开文件夹，很明显的垃圾邮件加密，使用网站 [spammimic - decoded](https://spammimic-decoded.com/) 解密



ZmxhZ3tITkNURkNudVRXbHpuRzcyVEZ9IA==



4、gift



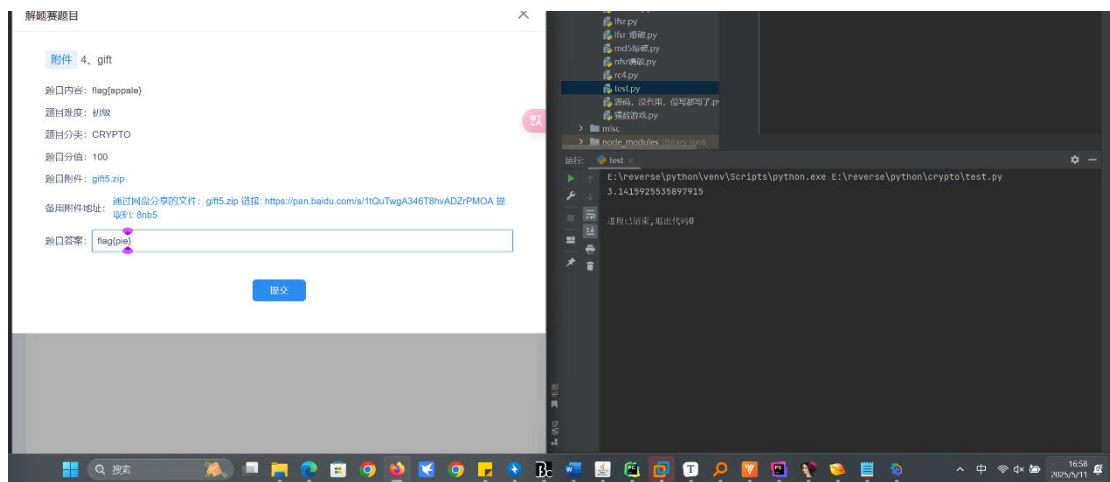
五一劳动节爸爸给家里人带了一个礼物。由于礼物不好拿，所以把礼物平均分成了四份，但是其中一份不小心掉在地上散落成了无数片，变成了 $1 - 1/3 + 1/5 - 1/7 + \dots$
 聪明的你能算出或猜出爸爸带的礼物是什么吗？ flag示例: flag{apple} flag{watermelon} 提交flag值凯撒密码加密，偏移量5在提交。

用代码逼近一下，得到的答案跟圆周率很接近

```
def gift(n=100):
    a = 0
    for i in range(n):
        a += (-1)**i * 1/(2*i + 1)

    return a * 4

print(gift(n=1000000))
```



所有礼物是 pie，凯撒密码偏移 5

```
def caesar_cipher(text, shift):
```

```
    encrypted = []
```

```
    for char in text:
```

```
        if char.isalpha():
```

```
            base = ord('a') if char.islower() else ord('A')
```

```
            encrypted.append(chr((ord(char) - base + shift) % 26 + base))
```

```
        else:
```

```
            encrypted.append(char)
```

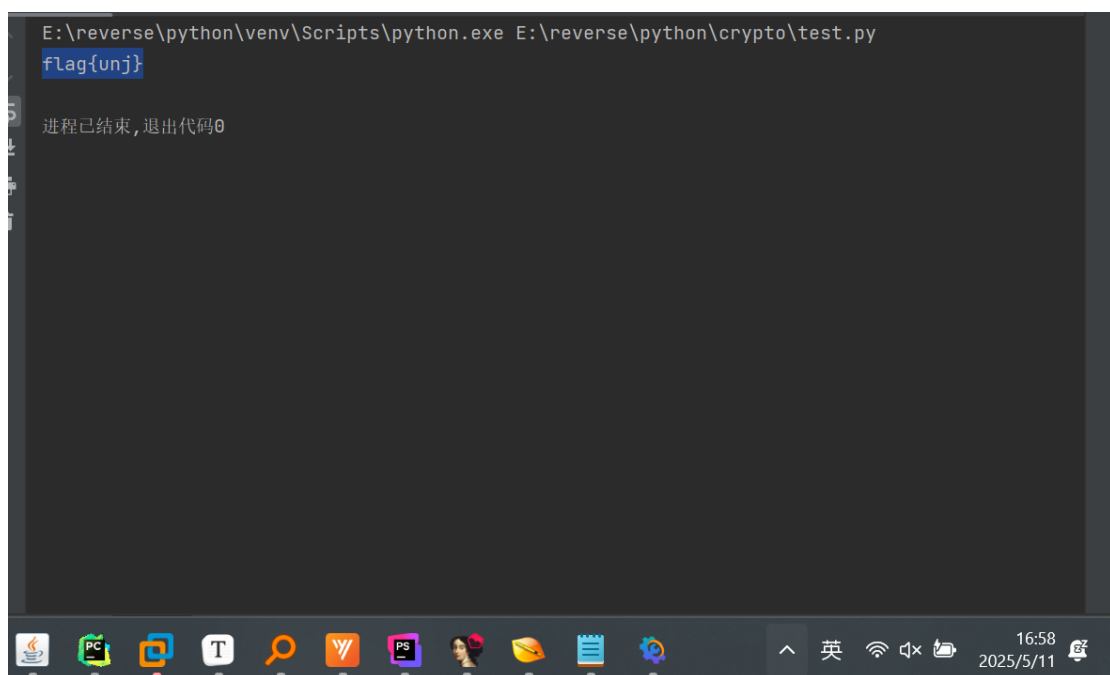
```
    return ".join(encrypted)
```

```
plaintext = "pie"
```

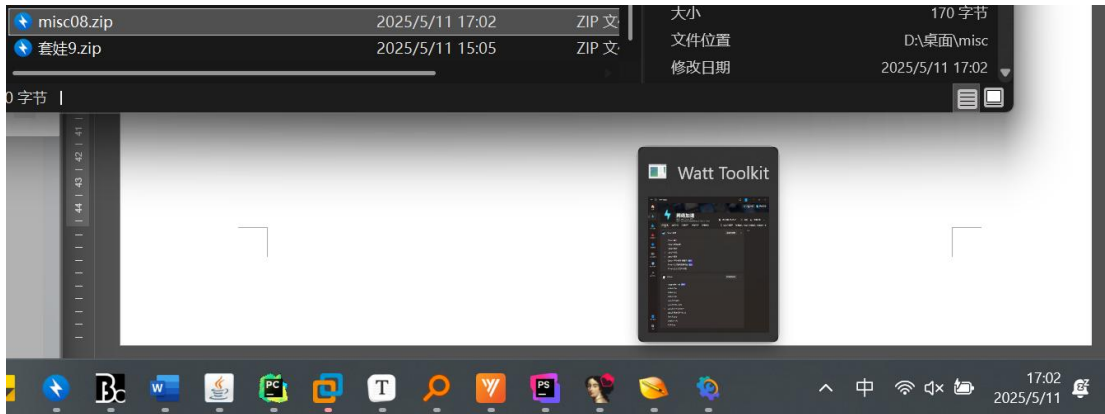
```
shift = 5
```

```
encrypted = caesar_cipher(plaintext, shift)
```

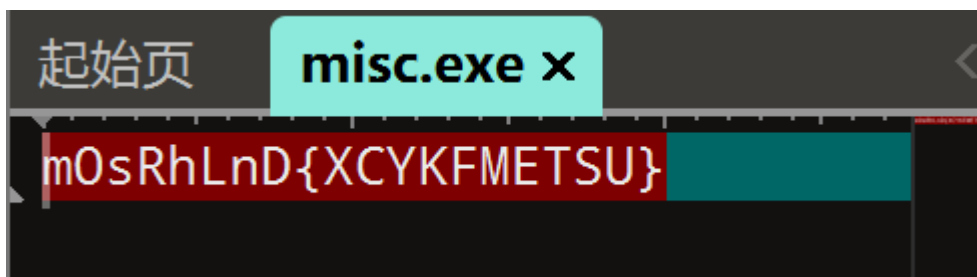
```
print(f"flag{{{encrypted}}}")
```



5、草甸方阵的密语



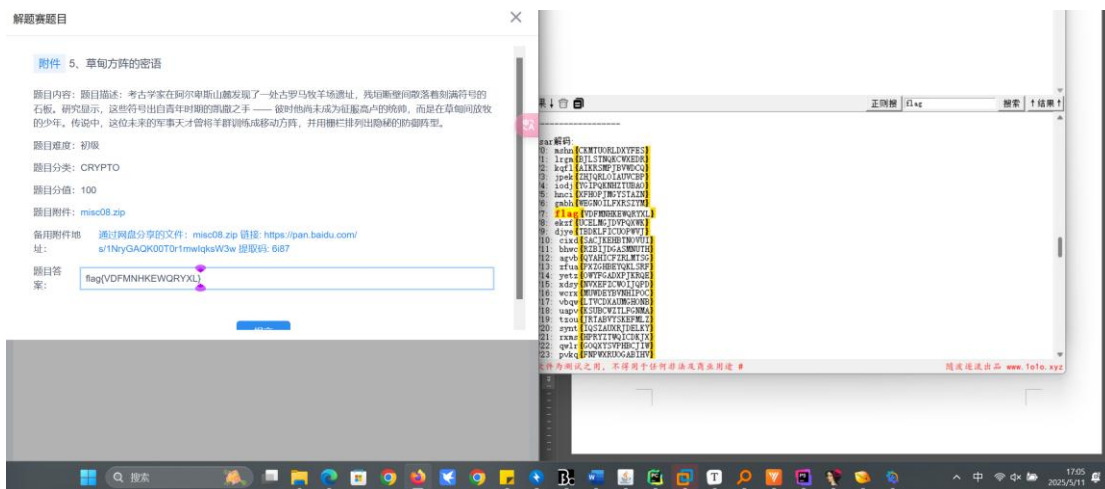
打开附件



根据题目提示为 langshan 密码，

栅栏fence解码：
因数 [2, 4, 5, 10]:
分为2栏时，解密结果为: mCOYsKRFhMLEnTDS {UX}
分为4栏时，解密结果为: mLCEOnYTsdKSR {FUhXM}
分为5栏时，解密结果为: mh {KTOLXFSsnCMURDYE}
分为10栏时，解密结果为: mshn {CKMTUORLDXYFES}

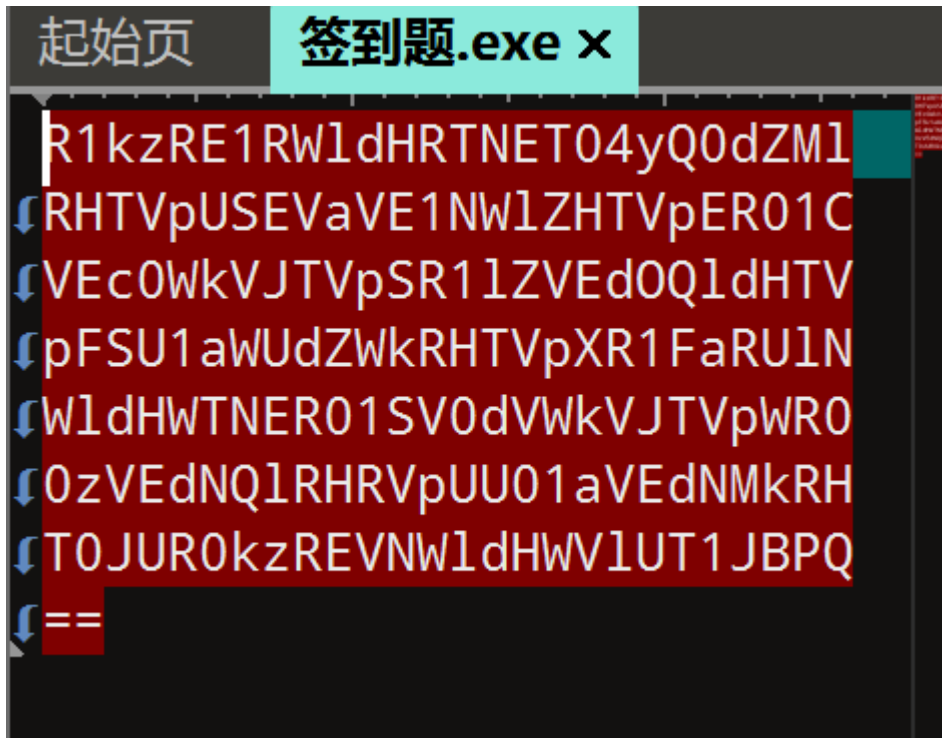
得到 10 栏时，因为大括号位置正确



6、easy-签到题



打开发现



使用 CyberChef

Recipe ^ [Save] [Folder] [Trash]

From Base64 ^ [Pause]

Alphabet
A-Za-z0-9+/=

Remove non-alphabet chars

Strict mode

From Base32 ^ [Pause]

Alphabet
A-Z2-7=

Remove non-alphabet chars

From Hex ^ [Pause]

Delimiter
None

Input + [Folder] [Copy] [Trash] [Grid]

```
R1kzRE1RWldHRTNET04yQ0dZM1RHTVpUSEVave1NW1ZHTVpER
1CVEc0WkVJTVpSR11ZVEdOQ1dHTVpFSU1aWUdZwKRHTVpXR1F
RU1NW1dHWTNER01SV0dVWkVJTVpWR00zVEdNQ1RHRVpUU01aV
dNMKRHT0JUR0kzREVNW1dHWV1UT1JBPQ==
```

Output [Save] [Copy] [Share] [Fullscreen]

```
flag{e3965207-1a4c-8b3d-6f2e-570193482b6a}
```

184 1 Raw Bytes