Bugku–flag在index里（本地文件包含）writeup

转载

xuchen16 于 2018-09-19 11:09:26 发布
分类专栏： ctf 文章标签： BugkuCTF flag在index里 writeup 本地文件包含

flag在index里

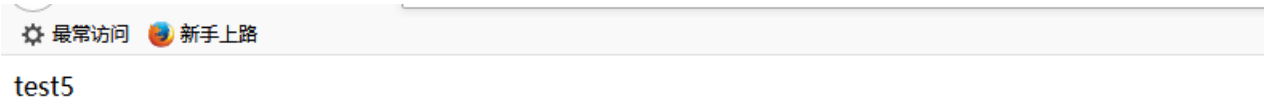
80

http://120.24.86.145:8005/post/

解题思路：

首先打开这道题，页面只给你click me? no

点击进去显示test5



第一步，查看源代码，无果

第二步bp，无果

结合到题目，flag在index里，大胆尝试http://120.24.86.145:8005/post/index.php，可惜和之前一样

注意到了传值为http://120.24.86.145:8005/post/index.php?file=show.php

file这个变量应该是关键，可惜无果

参考其他博主用到了php://filter

现在具体说说file=php://filter/read=convert.base64-encode/resource=index.php的含义

首先这是一个file关键字的get参数传递，php://是一种协议名称，php://filter/是一种访问本地文件的协议，/read=convert.base64-encode/表示读取的方式是base64编码后，resource=index.php表示目标文件为index.php。

通过传递这个参数可以得到index.php的源码，下面说说为什么，看到源码中的include函数，这个表示从外部引入php文件并执行，如果执行不成功，就返回文件的源码。

而include的内容是由用户控制的，所以通过我们传递的file参数，是include（）函数引入了index.php的base64编码格式，因为是base64编码格式，所以执行不成功，返回源码，所以我们得到了源码的base64格式，解码即可。

如果不进行base64编码传入，就会直接执行，而flag的信息在注释中，是得不到的。

我们再看一下源码中 存在对 ../ tp data input 的过滤，其实这都是php://协议中的其他方法，都可以结合文件包含漏洞执行，具体可以百度一下，或者参考我的博客。

0x00 前言

php://filter是PHP中的一个协议，利用这个协议我们可以解决一些ctf的题目，或者挖掘出一些漏洞。

0x01 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
resource=<要过滤的数据流>	指定了你要筛选过滤的数据流。	必选
read=<读链的筛选列表>	可以设定一个或多个过滤器名称，以管道符（|）分隔。	可选
write=<写链的筛选列表>	可以设定一个或多个过滤器名称，以管道符（|）分隔。	可选
<；两个链的筛选列表>	任何没有以 read= 或 write= 作前缀 的筛选器列表会视情况应用于读或写链。	

例子：

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

1.

```
<?php
/* 这会以大写字母输出 www.example.com 的全部内容 */
readfile("php://filter/read=string.toupper/resource=http://www.example.com");
```

5.

```
/* 这会和以上所做的一样，但还会用 ROT13 加密。 */
```

```
readfile("php://filter/read=string.toupper|string.rot13/resource=http://www.e
```

```
?>
```

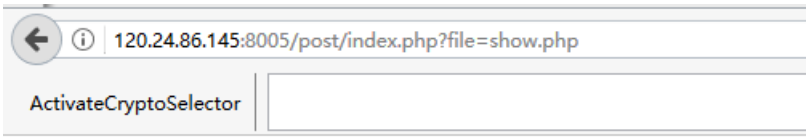
这里使用了管道符|来对多个过滤器的名称进行了分割

0x02 ctf中的应用

---

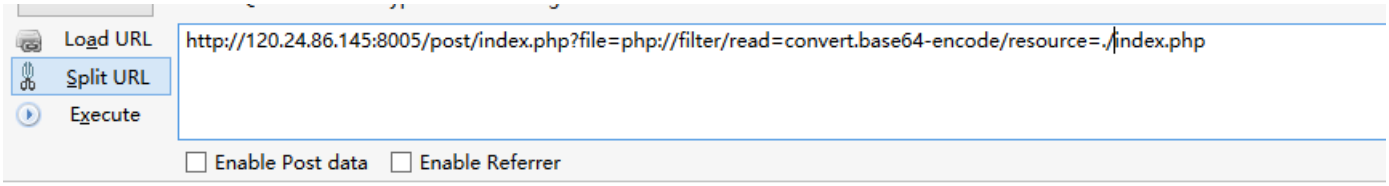
①使用base64获取源码

例：bugku ctf练习平台的 flag在index里面 题目



test5

题目提示flag在index里面，但是无法通过查看源码得到有效信息，因此尝试使用php://filter获取源码




```
PGh0bWw+DQogICAgPHRpdGxlPkJ1Z2t1LWN0ZjwvdGl0bGU+DQogICAgDQo8P3BocA0KCWVycm9yX3JlcG9yd
```


可以看到通过base64加密后的源码已经爆出
解密，排版，轻松得到flag

```
<html>

    <head>
        <title>
            Bugku-ctf
        </title>
        <!--?php error_reporting(0); if(!$_GET[file]){echo '<a href="./index.php?file=show.php"-->
    </head>

    <body>
        click me? no';} $file=$_GET['file']; if(strstr($file,&quot;../&quot;)||stristr($file,
        &quot;tp&quot;)||stristr($file,&quot;input&quot;)||stristr($file,&quot;data&quot;)){
        echo &quot;Oh no!&quot;; exit(); } include($file); //flag:flag{edulcni_elif_lacol_si_siht}?&gt;
    </body>

</html>
```


②绕过关键限制

这一部分思路来源于：leavesongs大牛 https://www.leavesongs.com/PENETRATION/php-filter-magic.html
我是勤劳的搬运工

思路一

```
<?php
```

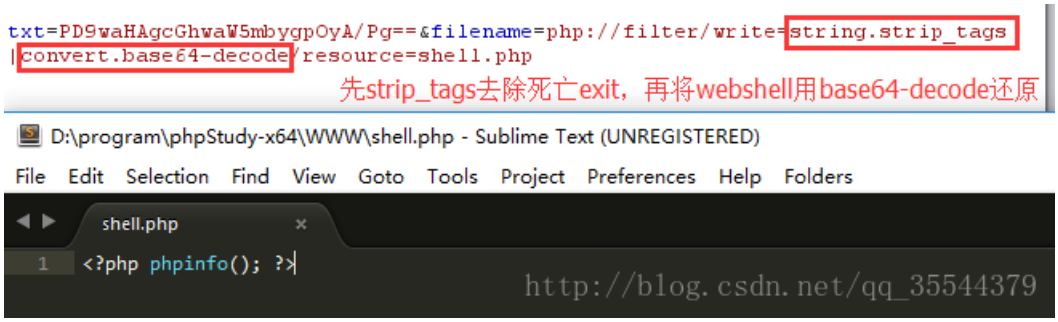
```
    $content = '<?php exit; ?>';

    $content .= $_POST['txt'];

    file_put_contents($_POST['filename'], $content);
```

这里的$content在开头增加了exit，使得我们即使可以成功写入一句话，也无法执行。
我们通过php://filter巧妙绕过这里的限制
1.对我们想要写入的一句话进行base64encode
2.php://filter/read=convert.base64-decode进行解码
那么\<\?php exit;?>这一部分在经过解码后，php不对其解析，而我们想要写入的一句话正好被成功解码。因此得以成功上传webshell。

思路二

上面的题目除了base64外，还有另一种做法。
我们可以发现我们想要去除的代码片其实是一个XML标签，因此我们可以想到一个函数去去除它：strip_tags

如果这样进行处理的话，我们想要写入的一句话也会被处理。
不过，我们的php://filter是可以使用多个过滤器的，因此我们可以先对我们想要的一句话进行编码，过滤器中先使用strip_tags函数进行处理再使用对应的解码方法即可！（如base64，rot-13



如果使用rot-13，则该方法仅限于当short_open_tag不开启的时候

转：1 https://blog.csdn.net/qq_35544379/article/details/78230629

2 http://www.cnblogs.com/lxz-1263030049/p/9315957.html

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

参考:https://www.securusglobal.com/community/2016/08/19/abusing-php-wrappers/