【Writeup】BUUCTF_Web_WarmUp

原创

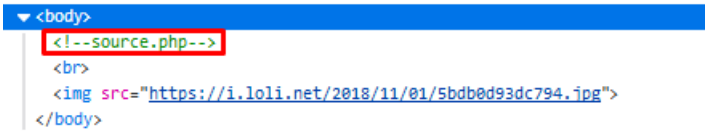
BAKUMANSEC 于 2019-08-14 22:48:56 发布 285 收藏

分类专栏： BUUCTF - Writeups 文章标签： CTF Web Writeup PHP审计

本文链接：https://blog.csdn.net/m0_38100569/article/details/98523425

0x01 解题思路

- F12查看网页源码：



;

- 直接访问获取到**source.php**源码：

```php
<?php
    highlight_file(__FILE__);
    class emmm
    {
        public static function checkFile(&$page)
        {
            $whitelist = ["source"=>"source.php","hint"=>"hint.php"];
            if (! isset($page) || !is_string($page)) {
                echo "you can't see it";
                return false;
            }

            if (in_array($page, $whitelist)) {
                return true;
            }

            $_page = mb_substr(
                $page,
                0,
                mb_strpos($page . '?', '?')
            );
            if (in_array($_page, $whitelist)) {
                return true;
            }

            $_page = urldecode($page);
            $_page = mb_substr(
                $_page,
                0,
                mb_strpos($_page . '?', '?')
            );
            if (in_array($_page, $whitelist)) {
                return true;
            }
            echo "you can't see it";
            return false;
        }
    }

    if (! empty($_REQUEST['file'])
        && is_string($_REQUEST['file'])
        && emmm::checkFile($_REQUEST['file'])
    ) {
        include $_REQUEST['file'];
        exit;
    } else {
        echo "<br><img src=\"https://i.loli.net/2018/11/01/5bdb0d93dc794.jpg\" />";
    }
?>
```

- 源码中得知**hint.php**的存在，直接访问：

flag not here, and flag in ffffllllaaaagggg

- 猜想flag在**ffffllllaaaagggg**文件中，需要通过source.php源码审计获取。

0x02 源码审计

souce.php需要用户通过任意REQUEST方式传入一个file参数，通过checkFile函数检查后传入include函数进行文件包含。

对source.php片段进行搜索，发现一个类似的phpMyadmin漏洞：CVE-2018-12613 PhpMyadmin后台文件包含分析

本题的漏洞主要在于对file参数进行白名单检查时，采用在末尾添加**?然后截取开始至第一个?**之间的子串，再检查该子串是否在白名单中的方式check。这种操作一共有两次，其中第二次操作之前会对原参数进行一次url解码。这样做的本意是想在file参数后即使跟参数也能够解析出文件名，但也因此遗留了漏洞。这样我们只需要构造`[name in whilelist]?xxxxx`格式的file参数就可以通过checkFile函数中的第一次检查返回True，构造`[name in whilelist]%253fxxxxxx`格式的file参数就可以通过checkFile函数中的第二次检查返回True（PHP会先自动进行一次url解码把%253f解码成为%3f，第二次操作中还含有一次url解码把%3f解码成为?）。这样就可以顺利地把file参数传递给include函数了，但是这样构造的file参数为什么能够成功包含呢？

PHP中的include支持三种路径：

- 绝对路径：以`/`或者盘符名如`C:`开头的路径
- 相对路径：以`./`（当前目录）或者`../`（上级目录）开头的路径
- 不确定路径：除去绝对路径和相对路径以外的路径，比如`a.php`

关于这三种路径的定义以及PHP对其解析方式的不同，以下这篇文章讲的很清楚：PHP中require和include路径问题总结

其中绝对路径和相对路径会直接检查，不经过与include_path中路径的拼接，但是不确定路径会与include_path中的路径拼接然后检查，我们传入payload就是一个不确定路径。对于这道题来说，最终payload如下：

```
source.php?file=source.php%253f/../../../../ffffllllaaaagggg
source.php?file=source.php%3f/../../../../ffffllllaaaagggg
source.php?file=source.php?/../../../../ffffllllaaaagggg
```

以上三种尝试了一下都可以。本题中的payload是不确定路径，浏览器会认为`source.php%253f/`这个部分是目录，然后加上一个../就会穿越到当前路径（include_path中的各项），后面的文件名是知道的，但是具体路径需要不断加上../尝试，最终catch flag：

?> flag{be7e3153-60f1-4d78-aa44-6d8220bb5652}