

代码注入漏洞入门



漏洞概述

PHP 代码执行漏洞指应用程序本身过滤不严格，用户可以通过请求将代码注入到程序中执行，类似于 SQL 注入漏洞，可以把 SQL 语句通过网页注入到 SQL 服务执行，而 PHP 代码执行漏洞则是可以把代码注入应用到网站后端中，如果漏洞没有特殊的过滤，相当于直接有一个 web 后门存在，该漏洞主要由动态代码执行函数的参数过滤不严格导致。

利用类型

1.eval 函数

eval()函数可以把字符串按照 PHP 代码来执行，换句话说，就是可以动态地执行 PHP 代码，使用 eval 函数需要注意的是：输入的字符串必须是合法的 PHP 代码，且必须以分号结尾。

```
[php]    
1. <?php  
2.     $myvar = "varname";  
3.     $x = $_GET['arg'];  
4.     eval("\$myvar = $x;");  
5. ?>
```

攻击者可以通过如下 Payload 实施代码注入：

```
[php]
1. /index.php?arg=1;phpinfo()
```

2. 命令注入

PHP 提供了部分函数用来执行外部应用程序，例如：`system()`、`shell_exec()`、`exec()`和 `passthru()`

```
[php]
1. <?php
2.     $command = $_REQUEST["command"];
3.     system($command);
4. ?>
```

攻击者通过以下的 payload 实施攻击

```
[plain]
1. vulnerable.php?command=ls
```

3. 动态函数调用

```
[php]
1. <?php
2.     $fun = $_GET['fun'];
3.     $par = $_GET['par'];
4.     $fun($par);
5. ?>
```

这样可以动态调用函数,但也易被黑客利用，如，
<http://www.example.com/funtion.php?fun=system>

&par=net user

最终的执行函数为 `system("net user")`

4.PHP 函数代码执行漏洞

在 PHP 中，代码执行漏洞出现较多，像 `preg_replace()`, `ob_start()`, `array_map()` 等函数都存在代码执行漏洞。

```
[php]
1. <?php
2.     $arr = $_GET['arr'];
3.     $array = array(1,2,3,4,5);
4.     $new_array = array_map($arr,$array);
5. ?>
```

攻击者可输入如下 URL：
`http://www.example.com/function.php?arr=phpinfo,`
发现 `phpinfo` 函数被执行。

防御方法

1. 尽量不要使用系统执行命令；
2. 在进入执行命令函数/方法之前，变量一定要做好过滤，对敏感字符进行转义；
3. 在使用动态函数之前，确保使用的函数是指定的函数之

一；

4.对 PHP 语言来说，不能完全控制的危险函数最好不要使用