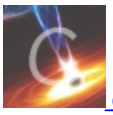
XCTF攻防世界web进阶练习_ 3_unserialize3

原创

Dar1in9 于 2019-04-30 23:35:11 发布 3389 收藏 19

分类专栏： ctf_web

本文链接：https://blog.csdn.net/silence1_/article/details/89716976



XCTF攻防世界web进阶练习—unserialize3

题目

题目是unserialize3，是反序列化的意思，应该是关于反序列化的题

打开题目，是一段残缺的php代码

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


代码定义了一个类，其中有一个魔法方法_wakeup()，_wakeup()会直接退出，并输出" bad requests " 末尾有" ?code= " 看样子是要构造url来绕过_wakeup()这个函数了。

首先_wakeup()方法会在使用函数unserialize()时自动调用。
我们先试试直接将定义的类序列化后的字符串传入code，看看是是啥反应
这里序列化操作可以自己写php或者找序列化工具
可以到这里在线写php
简单的写一段php代码，将所给的类序列化为字符串(注意：所给的代码中最后要加一个"}"才完整)

```php
<?php

class xctf{
public $flag = '111';
public function __wakeup(){
exit('bad requests');
}}
$a =new xctf();
$s = serialize($a);
echo $s;
?>
```


得到序列化后的对象字符串为`O:4:"xctf":1:{s:4:"flag";s:3:"111";}`
直接将它赋给code试试

可见，传入的code会被unserialize()反序列化，这就很简单了，我们的目的就是避免执行_weakup()函数，网上搜索一番，得到结论：

> **当序列化字符串当中属性个数值大于实际的属性个数时,就会导致反序列化异常,从而跳过__wakeup函数**

也就是说，可以将序列化后的字符串更改为

```
O:5:"xctf":5:{s:4:"flag";s:3:"111";}
```

即可绕过执行_weakup函数，当然把5改为比1大的任何数都行
按上面所说，构造url为
`?code=O:4:"xctf":5:{s:4:"flag";s:3:"111";}`
回车，得到flag！！

关于反序列化

序列化与反序列化

维基百科：

> 序列化（serialization）在计算机科学的数据处理中，是指将数据结构或对象状态转换成可取用格式（例如存成文件，存于缓冲，或经由网络中发送），以留待后续在相同或另一台计算机环境中，能恢复原先状态的过程。

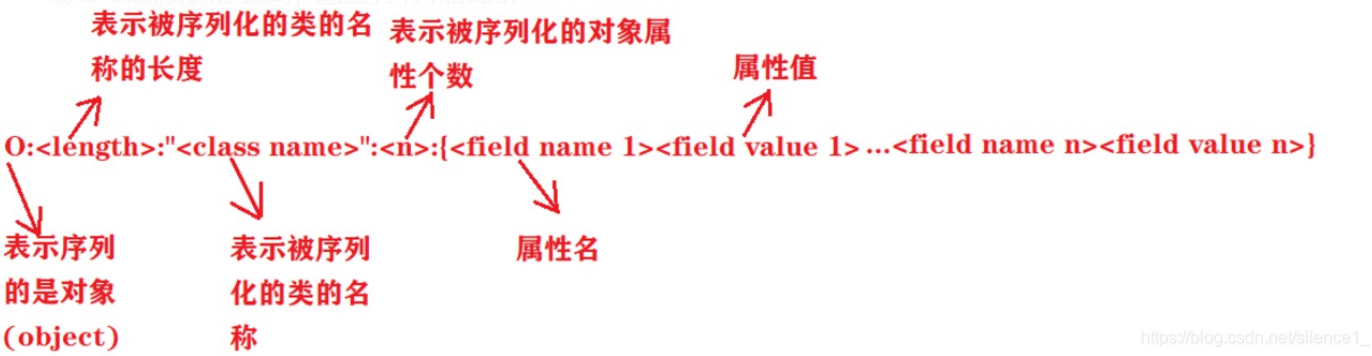
简单的说，序列化是将变量转换为可保存或可传输的字符串的过程。
而反序列化是在适当的时候吧这个字符串再转化成原来的变量使用。

php的序列化和反序列化

php的序列化和反序列化由serialize()和unserialize()这两个函数来完成
serialize()完成序列化的操作，将传入的值转换为序列化后的字符串
而unserialize()完成反序列化的操作，将字符串转换成原来的变量

serialize()函数将一个对象转换成字符串时，其返回的字符串有一定规则：

```
O:<length>:"<class name>":<n>:{<field name 1><field value 1>...<field name n><field value n>}
```



比如：`O:4:"xctf":1:{s:4:"flag";s:3:"111";}` 表示序列化的是一个对象，对象所在类名是"xctf"、该对象有一个属性，属性名为一个长度为4的字符串"flag"、该属性值为一个长度为3的字符串"111"

注意：
① 当属性为private属性时，它会在两侧加入空字节，导致其长度会增加2
② 序列化对象时只会序列化对象中的属性值，不会序列化其中的函数

魔术方法

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

与序列化和反序列化的魔术方法主要是：

```
__construct() //当一个对象创建时被调用
__destruct()  //对象被销毁时触发
__wakeup()  //使用unserialize时触发
__sleep()  //使用serialize时触发
__toString()  //把类当做字符串时触发
__get()  //用于从不可访问的属性读取数据
__set()  //用于将数据写入不可访问的属性
```

PHP反序列化漏洞

php反序列化漏洞又称对象注入，可能会导致注入，远程代码执行等安全问题的发生

php反序列化漏洞如何产生：

如果一个php代码中使用了unserialize函数去调用某一类，该类中会自动执行一些自定义的魔法方法，这些魔法方法中如果包含了某一些危险的操作，或者这些魔法方法回去调用类中带有危险操作的函数，如果这些危险操作时我们可控的，那么就可以进行一些不可描述的操作了