**php反序列化漏洞原理及利用浅析**

# 0x00实验环境

08\_r2\_dat\_zh-chs

phpstudy\_8.1

vscode

firefox

简单配置后，开始我们的动手实验

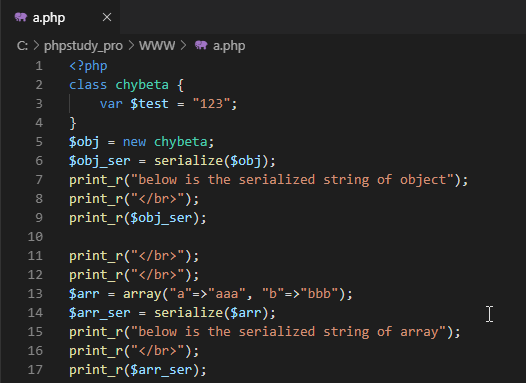
# 0x01序列化与反序列化原理

php反序列化漏洞，又叫php对象注入漏洞。

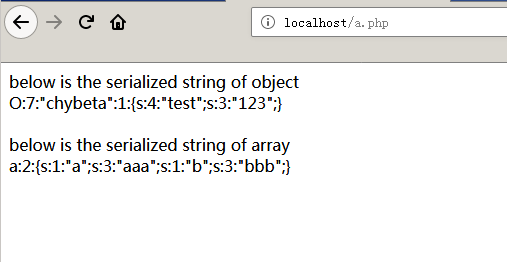
php中序列化与反序列化有两个函数serialize() 和unserialize()。

## serialize()

当在php中创建一个对象或数组后，可以通过serialize()把这个对象或数组转变成一个字符串，保存序列化后的值，方便之后的传递与使用。测试代码如下



将其序列化后的结果打印出来

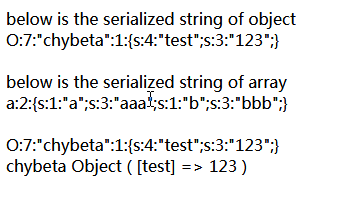


这里的O代表存储的是对象（object）,假如你给serialize()传入的是一个数组，那它会变成字母a。7表示对象的名称有7个字符。"chybeta"表示对象的名称。1表示有一个值。{s:4:"test";s:3:"123";}中，s表示字符串，4表示该字符串的长度，"test"为字符串的名称，之后的类似。

## unserialize()

与 serialize() 对应的，unserialize()可以从已存储的表示中创建PHP的值，单就本次所关心的环境而言，可以从序列化后的结果中恢复对象（object），测试代码如下





这里提醒一下，当使用 unserialize() 恢复对象时， 将调用 \_\_wakeup() 成员函数。

# 反序列化漏洞

由前面可以看出，当传给 unserialize() 的参数可控时，我们可以通过传入一个精心构造的序列化字符串，从而控制对象内部的变量甚至是函数。

## 利用构造函数等Magic function

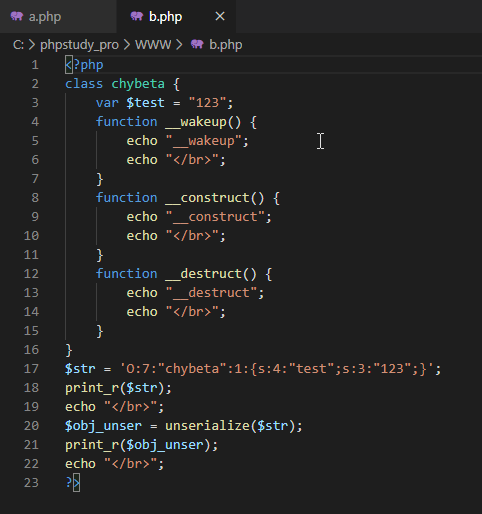
php中有一类特殊的方法叫“Magic function”， 这里我们着重关注一下几个：

构造函数\_\_construct()：当对象创建(new)时会自动调用。但在unserialize()时是不会自动调用的。

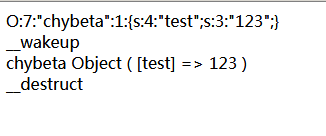
析构函数\_\_destruct()：当对象被销毁时会自动调用。

\_\_wakeup() ：如前所提，unserialize()时会自动调用。

测试代码如下



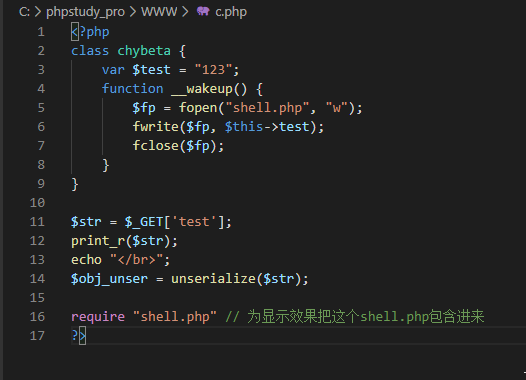
输出如下



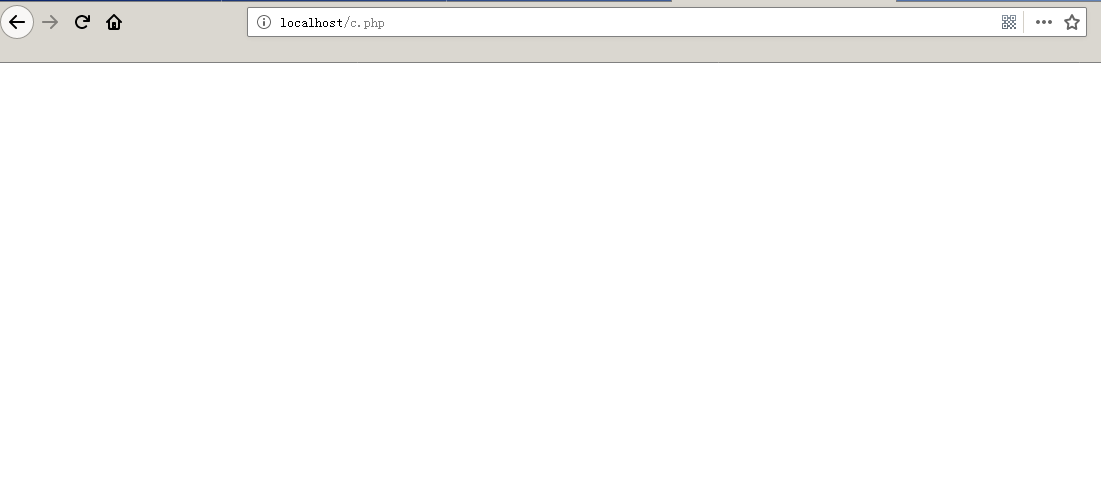
## 利用场景

\_\_wakeup() 或\_\_destruct()

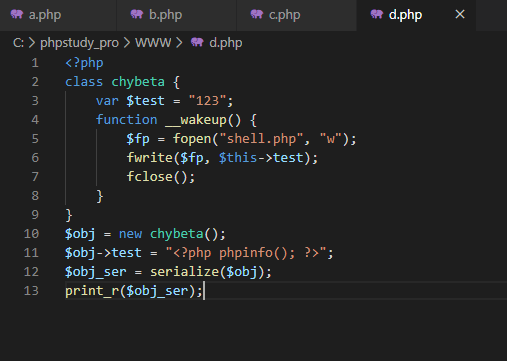
由前可以看到，unserialize()后会导致\_\_wakeup() 或\_\_destruct()的直接调用，中间无需其他过程。因此最理想的情况就是一些漏洞/危害代码在\_\_wakeup() 或\_\_destruct()中，从而当我们控制序列化字符串时可以去直接触发它们。这里针对 \_\_wakeup() 场景做个实验。假设源码如下：



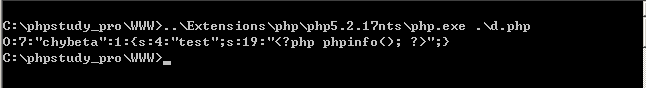
同目录下有个空的shell.php文件。一开始访问c.php。



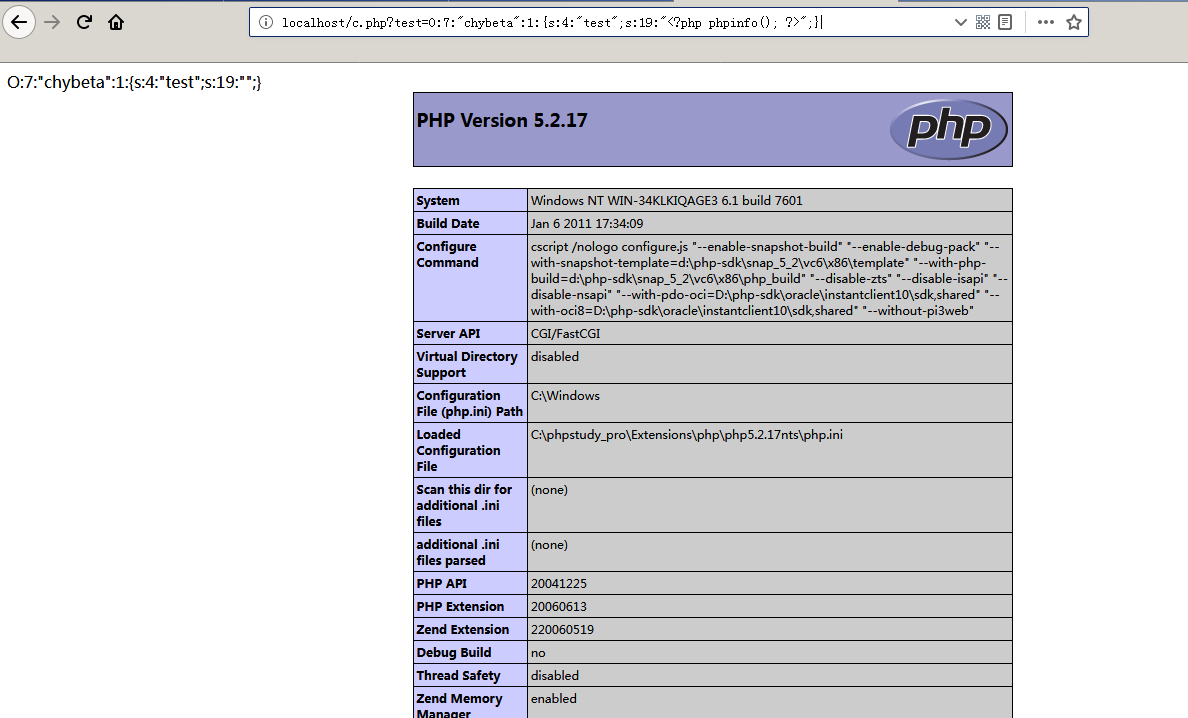
基本的思路是，本地搭建好环境，通过 serialize() 得到我们要的序列化字符串，之后再传进去。通过源代码知，把对象中的test值赋为 “<?php phpinfo(); ?>”,再调用unserialize()时会通过\_\_wakeup()把test的写入到shell.php中。为此我们写个php脚本：



由此得到序列化结果：

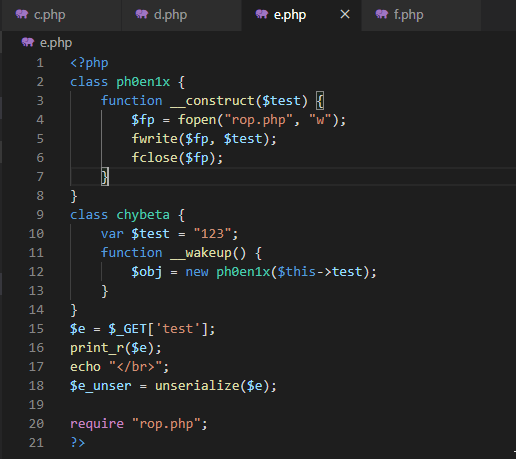


这里想执行phoinfo();会有一个坑，参见附录1

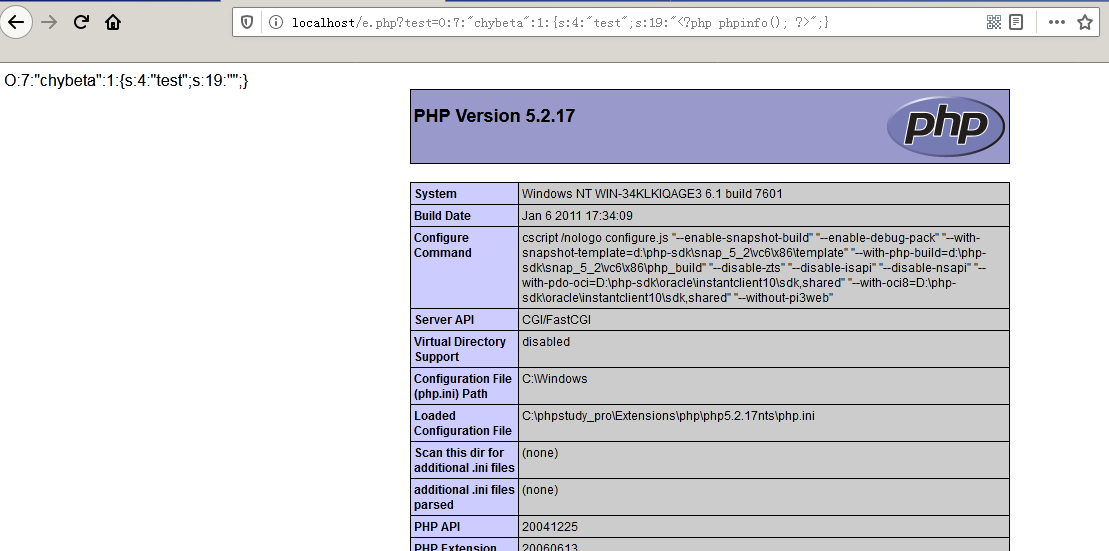


## 其他Magic function的利用

但如果一次unserialize()中并不会直接调用的魔术函数，比如前面提到的\_\_construct()，是不是就没有利用价值呢？非也。类似于PWN中的ROP，有时候反序列化一个对象时，由它调用的\_\_wakeup()中又去调用了其他的对象，由此可以溯源而上，利用一次次的“gadget”找到漏洞点。

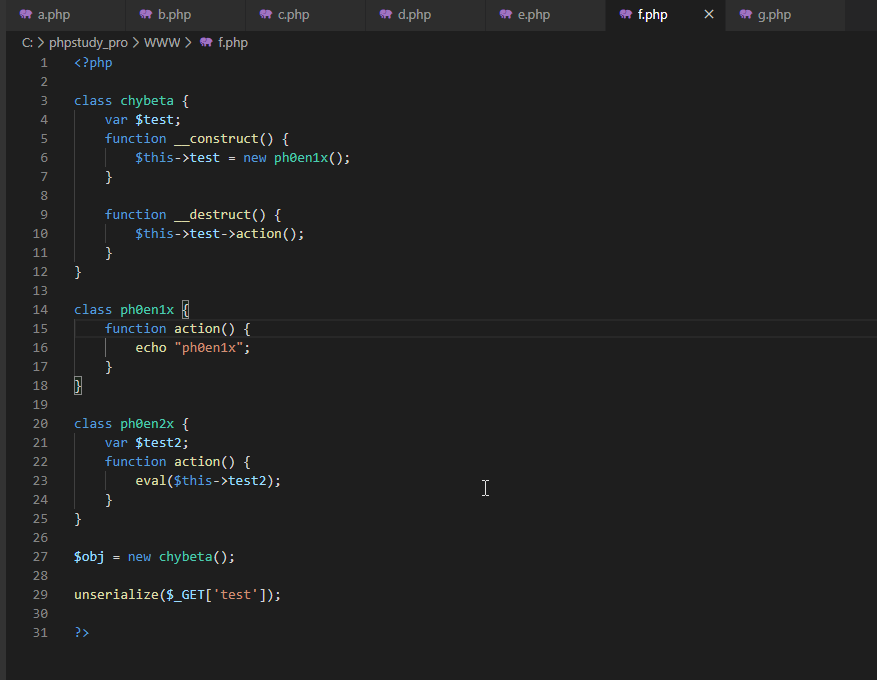


这里我们给test传入构造好的序列化字符串后，进行反序列化时自动调用 \_\_wakeup()函数，从而在new ph0en1x()会自动调用对象ph0en1x中的\_\_construct()方法，从而把<?php phpinfo() ?>写入到 shell.php中。



## 利用普通成员方法

前面谈到的利用都是基于“自动调用”的magic function。但当漏洞/危险代码存在类的普通方法中，就不能指望通过“自动调用”来达到目的了。这时的利用方法如下，寻找相同的函数名，把敏感函数和类联系在一起。

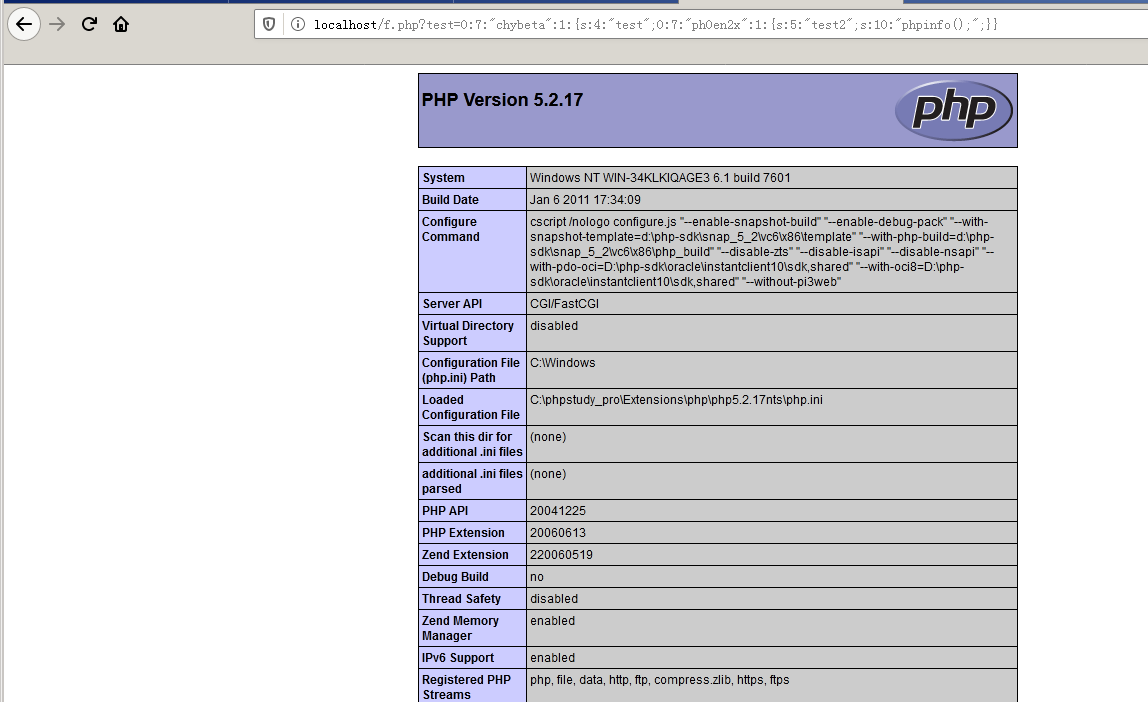


本意上，new一个新的chybeta对象后，调用\_\_construct()，其中又new了ph0en1x对象。在结束后会调用\_\_destruct()，其中会调用action()，从而输出 ph0en1x。

下面是利用过程。构造序列化。



得到：



# 附录1

1、浏览器访问解析后的php代码，不能显示<?php phpinfo(); ?>，进一步测试，不能显示<?、<php、等，所以想显示上述输出，需要在控制台下显示输出

2、通过get、post、cookie传入的数据，当包含【'】【"】【\】【NULL】这4个字符时，会被自动加入反斜线转义

应对方式stripslashes()、magic\_quotes\_gpc = Off

# 参考链接：

https://chybeta.github.io/2017/06/17/%E6%B5%85%E8%B0%88php%E5%8F%8D%E5%BA%8F%E5%88%97%E5%8C%96%E6%BC%8F%E6%B4%9E/