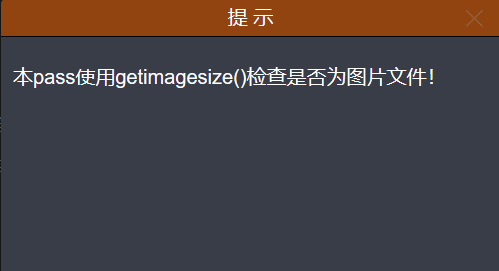
Pass14

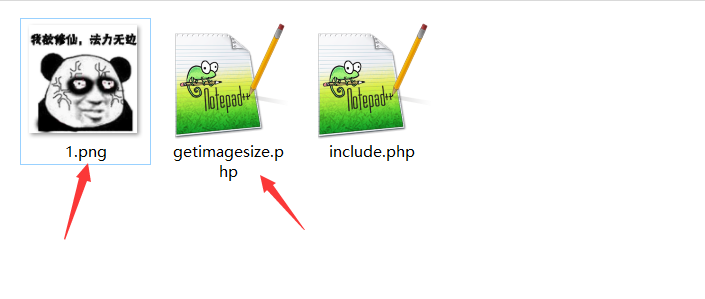
打开第十四关，发现任务要求跟上一关是一样的。都是要求上传图片马。所以再看看提示吧。



没有说要检查文件前两个字节了，但是用了一个新的函数来检查是否为图片文件。我们先了解一下这个getimagesize()函数是干嘛的吧。

getimagesize() 函数将测定任何 GIF，JPG，PNG，SWF，SWC，PSD，TIFF，BMP，IFF，JP2，JPX，JB2，JPC，XBM 或 WBMP 图像文件的大小并返回图像的尺寸以及文件类型及图片高度与宽度。函数成功返回的就是一个数组，失败则返回 FALSE 并产生一条 E\_WARNING 级的错误信息。

既然是这样的话，我们可以在本地新建一个php文件来测试该函数返回数组的各个值的含义。



先放入一个正常图片，再创建一个名为getimagesize的php文件，代码如下：

<?php

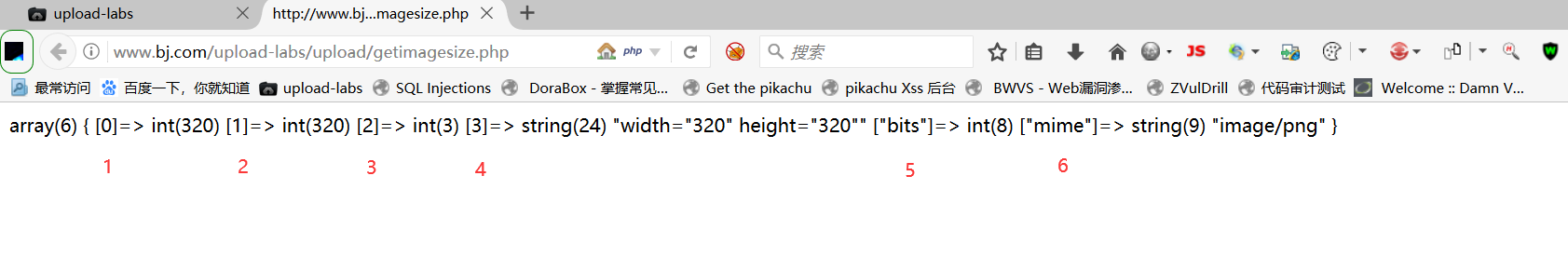
$local\_png ='./1.png';

$img\_data = getimagesize($local\_png);

var\_dump($img\_data );

?>

意思就是用getimagesize()函数对本地图片进行检查，然后将检查结果的数组打印出来。我们运行看看



可以看到数组一共有6个值。通过查阅资料了解到这6个值分别表示的含义如下：

1、索引 0 给出的是图像宽度的像素值

2、索引 1 给出的是图像高度的像素值

3、索引 2 给出的是图像的类型，返回的是数字，其中1 = GIF，2 = JPG，3 = PNG，4 = SWF，5 = PSD，6 = BMP，7 = TIFF(intel byte order)，8 = TIFF(motorola byte order)，9 = JPC，10 = JP2，11 = JPX，12 = JB2，13 = SWC，14 = IFF，15 = WBMP，16 = XBM

4、索引 3 给出的是一个宽度和高度的字符串，可以直接用于 HTML 的 <image> 标签

5、索引 bits 给出的是图像的每种颜色的位数，二进制格式

6、索引 channels 给出的是图像的通道值，RGB 图像默认是 3

7、索引 mime 给出的是图像的 MIME 信息，此信息可以用来在 HTTP Content-type 头信息中发送正确的信息，如： header("Content-type: image/jpeg");

在本次测试中并未出现索引channels，此处只做了解。

接下来我们再看看源码是对上传文件进行过滤的。

function isImage($filename){

$types = '.jpeg|.png|.gif';

if(file\_exists($filename)){

$info = getimagesize($filename);

$ext = image\_type\_to\_extension($info[2]);

if(stripos($types,$ext)>=0){

return $ext;

}else{

return false;

}

}else{

return false;

}}

$is\_upload = false;$msg = null;if(isset($\_POST['submit'])){

$temp\_file = $\_FILES['upload\_file']['tmp\_name'];

$res = isImage($temp\_file);

if(!$res){

$msg = "文件未知，上传失败！";

}else{

$img\_path = UPLOAD\_PATH."/".rand(10, 99).date("YmdHis").$res;

if(move\_uploaded\_file($temp\_file,$img\_path)){

$is\_upload = true;

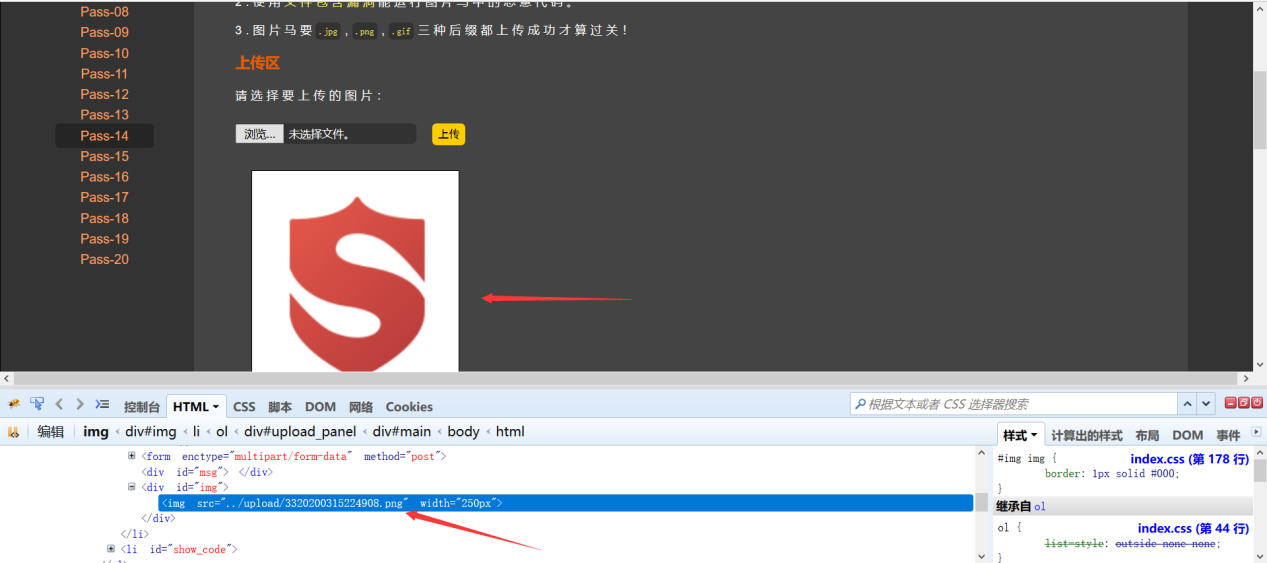
} else {

$msg = "上传出错！";

}

}}

从源码我们不难看出，服务器确实只是通过该函数索引2的值来判断文件是否为图片文件。后面并未做更多的过滤操作，所以上一关的图片马依然是有效的。个人估计函数检查文件类型也就是通过文件的头部来判断的。让我们测试一下之前的图片马是否能够直接使用。



上传是可以成功的，看来该函数对于图片马来说并没有起到任何的过滤防护作用。所以通过跟前一关同样的文件包含漏洞可以直接getshell。

Pass15

打开第十五关，先看看提示。



这一次换了个函数来做检查了。那么这个exif\_imagetype()函数又有哪些功能，又是如何检查文件的呐？

exif\_imagetype() 读取一个图像的第一个字节并检查其签名。如果发现了恰当的签名则返回一个对应的常量，否则返回 FALSE。返回值和 getimagesize() 返回的数组中的索引 2 的值是一样的，但本函数快得多。

这个函数的对应图像类型常量都是事先定义好的。如下：



那么它的语法是：

exif\_imagetype ( string $filename ) : int

$filename是文件位置和名称，如果检查是图片文件则返回一个跟图像类型对应的数字，否则返回 FALSE。

同样的我们在本地新建一个exif\_imagetype.php的文件来测试一下。代码如下：

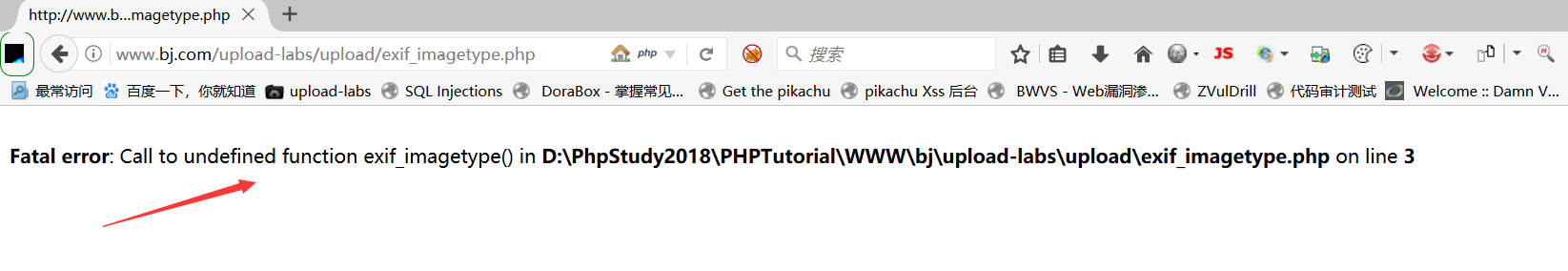
<?php

$name=exif\_imagetype("./1.png") ;

echo $name;

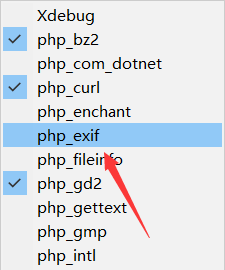
?>

但是当我访问是却出现了函数未定义这样的错误



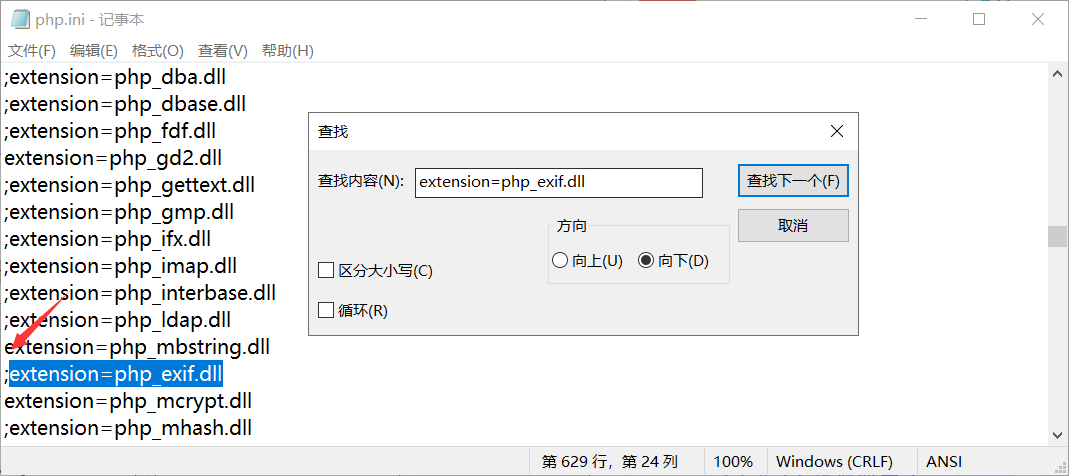
这里不得不提一下该函数的使用环境了：PHP 4 >= 4.3.0, PHP 5, PHP 7

检查一番之后发现phpstudy里面该扩展默认没有开启

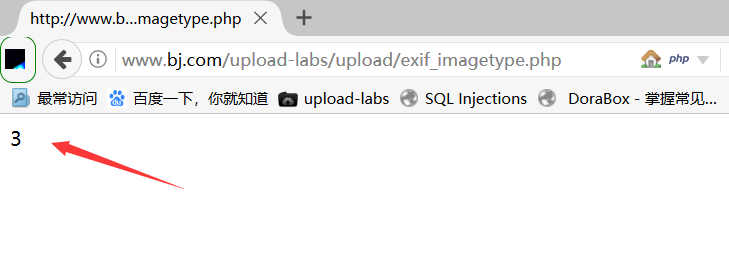


但是开启之后发现依然不能使用这个函数，百度一番之后才知道要在php.ini中修改配置。

在php.ini中搜索extension=php\_exif.dll，然后将前面的分号去掉



保存重启之后就能正常使用了。



既然是检查签名的话，跟getimagesize()函数就没什么区别了。只不过是返回值少一些，更快一些而已。之前的图片马依然是起效的。