漏洞演练平台之文件上传篇

一、 文件上传漏洞与 webshell

文件上传漏洞是指网络攻击者上传了一个可执行的文件到服务器并执行。这里上传的文件可以是木马，病毒，恶意脚本或者 WebShell 等。这种攻击方式是最为直接和有效的，部分文件上传漏洞的利用技术门槛非常的低，对于攻击者来说很容易实施。

文件上传漏洞本身就是一个危害巨大的漏洞，WebShell 更是将这种漏洞的利用无限扩大。大多数的上传漏洞被利用后攻击者都会留下 WebShell 以方便后续进入系统。攻击者在受影响系统放置或者插入 WebShell 后，可通过该 WebShell 更轻松，更隐蔽的在服务中为所欲为。这里需要特别说明的是上传漏洞的利用经常会使用 WebShell，而 WebShell 的植入远不止文件上传这一种方式。

1． webshell 简介

webshell 就是以 asp、php、jsp 或者 cgi 等网页文件形式存在的一种命令执行环境，也可以将其称之为一种网页后门。攻击者在入侵了一个网站后，通常会将这些 asp 或 php 后门文件与网站服务器 web 目录下正常的网页文件混在一起，然后使用浏览器来访问这些后门，得到一个命令执行环境，以达到控制网站服务器的目的（可以上传下载或者修改文件，操作数据库，执行任意命令等）。

webshell 后门隐蔽较性高，可以轻松穿越防火墙，访问 webshell 时不会留下系统日志，只会在网站的 web 日志中留下一些数据提交记录，没有经验的管理员不容易发现入侵痕迹。攻击者可以将 webshell 隐藏在正常文件中并修改文件时间增强隐蔽性，也可以采用一些函数对 webshell 进行编码或者拼接以规避检测。

2 .文件上传漏洞原理

大部分的网站和应用系统都有上传功能，如用户头像上传，图片上传，文档上传等。一些文件上传功能实现代码没有严格限制用户上传的文件后缀以及文件类型，导致允许攻击者向某个可通过 Web 访问的目录上传任意 PHP 文件，并能够将这些文件传递给 PHP 解释器，就可以在远程服务器上执行任意 PHP 脚本。

当系统存在文件上传漏洞时攻击者可以将病毒，木马，webshell，其他恶意脚本或者是包含了脚本的图片上传到服务器，这些文件将对攻击者后续攻击提供便利。根据具体漏洞的差异，这里上传的脚本可以是正常后缀的 PHP，ASP 以及 JSP 脚本，也可以是篡改后缀后的这几类脚本。

上传文件是病毒或者木马时，主要用于诱骗用户或者管理员下载执行或者直接自动运行；

上传文件是 webshell 时，攻击者可通过这些网页后门执行命令并控制服务器；

上传文件是其他恶意脚本时，攻击者可直接执行脚本进行攻击；

上传文件是恶意图片时，图片中可能包含了脚本，加载或者点击这些图片时脚本会悄无声息的执行；

上传文件是伪装成正常后缀的恶意脚本时，攻击者可借助本地文件包含漏洞(Local File Include)执行该文件。（LFI 详见 file_inclusion.pdf）

二、 文件上传漏洞防御

1. 文件上传的目录设置为不可执行

只要 web 容器无法解析该目录下面的文件，即使攻击者上传了脚本文件，服务器本身也不会受到影响，因此这一点至关重要。

2. 判断文件类型

在判断文件类型时，可以结合使用 MIME Type、后缀检查等方式。在文件类型检查中，强烈推荐白名单方式，黑名单的方式已经无数次被证明是不可靠的。此外，对于图片的处理，可以使用压缩函数或者 resize 函数，在处理图片的同时破坏图片中可能包含的 HTML 代码。

3、使用随机数改写文件名和文件路径

文件上传如果要执行代码，则需要用户能够访问到这个文件。在某些环境中，用户能上传，但不能访问。如果应用了随机数改写了文件名和路径，将极大地增加攻击的成本。再来就是像 shell.php.rar.rar 和 crossdomain.xml 这种文件，都将因为重命名而无法攻击。

4、单独设置文件服务器的域名

由于浏览器同源策略的关系，一系列客户端攻击将失效，比如上传 crossdomain.xml、上传包含 Javascript 的 XSS 利用等问题将得到解决。

三、 设计目标

以上传图片功能为例，分别实现文件头 content-type 字段校验，服务端检测文件头，文件扩展名检测，JS 验证四种常见文件上传验证方式，让使用者能去了解并且利用文件上传漏洞，通过了解文件上传的原理，掌握文件上传的验证与绕过技巧。

四、 作品演练

1. /file_upload/1.html

该关卡并未设置文件上传过滤，关键代码如下：

```php
$filename = $_FILES['file']['name'];
$tmp_name = $_FILES['file']['tmp_name'];
$error = $_FILES['file']['error'];
$des_addr = "./upload/".$filename;
move_uploaded_file($tmp_name,$des_addr);
if (error == 0 ){
    echo "上传成功。";
}
```

选择上传文件:logo.jpg

可使用抓包工具查看所上传文件的文件头 content-type 字段，如图为 logo.jpg 的 content_type 字段：

```
Content-Disposition: form-data; name="file"; filename="logo.jpg"
Content-Type: image/jpeg
```

2. /file_upload/2.html

该关卡设置文件头 content-type 字段校验，校验代码如下：

```
if($type != "image/png" && $type != "image/gif" && $type != "image/bmp"
    && $type != "image/jpeg" && $type != "image/pjpeg" && $type != "image/x-png"){
    echo "请选择正确类型的文件! "."<br />";
    exit;
}
```

绕过方法：可以通过抓包工具抓包，将 content-type 字段改为允许上传的类型。

例如上传内容为<?php phpinfo(); ?> 的 phpshell.php，该文件文件头 content-type 字段为 application/x-php ，如图：

```
Content-Disposition: form-data; name="file"; filename="phpshell.php
"
Content-Type: application/x-php

<?php
phpinfo();
?>
```

自然无法成功上传。我们可以通过抓包工具抓包，修改文件头 content-type 字段为 image/gif：

```
Content-Disposition: form-data; name="file"; filename="phpshell.php
"
Content-Type: image/gif

<?php
phpinfo();
?>
```

进行文件类型绕过，从而成功上传。

3. /file_upload/3.html

该关卡设置文件头校验，PHP 使用 getimagesize 函数验证图片文件头。校验部分源码如下：

```
$imageinfo = getimagesize($tmp_name);
if($imageinfo["mime"] !="image/jpeg" && $imageinfo["mime"] != "image/png"
    && $imageinfo['mime'] != "image/gif"){
    echo "请选择正确类型的文件! "."<br />";
    exit;
}
```

文件头简介：不同的图片文件都有不同文件头，如：

PNG： 文件头标识 (8 bytes) 89 50 4E 47 0D 0A 1A 0A

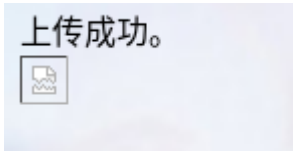
JPEG： 文件头标识 (2 bytes): 0xff， 0xd8 (SOI) (JPEG 文件标识)

GIF： 文件头标识 (6 bytes) 47 49 46 38 39(37) 61

绕过方式：在所需上传文件前加上允许上传文件的头标识。

测试文件为：test3.php（向某图片添加注释<?php phpinfo(); ?>,保存图片后将其扩展名改为 php)如下图：

选择上传，可访问该文件。



4. /file_upload/4.html

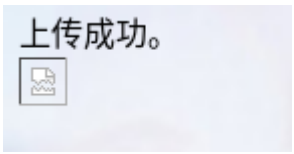
该关卡设置文件扩展名检测，设置黑名单检测文件扩展名，源码如下：

```
$file_ext = substr(strrchr($filename, '.'), 1);
$type = array("php","php3");
if(in_array($file_ext,$type)){
    echo "请选择正确类型的文件！"."<br/>";
    exit();
}
```

绕过方法：当黑名单不全，构造特殊文件名可以绕过扩展名验证。

例如：构造 phpshell.PHP：

上传：



5. /file_upload/5.html

设置客户端校验，一般都是在网页上写一段 javascript 脚本，校验上传文件的后缀名：

```
var allow_ext = ".jpg|.jpeg|.png|.gif|.bmp|";
var ext_name = file.substring(file.lastIndexOf("."));
if (allow_ext.indexOf(ext_name + "|") == -1) {
    var errMsg = "该文件不允许上传，请上传" + allow_ext
        + "类型的文件，当前文件类型为：" +     ext_name;
    alert(errMsg);
    return false;
}
```

判断方式：在浏览加载文件，但还未点击上传按钮时便弹出对话框，内容如：只允许上传.jpg/.jpeg/.png 后缀名的文件，而此时并没有发送数据包。

绕过方法：

1、通过 firefox 的 F12 修改 js 代码绕过验证

2、使用 burp 抓包直接提交，绕过 js 验证

例如，选择 php 文件上传：

修改允许上传的文件类型为：

```
//定义允许上传的文件类型
var allow_ext = ".jpg|.jpeg|.png|.gif|.bmp|.php|";
```

则可成功上传。