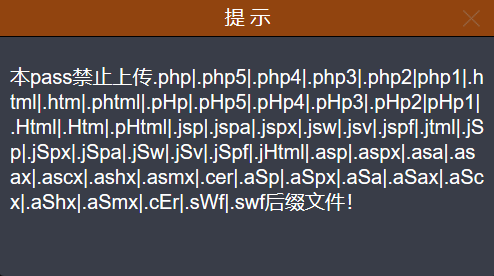
Pass06

打开第六关先查看提示



可以看到依然是黑名单禁止了很多的文件后缀，接下来再看看源码吧

$is\_upload = false;$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';

}

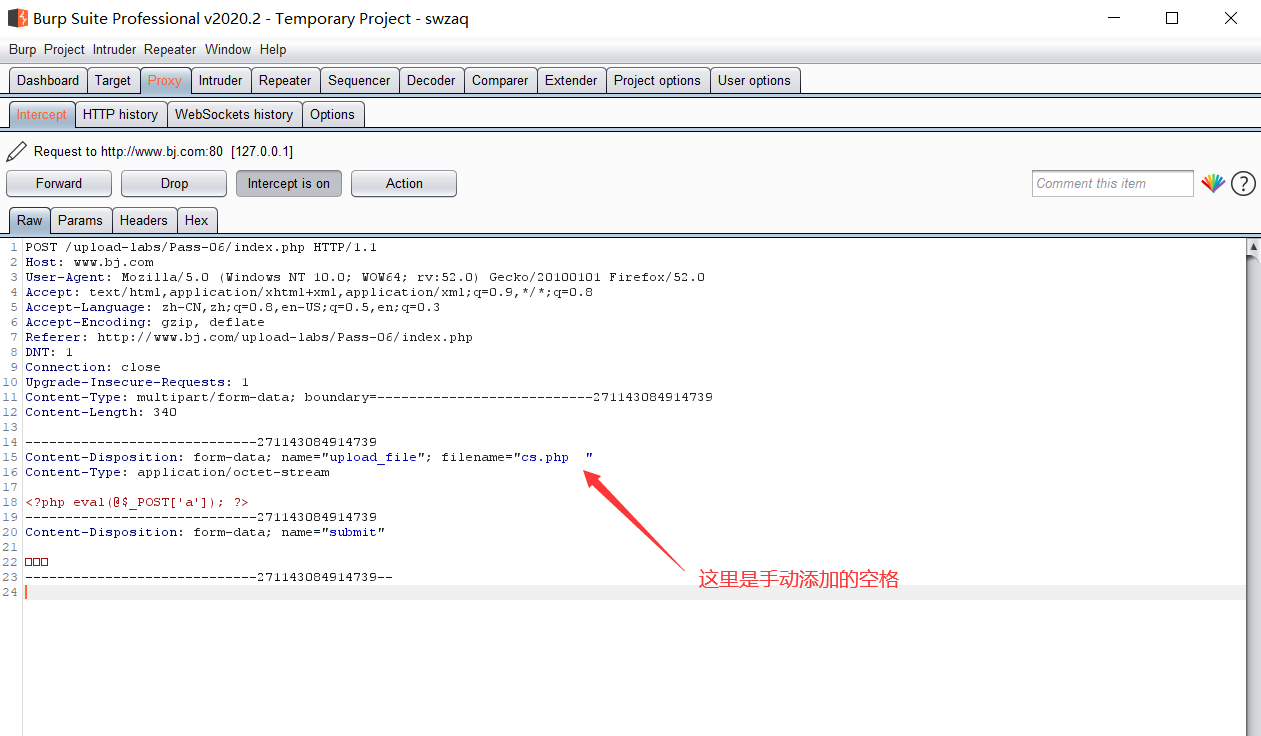
} else {

$msg = UPLOAD\_PATH . '文件夹不存在,请手工创建！';

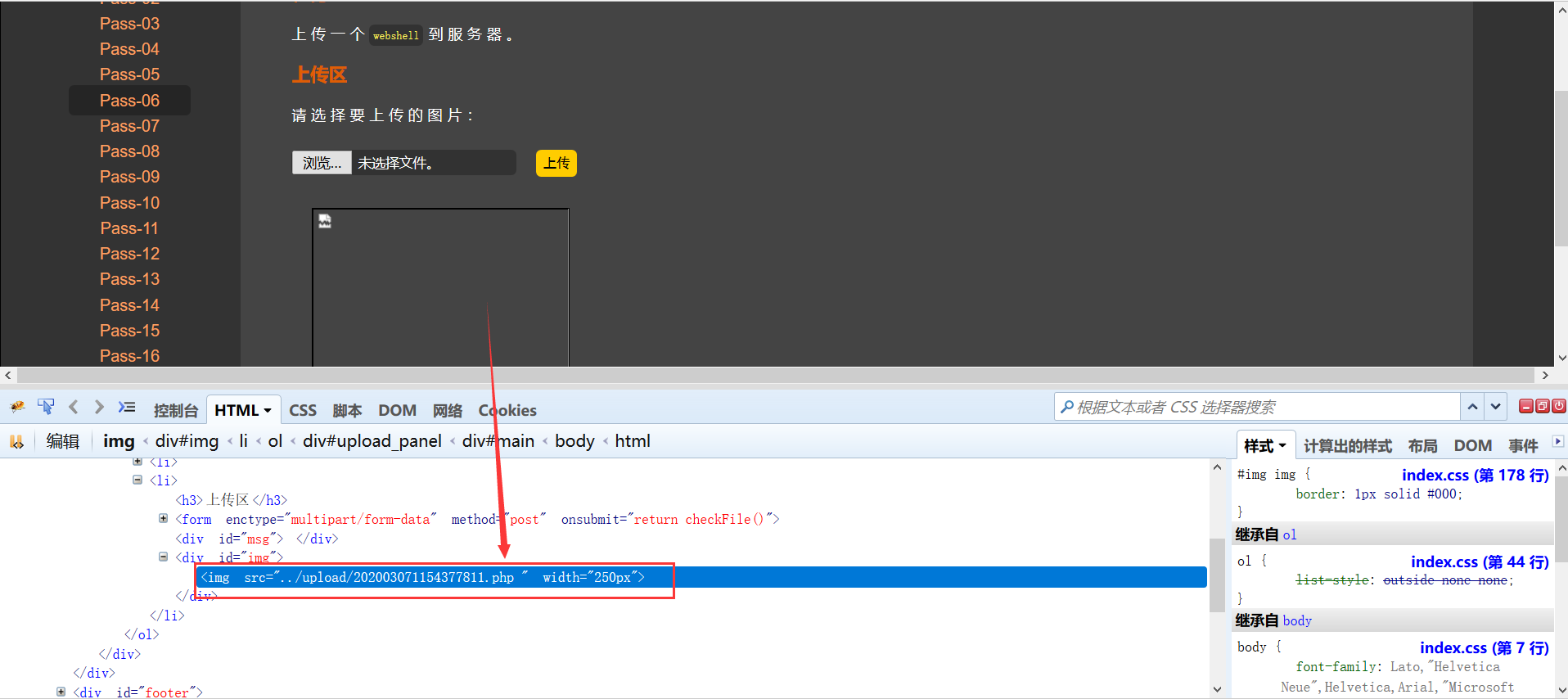
}}

从源码中可以看到，服务器端对文件后缀名仅仅是进行了去除末尾的”.”以及小写转换操作。但是并没有进行去除空格的操作。这里值得一提的是在windows系统中会将文件扩展名后的空格做空处理，并不会被当成另一种不可识别的文件类型。因此可以利用这个特性来绕过这一关的黑名单。

第一步：上传一句话shell，抓取数据包添加空格



第二步：shell成功上传，但是会对文件名重命名

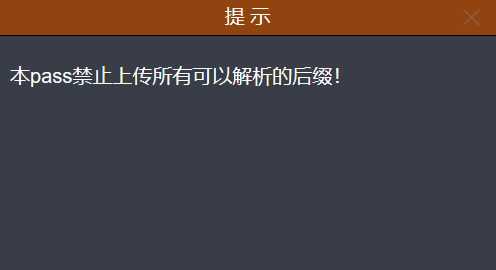


直接用蚁剑连接测试（成功）



Pass07

打开第七关直接查看提示



这么狠的吗，通过源码看看服务器端都干了什么

$is\_upload = false;$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！';

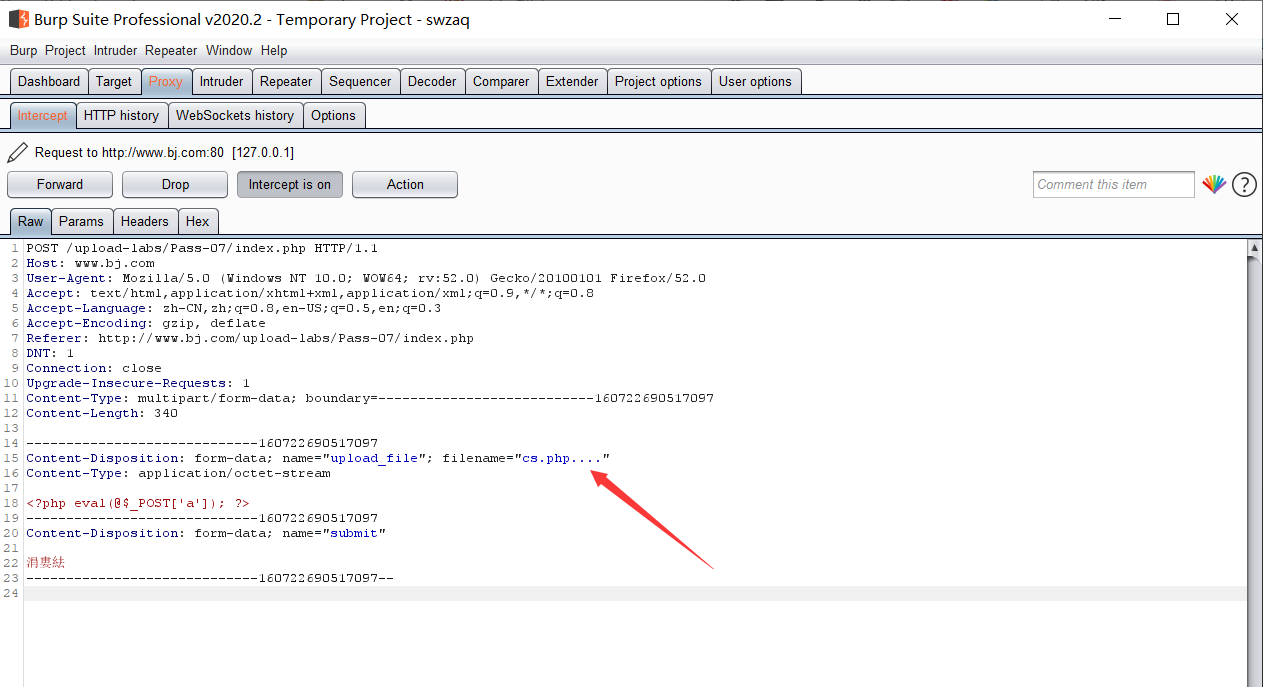
}}

原来是对文件名进行了大小写转换并且把常见配置中允许解析的都禁止掉了，甚至.htaccess文件也没落下。不过总感觉跟上一关的过滤有点差别，仔细对比之后发现原来这里没有对文件后缀名后面进行去除”.”操作。

这里涉及到一个知识点就是windows系统对于文件后缀名后面跟的”.”会自动去除，不管是添加一个还是多个效果都是一样的。

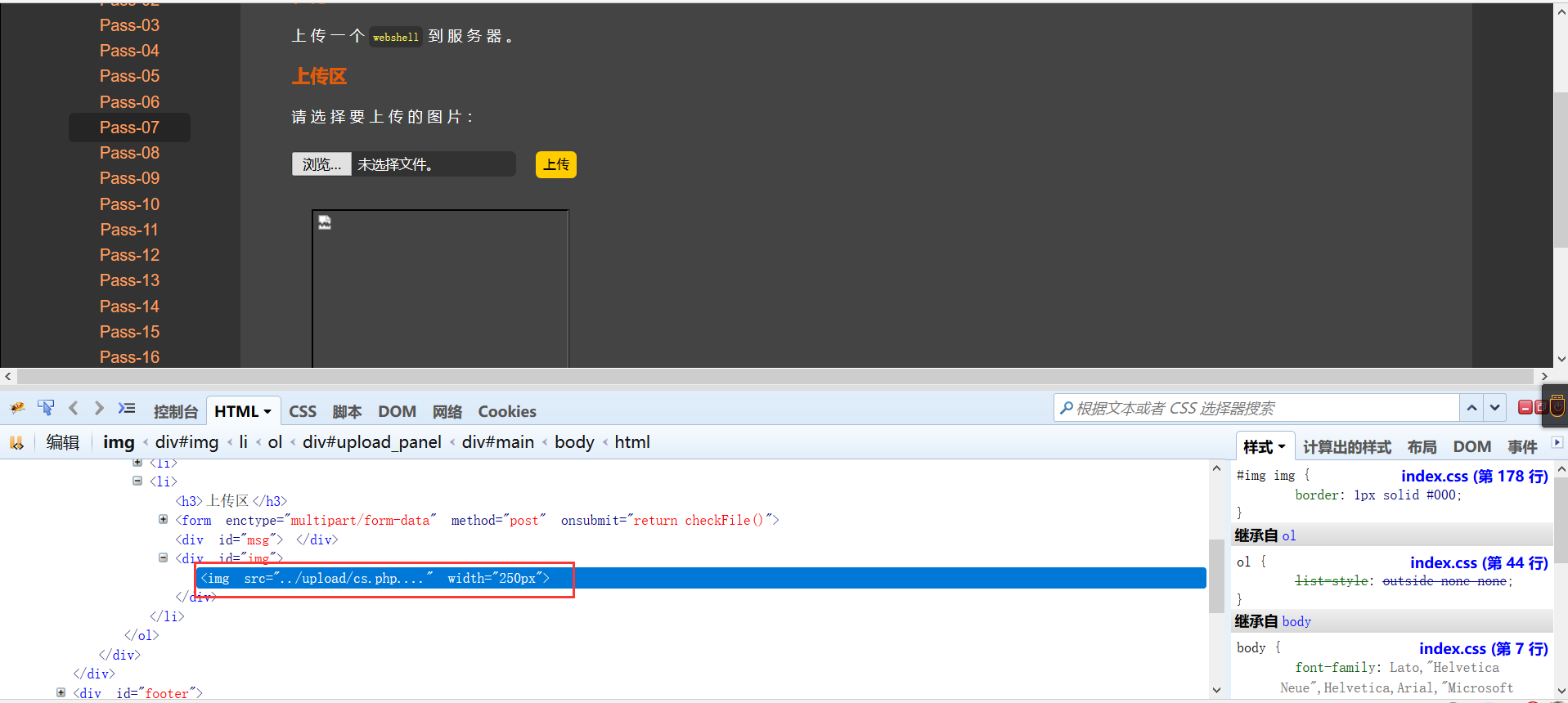
根据上面的思路可以看到，服务器端的代码虽然对文件后缀作出了诸多限制，但是对后缀名加”.”的这种方式却没有防范。可以进行测试。

第一步：上传shell，抓包修改文件名



第二步：分析返回的上传成功文件名和服务器的文件名

首先看页面中的显示



可以看到文件确实是上传成功了，但是上传的文件依然是cs.php...（这样的文件肯定是无法解析的啊）

再来看看服务器中的文件是否如同页面显示的那样



可以看到在服务器中文件其实是去除了后缀名后的”.”的，所以是可以正常解析的。用蚁剑连接测试。

