《漏洞利用及渗透测试基础》实验报告

姓名：姜志凯 学号：2011937 班级：信安班

**实验名称：**

跨站脚本攻击

**实验要求：**

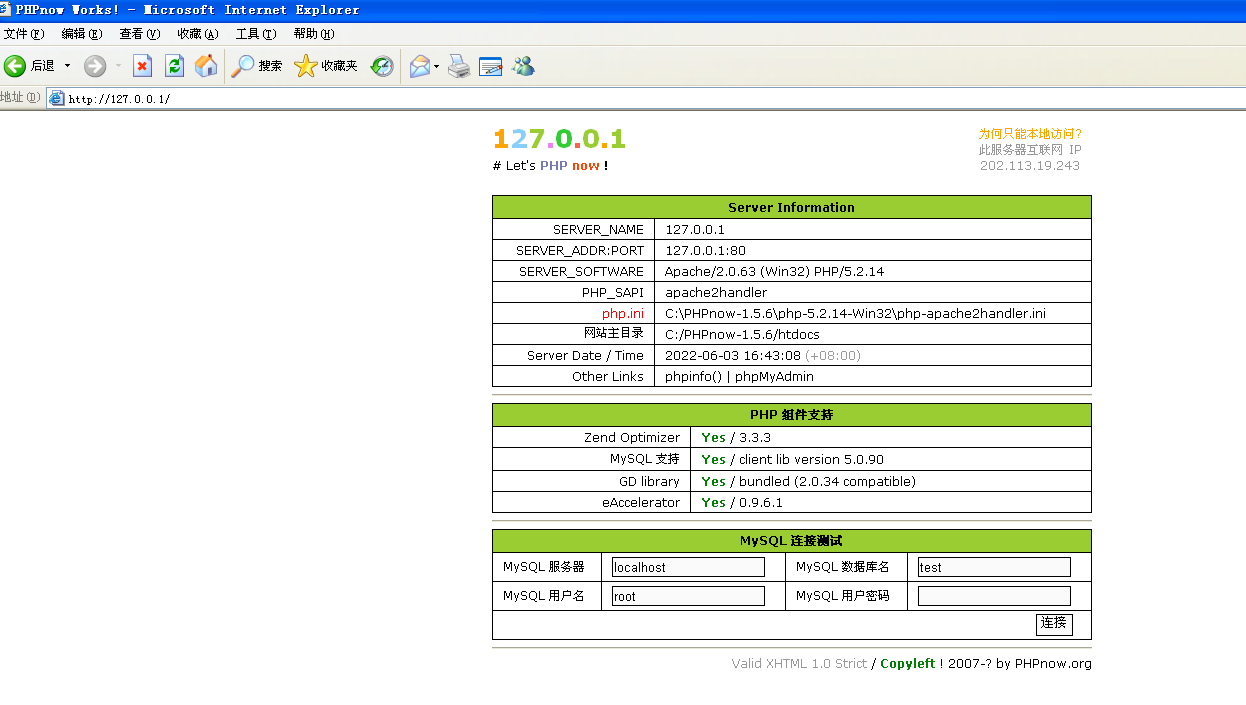
复现课本第十一章实验三，通过img和script两类方式实现跨站脚本攻击，撰写实验报告。有能力者，可以自己撰写更安全的过滤程序。

**实验过程：**

利用phpnow的实现跨站脚本攻击

下图为phpnow的主页面127.0.0.1，用dreamweaver编写PHP文件，保存在本机上的网页文件htdocs的目录下，然后再网页url后加上文件名就可以执行php文件，访问自己编写的网页了。

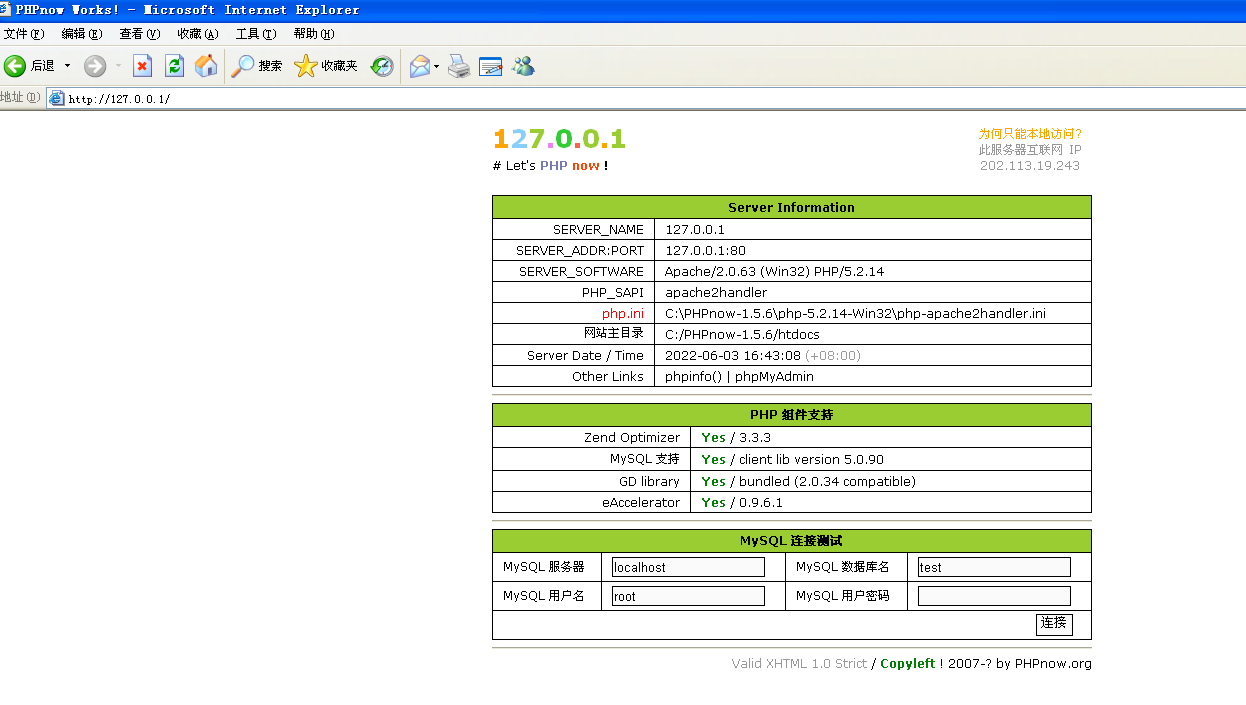
然后在网页内输入脚本，，实现跨站脚本攻击。





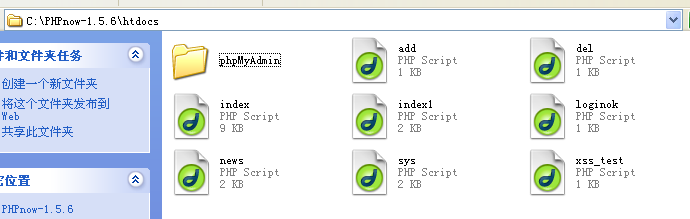


1. 进入PHP主页

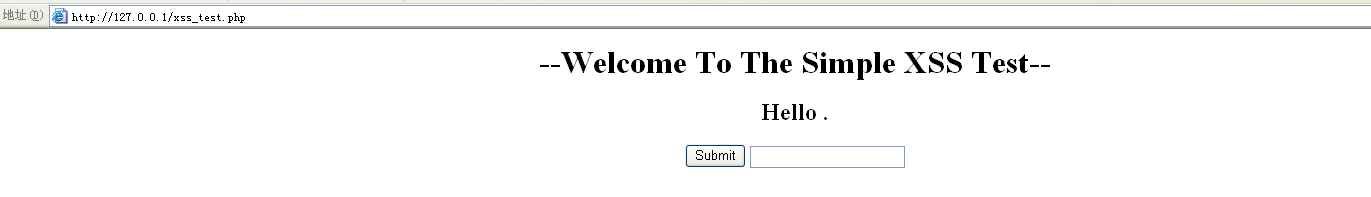


1. 编写php网页文件，保存在主页目录下，后序对其进行XSS攻击





1. 访问xss\_test.php



1. 先从黑盒测试的角度进行xss攻击

最简单的<script>alert('xss')</script>，然后提交。



失败了！

结果发现Hello后面出现了我们输入的内容，并且输入框中的回显过滤了script关键字，这个时候考虑后台只是最简单的一次过滤。于是可以利用双写关键字绕过，构造脚本：<scrscriptipt>alert('xss')</scscriptript>测试。执行效果如下：



现虽然输入框中的回显确实是我们想要攻击的脚本，但是代码并没有执行，并没有执行alert函数弹出消息框。因为在黑盒测试情况下，我们并不能看到全部代码的整个逻辑，所以无法判断问题到底出在哪里。所以我们需要分析源码的逻辑，尝试从源码中分析问题，编写可以成功执行弹窗代码的脚本。

1. 分析源码



·如果可以成功执行alert函数的话，页面将会跳出一个确认框，显示Congratulations~

·Htmlspecialchars函数可以有效防止XSS脚本攻击，是一个过滤函数，实现预定义字符的转换

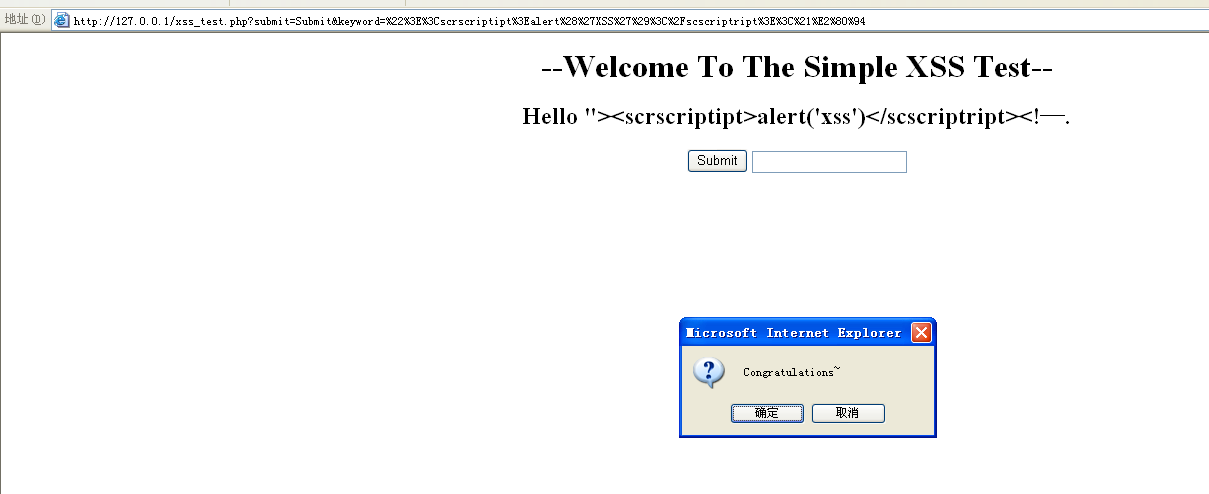
·虽然我们成功的插入了<script></script>标签组,但是我们并没有**跳出input的标签**，使得我们的脚本仅仅可以回显而不能利用。

1. 尝试用script方法

这个时候的思路就是想办法将前面的<input>标签闭合，于是构造如下脚本：

"><scrscriptipt>alert('XSS')</scscriptript><!—

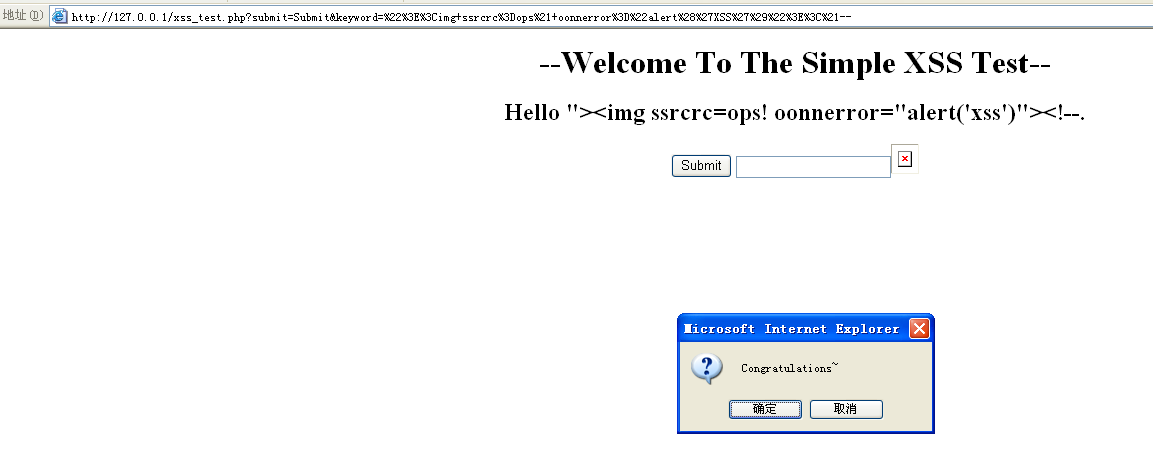
执行结果如下：



1. 用img方法

编写脚本 "><img ssrcrc=ops! oonnerror="alert('XSS')"><!--

执行结果如下：



成功！

**心得体会：**

原理：

·从输入框输入的字符串会经过过滤器过滤，过滤掉某些关键字，所以要对输入的内容进行双写关键字，页面源码如下：

<?php

ini\_set("display\_errors", 0);

$str =strtolower( $\_GET["keyword"]);

$str2=str\_replace("script","",$str);

$str3=str\_replace("on","",$str2);

$str4=str\_replace("src","",$str3);

echo "<h2 align=center>Hello ".htmlspecialchars($str).".</h2>".'<center>

<form action=xss\_test.php method=GET>

<input type=submit name=submit value=Submit />

<input name=keyword value="'.$str4.'">

</form>

</center>';

?>

"><img>ssrcrc=ops! oonnerror="alert('XSS')"<img><!—

由源码可知，过滤器过滤了script、on、src关键字，所以脚本涉及这些词的需要双写，规则为“scrscriptipt”、“oonn”、“ssrcrc”，这样就算过滤掉这些词，前后剩下的还能组成关键字

·如果不使<input>闭合，跳出input，则只会回显，不会执行，所以必须构造以使input闭合：**”>**<......>**<!--**,加粗的是使其闭合的方法，"> 用来闭合前面的<input>标签。而 <!-- 其实是为了美观，用来注释掉后面不需要的 "> ,否则页面就会在输入框后面回显 ">

·script方法在双写关键字和跳出input后，直接就能执行

·<img>方法原理：

"><img ssrcrc=ops! oonnerror="alert('XSS')"><!--

<img>标签是用来定义HTML中的图像，src一般是图像的来源。而onerror 事件会在文档或图像加载过程中发生错误时被触发。所以上面这个攻击脚本的逻辑是，当img加载一个错误的图像来源ops!时，会触发onerror事件，从而执行alert函数。

此外，通过本实验，掌握了多个汇编语言的用法