南開大學
Nankai University

计算机学院

软件安全实验报告

实验十一：跨站脚本攻击实验

姓名：林盛森

学号：2312631

专业：计算机科学与技术

2025 年 5 月 31 日

目录

1 实验名称

跨站脚本攻击实验

2 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。

3 实验过程

3.1 script 方法

3.1.1 编写 xss_test.php

```
<!DOCTYPE html>
<head>
<meta http-equiv="content-type" content="text/html;charset=utf-8"> <script>
window.alert = function()
{
  confirm("Congratulations~");
}

</script>
</head>
<body>
<h1 align=center>—Welcome To The Simple XSS Test—</h1>
<?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>';
?>
</body>
</html>
```

编写完成之后，把这个文件放在 phpnow 的 htdocs 文件夹下，之后我们访问 http://127.0.0.1/xss_test.php，显示如下页面：

图 3.1: Enter Caption

3.1.2 黑盒测试

我们可以看到在这个页面有一个 submit 按钮和输入框，并且还有标题提示 XSS。我们输入我们刚学过的 XSS 脚本：<script>alert('xss')</script> 来进行测试。点击 Submit 按钮以后，效果如下：

图 3.2: Enter Caption

我们发现在 Hello 后面出现了我们输入的内容，但是输入框中的 script 却没有了，这说明网站开发者在进行设计时为了避免跨站脚本攻击回显过滤了 script 关键字。我们可以利用双写关键字绕过这个过滤操作，构造脚本：<scrscriptipt>alert('xss')</scscriptript> 测试。执行效果如下：

图 3.3: Enter Caption

虽然输入框中出现的内容是我们希望看到的内容，但是这个脚本并没有被执行，说明没有被解析运行。我们在这个页面右键查看源码，可以看到第五行的这里的逻辑是当 alert 函数成功执行时，页面将会跳出一个确认框，显示 Congratulations ，这也就是我们进行跨站脚本攻击成功的标志，在代码的第 16 行，可以看到我们成功把我们希望插入的内容插入到的客户端中，但是并没有跳出 input 的标签，使得我们的脚本仅仅可以回显而不能利用。所以我们重新设计脚本，来闭合前面的 input 标签。

```
1  <!DOCTYPE html>
2  <head>
3  <meta http-equiv="content-type" content="text/html;charset=utf-8"> <script>
4  window.alert = function()
5  {
6      confirm("Congratulations~");
7  }
8
9  </script>
10 </head>
11 <body>
12 <h1 align=center>--Welcome To The Simple XSS Test--</h1>
13 <h2 align=center>Hello &quot;&gt;&lt;scrscriptipt&gt;alert('xss')&lt;/scscriptript&gt;&lt;!--.</h2><center>
14 <form action=xss_test.php method=GET>
15 <input type=submit name=submit value=Submit />
16 <input name=keyword  value=""><script>alert('xss')</script><!--">
17 </form>
18 </center>
19 </body>
20 </html>
```

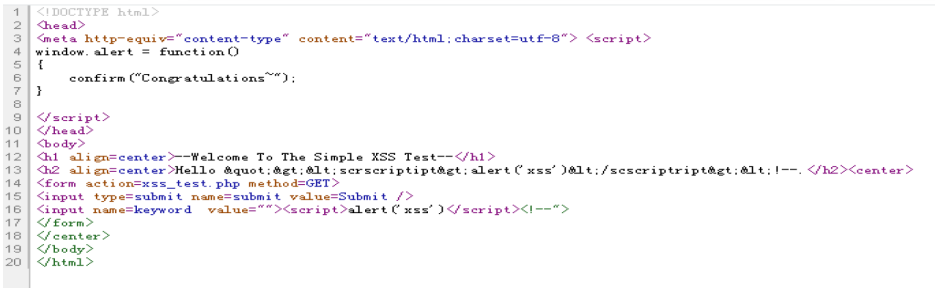
图 3.4: Enter Caption

我们构造新的脚本”><scrscriptipt>alert('XSS')</scscriptript><!– ，并点击提交，但是发现仍然没用弹出提示框，通过查看教材，我发现，php 服务器自动会对输入的双引号等进行转义，以预防用户构造特殊输入进行攻击，为了让实验能正常运行，我们需要在 phpnow 安装目录下搜索文件 php-apache2handler.ini，并将 “magic_quotes_gpc = On” 设置为 “magic_quotes_gpc = Off”。

这样修改之后，再次点击 Submit，可以看到成功弹出了提示 Congratulations ，说明我们成功实现了跨站脚本攻击。

图 3.5: Enter Caption

这时候再查看源码，可以看到在第 16 行，我们实现了 input 标签的闭合，我们的脚本独立在 input 之外，能够正常运行。



```
1  <!DOCTYPE html>
2  <head>
3  <meta http-equiv="content-type" content="text/html;charset=utf-8"> <script>
4  window.alert = function()
5  {
6      confirm("Congratulations~");
7  }
8
9  </script>
10 </head>
11 <body>
12 <h1 align=center>--Welcome To The Simple XSS Test--</h1>
13 <h2 align=center>Hello &quot;&gt;&lt;scrscriptipt&gt;alert('xss')&lt;/scscriptript&gt;&lt;!--. </h2><center>
14 <form action=xss_test.php method=GET>
15 <input type=submit name=submit value=Submit />
16 <input name=keyword  value=""><script>alert('xss')</script><!--">
17 </form>
18 </center>
19 </body>
20 </html>
```

图 3.6: Enter Caption

3.1.3 白盒测试

通过查看 xss_test.php 文件，我们从源码的角度来看一下页面的核心逻辑。

```
1  <?php
2    ini_set( "display_errors", 0);
3    $str=strtolower( $_GET[ "keyword"]);
4    $str2=str_replace( "script", "",$str);
5    $str3=str_replace( "on", "",$str2);
6    $str4=str_replace( "src", "",$str3);
7    echo "<h2 align=center>Hello ".htmlspecialchars($str). ".</h2>". '<center>
8    <form action=xss_test.php method=GET>
9    <input type=submit name=submit value=Submit />
```

```
  <input name=keyword value="'.$str4. '">
  </form>
  </center>';
?>
```

我们发现，除了对 script 进行过滤之外，还会对 on、src 等关键词都进行了过滤，将其替换成了空，另外 Hello 后面显示的值是经过小写转换的。

3.2 image 方法

我们尝试使用 <img src=ops! onerror=”alert('XSS')”>

由于我们之前已经知道会过滤掉 on、src 等关键词，所以我们可以利用双写关键字绕过这个过滤操作，构造脚本：

”><img ssrcrc=ops! oonnerror=”alert('xss')”> <!–

图 3.7: 结果图

输入之后，成功弹出了提示 Congratulations ，说明我们完成了 image 方法的跨站脚本攻击。

4 心得体会

通过这次实验，我了解了如何从黑盒和白盒两个角度实现跨站脚本攻击，并学会了 script 和 image 两种攻击方法，通过精心构造脚本，实现简单的弹窗效果。另外，要想保证网站的安全性，一定要有对输入信息进行检测机制。