基于 Ajax 的 Web 应用安全性研究

杨洁，王虹
武汉理工大学 信息工程学院,武汉市 （430070）
E-mail: yangjie0119@126.com

摘 要：Ajax 是 Web2.0 时代新兴流行的网络技术。Ajax 使得新一代互联网应用系统响应更敏捷、交互性更强、用户体验更丰富。但随着它的广泛应用，也给 Web 应用带来新的安全隐患。本文介绍了 Ajax 的主要技术和原理，然后分析其优势和存在的安全威胁。重点研究针对由于脚本语言、XML 注入和 XSS 攻击问题引起的负面安全影响及其相应的防御措施，使 Web 应用的安全性得到了改善和提高。

关键词：Ajax；Web 应用；安全性；防御

1．引言

Ajax是新兴网络开发技术的代表。它一出现，便迅速应用到Web开发领域，使得新一代互联网应用系统响应更敏捷、交互性更强、用户体验更丰富。目前Ajax正成为Web2.0应用领域的研究热点。最典型的应用是Google Maps、Google Suggest和Google Gmail。但随着它的广泛应用，也给Web应用带来新的安全隐患。2006年6月13日的Yahoo worm[1]是第一个基于Ajax的蠕虫病毒，这对怎样安全地使用Ajax技术提出了紧迫的要求。目前， Ajax技术的安全性受到越来越多的关注。XML安全厂商Forum Systems公司曾提出随着越来越多的Ajax风格的应用出现，很多组织需要考虑潜在的安全缺陷以及性能问题。Ajax in Action 的作者之一Eric Pascarello也谈到Ajax安全方面的议题，并围绕着服务器端的验证和Ajax蠕虫提出了相关的经验法则。

2．Ajax技术

2.1 Ajax 的定义

Ajax 是 Asynchronous JavaScript and XML（异步 JavaScript 和 XML）的缩写。它不是一门新的语言或技术，实际上是由几种蓬勃发展的技术以新的强大方式组合而成。这些技术包括 JavaScript、XHTML、CSS 、DOM、 XML 、 XSTL 和 XMLHttpRequest。其中：使用 XHTML 和 CSS 进行标准化呈现；使用 DOM 实现动态显示和交互；使用 XML 和 XSTL 进行数据交换与处理；使用 XMLHttpRequest 对象进行异步数据读取；使用 JavaScript 绑定和处理所有数据[2]。

2.2 Ajax 的工作原理

Ajax 的工作原理，即基于它的 Web 交互模式，相当于在用户和服务器之间加了一个中间层，通常称为 Ajax 引擎，由一系列 JavaScript 代码组成。Ajax 引擎的责任包括构造用户操作界面以及与服务器的沟通。Ajax 引擎允许用户与应用程序的交互异步进行，即无须直接访问服务器。所以用户永远不会在服务器处理数据期间面对一个白屏和沙漏图标。用户操作的处理由传统的表单提交来激发一个 HTTP 请求，变为 JavaScript 调用 Ajax 引擎。给用户的回应不用等到服务器处理后再返回，简单的数据校验和数据处理，甚至一些导航功能都直接交给 Ajax 引擎来处理[3]如果 Ajax 引擎需要从服务器获取新数据或者加载额外的界面代码，Ajax 引擎通常以 XML 格式激发一个异步的请求，用户完全没有被中断的感觉。Ajax 的 Web 模式与传统的 Web 模式对比如图 1 所示。

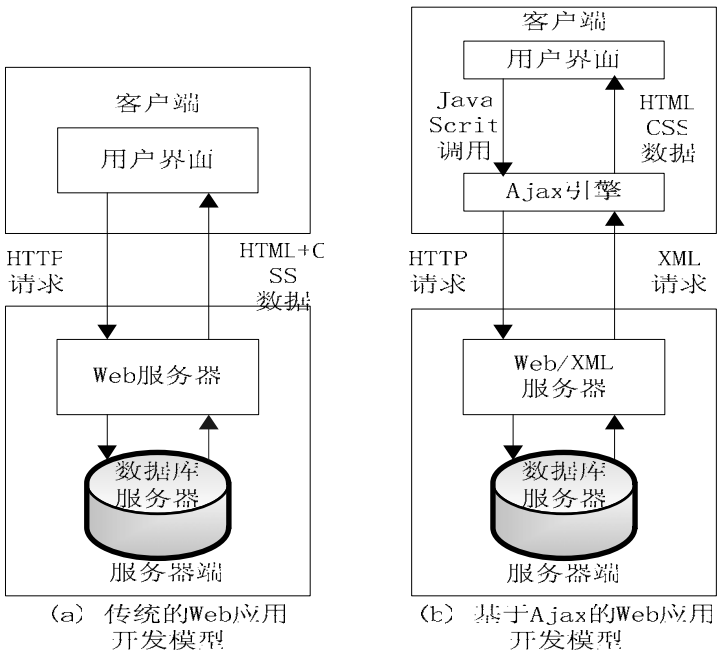

图 1 传统的 Web 模式与 Ajax 的 Web 模式对比

3. Ajax的技术优势

传统的 Web 应用处理模式都是采取同步交互的，用户首先在界面上激发一个 HTTP 请求到 Web 服务器，服务器分析用户请求的内容后执行响应任务并向用户返回结果。由于是“请求-等待-请求”模式，在这循环过程中浏览器显示空白页，用户必须等待，直到服务器返回数据后才重新绘制页面。因为用户得不到立即的反馈，感觉上不同于桌面应用。这是一种不连贯的用户体验，也是 Web 应用交互性差的原因所在。Ajax 不同于传统的 Web 模式，它采用异步交互，服务器处理提交数据的同时客户端无需等待。由于数据的发送和接收在后台完成，用户浏览器端显示的内容不会闪烁、延迟或消失，不会出现“白屏”现象。Ajax 应用的优势主要表现在以下几个方面：

(1)减轻服务器的负担。因为 Ajax 的根本理念是“按需取数据”，所以最大可能地减少了冗余请求和响应对服务器造成的负担。

(2)无刷新地更新页面，减少用户实际和心理等待时间，是更好的用户体验。

(3)把以前一些服务器所负担的工作转嫁到客户端，利于客户端闲置的处理能力来处理，减轻服务器和带宽的负担，节约了空间和带宽租用成本。

(4)是基于标准化并被广泛支持的技术，不需要插件或下载小程序。

(5)Ajax 使 Web 中的界面与应用分离（也可以说是数据与呈现分离）。

由此可见，Ajax 使得 Web 应用既保留了 B/S 结构的优点，又具有 C/S 结构应用的强大功能和用户感受。它极大地改善了 Web 应用的交互性和可用性，使 Web 应用更加动态、更加智能，最终得到了用户和市场的广泛认可。

4. Ajax应用安全与防范措施

Ajax 技术存在的诸多优点及其应用的适应性，它将在 Web 应用中起着重要作用。但随着 Ajax 技术的广泛应用，已经出现了基于 Ajax 的蠕虫（Samy worm、Yamannerworm[4]）和信息泄露事件[5]，因此 Ajax 应用的安全性及其相应的防御措施越来越受到重视和研究。

4.1 恶意 JavaScript 问题

Ajax 最基本的开发语言是 JavaScript,这就决定了 Ajax 在 Web 程序中是以脚本形式存在于客户端的， JavaScript 脚本可能暴露用户标识、产品标识、数据库表结构等重要信息。恶意攻击者利用这些信息，然后构造特定的 HTTP 请求，直接访问服务器来获取敏感信息。这样信息就很容易被泄露。

防御措施：

（1）把交互脚本写进 js 文件，对 URL 实施过滤，拒绝直接对 js 文件的访问请求，防止 js 文件被下载。

（2）使用 JavaScript 源代码迷惑工具，使得释放到客户端的 JavaScript 代码不可读。

（3）对含重要信息的资源进行安全访问限制，在配置文件 web.xml 设定需要保护的资源，对该资源的访问要进行认证。

另外，用户输入进行校验，可以减少各种注入漏洞、缓冲区溢出和拒绝服务攻击等。但是 Ajax 技术在客户端 JavaScript 中执行输入校验是非常不安全的，例如：对于 Get 请求，可以直接在地址栏中输入请求参数，使得校验失效；还可以自己构造页面，然后把请求提交给服务器处理。因此，校验变为不可靠。

防御措施：

（1）在服务器端执行输入校验。

（2）对特殊字符进行过滤，防止注入。

（3）对输入长度进行限制，防止缓冲区溢出或拒绝服务攻击。

4.2 严重的 XML 注入问题

XML 注入问题在传统 Web 应用中很少被提起，非常容易被忽略。大多数 Ajax 框架使用 XML 来传输数据，这使得 XML 注入的方法更自动化，更为隐蔽。故也会引起的负面安全影响。

例如下面是一个 Web 应用的用户数据结构：

```
<?xml version="1.0"encoding="utf-8"?>
<Students>
< Student ID="1">
<UserName>张强</UserName>
<Password>123</Password>
</ Student >
< Student ID="2">
<UserName>王芳</UserName>
<Password>456</Password>
</ Student >
……
</ Students >
```

此服务端程序使用 C#作为后台语言，则查询的字符串，作为最普通的情况，用户名和密码被作为认证的标志字符串，于是，查询的语句如下：

```
String FindUserXPath;
```

FindUserXPath="// Student [UserName/text () =’"+Request ("Username") +"’
And Password/text () =’"+Request ("Password") +"’]";

其中，Request("Username")和 Request("Password")是用户在 Web 表单内输入的值，结果为真，则说明用户可以被认证，反之则说明用户名或者密码错误，不给与用户权限.

然而，这个查询语句没有对用户的输入做检查，若用户在表单中输入：

UserName: hi ’or 1=1 or’ a’=’a

Password: hi

则查询字符串即刻便成了：

"// Student [UserName/text () =’hi’or 1=1 or’ a’=’a’ And Password/text () =’hi’]";

而这个字符串，在逻辑上等价于：

"// Student [(UserName/text () =’hi’or 1=1) or (’a’=’a’ And Password/text () =’hi’)]";

而这个字符串无论如何查询，返回结果总为真，导致恶意攻击者取得了不应有的权限。存在安全隐患。

防御措施：对特殊字符进行过滤，防止注入。并根据 XPATH 查询字符串语法进行替换，在上述例子中只要将单引号替换成&apos 就可以在一定程度上解决注入的问题。

4.3 严峻的 XSS 跨站攻击问题

在基于 Ajax 的 Web 应用中，由于 Web 页面在 JavaScript 和 XMLHttpRequest 对象的控制下，拥有了无需用户交互而直接向 Web 服务器进行 HTTP 请求的能力，这使 XSS 攻击变得更加容易，因此危险程度较传统 Web 架构下大大提高了。

XSS 又叫 CSS(Cross Site Script)，跨站脚本攻击。它指的是恶意攻击者往 Web 页面里插入恶意 html 代码，当用户浏览该页之时，嵌入其中 Web 里面的 html 代码会被执行，从而达到恶意用户的特殊目的[6]。其流程往往如图 2 所示

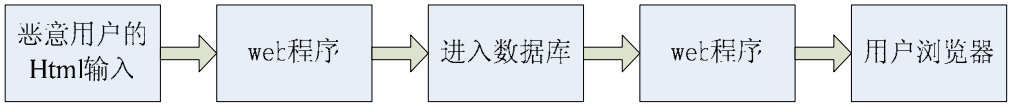

图 2　XSS 攻击的流程图

例如当攻击者将被攻击的网页注入以下情况时：

<iframe src="http://www.poo.com/search.asp?text=<script>alert (document. cookie) </script>">

用户的浏览器便会在装载网页时，自动跳转到 www.poo.com 的网站，并弹出包含当前网页 cookie 信息的提示框。若改变一下程序的结构，把弹出提示框中的内容提交到攻击者预先设置的 www.poo.com 网站，那么用户在当前 cookie 中的所有隐秘信息都将完全暴露给攻击者，XSS 攻击就可以收集受害者的敏感信息（如 cookie）和记录用户键盘操作；可以挟持用户会话；可以实施网络钓鱼等诈骗活动可以篡改网页内容，严重破坏组织的声誉，给 Web 应用带来严重的负面安全影响。

防范措施：

（1）对输入的字符进行转义处理不解析其中的标签。

（2）对输入长度作限制。

5．结束语

本文在基于 Ajax 的 Web 应用安全性研究中，逐一举例分析针对由于脚本语言、XML 注入和 XSS 攻击问题对 Web 应用的安全威胁的成因，并研究其相应的防御措施，使 Web 应用的安全性得到了改善和提高。正如 Open Ajax 联盟主席 Bo-loker 所说,Ajax 的安全问题实际上就是 Web 的安全问题。随着 Ajax “最佳实践”的出现，相信 Ajax 技术所带来的负面影响可以降到最低。同时我们可以预见在不久的将来安全高效的基于 Ajax 的 Web 开发及应用会得到更好的发展。

参考文献

[1] Roberts P.Yahoo worm demonstrates AJAX threat [EB/OL]. http://www.macworld.com/news/2006/06/16/ajax/index.php,2006-06-16.
[2] Ryan Asleson,Nathaniel T.Schutta.Ajax 基础教程[M].北京:人民邮电出版社.2006.
[3] 柯昌正,黄厚宽.AJAX 技术的原理与应用[J].铁路计算机应用.2007.
[4] Steven Holzner.Ajax Bible[M].WileyPublishing, Inc.2007.
[5] Billy Hoffman, Ajax Security Dangers[EB/OL].http://www.spidynamics.com/ assets/documents/AJAXdangers.pdf.2006.
[6] T_Torchidy,也谈跨站脚本攻击与防御,XFocus,2006.

The Web Application Security Research Based on Ajax

Yang Jie, Wang Hong
School of Information Engineering, Wuhan University of Technology, Wuhan, (430063)

Abstract

Ajax is the Web2.0 era emerging popular network technology. Ajax made the new generation of internet application system can respond more agilely, interact more rapidly, make user-experience more bountifully. But with its wider application, also bring security risks to new Web application. In this paper, introduced the Ajax main technology and principles, and then analyze the advantages and the security threat. As focus on scripting language, XML injection and XSS attack that has caused negative impact on security and defense measures corresponding to the security, enabled the Web application security to be improved and enhanced.

Keywords: Ajax, Web application, security, defense