WEB 应用程序漏洞发掘原理

黄玮
北京邮电大学信息工程学院，北京（100876）
E-mail：huangwei1983@gmail.com

摘　要：Web 应用程序由于其良好的跨平台性和交互性特点，正在得到越来越普遍的应用。伴随着 Web 应用程序发展的过程中，各种针对 Web 应用程序的攻击也在逐步的发展。本文就目前 Web 应用程序所面临的各种安全威胁从漏洞发掘的角度进行了归纳和分析，从漏洞产生原因上去介绍构建安全 Web 应用程序需要注意的问题。

关键词：WEB 应用程序，漏洞，安全
中图分类号：TP393

1. 引言

Web 应用程序在发展的开始仅仅局限于静态页面的展示，用户可以做的仅仅是“浏览网页”。直到 20 世纪 90 年代初期开始，基于服务器端的脚本和 CGI 技术的发展，使得 Web 应用第一次实现了用户交互。今天我们所看到的和经常使用的 Web 应用程序，如 Gmail，Google Maps，Blog 等等，都得归功于 Web 2.0 技术带给我们的全新的 Web 应用程序用户体验。

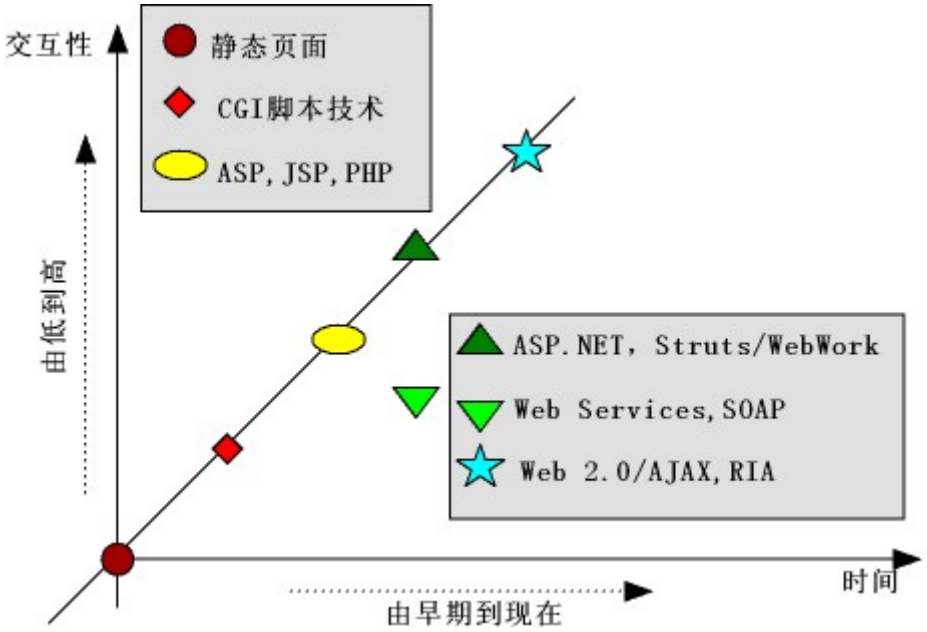

图 1 Web 应用发展简史

图 1 简要的总结了 Web 应用程序从最初的静态页面到今天的 Web 2.0 技术的发展历程。从图上我们可以看出，Web 应用程序发展的一个重要的趋势就是越来越高的用户交互性。不论是 AJAX 技术也好，还是 Flash 也罢，Web 应用程序从来都没有像今天这样发展的越来越接近桌面应用程序。更高的用户输入响应速度，完善和灵活的用户 UI 设计，客户端“0 维护”成本，等等，越来越多的理由让我们去普及和应用最新的 Web 技术去获得更高的用户交互性体验。而作为 Web 2.0 技术发展的基础，从微软的 ASP.NET 到 Sun 的 JSF 框架，再到最新的 PHP 的 Smarty 模版技术，“动态网页”仍然将在今后很长一段时间内主导 Web 应用程

序的主流。

然而，自从在 Web 应用程序中引入了“交互性”，各种针对 Web 应用程序的攻击和漏洞利用也开始随着“交互性”的发展而迅速发展。这里面既有 Web 应用程序的基础——HTTP 协议和其他相关 TCP/IP 协议的脆弱性，也有 Web 应用程序本身在编码实现时候的不规范性，更有 Web 应用程序部署和管理上的疏忽和大意原因。

2. Web 应用程序风险分类及漏洞发掘原理

根据 SANS Institute 2006 年的一份最新的安全报告[1]，在过去的一年中，互联网上的安全攻击目标中，Web 应用程序已经成为跨平台的应用程序中最频繁遭受攻击的对象。对比 2005[2]和 2004[3]年的情况，Web 应用程序风险经历了从 Web 服务器的频繁遭受攻击，到针对 PHP、ASP 等动态脚本网页技术本身的漏洞攻击，再到现在的不局限于具体动态网页实现技术的各种跨站脚本攻击(XSS)和各种数据注入攻击(Data Injection)，攻击的目标和手段不断的变化，但基本的漏洞利用原理仍然无非是 2 个方面，一个就是 Web 应用程序对用户输入和请求参数的校验漏洞，而另一个则是 Web 应用服务部署和管理的缺乏安全意识和粗心大意。

2.1. 输入和参数校验

在各种 Web 应用程序的漏洞利用中，排除 Web 应用程序服务器本身的安全漏洞，对用户提交的表单请求和 GET 请求中参数的过滤不严格导致的 Web 应用程序漏洞构成了当今 Web 应用程序漏洞利用的主要方式。下面重点讨论各种可能的漏洞利用情况。

2.1.1. 缓冲区溢出漏洞

这种漏洞利用多见于针对 CGI 脚本构建的动态 Web 应用程序，具体的 CGI 脚本可能是 C，漏洞利用一般是通过构造超长请求字符串来测试是否存在漏洞。

实际上，缓冲区溢出攻击几乎可以应用于所有的 Web 应用程序。对于使用 Java/J2EE 的应用程序来说，如果启用了本地方法调用或者允许系统调用就使得其存在被缓冲区溢出攻击的可能；对于.NET 应用程序来说，如果使用 P/Invoke 方式进行了 Win32 本地 API 调用或者是使用 COM Interop 技术使用了 COM 组件，也会可能存在上述漏洞；Perl、Python、PHP 等脚本如果调用了外部应用程序或者使用了一些不安全的程序库，也会可能存在上述漏洞。

2.1.2. 跨站脚本漏洞

现在只要谈到跨站脚本漏洞，大家一般会想到的危害就是偷取用户 cookie，重定向用户的表单请求偷取用户登陆信息等等。一个简单的跨站脚本漏洞的利用方法是在正常的 URL 请求后添加一小段 JS 代码，如下所示：

```
<a
HREF="http://www.example.com/?tw=<script>document.location.replace('http://freewebhost.com/steal.cgi?'
+document.cookie);</script>">点击参加抽奖</a>
```

如果用户点击了上述这段代码生成的超链接，将会把用户在www.example.com的cookie发送到freewebhost.com的steal.cgi，steal.cgi用于收集存储这个cookie的信息，如果这个cookie中包含了用户的登陆信息，则用户登陆信息就被窃取了。

一些安全性较高的站点都会对URL请求中的请求参数进行过滤和监控，所以一般高级的跨站脚本漏洞都会采用一些变形方法，如16进制编码、随机空白填充等等，这些变形方法给我们的检测带来了很大的麻烦。

2.1.3. SQL 注入

最早的 SQL 注入就是经典的“1’ or ‘1’=’1”漏洞了，曾经有无数的站点登陆控制就被这样一个简单的 SQL 注入攻击而攻破。

现在的 SQL 攻击已经不仅仅局限于绕过登陆验证了。通过构造查询参数中包含 SQL 语句，可以对 Web 应用程序的后台数据库进行增、删、改，查询非授权的数据。对于某些配置不当的数据库系统，甚至有可能获得对 Web 应用程序所运行的操作系统的控制权。

通常的 SQL 注入漏洞利用包含以下几个步骤：

- 数据库类型信息检测，由于不同的数据库的 SQL 语法的差异，需要先明确 SQL 注入的数据库类型是 SQL Server 还是 Oracle 等等。检测方法可以是数据库出错提示信息、使用特定于指定数据库的语法观察返回结果、使用专业的扫描器等。
- SQL 注入尝试，常用的构造 SQL 注入的方法包括利用数据库注释标记、“短路”查询、union 查询等。

2.1.4. cookie 漏洞

在前面的跨站脚本漏洞部分已经提到，利用 cookie 可以登陆某些站点而无需输入用户口令，实际上，在现代的 Web 应用服务器实现中，几乎所有的 Web 应用服务器都支持 HTTP Session 机制，而区分不同 Session 的标记通常都记为 SessionID，服务器端通过 SessionID 来区别不同的用户请求对象，许多 Web 应用程序的权限系统设计就是基于 Session 和 Cookie 的。本来，Cookie 作为客户端的验证技术，和 Session 这种服务器端的验证技术是独立存在的，但是由于一些 Web 应用程序设计上的缺陷，经常出现 Cookie 校验成功之后就不进行 Session 验证，从而导致通过构造合法的 Cookie 来提升权限的目的。

常用的 Cookie 漏洞技术就是 Cookie“重放”攻击，攻击者通过在 HTTP 的请求头中包含合法的 Cookie，来绕过验证，或者提升权限等。

2.1.5. 脚本注入和 HTML 滥用

前面已经提到了跨站脚本漏洞问题，从技术类型上来说，也是一种脚本注入的漏洞。但跨站脚本攻击和脚本注入攻击的一个最显著的区别就是前者一般不会用到持久化存储，而后者一般是持久化的存储在受害 Web 应用服务器上或 Web 应用程序的后台数据库中。一个典型的漏洞利用过程如下。

原始的文本输入框代码是这样的：

```
<input type="text" name="victim" value=""/>
```

攻击者在输入框中输入以下代码并提交包含这段代码的表单：

```
"/><script>alert(“vulnerable html")</script><input type="hidden" name="nosense
```

服务器处理表单并返回的结果代码将会变成这样：

```
<input type="text" name="victim" value=""/><script>alert(“vulnerable html")</script><input type="hidden" name="nosense"/>
```

上述代码经过浏览器解释执行后的结果就是弹出一个提示框显示“vulnerable html”。攻击者只需要改变脚本的内容，就可以达到执行任意脚本代码的目的。

许多有安全意识的 Web 应用程序开发者已经注意到了这个问题，他们对所有的用户输入进行了过滤或者 HTML 转义输出。但是，有些时候，比如需要丰富文本编辑器支持来接受用户输入的场景，一个典型的应用就是许多论坛采用的发表文章用到的嵌入在网页中的编辑器。这些编辑器的主要用途就是需要直接或间接支持 HTML 代码的可视化编辑，来达到

丰富网页显示效果的目的。不论这些编辑器是否直接支持 HTML 代码编辑，或是采用更加安全的 UBB[5]标签，只要网站支持用户自定义的动态页面显示效果，脚本注入和 HTML 滥用的危险都会存在。

在这儿再给一个 UBB 标签的脚本注入的示例代码：

```
[COLOR=[IMG]http://aaa.aa/=`aaa.jpg[/IMG]]`style=background:url("javascript:document.location.replace('http://hacker.com');") [/COLOR]
```

这段代码只是简单的将当前用户的浏览页面重定向到http://hacker.com，实际上如果对JavaScript脚本技术熟悉的话，稍加改动就可以进行更复杂的脚本注入或跨站脚本攻击。

2.2. 应用部署和管理[6]

前面讨论的这些漏洞的产生主要集中在 Web 应用程序的设计和开发过程中，实际上，一个安全的 Web 应用程序不仅仅需要经过规范和严谨的设计和编码实现，Web 应用程序的部署和运行期维护管理也是至关重要的。

下面简要列举一下在 Web 应用程序部署和运行维护阶段可能会出现的安全风险和漏洞利用方法。

- 配置管理

不恰当的配置管理在Unix或Linux系统中最常见的就是以root身份运行Web应用程序，在 Windows 系统中最常见的就是 SQL Server 服务器配置不当，如 SA 账户连接数据库、没有禁用或屏蔽对系统默认存储过程的访问等等。对于这种类型的安全风险，最常用到的漏洞利用方法就是通过溢出程序直接获得管理员权限的 shell，直接对数据库服务器的攻击等等。除此之外，不恰当的授权和 Web 应用程序后台管理界面的远程访问地址暴露，将会允许攻击者采用各种暴力破解的方法进行枚举攻击。

- 敏感数据保护

敏感数据包括用户登陆 Web 应用程序时的口令，Web 应用服务器所在操作系统的系统文件等。对于敏感数据的主要威胁有：网络监听、Web 服务器目录遍历和任意文件下载。

- 审计和日志

一个良好的健壮的日志系统应该可以记录所有必要的用户请求历史，以备攻击后的取证等工作。不幸的是，现在的许多 Web 应用程序日志普遍采用的是本地日志和周期覆盖方式的日志，Web 应用程序的审计工作一般也仅由操作系统提供，所以如果在短时间内产生了大量的日志，就很有可能会覆盖一些历史日志，导致攻击者的一些行为无法被记录。更有一些漏洞利用程序，专门用于破坏 Web 应用程序的审计和日志系统，从而达到擦除攻击痕迹的目的。

3. Web 应用程序漏洞利用新趋势

随着 Web 2.0 技术的普及应用，Web 应用程序相比本地应用程序的交互性差的状况正在得到逐步的改善，无刷新技术和实时按需加载数据正得到迅速的推广和应用。在众多 Web 2.0 技术的应用中，博客（Blog）和新闻聚合（RSS）是普及程度最高的两种应用。人们在享受 Web 2.0 所带来的这些前所未有的 Web 应用体验的同时，也在经历着各种新的 Web 漏洞利用所带来的危害。

著名的 MySpace 蠕虫病毒和 2006 年 7 月的百度空间 XSS 漏洞都是典型的 Web 应用新漏洞的代表，从这 2 次漏洞的被利用情况可以发现一个 Web 应用程序漏洞利用的新趋势，即 CSS 注入和异步 JavaScript 的利用。通过在 CSS 中嵌入 JavaScript 脚本的方式来进行注入

的方式比之以往的任何一种脚本注入都要难以侦测，而异步 JavaScript 的使用使得用户隐私数据的窃取变得更加隐蔽。

4. 结束语

我们在欢呼 Web 应用程序的迅速发展所带给我们的更加愉悦的用户体验的同时必须看到，Web 应用程序的安全性也正在变得越来越复杂。Web 应用程序由于 HTML 代码的透明性和 HTTP 协议本身的脆弱性，其漏洞利用技术难度要比传统的蠕虫病毒和木马技术要低的多，用户甚至不需要编写一行代码，仅仅通过浏览器就可以对 Web 站点发起各种各样的攻击。而 Web 应用程序的跨平台和公开访问特点，使得互联网上所有可以访问该 Web 应用程序的用户都能成为潜在的攻击者，所以 Web 应用程序的安全性控制可谓任重而道远。

参考文献

[1] The SANS Institute．SANS Top-20 Internet Security Attack Targets (2006 Annual Update)[OL]．http://www.sans.org/top20/?ref=1697，2006.11．
[2] The SANS Institute．The Top 20 Most Critical Internet Security Vulnerabilities[OL]．http://www.sans.org/top20/2005/?portal=0b6226184486b13edda77d2dd48a1f70#c3，2005．
[3] The SANS Institute．The Top 20 Most Critical Internet Security Vulnerabilities[OL]．http://www.sans.org/top20/2004/?portal=0b6226184486b13edda77d2dd48a1f70#w1，2004．
[4] Scott D, Sharp R. Developing secure Web applications[J]. Internet Computing, IEEE, 2002, Volume 6, Issue 6:38 - 45
[5] UBB 帮助[OL]. http://www.dwfz.net/liuyan/ubbhelp.htm
[6] Microsoft Corporation. Improving Web Application Security: Threats and Countermeasures [M]. Microsoft Press. 2003

Web Application Vulnerability Exploit Theory

Huang Wei
Department of Information Engineering, Beijing University of Posts and Telecommunications, Beijing, PRC (100876)

Abstract

Web application is widely used today due to its great characteristic of platform-independent and interactive. At the same time, attacks on Web applications are being progressed too. This paper does some research and analysis on the kinds of security risks of web applications from the vulnerability exploit point of view. It introduces the issues that should be paid attention to in the process of secure web application design by explaining the cause of security holes.

Keywords: Web Application Vulnerability Security