文章编号：2095-2163(2019)06-0322-06 中图分类号：TP393.08 文献标志码：A

基于 Web 会话管理漏洞检测技术的研究

谢品章，曾德生，庞双龙
（广东创新科技职业学院 信息工程学院，广东 东莞 523960）

摘 要：WEB 会话管理涵盖了从登录到离开 WEB 应用程序的所有用户控制，在提高应用的易用性和友好性方面发挥了重要作用，同时也增加了在无须提供正确凭证就能非法访问 WEB 应用程序的风险。本文首先对 WEB 会话管理漏洞进行详细分析，针对不同类型的 WEB 会话管理漏洞设计相对应的检测方法，并在此基础上提出 WEB 会话管理漏洞的防范措施。

关键词：Web 安全；会话管理；漏洞检测；防范措施

Research on Web-based Session Management Vulnerability Detection Technology

XIE Pinzhang, ZENG Desheng, PANG Shuanglong
(Department of Information Engineering, Guangdong Innovation and Technology Vocational College, Dongguan 523960, Guangdong, China)

【Abstract】 Web session management covers all user control from login to leave the Web application. It plays an important role in improving the usability and friendliness of the application. It also increases the risk of illegal access to the Web application without providing correct credentials. Firstly we make a detailed analysis of the Web session management vulnerabilities, then according to different types of Web session management vulnerabilities, corresponding detection methods are designed, and on this basis the prevention measures of Web session management vulnerabilities are proposed.

【Key words】 web security; session management; vulnerability detection; preventive measures

0 引 言

HTTP 是一种无状态协议，WEB 服务器并不需要关联客户端请求就会做出响应[1]。为了避免不断地认证网站或者服务的每一个页面，WEB 应用程序通常采用某种会话管理机制将多个客户端请求关联起来组成一个“会话”，每个会话使用一个会话标识符或者 Cookie 来标识。通过在预先确定的时间范围内存储和验证其会话凭证，实现对网站用户交互方式的控制，会话管理涵盖了从登录到离开 WEB 应用程序的所有用户控制，在提高应用的易用性和用户的友好性方面发挥了重要作用，同时也增加了在无须提供正确凭证的情况下进入用户账号，非法访问 WEB 应用程序的安全风险。

大多数的 WEB 应用程序都是通过 Cookie 来关联一个会话中的多个请求，RFC6265 文档对这一方法给出了详细的说明。一个典型的应用例子就是在线购物车，整个用户会话期间，WEB 应用程序必须跟踪用户身份、个人信息以及选择购买的产品、数量、单价、折扣等信息，这些信息存储在 Cookie 中。WEB 应用程序在 HTTP 应答中使用 SET-COOKIE 指令来创建相关的 Cookie，当 WEB 应用程序指示客户端浏览器使用某个 Cookie 后，浏览器就会在以后的每个请求中发送该 Cookie。由于 Cookie 中的数据非常重要，如果会话管理存在漏洞，则可能导致用户会话被劫持，利用当前活动会话非法进入用户账号，获得访问 WEB 应用程序的权限，在未经授权的情况下实施非法操作。

会话管理漏洞在 2010 年的 OWASP TOP 10 中排名第三，而在 2013 年的 OWASP TOP 10 中排名第二，说明会话管理攻击事件在不断增加，安全风险在上升。本文对 WEB 会话管理漏洞进行了详细分析，然后提出 WEB 会话管理漏洞的检测方法，最后针对该漏洞提出一些防范措施。

1 Web 会话管理漏洞分析

会话管理漏洞主要是由于 WEB 应用程序的会话管理缺陷。Cookie 属性设置不当等原因造成的。

基金项目：广东创新科技职业学院 2018 年科研项目（2018XJYB027）；2018 年度广东省普通高校重点科研平台和科研项目（2018GkQNCX065）。

作者简介：谢品章（1982-），男，硕士，高级工程师，主要研究方向：信息安全、数据存储；曾德生（1983-），男，硕士，高级工程师，主要研究方向：云计算、大数据技术；庞双龙（1988-），男，硕士，讲师，主要研究方向：云计算、虚拟化技术。

收稿日期：2019-08-06

例如:

(1)WEB 应用程序或客户端使用非加密信道传送 Cookie,攻击者通过网络监听,非法获取 Cookie,导致 Cookie 内容泄露或者被篡改。

(2)WEB 应用程序的会话 ID 生成算法缺乏足够的随机性,攻击者通过逆向分析,伪造一个有效的 Cookie,实施未授权的非法访问。

(3)WEB 应用程序在会话中使用了固定的会话 ID, 攻击者通过会话劫持攻击,假冒合法用户绕过 WEB 应用程序的身份认证机制,非法进入该用户账户。

(4)Cookie 属性设置不当,导致 Cookie 属性设置漏洞,为攻击者提高了攻击 Cookie 的机会。

(5)WEB 应用程序提供了不安全的请求方法,允许客户端使用 GET 请求来传送 Cookie。与 POST 请求的方法相比,GET 请求方法容易被操控,引发安全风险。

会话管理的攻击方法主要有 Cookie 篡改、Cookie 溢出、会话劫持、跨站请求伪造等。

1.1　COOKIE 篡改攻击

Cookie 篡改攻击一般采用如下步骤:

(1)Cookie 收集。收集足够数量的 COOKIE 样本。

(2)Cookie 逆向分析。分析会话 ID 生成算法。

(3)Cookie 伪造。当加密的 Cookie 被第三方获取, 并通过密文分析、暴力破解等方式获取了相应的加密密钥, 第三方就可以伪造 Cookie[2],获得访问 WEB 应用程序的权限,实施非法操作。

攻击者一旦从 Cookie 中得到足够多的用户信息,就能够对 Cookie 内容进行篡改。例如:移动电信运营商用户通过互联网发送彩信消息,在身份认证过程结束后,在 Cookie 中包含了发件人的电话号码,这个 Cookie 是服务收费程序用来识别用户的。如果攻击者得到电话号码以明文方式存储的,并且没有任何保护,则可以将 Cookie 中的电话号码改成攻击者的电话号码,那么被攻击者将会为攻击者支付彩信的费用。

1.2　COOKIE 欺骗攻击

由于在 Cookie 中存在用户的敏感信息,所以开发者往往采用 MD5 对 Cookie 内容进行加密处理,即使攻击者获得了 Cookie 也很难破解其中的内容。但在某种情况下,攻击者并不需要知道 Cookie 的明文内容,就可以采用 Cookie 欺骗的方式进行攻击,将截获的 Cookie 提交给服务器,从而假冒他人的身份登录到网站。

实施 Cookie 欺骗攻击的前提条件是服务器的验证程序存在漏洞,并且攻击者能够获得被假冒者的 Cookie 信息。网站的验证程序要验证所有的非法登录是非常困难的,并且编写验证程序的语言也可能存在漏洞[3]。获得他人的 Cookie 信息比较容易,下面是利用 PHP 脚本语言编写的用于收集 Cookie 的程序代码:

```
$info=getnev("QUERY_STRING");
if ($info){
$fp=fopen("info.txt","a");
fwrite($fp,$info."\n");
fclose($fp);
}
```

攻击者把以上代码放到论坛里,并附上让人感兴趣的话题,吸引大家点击浏览,这样就可以收集到大量的 Cookie,攻击者利用这些信息尝试提交给服务器验证,如果服务器验证成功,就可以成功登录到该网站。

1.3　会话固定漏洞攻击

会话固定(Session Fixation)漏洞是指 WEB 应用程序没有废止当前会话 ID,而是继续使用同一个会话 ID 来认证其它用户身份,导致会话劫持攻击。攻击者在 WEB 应用程序上创建一个新的会话并记录相关的会话 ID,当一个用户使用同一个会话 ID 通过了 WEB 应用程序的身份认证后,攻击者就有可能利用当前的活动会话进入该用户账户,假冒该用户非法访问 WEB 应用程序。

1.4　URL 重写

WEB 应用程序支持 URL 重写,将用户的会话 ID 直接放在 URL 中返回给用户,如果用户不小心泄露了该 URL,则会导致会话 ID 泄露和被恶意利用。

例如,一个网站的机票预订程序支持 URL 重写,将用户的会话 ID 直接放在 URL 中返回给用户,即:

http://www.xxxx.com/ex;lsessionid=2POc2JDXMD0OSQXIEMDPSSDNXSHCJKJVMSL? dest=huawei

该网站一个用户通过认证,该用户想通知朋友知道机票打折信息,于是将这个链接发给朋友,但该用户并不知道已经泄露了自己的会话 ID。当该用户的朋友点击该链接时,将会使用该用户的会话登录这个网站,并可以恶意消费该用户的信用卡。

1.5 CSRF 攻击

跨站请求伪造(CSRF)是一种恶意利用网站的攻击方式。从字面来看,跨站请求伪造与跨站脚本有些相似,然而二者是两种不同的攻击方式[4-5]。XSS 攻击的对象是网站用户,而 CSRF 攻击的对象则是网站或者 WEB 应用程序[6]。与 XSS 攻击相比,CSRF 攻击更难防范,比 XSS 攻击更具危险性[7]。

CSRF 攻击主要通过在用户访问的页面包含恶意链接或者脚本的方式来实施。假设一个用户使用 GET 请求来访问一个网站,如果该用户通过了网站 WEB 应用程序身份验证,则下次提交的 GET 请求可以由如下的用户来产生[8]。

(1)由实际使用 WEB 应用程序的用户来产生。

(2)由直接在浏览器输入 URL 的用户来产生。

(3)由使用外部链接访问网站的用户来产生。

由于 WEB 应用程序无法判断由谁产生的 GET 请求,因此攻击者可以利用这一特征来实施跨站请求伪造攻击。攻击者首先将一个执行恶意操作的外部链接通过嵌入在一个电子邮件、图片、或者网站等形式发布出来,引诱用户点击。如果一个通过了 WEB 身份验证的用户点击了该链接,则浏览器就会向 WEB 应用程序发出包含该用户 Cookie 的 GET 请求,由于该用户的 Cookie 是有效的[9],因此 WEB 就会执行该外部链接所规定的操作,导致跨站请求伪造攻击。

1.6 隐私信息泄露

跨站 Cookie 和超级 Cookie 的存在,可能导致用户隐私信息泄露。

1.6.1 跨站 Cookie

由于 Cookie 的敏感性,使 Cookie 具有专属性质,即 A 网站存在 Cookie 中的信息,B 网站是没有权限直接获取的。然而,一些第三方广告联盟的代码使用范围非常广,这就可能通过第三方广告代码形成跨站 Cookie。例如,A 网站和 B 网站都使用了同一家第三方广告代码,如果用户首先在 A 网站搜索了一个“心脏病”关键词,然后再去访问 B 网站,第三方广告代码就可以从 Cookie 中获取到用户在 A 网站的搜索行为,在用户浏览器上就会即刻出现治疗心脏病的广告信息,虽然实现了精准投放广告,但是未经用户同意,对用户的隐私构成了侵犯,造成用户隐私信息泄露[10]。

1.6.2 超级 Cookie

超级 Cookie 存在于某些浏览器中,如 Mozilla、谷歌 Chrome、苹果 IOS 浏览器等,能够在浏览器隐私模式下执行普通会话,导致浏览器隐私模式的失效。IE 浏览器不存在这种问题,因为 IE 浏览器内部并没有提供对特殊操作的记忆功能。例如,用户在浏览器地址栏使用前缀 https://,对某个网站的通信进行加密保护。一些浏览器对此进行记忆,保存一个“超级 Cookie”,当用户下次访问该网站时,浏览器会自动进入 HTTPS 通道,即使用户在浏览器设置了隐私模式,禁用了 Cookie,但是这个超级 Cookie 依然存在。

2 Web 会话管理漏洞检测

会话管理漏洞主要是由于 WEB 应用程序的会话管理缺陷和 Cookie 属性设置不当等原因造成的[11],通常采用动态检测技术来检测会话管理漏洞。检测系统由两部分组成:检测主机和被检测网站。两者之间通过网络连接起来,在检测主机运行检测程序,对被测网站进行测试;在被测网站上运行 WEB 服务器及应用程序,图 1 是会话管理漏洞检测系统模型。

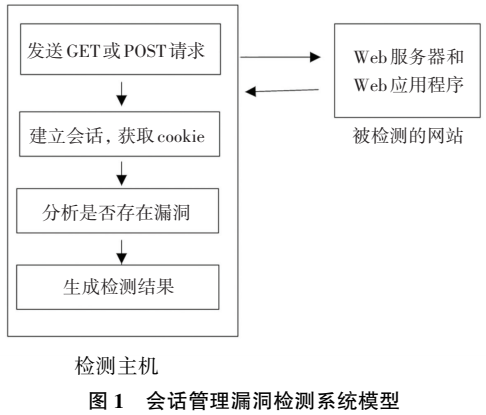

图 1 会话管理漏洞检测系统模型

Fig. 1 Model of session management vulnerability detection system

动态检测方法的一般步骤如下:

(1)Cookie 获取。根据测试项目,检测主机采用 GET 或者 POST 请求方法向被测网站发出 HTTP 请求包。

(2)Cookie 分析。对被测网站返回的 Cookie 信息进行分析,确定是否存在相应的漏洞,给出检测结果。

2.1 COOKIE 属性设置漏洞检测

分析被测网站返回的 Cookie,对 Cookie 属性设置进行逐项检测,检查是否存在 Cookie 属性设置漏洞,即:

(1)Secure 属性。检查是否使用安全方式来传

送含有敏感信息或会话 ID 的 Cookie。例如，当登录 WEB 应用程序后，应用程序在 Cookie 中设置了会话 ID。检测是否设置了 secure 属性，如果没有设置，则存在 Cookie 信息泄露风险。

（2）HTTP Only 属性。检查是否设置了 HTTP Only 属性，如果没有设置，则存在客户端脚本利用该 Cookie 进行攻击的风险。

（3）Domain 属性。检查 Domain 设置情况，应该将其设置为需要接收该 Cookie 的服务器。例如，如果 WEB 应用程序存储在 app.myweb.com 服务器上，则应该设置成"Domain = app.myweb.com"，而不能设置成""，因为这种设置允许其它存在漏洞的服务器接收到 Cookie。

（4）Path 属性。检查 PATH 属性，如果 PATH 是设置在根目录"/"下，则同样允许其它存在漏洞的 WEB 应用程序接收该 Cookie。例如，如果 WEB 应用程序存储在"/myapp/"目录，则 Cookie 路径应设置为"path = /myapp/"，而不能设置为"path = /"或"path = /myapp"。

（5）Expires 属性。检查 Expires 属性情况，如果该属性设置成一个未来的截止日期，则需要确认该 Cookie 不包含任何敏感信息，否则有权读取这个 Cookie 的用户就有可能在截止日期前通过重复提交这个 Cookie 进入 WEB 应用程序。

2.2　会话固定漏洞检测

会话固定漏洞检测方法如下：

（1）向被测试网站（例如：www.xxxx.com）发送 get 请求：

GET www.xxxx.com

（2）被测试网站响应如下信息：

GET / HTTP/1.1
Host：www.xxxx.com
User-Agent：Mozilla/5.0（X11；Linux x86_64；rv：60.0）Gecko/20100101 Firefox/60.0
Accept：
text/html，application/xhtml+xml，application/xml；q = 0.9，*/*；q = 0.8
Accept-Language：en-US，en；q = 0.5
Accept-Encoding：gzip，deflate
Set-Cookie：
JSESSIONID = CD7F8599FDD1BB09FDB2F9DFF3276EA2.jvm_0
keep-alive：timeout = 5，max = 100
Connection：keep-alive

从网站服务器响应的信息可知，WEB 应用程序为客户建立了一个新的会话 ID：JSESSIONID = CD7F8599FDD1BB09FDB2F9DFF3276EA2.jvm_0.

（3）利用 POST 方法向 WEB 服务器提交身份认证请求：

POST /aa/denglu.php HTTP/1.1
Host：www.xxxx.com
User-Agent：Mozilla/5.0（X11；Linux x86_64；rv：60.0）Gecko/20100101 Firefox/60.0
Accept：
text/html，application/xhtml+xml，application/xml；q = 0.9，*/*；q = 0.8
Accept-Language：en-US，en；q = 0.5
Accept-Encoding：gzip，deflate
Referer：http：//www.xxxx.com
Content-Type：
application/x-www-form-urlencoded
Content-Length：53
Cookie：
JSESSIONID = CD7F8599FDD1BB09FDB2F9DFF3276EA2.jvm_0
name = xiaoli&password = xie123456&Submit = %B5%C7+++%C2%BC

（4）提交身份认证后，WEB 服务器响应如下信息：

HTTP/1.1 200 OK
Date：Wed，10 Jul 2019 01：14：21 GMT
Server：Apache/2.4.18（Win32）PHP/5.3.29
X-Powered-By：PHP/5.3.29
location：new.php
Content-Length：8
Connection：close
Content-Type：text/html
…

利用 POST 方法提交身份认证成功之后，从 WEB 响应的信息可知，WEB 服务器并没有重新产生新的会话 ID，因此，存在会话固定漏洞。攻击者利用这种漏洞就可以实施会话劫持攻击，假冒该用户进行各种非法操作。

2.3　CSRF 漏洞检测

CSRF 漏洞检测方案如下：

（1）使用 U 代表被测 URL，例如，U = http = www.mytest.com/action.

（2）建立一个包含 URL 的 HTTP 请求的 HTML

页面,并列出所有相关参数。如果使用 GET 请求方法,则可以直接完成,如果使用 POST 请求方法,则需要通过 JavaScript 脚本来完成。

(3)确保有效的用户能够登陆到 WEB 应用程序。

(4)模拟用户点击到一个指向被测网站的链接。

(5)检查被测网站是否执行了所指定的操作。

2.4 请求方法弱点检测

由于 GET 请求方法容易被操控,存在安全弱点,因此在传输 Cookie 时,最好使用 POST 请求[12]。虽然 POST 请求可以通过 JavaScript 脚本等手段来模拟,但是增加了攻击的难度。

请求方法弱点检测是对使用 POST 请求方法传送数据的 WEB 程序进行检测,检查 WEB 应用程序是否接受通过 GET 请求方法传送的数据。

例如:一个 WEB 登陆产生如下的 POST 请求:

```
POST / denglu.php HTTP/1.1
Host:www.xxxx.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64;
  rv:60.0) Gecko/20100101 Firefox/60.0
Accept:
text/html,application/xhtml+xml,application/
  xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://www.xxxx.com
Content-Type:
application/x-www-form-urlencoded
Content-Length: 53
Cookie:
UM_distinctid=16bd95270ed315-0f756868b
  393a58-396b4645-fa000-16bd95270ef132
name=xiaoli&password=xie123456&SessionID=
  987654321
```

根据以上 POST 请求信息,经过修改后采用 GET 方法提交请求,直接在浏览器地址栏输入如下内容:

http://www.xxxx.com/denglu.php?name=xiaoli&password=xie123456&SessionID=987654321

如果能够成功登陆,说明该 WEB 程序存在安全漏洞,容易被攻击者利用。

3 Web 会话管理漏洞防范

由于 WEB 应用程序的会话管理缺陷,以及 Cookie 属性设置不当等原因导致产生会话漏洞[13]。因此需要从正确使用会话管理功能,合理设置 Cookie 属性等方面提高会话管理的安全性,防止攻击者利用会话管理漏洞实施攻击。可以采用如下防范措施:

(1)Web 应用程序和客户端应避免使用非加密信道传送 Cookie,防止攻击者通过网络监听来获取 Cookie,导致 Cookie 内容泄露[14]。

(2)应避免使用缺乏足够随机性的会话 ID 生成算法,防止攻击者通过逆向分析来伪造 Cookie。

(3)避免使用固定的会话,防止攻击者利用固定会话漏洞实施会话劫持攻击。

(4)正确设置 Cookie 属性对 Cookie 进行保护,防止攻击者利用 Cookie 属性设置漏洞实施 Cookie 攻击。

(5)避免使用 GET 请求方法来传送 Cookie,防止攻击者利用 GET 请求来实施攻击。

(6)避免使用持久性 Cookie,尽量使用一次性 Cookie,使 Cookie 只在当前活动会话中有效,并设置合理的 Cookie 有效期,浏览器应当及时删除过期的 Cookie。

(7)不要在 URL 错误信息或者日志中暴露会话 ID ,会话 ID 应当只出现在 HTTP Cookie 头信息中,不要以 GET 参数来传递会话 ID。

(8)通过在每个请求或者每个会话中使用强制随机令牌或者参数,为敏感或者关键的操作提供标准的会话管理。

(9)在身份认证时,如果连接从 HTTP 变为 HTTPS,则应当生成一个新的会话 ID。在 WEB 应用程序中,推荐持续使用 HTTPS 而并非在 HTTP 和 HTTPS 之间切换使用。

(10)禁止连续的登录并强制执行周期性的会话终止。即使是活动的会话,特别是对于支持网络连接或者连接到关键系统的 WEB 应用程序,终止时间应当根据业务需求做适当的调整。

(11)将 Cookie 设置为 HTTP only 属性。除非在 WEB 应用程序中明确要求客户端脚本程序读取或者设置 Cookie 的值。

4 结束语

本文介绍了 WEB 会话在实际应用中的作用,详细分析了会话管理漏洞产生的原因以及利用会话漏洞实施攻击的各种手段;提出了由于不同原因造

(下转第 331 页)