

基于 AOP 的角色访问控制模型设计与实现

尹涛,李翔,林祥,魏诚

(上海交通大学信息安全工程学院,上海 200240)

摘要:随着信息技术的不断发展,用户可访问的信息资源越来越复杂,越来越难以管理。当前,在 Web 信息系统开发中,角色访问控制已经成为了一个研究热点。介绍了角色访问控制(RBAC)、AOP 以及 AspectJ 的相关知识;根据 RBAC 的基本理论,以笔者曾参与开发的一个 Web 信息系统为原型,给出了一个基于角色-表单模型的设计,并采用 AOP 技术,给出了相应的实现方案。

关键词:基于角色的访问控制;面向方面编程;AspectJ

中图分类号: TP393

文献标识码: A

文章编号: 1673-629X(2008)10-0136-03

Design and Implementation of Role Access Control Model Based on AOP

YIN Tao, LI Xiang, LIN Xiang, WEI Cheng

(School of Information Security Engineering, Shanghai Jiaotong University, Shanghai 200240, China)

Abstract: With the development of information technology, the information resources that users can access are more and more complicated, more difficult to manage. At present, role-based access control based on Web has become a hotspot. Introduces the role-based access control (RBAC), the AOP and AspectJ. According to the basic theory of RBAC, to a Web information system that author involved in the development as an example, gives a role-form access control module design and implementation methods based on AOP.

Key words: RBAC; AOP; AspectJ

0 引言

当前,随着信息技术的不断发展,对于资源的安全访问与管理已经成为了一个急需解决的问题。访问控制正是基于通过对主体进行相应的权限设计,让主体在一定的权限设计框架内对客体信息资源进行访问的一种方法,它依赖于鉴别使主体合法化,并将组成员关系和特权与主体联系起来。只有经授权的用户,才允许访问特定的系统资源^[1]。对于现在所有开发的 Web 信息系统来说,没有访问控制的功能将是不可想象的。

访问控制的实现方法有很多种,传统的访问控制方法主要有自主型访问控制和强制型访问控制,这两种访问控制方法虽然解决了 Web 系统的安全问题,但也会使系统的安全管理更为复杂,管理开销增大。角色访问控制理论的提出,为访问控制提供了一种良好

的解决方案。角色访问控制能够较好地减少权限管理复杂性,并能够很好地提供与企业组织结构相一致的安全策略。

AOP 技术是一种新型的软件设计方法,它通过将访问控制、日志记录和事务管理等非系统功能模块从主功能模块中分离出来,使软件具有更好的松散耦合性、适应性和可维护性,是当前软件开发中的发展方向和趋势。文中根据 RBAC 的基本模型,给出了一个基于角色-表单的访问控制模型的设计,并采用了 AOP 技术,给出了实现方案。

1 基础知识概述

1.1 角色访问控制(RBAC)

基于角色的访问控制是美国 NIST(National Institute of Standards and Technology)于 20 世纪 90 年代初提出的一种新的访问控制技术。该技术主要研究将用户划分成与其在组织结构体系相一致的角色,以减少授权管理的复杂性,降低管理开销和为管理员提供一个比较好的实现复杂安全政策的环境^[1]。

RBAC 主要由用户集(user)、角色集(role)、权限集

收稿日期:2008-01-25

基金项目:国家自然科学基金项目(60502032);上海市科委项目(065115020)

作者简介:尹涛(1979-),男,湖北天门人,硕士研究生,研究方向为互联网内容安全;李翔,副教授,研究方向为网络内容安全。

(permission)及会话集(session)等实体组成,用户与角色之间、角色与权限之间都是多对多的关系,采用 RBAC 的最大好处在于将用户和它所具有的权限分离开来,可以将用户的授权和权限的划分进行分别处理。图 1 为 RBAC 的基本模型图。

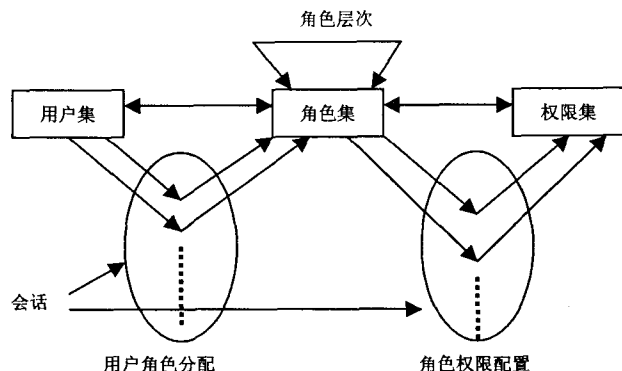


图 1 RBAC 的基本模型图

现阶段, RBAC 主要有两种实现级别:一种是操作系统级别,一种是应用程序级别。当前,由于 B/S 架构的流行,针对 Web 信息系统的角色访问控制已经成为了一个研究热点。文中主要讨论基于应用程序级别的 Web 信息系统中的角色访问控制模型的设计与实现。

1.2 面向方面编程(AOP)及 AspectJ

在传统的编程方法中,常常将软件的业务逻辑作为软件设计的主线,即核心关注点(也称问题领域关注)^[2],但是,在软件中常常还包含许多系统级的关注点(也称解题领域关注)^[2],比如系统安全性、日志记录、事务完整性等。按照面向对象的编程思想,很难将核心关注点与系统级关注点完全分离,从而产生了实现多种关注的代码相互穿插,程序因缺乏抽象性、封装性而导致程序难以理解、难以维护。通常将这类混合与散乱于多个模块的系统关注点,称为横切(crosscutting)关注点。

AOP 是一种能有效管理、分离横切关注点的程序设计方法学,是面向对象编程思想的一种有效补充。AOP 通过引入方面(aspect)这种新的模块单元来实现横切关注点。在设计和实现系统时,将横切关注模块织入(weaving)与其相关的核心关注模块中,从而实现了横切关注模块与核心关注模块的有效分离,使开发人员能够轻松地构造松散耦合的软件系统^[3]。

AspectJ 是 AOP 在 Java 语言上的一种实现,是最成熟与最具代表性的 AOP 语言工具^[4]。AspectJ 是 Java 语言的无缝扩展,这使得所有符合规范的 Java 虚拟机都可以解释、执行其所生成的代码。同时,Aspect

还提供了与流行 IDE 的集成,因此,对于 Java 开发者而言,Aspect 是一个很有用的 AOP 实现。

2 基于 AOP 的角色 - 表单访问控制模型的实现

2.1 基于角色 - 表单的角色访问控制模型

近年来,B/S 架构已经成为信息系统的主流,而由于在 RBAC 模型中,用户与角色、角色与权限之间多对多关系的存在, RBAC 模型在 Web 应用中的复杂性也越来越明显。一般基于 Web 的信息系统体系结构分为三层:第一层为页面表示层,第二层为业务逻辑层,第三层为数据层。在三层架构中,与用户直接交互的就是页面表示层,而用户主要是通过页面中的表单(Form)或按钮来与服务器进行交互,因此,一个系统中访问控制的最小单位应该是表单或按钮,基于此,以笔者曾参与开发的一个 Web 信息系统为原型,提出了一种基于表单的角色访问控制模型(见图 2)。

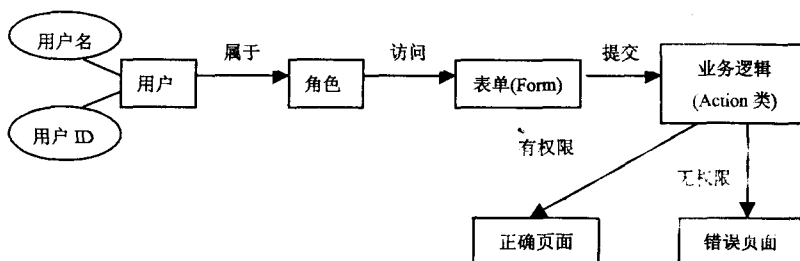


图 2 基于表单的角色访问控制模型

在该系统中,笔者采用了 struts 框架和 hibernate 框架, struts 框架主要应用于业务逻辑层, hibernate 主要应用于数据层,在 struts 中,主要由 Action 类来控制业务逻辑, Action 类将用户输入到表单中的数据提交给 Action 类,由 Action 类来判断用户权限并执行相关业务逻辑。在图中,根据 Web 应用的需求形成了用户 - 角色 - 表单的关系,以实现角色访问控制。这种模型相对于一般所应用的基于角色 - 页面的角色访问控制模型来说,更具有灵活性,能够实现细粒度的访问控制。

2.2 数据库设计

根据以上基于表单的角色访问控制模型,在实际的系统开发中,建立了相应的角色 - 用户 - 权限的数据表单,详细说明如下:

用户表(user):保存用户 ID、用户名及密码等字段;

角色表(role):保存角色 ID、角色名称等字段;

用户角色分配表(userRole):用于保存用户的角色信息,主要有用户 ID、角色 ID、用户名、角色名称等字段。

权限表(permission):用于保存系统中相关的表单功能信息,与后台中的表单处理功能相对应。

角色权限配置表(rolePermission):保存不同角色对应的不同权限信息,主要有角色 ID、角色名称、是否有访问权限等字段信息。

2.3 具体实现

在传统的 OOP 实现方法中,往往会将所有的角色权限检查等相关方法都放在 Action 类中,致使所有的访问控制相关代码都散乱地分布在响应的业务类中,而且在各个业务类中都必须加入相关的权限检查代码,导致程序可重用性低,调试及维护代码都变得十分繁杂,不利于软件开发。针对于此,在该 Web 信息系统中,将访问控制模块作为一个横切关注点,通过引入 aspect,将角色访问控制相关代码从相关的功能模块中抽取出来,单独形成一个 aspect,然后用 AspectJ 的编译器将访问控制模块织入主功能模块中,有效地分离了主要功能模块和访问控制模块,取得了较好的效果。

一般来说,安全相关的模块或者说作为横切关注点的模块总是放在最后完成的,当完成了主程序功能之后,可以选择在 IDE 中将普通的 Web 工程转换为 AspectJ 工程,这样 AspectJ 程序就能在编译时被织入应用程序,并得到应用。

下面以对程序中的一个模块的角色访问控制的实现为例来展现如何利用 AspectJ 来实现对应用程序的角色访问控制。首先介绍几个主要的类:

LoginUsrForm:用户登陆系统时,该类获取用户输入的用户名及密码等相关信息;

LoginUsrAction:执行相关的业务逻辑;

LoginUsrAspect:定义一个方面,扩展 LoginUsrAction 类,加入 LoginPermission()方法,判断用户是否为系统的合法用户;加入 FormPermission()方法,根据用户-角色-权限表关系,判断是否用户有访问表单的权限;声明一个连接点,当用户访问登录表单时,通知该方面进行权限检查;

FunctionOneAction:执行相关的业务逻辑;

FunctionOneAspect:定义一个方面,声明一个连接点,当用户访问 FunctionOneForm 表单时,通知该方面进行权限检查。

在 AspectJ 中,可以使用静态横切技术,以静态方式使用 AspectJ 中的方面把方法和属性引入现有的类中,相当于在现有的类中加入了新的方法和属性。

在 LoginUsrAspect 中,首先扩展 LoginUsrAction 类,为其增加方法 LoginPermission(String name, String password)(根据用户名和密码判断用户是否有合法登陆系统的权限)和方法 permission(String role)(根据用

户所属角色拥有的权限来发还给用户相对应的页面)。代码片段如下:

```
Public Boolean LoginUsrAction. LoginPermission (String name,
String password)
{
LoginUsrDAO logindao = new LoginUsrDAO();
Permission = logindao. bool(name,password);
Return permission;
}

public Boolean LoginUsrAction. permission(String role)
{
Permission = rolePermissionDAO. getPermissionByRole(role);
return permission;
}
```

然后,在 FunctionOneAspect 中声明一个连接点(joinpoint),当执行 FunctionOneAction 的 execute()方法时,通知该方面进行权限检查。以下是代码片段:

```
//当执行 execute()方法时,调用该 aspect
public pointcut PermissionCheck():execution(ActionForward FunctionOneAction. execute());
ActionForward around():PermissionCheck()
{
Boolean permission = loginUsrAction. permission(role);
if (permission){
//返回原始连接点继续执行;在 FunctionOneAction 类中,默认有权限访问该页面
return proceed(mapping,form,request,response);
}else
{
//用户无权限,调用 failure.jsp
Return mapping. findForward("failure");
}
}
```

从以上分析可以看出,利用面向方面编程可以将角色访问控制模块与主功能模块完全分离,但在一个 Web 信息系统中,页面表单的数量常常很大,如果表单过多,容易造成管理和维护上的困难,基于此,AspectJ 提供了强大的连接点捕获功能,能够通过利用通配符以及布尔表达式来实现在一个方面中一次捕获多个连接点。比如,要捕获在一个 package 中的所有以 Action 结尾的类的 execute()方法,可以作如下声明^[5]:

pointcut callPointcut():call(void * Action.execute())

通过这种方法,可以把类似的访问控制代码都放在一个方面中,极大地提高了程序的可重用性。

总的来说,通过应用 AspectJ 进行角色访问控制的实现,能够减少代码的繁杂性,提高软件代码的可重用性及可维护性。同时,如果当安全策略需要变动时,只

(下转第 142 页)

阶次具有很高的敏感性,在不知道确切的分数阶微积分运算阶次的情况下,无法提取出嵌入的水印信息。

该仿真实例中,水印提取端与水印嵌入端使用的分数阶微积分运算阶次差值仅为 $v_1 - v_2 = 0.000\ 01$ 。如果运算阶次的差值 $v_1 - v_2$ 进一步地减小,那么在水印嵌入过程中,应该选择伪随机序列 $x(n)$ 的起始 n 值更大的一段取值范围,以保证水印提取端无法恢复水印信息。理论上对于这种敏感性的精度为无限小,因为无论 $(v_1 - v_2)$ 多么小,总可以找到足够大的一段 n 值范围,使其满足两组序列的差值足够大,从而无法提取出水印。

5 结束语

提出了基于分数阶微积分运算数字水印系统,该种水印系统为不可见、私有水印实现方案。并采用计算机进行数值仿真实验,证明了水印系统是可行的,而且是有效的。

分数阶微积分运算对于正弦信号进行处理,通过离散化操作,得到离散伪随机序列 $x(n)$,该伪随机序列对于分数阶微积分运算的阶次具有初始值敏感性。表现在水印系统中,对于两个非常接近的分数阶微积分运算阶次,可以找到合适的一段 n 值取值范围,使产生的两个伪随机序列差值尽可能大,从而在接收端无法恢复出原始水印信息。因此,通信双方可以利用分数阶微积分运算阶次作为保密密钥,进行保密通信。仿真结果表明,在分数阶微积分运算阶次未知的情况下,将导致接收端无法正确恢复原始信号,不能重建水

印图像,从而达到了信息掩盖的目的。

而在传统的水印嵌入与提取算法中,大部分用到的伪随机序列是 $(0,1)$ 随机数,该算法虽然也有一定的初始值敏感性,但很难达到 10^{-5} 以上的精确度,而如果利用文中所说的分数阶微积分的方法,只要适当选取伪随机序列的取值范围,理论上该精确度可以达到无限小,几乎没有非法破解的可能性。

如何进一步地对基于分数阶微积分运算的水印系统进行完善,以及设计鲁棒性能更好的水印嵌入与提取算法,将是下一步需要研究的课题。

参考文献:

- [1] van Schyndel R G, Irkl A Z, Osborne C F. A digital watermark[C]//Proc of IEEE International Conference on Image Processing. Austin: IEEE Press, 1994: 86 - 90.
- [2] Bender W, Gruhl D, Morimoto N. Techniques for data hiding [C]//Proc of the SPIE Conference on storage and retrieval for image and video database. San Jose: SPIE Press, 1995: 164 - 173.
- [3] Kleinz M, Osler T J. A child's garden of fractional derivatives [J]. The College Mathematics Journal, 2000(31): 82 - 88.
- [4] 赵元英,袁晓,滕旭东,等. 常用周期信号的分数阶微积分[J]. 四川大学学报:工程科学版, 2004(36): 94 - 97.
- [5] 王剑,林福宗. MATLAB 在数字水印技术研究中的应用[J]. 计算机工程与应用, 2003(11): 156 - 158.
- [6] Trancevski K, Tomovski Z. On some fractional derivatives of functions of exponential type[J]. Univ. Beograd. Publ. Elektrotehn. Fak. Ser. Mat., 2002(13): 77 - 84.

(上接第 138 页)

需要改变 aspect 相关代码,而不必改动核心关注模块代码;这种独立的访问控制策略,使软件具有更好的复用性。另外,由于访问控制模块与主功能模块代码完全分离,使软件具有更好的可读性,提高了软件的质量。

3 结束语

面向方面编程思想的提出,是为了降低软件的复杂性,更好地分离核心关注模块和横切关注模块,可以说,软件系统的访问控制模块是一个相当理想的面向方面编程的实例。文中基于 AspectJ 构建了一个 Web 信息系统的角色访问控制模块,该模块已经应用于实践,取得了较好的效果。相信随着 AOP 技术的不断发展,会有越来越多的软件系统利用面向方面的编程思

想来进行开发、测试,以构建更好的松散耦合的软件系统。

参考文献:

- [1] 叶锡君,许勇,吴国新. 基于角色的访问控制在 Web 中的实现技术[J]. 计算机工程, 2002, 28(1): 167 - 169.
- [2] 邓阿群,厉小军,俞欢军,等. 一种新型软件设计方法 AOP 的研究[J]. 系统工程与电子技术, 2004, 26(7): 970 - 975.
- [3] 李志纯,张南平. 面向 Aspect 编程的应用研究[J]. 计算机技术与发展, 2006, 16(5): 217 - 222.
- [4] Kiczales G, Hilsdale E, Hugunin J, et al. An Overview of AspectJ[C]//In Proceedings of the European Conference on Object - Oriented Programming. Hungary: Springer - Verlag, 2001.
- [5] Miles R. AspectJ cookbook 中文版[M]. 北京:清华大学出版社, 2006.