文章编号:1001 - 9081(2006)12 - 2945 - 03

RBAC 模型研究、改进与实现

李志英[1],黄 强[2],楼新远[1],冉 鸣[3]
(1. 西南交通大学 信息科学与技术学院,四川 成都 610031;
2. 四川农业大学 信息工程与技术学院,四川 雅安 625014;
3. 四川师范大学 计算机软件重点实验室, 四川 成都 610068)
(xjlzy619@163.com)

摘 要:针对传统 RBAC 模型的角色管理过于复杂,权限粒度不够细化等不足,提出了一种改进模型,并详细描述了该改进模型的特点,引入了限制元素和特有权限等新概念。通过将功能模块及其相关操作映射为权限数据,使得权限代码和业务逻辑代码完全解耦。最后结合实例,详细阐述了该改进模型应用于权限管理系统的实现过程,以及该模型的优缺点。

关键词:基于角色的访问控制;限制元素;MVC
中图分类号: TP309 **文献标识码**:A

Research, improvement and implementation of RBAC model

LI Zhi-ying[1], HUANG Qiang[2], LOU Xin-yuan[1], RAN Ming[2]
(1. *School of Information Science and Technology, Southwest Jiao tong University, Chengdu Sichuan* 610031, *China*;
2. *College of Information and Engineering Technology, Sichuan Agricultural University, Yaan Sichuan* 625014, *China*;
3. *Computer Software Laboratory, Sichuan Normal University, Chengdu Sichuan* 610068, *China*)

Abstract: An improved model was proposed for the deficiencies of traditional RBAC model in role-management such as over complexity and rough granularity. The characteristics of this model were described in detail, in which permission constraint elements and special permissions were introduced. Acting as permission data, other functional modules and their related operations can make the couple of the permission codes and operation logics eliminated. As a conclusion, implementation of this improved model in system of permission management was described in detail, also including its advantages and disadvantages.

Key words: Role-based Access Control (RBAC); constraint element; MVC

基于角色的访问控制(Role-based Access Control,RBAC)是实施企业权限控制的一种有效的访问控制方法,既弥补了自主式访问控制安全性方面的不足,同时也解决了强制式访问的灵活性不够的问题。但是它也存在一些问题,如其角色管理过于复杂,权限粒度不够细化等,本文针对这些不足提出了改进模型。

1 RBAC 模型改进

1.1 RBAC96 模型简介

RBAC96 模型包括了 RBAC0、RBAC1、RBAC2、RBAC3 四个模型。

RBAC0 包含四个基本要素:用户、角色、会话和访问权限。用户在一次会话中激活所属角色的一个子集,获得一组访问权限,即可对相关客体执行规定的操作,任何非显示授予的权限都是被禁止的。

RBAC1 是对 RBAC0 的扩充,在 RBAC0 的基础上加入了角色层次关系,根据组织内部权力和责任的结构来构造角色与角色之间的层次关系。

RBAC2 也是 RBAC0 的补充,但与 RBAC1 不同,RBAC2 加进了约束的概念。RBAC2 中的约束规则主要有:最小权限、互斥角色、基数约束与角色容量、先决条件、等级间的互斥角色等。

而 RBAC3 模型是对 RBAC1 和 RBAC2 的集成,它不仅包括角色的层次关系,还包括约束关系。RBAC96 基本结构如图 1 所示。

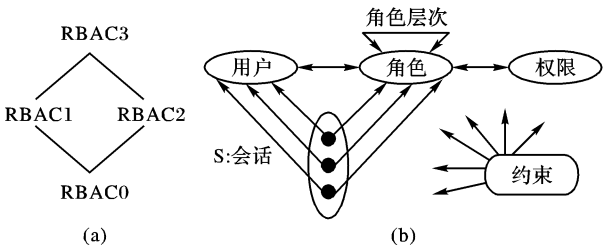

图 1 RBAC 96 模型

RBAC96 模型为 RBAC 思想的具体实现提供了一个基本框架,具有非常重要的作用。它具有以下优点:便于授权管理,便于根据工作需要分级,便于赋予最小特权,便于任务分担,便于文件分级管理,便于大规模实现等。但也存在一些问题:例如角色管理过于复杂,角色继承机制不完善,没有具体权限分配方法等。许多研究者对 RBAC96 模型进行了改进,如 RBAC97 模型在 RBAC 中加入管理员角色,将管理和使用分开。EHRBAC 提出一种改进的角色层次化关系模型。该模

收稿日期:2006 - 06 - 28;**修订日期**:2006 - 08 - 28

作者简介:李志英(1979 -),女,河南鹤壁人,硕士研究生,主要研究方向:软件工程、网络数据库; 黄强(1981 -),男,四川成都人,讲师,主要研究方向:软件工程、虚拟机、人工智能、神经网络; 楼新远(1963 -),男,四川成都人,副教授,主要研究方向:网络数据库、图形图像处理、软件工程; 冉鸣(1962 -),男,四川成都人,教授,主要研究方向:原子分子学研究、化学 CAI.

型遵循按现实世界模型建模的思想,定义了角色的公共权限和私有权限,引入一般继承和扩展继承机制,形成了功能更加完善和更易于扩充的角色层次化管理模型,很好地解决了使用私有角色所出现的问题。基于任务的工作流访问控制模型,对任务分配相应的权限,角色执行任务,权限则传递给角色,实现了权限的按需分配。RBAC 模型逐渐被完善,但仍有一些不足之处:

1)把角色指派给用户需要多步骤。角色级别越高,授权步骤越多,这可能加重安全管理员的工作量,同时也会造成数据的冗余。

2)权限的细粒度控制不够。对权限的定义无论是指众所周知的存取操作,或者被后来研究者提出的特殊操作权限,如签名密钥的使用权限(只能用于签名)等,都是定义到模块级别,但是对数据级别却显得软弱,比如对一些敏感数据需要只能对表中部分行数据,或者部分列数据根据不同级别的用户进行操作,对这些细粒度要求比较高的系统来说传统模型就显得无力。

为了解决这些问题,本文提出一个改进的角色控制模型,并且基于新兴的 AOP 技术支持来实现。

1.2 改进模型

1) 基本定义:

用户集合 $U = \{U1, U2, U3, \cdots\}$;

角色集合 $R = \{R1, R2, R3, \cdots\}$;

组集合 $G = (G1, G2, G3, \cdots)$;

权限集合 $P = (P1, P2, P3, \cdots)$;

权限限制元素集合 C, 该限制与 RBAC2 中的约束条件不同,RBAC2 中强调基数约束,角色动态/静态职责分离,以及角色、用户及权限之间的对应、并、交、差关系等,而该权限限制元素是对权限集合的补充,作为权限控制粗粒度的细化,用以实现对数据非常敏感的系统;

AXB:表示 A 与 B 之间是多对多的关系;

会话: 用户是一个静态的概念, 会话则是一个动态的概念。一次会话是用户的一个活跃进程, 它代表用户与系统进行交互.

粒度: RBAC 将访问控制概括地分为粗粒度访问控制和细粒度访问控制两个控制级别。粗粒度的访问控制过程仅考虑客体的类别, 不区分客体特定的实例。细粒度的访问控制过程在考虑客体的类别之后, 还要考虑客体、用户、操作等特定实例的细节。

2) 模型结构

对于传统模型中细粒度控制不够的问题,在权限部分做了补充,引入了权限限制元素 C;

对于传统模型中为用户指派角色复杂的问题,引入了组的概念,组可以映射为业务系统中的业务部门,比如化学教研组,在模型中可以定义为一个或多个角色的集合。它可以拥有多个角色,但是对于不是拥有一个角色所有权限的问题,比如拥有角色中的部分权限,引入特有权限的概念,特有权限也是权限原子,直接分给组或用户,而不通过角色来分配, 这样大大简化了以往只能通过角色继承或者复杂的角色交差等运算来实现的机制。用户、组、角色和权限之间是多对多的关系:

1) 对于用户:一个用户可以属于多个组,也可以拥有多个角色,并且可以拥有多个特有权限,同时对它拥有的某个权限可以具有多个限制元素。

U——UXG——UXR——UXP——UPXC

2) 对于角色:一个角色可以被多个用户拥有,也可以被多个组拥有,并且它可以拥有多个权限,同时对它拥有的某个权限可以具有多个限制元素。

R——RXU——RXG——RXP——RPXC

3) 对于组:一个组可以拥有多个用户,也可以拥有多个角色,并且可以拥有多个特有权限,同时对它拥有的某个权限可以具有多个限制元素。

G——GXU——GXR——GXP——GPXC

4) 对于权限:一个权限可以被多个用户、多个角色以及多个组所拥有。

P——PXU——PXG——PXR

该改进模型映射为五层结构,如图 2 所示。

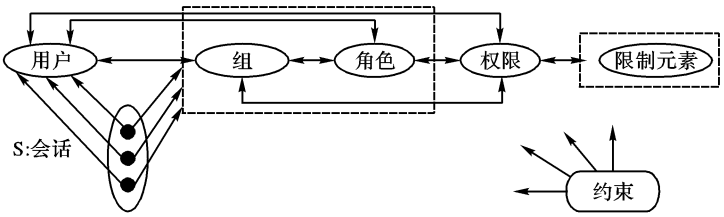

图 2 改进模型结构

由 2 图可见,改进模型在吸取了传统 RBAC 的许多优点以外,也针对其不足之处做了改进,比如在以往角色管理机制过于复杂方面引入了组和特有权限大大简化了角色继承以及角色间并、交、差等运算;同时在权限细化部分做了改进。为了满足一些对权限控制非常细的业务需求,模型中增加了限制元素。

2 RBAC 改进模型实现

2.1 相关技术简介

本文把改进模型应用于项目(交互式教学系统)的权限管理系统中,采用的框架是 FrameServer[8] 的 MVC 框架,它以 ControlServlet 核心作为控制器,它负责拦截所有客户请求,根据用户请求的命令和用户类型(Web,J2ME,GPRS……)在类型配置文件中查找相应的请求处理器以及配置信息。然后将用户请求统一交给请求处理器处理。请求处理器对处理进行拦截,调用 Action 进行请求处理,Action 调用业务处理模块进行业务逻辑处理,最后获得逻辑处理模块的数据模型并根据不同的终端类型组织结果数据并将结果返回给客户端。

2.2 建立实体关系模型

把理论模型转化为实际应用,首先需要建立对应的实体关系模型结果,如图 3 所示。

由图 3 可见,安全模型已经由实体关系模型进行了物理结构描述,下面具体说明权限管理基于 FrameServer 框架的实现过程。

2.3 实体关系模型基于框架的应用

传统的做法是把判断权限代码直接插入到具体业务代码前面,如果具有权限则访问业务逻辑,否则跳出该模块,这样在所有的功能模块中都要重复插入权限逻辑,使得代码冗余,并且会出现我们经常所说的代码混编的情况。而本文则将权限逻辑作为一个独立的模块抽取出来进行统一管理。

以任课老师查询本班学生的期末考试成绩模块为例,其权限逻辑的处理过程如下:

在基于 MVC 的 FrameServer 框架中,Action 模块是基于命令模式实现,其配置文件 mx-config. xml 如下:

```
…
< action name = "scoreManager" >
  <! -- >
  < interceptor > org. javawing. security. interceptor.
  SecurityInterceptor </interceptor >
```

```
< mxprocess > org. javawing. action. ScoreManagerAction </ mxprocess >
< exception global-ref = "error"/ >
...
</action >
...
```

一个功能模块对应一个这样的 action 配置,参照上面配置访问过程如图 4 所示。

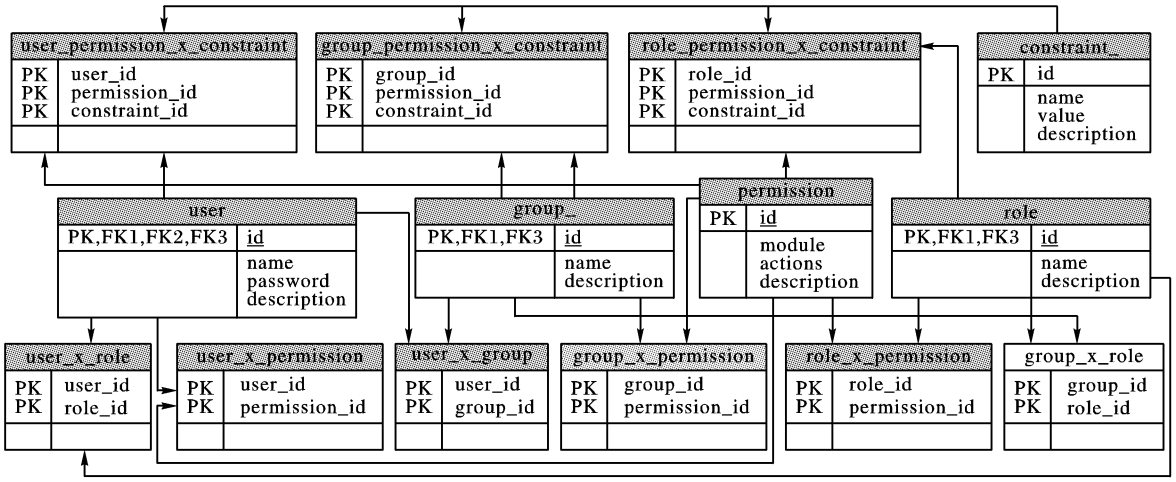

图 3　实体关系模型

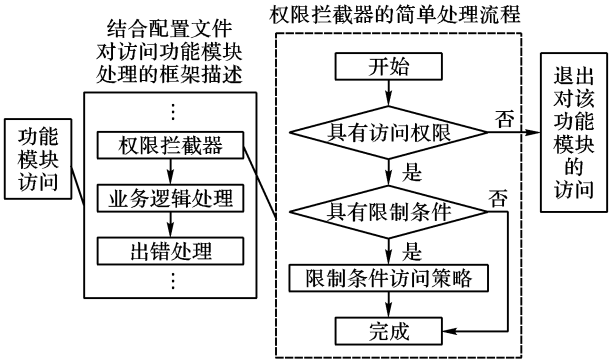

图 4　访问功能模块处理流程

结合实体模型图,其详细的处理流程如下:

1）用户登录。如果通过身份验证,则加载该用户所具有的组、角色、特有权限域模型并放入缓存。

2）访问学生成绩管理模块。请求处理器对处理进行拦截,调用 Action,对应 mx-config. xml 中 name 为 scoreManager 的 action,进行查询操作,即调用类 ScoreManagerAction 中的 query 方法。对应实体模型中的权限表 permission 中的 module 和 actions 字段,其对应数据为 scoreManager 和 query。这两个字段可以共同确定一个具体的权限,即对成绩管理模块的查询权限。

3）与步骤 1 中加载的域模型对比是否具有该权限,如果存在则转步骤 4,否则退出对该功能模块的访问。

4）根据用户登录 id 和步骤 2 对应的权限 id 查找是否具有限制条件(对应实体模型中的 user_permission_x_constraint 表),如果存在限制转步骤 5,否则转步骤 6。

5）加载限制策略。在本例中,该用户有本单位和本年度两个限制条件,对应实体业务成绩表中就是表中的一块(部分行,部分列)数据,本单位与具体用户密切相关,本年度与用户的权限级别有关。则本文的改进模型中限制元素就是为这种数据级别的操作所引入的。根据步骤 4 中查找到的限制 id,查找具体的限制条件。对应实体模型中的 constraint_。在表中有两个关键的字段 name 和 value,name 指的是限制属性,value 指限制属性值,正如在改进模型中介绍的一个用户对一个权限可以有多个限制,本实例有两个限制数据如表 1 所示。

6）对功能模块进行处理,即查询本班本年度的学生成绩。

7）显示查询结果。

上面处理流程可以看出,步骤 1 是用户登录模块的处理,即身份验证;步骤 2、3、4、5 则是访问成绩管理模块的权限控制策略,6 是对业务逻辑的调用。

表 1　限制数据表

Id	name	Value
1	class	${class}
2	time	${year}

3　结语

对 RBAC 模型进行了扩展,并将它用于权限管理中。权限管理像其他模块一样,抽取成一个独立的模块处理,把其他功能模块标记作为权限数据,使得权限代码和业务逻辑代码完全解耦,复用性和自动控制能力也大大提高。与其他功能模块不同的是,权限逻辑作为系统级模块在整个业务系统中作为拦截器而存在。当然采用该方案也有一些缺点,比如角色与权限、用户与权限、组与权限等之间的关系模型与域模型的转换过程会导致性能降低,但是其影响在目前的计算机硬件环境中可以忽略不计。

参考文献:

[1] SANDHU RS, COYNE EJ, FEINSTEIN HL, *et al.* Role-based access control models[J]. IEEE Computer, 1996; 29(2): 38 - 47.

[2] 陈波, 于冷, 肖军模, 等. 计算机系统安全原理与技术[M]. 北京: 机械工业出版社, 2006.

[3] 余文森, 张正秋, 章志明, 等. 基于角色的访问控制模型中私有权限问题的研究[J]. 计算机应用研究, 2004, 21(4).

[4] 钟华, 冯玉琳, 姜洪安. 扩充角色层次关系模型及应用[J]. 软件学报, 2000, 11(6): 779 - 784.

[5] 冯德民, 王小明, 赵宗涛. 一种扩展角色存取控制模型[J]. 计算机工程与应用, 2003, 39(3): 87 - 89.

[6] FILMAN RE. 面向方面的软件开发[M]. 莫倩, 等译. 北京: 机械工业出版社, 2006.

[7] 雅各布森, 黄邦伟, 徐锋. AOSD 中文版——基于用例的面向方面软件开发[M]. 北京: 电子工业出版社, 2005.

[8] 黄强, 蒋茂. FrameServer 框架设计说明书[Z]. 2004.