DOI:10.16208/j.issn1000-7024.2013.02.030

基于角色的细粒度访问控制模型的设计与实现

赵卫东，毕晓清，卢新明

（山东科技大学 信息科学与工程学院，山东 青岛 266590）

摘 要：为有效解决目前权限管理中存在的授权复杂、不灵活等问题，提出了基于角色的细粒度访问控制模型。改进基于角色的访问控制模型，在其中引入属性约束和用户组，从访问控制的粒度出发分析并设计出基于角色的细粒度模型。新模型有效减少了角色和权限的授予数量，降低了权限管理复杂性，确保了细粒度的访问控制。通过将新模型具体运用到基于 J2EE 的 Web 系统并以用户权限树和存储过程实现，表明了新模型的可行性、灵活性和高效性。

关键词：细粒度；基于角色的访问控制；属性约束；用户权限树；存储过程

中图法分类号：TP311 **文献标识号**：A **文章编号**：1000-7024（2013）02-0474-06

Design and implemention of fine-grained RBAC model

ZHAO Wei-dong，BI Xiao-qing，LU Xin-ming

（School of Information Science and Engineering，Shandong University of Science and Technology，Qingdao 266590，China）

Abstract：To effectively solve the problems as complex and inflexible authorize for authorization management，a model of fine-grained role based access control is presented. Adding attribute-constraint and group module to RBAC model from the grain of access control and forward the fine-grained RBAC model through detailed analysis and design. The new model effectively decreases the number of roles and authorities，reduces complexity for authorization management and ensures fine-grained access control. Finally，the validity，flexibility and efficiency of presented model are demonstrated by the application of a J2EE based web system with the accomplishment of user-authority-tree and stored procedure.

Key words：fine-grained；RBAC；attribute-constraint；user-authority-tree；stored procedure

0 引 言

早期传统的访问控制技术如自主访问控制、强制访问控制等技术虽在一定时期内确保了基于 Web 的管理信息系统（management information system）权限管理（包括系统级安全管理如操作系统和数据库安全和应用级安全管理如应用程序和 Web 系统安全[1]）的安全，但其所能操作的用户、资源数量有限；基于角色的访问控制作为目前常用的访问控制技术虽具有较灵活的授权机制，但其无法解决组织结构复杂、权限变动频繁的管理信息系统。总之，不同的访问控制模型各有优缺点并且都无法完全确保灵活、高效的权限管理。本文通过分析多种模型，改进基于角色的访问控制模型，在该模型基础上增加属性约束和用户组，提出基于角色的细粒度访问控制模型。通过对细粒度模型进行分析和设计，将其进行具体应用并通过基于 SSH2 框架的 Web 系统实现。新模型确保了系统高效的权限访问控制和灵活的权限管理。

1 传统访问控制模型分析

传统的访问控制模型有基于对象的访问控制 OBAC、基于任务的访问控制 TBAC 和基于角色的访问控制 RBAC 模型。其中 OBAC 以用户为基本对象进行权限授予，授权过程虽然简单却存在对相同权限用户的重复授权及重复数据冗余等问题。自主访问控制 DAC 作为 OBAC 的一种，通过访问控制表控制主体对客体的直接访问，但由于访问存在传递性，系统无法控制主体对客体的间接访问，由此引发授权混乱。另一种基于对象的强制访问控制 MAC 通过为主体和客体设定安全级来保障信息的单向流动，虽能确保访问的高度安全但灵活性较差且工作量大；TBAC 从任务角度建立模型，角色通过任务和权限关联，实现动态实时的安全控制，却存在任务分配复杂等问题；RBAC 作为目前应用最广泛的模型，通过创建角色，分别为角色分配不同操

收稿日期：2012-04-11；修订日期：2012-06-16

作者简介：赵卫东（1967-），男，山东泰安人，博士研究生，副教授，研究方向为软件工程、数据库；毕晓清（1986-），男，山东临沂人，硕士研究生，研究方向为软件工程；卢新明（1961-），男，河南济源人，博士，教授，研究方向为数字矿山。

E-mail：bixiaoqingvip@163.com

作权限，然后为用户指定一个或多个角色，使用户通过角色间接地获得操作权限。该模型通过引入角色减少了权限授予工作量，并对组织变化具有灵活的伸缩性，但却存在着如下问题:

(1) 权限粒度约束不够细化。粗粒度的权限(尤其是功能权限) 控制导致用户权限树过宽却不深，树的层次不分明。

(2) 权限授予过程复杂。例如对于某高校软件系中 N 名教授不同课程的教师，他们具有相同的 S 种功能权限，管理 M 个不同的数据对象(如课程/学生信息)，RBAC 将分别为 N 教师分配角色并授予相应 S 种功能权限和对应 M 个数据对象，由此进行 M × S × N 次授权。这对于人员众多、部门设置复杂且同级部门中的用户具有相同功能权限的高校，会造成大量的重复授权并导致角色和权限的冗余，增加授权的复杂性。

(3) 功能权限和数据权限未实现完全分离。RBAC 中对数据对象的操作包含在权限管理中，数据对象的访问和管理较乱，无法准确获取角色不同功能权限对应的不同数据对象。如对于具有授课教师 A 角色的某用户，具有查看和修改学生信息的功能权限，但无法确保他能查看某学院 1 班和 2 班学生的信息，却只能修改 1 班学生的信息。

(4) 具体应用复杂，效率低下。对于登录验证、访问数据库等操作存在重复性，且客户端不断向服务器提交重复的 T-SQL 语句，由此增加服务器负担，降低服务器处理效率。

针对 RBAC 存在的问题，唐金鹏等人[3] 提出面向用户属性的 RBAC 模型，增加属性集合和约束，通过用户属性实现动态的分配角色。然而此方法无法确保通过多个属性组合得到的角色间不存在冗余，不会引起角色分配混乱等问题; 赵静[3] 等人提出的基于数据对象的 RBAC 模型，实现了数据对象和功能权限的分离。但此模型以数据对象为主，通过 UD、RD 和 PA 三次才完成权限分配，授权过程复杂且当数据对象数量过多时依次为数据对象分配角色的工作量巨大，极易出错。对于数据对象变动较大的单位部门，每增加或删除一个数据对象可能牵扯多个角色，由此可能引发 RD 混乱，并删除或增加很多角色，造成角色冗余。

2　基于角色的细粒度访问控制模型设计

2.1　设计思想

本文从访问控制的粒度出发，提出基于角色的细粒度模型解决 RBAC 存在的上述问题:

(1) 通过细粒度权限管理(即将权限分解为原子链接或按钮等可访问集合并按此粒度管理)，按尽量小的粒度将功能权限分解为增加/修改/删除/打印等原子操作，并将权限以字符串形式存储到数据库，最后以用户权限树叶子节点的形式展现。

(2) 引入用户组，将具有相同角色(功能权限相同，操作数据对象不同) 的用户集合成组，通过为用户组直接分配角色，避免权限/角色的重复授予和冗余并保留角色间的继承关系，允许直接为用户指定角色。

(3) 分离功能权限和客体对象，从功能权限和其对应数据对象两方面入手，授权时先分配功能权限，然后分配其所能操作的数据对象。通过编写包含属性约束的存储过程，到数据库中获取用户不同的功能权限所对应的不同数据对象，并转换获取到的结果加载到前台显示; 当用户数据对象改变时，通过权限管理树更新权限数据表中的数据对象范围即可。

(4) 封装常用控件成复合控件以完成基本登陆验证功能; 编写存储过程集合封装用户对任务的重复操作避免每次写重复访问语句，由此提高批量语句的执行速度并能避免越权访问。

2.2　模型设计

细粒度模型确保了权限访问控制的细粒度实现，提高了动态访问控制的安全性。作为基于角色的通用模型 RBAC96[4] 的扩展，其模型图如图 1 所示。

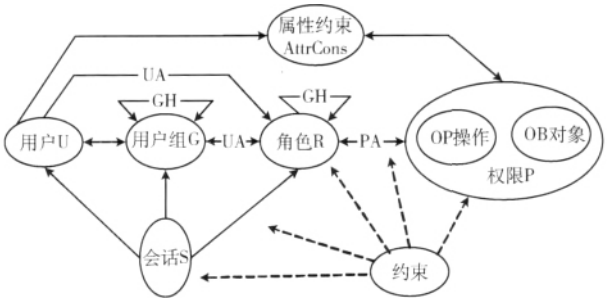

图 1　基于角色的细粒度访问控制模型

模型中各元素的定义和描述如下:

(1) 用户 U、用户组 G、角色 R、权限 P、会话 S 分别表示用户集、用户组集、角色集、权限集和会话集; OP 操作表示功能权限集 add/delete 等，OB 对象表示客体对象集(客体对象作为系统保护的资源如数据库中数据、内存中的片段等。本文中的客体对象为数据对象)。

(2) UA ((G × R) ∪ (U × R) 表示为用户分配角色。一个用户组可被分配多个角色，一个角色可被分配给多个用户组，对具有独立访问权限的部分用户直接为其分配角色; GH (G × G，表示用户组间的层次关系; RH (R × R 则表示角色间的层次关系。授权关系图如图 2 所示。

(3) PA⊆P × R，表示角色的权限分配。一个角色可拥有多个权限，一个权限可属于多个角色; 其中 P = OP × OB，OP 是管理员为角色分配的操作权限，OB 是管理员为角色中的功能权限单独分配的对应数据对象(通过增加 check-

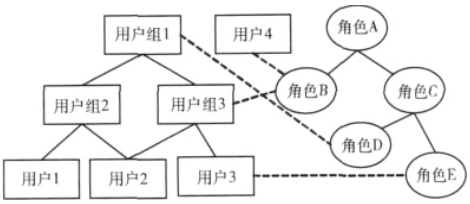

图 2 授权关系

box 属性的用户权限树实现)。

(4) user(Si): S→(U∪G),表示会话与用户的映射。用户 User(Si)创建的会话 S 中包含用户激活的角色集合(一个用户可同时拥有多个会话)。

(5) roles(Si): $S \rightarrow 2^R$,表示会话与角色集的映射。roles(Si)({r|(MYMr' ≥r) 〖(user(Si),r) ∈UA〗}是会话 Si 对应的角色集合;会话 Si 对应权限为: $U_{r \in roles(Si)}$ {p|(MYMr' ≥r) 〖(p,r) ∈PA〗},其中用户具有权限为会话集中所有权限并集。

(6) 约束:同 RBAC2 中的约束,包括互斥角色约束(即不能在一次会话中为用户组/用户分配互斥的角色)、基数限制(即限定授予给用户组/用户的角色数量)、先决条件约束(即仅当具有某种操作许可时才可分配角色)及时间、频度约束等。

(7) 属性约束:表示约束条件。以用户属性、功能权限等作为约束条件,将其封装到包含众多 T-SQL 语句的存储过程中,通过调用该存储过程(具体介绍如 3.3)获取不同功能权限操作的不同数据对象。

3 基于角色的细粒度访问控制的实现与应用

基于角色的细粒度访问控制模型具有较强的通用性,能确保高效灵活的访问控制。其模型类图[5]如图 3 所示。

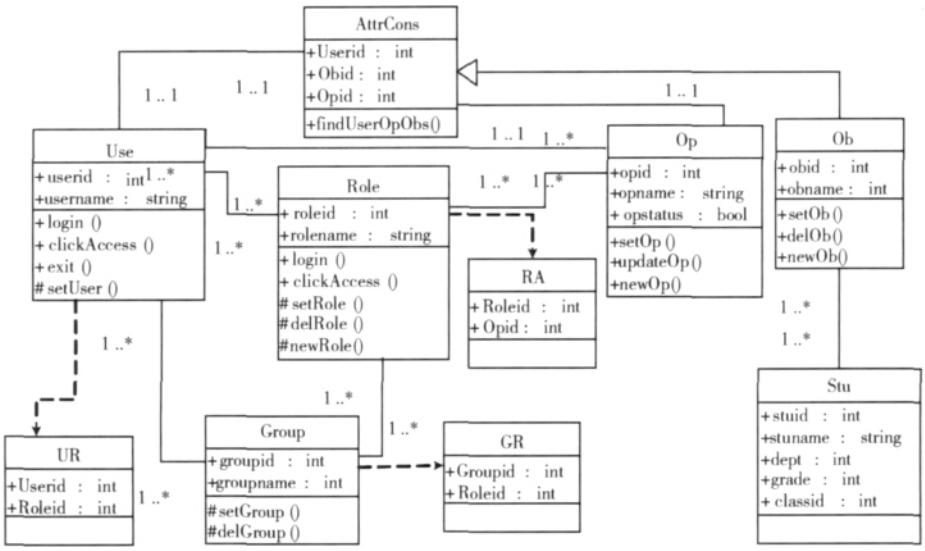

图 3 细粒度模型的类图

类图中的 User 对应数据库中的用户表,Group 对应用户组表,Role 对应角色表,Op 对应功能权限表,Ob 对应数据对象表,Stu 对应学生表;UR 表示用户角色的授予表,GR 表示用户组角色的授予表,RA 表示角色权限的授权表,AttrCons 表存储用户所有功能权限及其所能操作的数据对象,通过 Userid 和 OPid 来确定 OBid。最后分解获取到的 OBid 并在数据库中的表中查找符合 OBid 属性的数据信息返回到前台[6]。

该模型在具体的应用中包括登录验证、权限访问控制和权限管理三部分。用户登陆进行身份验证,然后系统为合法用户加载初级用户权限树,用户点击初级加载树中的叶子节点来访问对应的功能权限,最后调用存储过程获取功能权限对应的数据对象并返回给前台界面显示。具体如下:

3.1 登录验证

登录验证能确保用户的合法登录。封装常用验证控件[7]成复合控件 checkUserForm,该控件扩展 FormPanel,能确保用户输入验证及合法身份验证。控件的构造如下:

```
Ext. namespace ("Ext. ux. form"); //为新控件 checkUserFormt 创建命名空间:
Ext. ux. form. userCheckForm = Ext. extend(Ext. form. userCheckForm,{ //扩展 Form 控件
… // 复合控件中所包含的控件及其属性与方法
url 'check. Action'; //提交用户输入的信息到 check. Action 验证
});
Ext. reg('userCheckForm', Ext. ux. Checkform); //完成该组件类型的注册
```

通过在 Ext. onReady（function（）{...}）；中增加{xtype：'userCheckForm'}完成复合控件的调用。复合将获取到的用户信息提交的具体的 url 进行用户身份验证，具体实现过程如图 5 所示。

3.2 权限的访问控制

3.2.1 功能权限的访问控制

权限的访问控制能确保用户访问指定的权限资源，避免越权访问。通过用户权限树（Extjs 中的异步树组件）实现为不同用户加载显示不同的树节点，从而确保安全的访问控制。该权限树的实现包括初始权限树的加载和权限树的级联加载：

（1）初始权限树的加载：为保证用户不花长时间等待用户权限树的数据加载和首次渲染，设计权限树先向用户加载包含一级节点（根的子节点）的初始权限树。用户点击一级树节点，通过层层的级联加载直至访问叶子节点。此方式与一次性加载整棵权限树相比不但省时而且确保页面不会频繁刷新，提高了用户访问体验。用户通过登录验证后，系统以获取到的用户 Userid 作为参数，获取该用户对应的角色并查找角色所对应的功能权限，最后获取根节点的一级子节点返回前台显示，完成初始权限树的加载。

（2）权限树级联加载：用户点击初始权限树中的节点，层级的加载其对应子节点直至叶子节点，点击叶子节点对功能权限的访问。在层级加载过程中，需要获取到所有功能权限对应的叶子节点到根节点的权限路径，在加载节点的过程中，通过判断用户点击所节点的加载路径是否在权限路径中来确定是否加载相应子节点。其中获取权限路径的递归算法 findNodePath（List < resource >）；实现的思想为：查找用户权限集 listResource 中每个叶子权限节点到根节点的路径（所经过的所有 resource 节点的 id），并将路径保存到数组 a 中，由此得到权限加载路径数组。算法内容如下：

```
For（Resource r： re）{// 依次对每个权限/Resource 进行处理
    for（int j = 0； j < a. length-1； j + +）{//指定数组初始长度：a [200]
    a [j] = r. getResourceId（）； //获取 Resource 的节点 id
    a [j + 1] = r. getParentId（）； //获取该节点的父节点的 id
    r = resourceDao. getReourceById（r. getParentId（））； //查找该父节点的父节点并判断其是否存在
    if（r = = null）{//不存则说明为根节点，跳出这个循环
    j + +； //若存在，则继续循环直至找到根节点
    break；}}}
```

当用户点击权限树中的节点时，系统调用 resourceDao. findByParentIdAndIds（id，a）；方法查询该节点是否具在合法的访问路径中，若有则返回该节点的路径加载信息继续加载；若无则提示用户不具有继续操作的权利。其中 id 为通过树的 depTreeLoader. on（'beforeload'，function（depTreeLoader，node））；方法获取用户点击的树节点的 id。路径检查方法如下：

```
public List < Resource > findByParentIdAndIds（int parentId，int ids []）{//查找点击节点的子节点
    String paramsInt = "（"； //将用户权限数组 a 转换成字符串形式，方便比对
    for（int i = 0； i < ids. length-1； i + +）{
        paramsInt = paramsInt + ids [i] + ",";}
        paramsInt = paramsInt + ids [ids. length-1] + "）";
        String queryString = " from Resource r where r. parentId = " + parentId + " and r. resourceId in" + paramsInt； //查询以 parentId 为父节点的该用户所具有的子节点的字符串，防止越界访问
        List < Resource > listResource = getHibernateTemplate（）. find（queryString）；
        return listResource；}
```

（3）用户权限树的显示：由于 Extjs 通过 Json 数据与后台异步交互，故用户权限树只接受格式为：[{text:"TreeName"，id："""，parentId："0"，leaf：true}]}]（id 为树节点，text 为树结点内容，leaf 指节点是否为叶子，parentId 是父节点）的 Json 数组才能确保树形结构的显示。由于 Resource 的属性格式与 JSON 形式不符，由此增加 Tree 表，将 Resource 表中的属性列与树所需的 Json 数组对象一致，将 listResource 依次转换成 Tree 表的标准 Json 格式，然后将结果存放到 struts. xml 中配置的 Json 类型的参数 menuList 中自动加载给前台异步树显示。

由此完成了权限树的级联加载。按照此方法，用户可按照自己的需要显示相应功能权限，无需每次登陆都完全展示整棵权限树，确保高效的异步访问控制。

3.2.2 数据对象的访问控制

用户权限树确保了用户功能权限的安全访问控制。按照访问控制的细粒度，不同的功能权限对应不同的数据对象。由此整合 T-SQL 语句编写存储过程并将其存储到数据库中，然后调用存储过程获取角色不同的功能权限所对应的数据对象。

存储过程封装了用户对任务的重复操作，通过选择性调用此预先编译的存储过程，不必每次写重复语句，提高批量语句的执行速度。在编写过程中，需要确定存储过程的输入和输出参数并完善主体内容。其中输入参数包括用户名 Username（通过它获取功能权限操作对应的数据对象），权限树的节点编号 treenode（通过它获取用户选择的功能权限），数组 DataObjects（权限数据表获取的数据对

象）。例如通过存储过程 findUserObjects 查找具有辅导员角色的用户所能编辑的学生信息:

```
Use RuanjianShijianPingtai //存放用户表、角色表等数据信息的数据库
Create Procedure findUserObjects @ Username char (8), @ treenode char (8), @ Data Objects List Output
AS
BEGIN
Select obj from AttrCons u where u. username = @ Username and u. fun = @ treenode
```

在调用时，执行语句 EXEC pro findUserObjects @ ' zhangsan', @ ' update', @ Datalist OUTPUT 即可。当返回形式如｛D200901，F201203｝的数据对象信息时，表示具有辅导员角色的该用户能够编辑的学生对象为 D200901（信息学院 2009 级 1 班）的学生和 F201203（外语学院 2012 级的 3 班）的学生信息。

然而，存储过程中所查询出来的功能权限对应的数据都是对象的形式，要输出该数据对象信息，需要以其属性为条件到数据库中查询符合该条件的数据表中的数据信息，并将其返回到前台 Grid 数据面板显示。对如 D200901 的数据对象，按以下属性进行分解: D 代表学院如信息学院、2009 代表年级、01 代表班级。根据学院/年级等属性，到 stu 表中读取符合条件的数据并加载显示。通过调用存储过程获取数据与通过查表方法所得到的结果是一致的，但是此方法大大提高了执行效率。

3.3 权限的授予和管理

在用户权限树中设定 Checkbox 属性为 true，构造出用户权限管理树。通过选中或取消用户权限管理树中节点前的复选框，能方便的实现角色权限的授予、权限数据对象的指定以及用户组的分配等。其中角色的权限授予如图 4 所示。

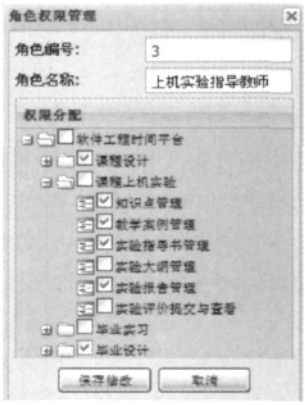

图 4 用户权限管理树

选中一级节点前的复选框则其所有子节点被选中。选择完成后点击保存修改按钮，修改信息会自动保存并映射到数据库中的 AttrCons/RA/GR/PA/UR/RA 表中，下次加载用户权限树的时候会读取更新的表中的信息以实现权限树的更新加载，确保角色/权限的灵活授权和管理。

3.4 基于角色的细粒度模型的实现

选取基于 B/S 的软件工程实践平台实现细粒度模型。该 Web 系统采用 Java 语言，选取 myeclipse 集成环境、mysql 数据库等进行开发。通过搭建 SSH2 框架并整合 Extjs 技术[8]，将系统按照 MVC 设计思想分为显示层、业务逻辑层、数据访问层等，各层间调用完全面向接口，实现了插件式编程，降低耦合度。系统权限访问控制实现过程的顺序图[9-10]如图 5 所示。

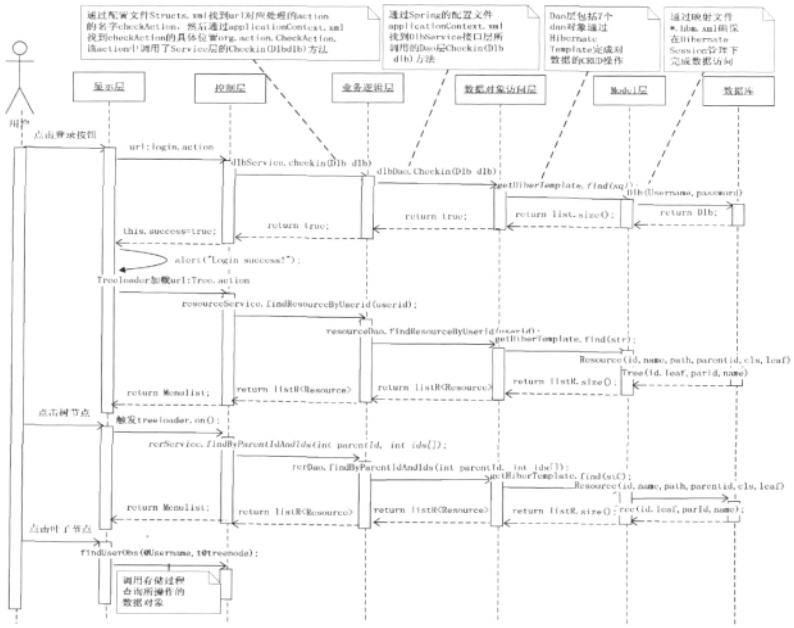

图 5 权限访问控制实现顺序

系统最终实现结果如图 6 所示。

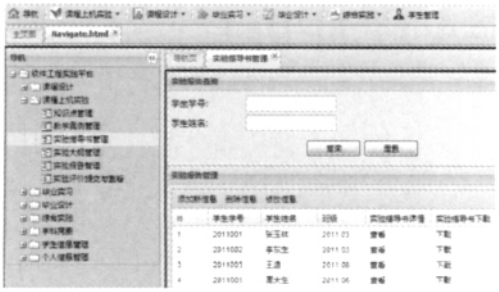

图 6 系统的实现

4 结束语

本文通过分析、比较各种常用的访问控制技术，选取基于角色的访问控制模型，在模型中增加属性约束和用户组所构建的基于角色的细粒度访问控制模型有效解决了目前管理信息系统的权限授予和管理过程中存在的授权复杂性、角色冗余性问题，减少了权限授予的数量，提高了权限授予的效率并确保了准确、灵活的权限访问控制。然而，此细粒度模型在解决模型冲突、角色继承关系的私有权限等问题上仍存在缺陷，需通过深入的分析、研究加以改进和完善。

参考文献:

[1] WU Jiangdong，LI Weihua，AN Xifeng. Method of finely granular access control based on RBAC [J]. Computer Engineering，2008，34 (20)：52-54 (in Chinese). [吴江栋，李伟华，安喜锋. 基于 RBAC 的细粒度访问控制方法 [J]. 计算机工程，2008，34 (20)：52-54.]

[2] ZHAO Jing，YANG Rui，JIANG Luansheng. RBAC permission access control model based on data objects [J]. Computer Engineering and Design，2010，31 (15)：3353-3355 (in Chinese). [赵静，杨蕊，姜滦生. 基于数据对象的 RBAC 权限访问控制模型 [J]. 计算机工程与设计，2010，31 (15)：3353-3355.]

[3] TANG Jinpeng，LI Linglin，YANG Luming. User attributes oriented RBAC model [J]. Computer Engineering and Design，2010，31 (10)：2184-2186 (in Chinese). [唐金鹏，李玲琳，杨路明. 面向用户属性的 RBAC 模型 [J]. 计算机工程与设计，2010，31 (10)：2184-2186.]

[4] SHEN Lei，ZHANG Yuan，JIANG Ping. Design and implementation of rights management based on RBAC in Web [J]. Computer Era，2010 (10)：49-51 (in Chinese). [沈磊，张媛，蒋平. 基于 RBAC 的权限管理在 Web 中的设计与实现 [J]. 计算机时代，2010 (10)：49-51.]

[5] LIU Hongbo，LUO Rui，WANG Yongbin. Competence system based on RBAC design and implementation [J]. Computer Technology and Development，2009，19 (9)：154-156 (in Chinese). [刘宏波，罗锐，王永斌. 一种采用 RBAC 模型的权限体系设计 [J]. 计算机技术与发展，2009，19 (9)：154-156.]

[6] DU Hongyan，PAN Yi，HUANG Caixia，OU Xinliang. Application of OC-RBAC model in MIS [J]. Huazhong Univ of Sci & Tech (Natural Science Edition)，2009，37 (9)：53-55 (in Chinese). [杜红艳，潘怡，黄彩霞，等. OC-RBAC 模型在管理信息系统中的应用 [J]. 华中科技大学学报 (自然科学版)，2009，37 (9)：53-55.]

[7] FAN Minghu，FAN Hong，WU Xiaojin. Competence system based on RBAC design and implementation [J]. Computer Engineering，2010，364 (1)：143-145 (in Chinese). [范明虎，樊红，伍孝金. Universal Privilege Management System Based on RBAC in ASP.net [J]. 计算机工程，2010，364 (1)：143-145.]

[8] LI Tianming，HE Yueshun. Research on privilege management based on ExtJS technology and SSH framework [J]. Computer Applications and Software，2011，28 (5)：165-167 (in Chinese). [李天鸣，何月顺. 基于 Extjs 技术与 SSH 框架的权限管理研究 [J]. 计算机应用与软件，2011，28 (5)：165-167.]

[9] SONG Yun，LI Feng. Design and implementation of access control under B/S system based on RBAC [J]. Software Guide，2009，8 (7)：28-30 (in Chinese). [宋云，李峰. 基于 RBAC 的 B/S 系统访问控制设计与实现 [J]. 软件导刊，2009，8 (7)：28-30.]

[10] LIAO Junguo，HONG Fan，XIAO Haijun，et al. Authorization management research based on extjs and SSH framework [J]. Computer Engineering and Applications，2007，43 (34)：138-140 (in Chinese). [廖俊国，洪帆，肖海军，等. 细粒度的基于角色的访问控制模型 [J]. 计算机工程与应用，2007，43 (34)：138-140.]