Computer Science and Application 计算机科学与应用, 2014, 4, 307-313
Published Online December 2014 in Hans. http://www.hanspub.org/journal/csa
http://dx.doi.org/10.12677/csa.2014.412042

Hans汉斯

Privilege Management Research and Practice under B/S Structure

Xin He[1*], Xiaowei Wang[2], Chuncheng Yang[2]

[1]School of Economics and Management, Beijing Institute of Petrochemical Technology, Beijing
[2]Jidong Oilfield Energy Company, Tangshan
Email: *gongkong@126.com

Received: Oct. 22nd, 2014; revised: Nov. 24th, 2014; accepted: Dec. 2nd, 2014

Abstract

In this B/S structure of the application system, privilege management used broad. According to the different requirements, privilege management functions have different ways of design and implementation, and RBAC (Role Based Access Control) provides a common way to achieve guidance, RBAC based on the uniqueness of the next B/S URL environmental resources, the use of access control algorithm and role-core to achieve the user's permission to perform operations managed by the role to perform operations.

Keywords

RBAC, Privilege Management, Privilege Control Logic, URL, B/S

B/S结构权限管理的研究与实现

何　信[1*]，王小伟[2]，杨春成[2]

[1]北京石油化工学院经管学院，北京
[2]冀东油田能源公司，唐山
Email: *gongkong@126.com

收稿日期：2014年10月22日；修回日期：2014年11月24日；录用日期：2014年12月2日

*通讯作者。

摘　要

在B/S结构的应用系统中，权限管理功能有着广泛的应用。由于需求的不同权限管理功能有不同的设计和实现方式，RBAC (基于角色的访问控制)作为一种比较成熟的模型，为权限管理的设计和实现方式提供了参考。基于Web环境下资源标示的唯一性和权限控制算法，并参考RBAC和学习多种权限管理的实现，总结出一种简单实用的权限管理实现方式。

关键词

RBAC，权限管理，权限控制算法，URL，B/S

1. 引言

权限管理在软件系统中有着广泛的应用，从操作系统到企业网站，都能发现权限的模块或功能。常用的权限管理主要有以下几个方面的作用：保护系统的安全，确保系统的健壮性；保护数据的安全，防止数据被非法修改；提供资源的使用权，使一部分人有权使用某一部分资源；记录用户的操作情况等。可见权限管理功能设计的成功与否直接影响系统的安全性。

权限管理在不同形式的系统中有着不同的设计和实现方式，通常在系统中可能包括但不限于如下功能，系统登录、用户管理、权限分配、日志等。B/S结构的应用系统有着特定的资源访问型式，即所需访问的资源或操作都是使用浏览器访问指定的URL来实现[1]-[4]。

2. 文献综述

在实际项目中，往往存在多个系统，每个用户访问各个系统需要统一的账号口令。各个系统之间都要为用户提供统一的授权，保持用户信息的一致性。由于用户管理和授权管理的复杂性，美国国家标准与技术研究院(National Institute of Standards and Technology, NIST)提出了RBAC (Role-Based Access Control，基于角色的访问控制)模型[5] [6]。基于角色访问控制的基本思想就是：根据企业或单位内部不同的职务划分成与之相匹配的角色，权限不是直接授予用户主体，而是授予角色。再为每一位用户分配相应的角色。这样角色实体实现了连接用户实体和权限实体的中介。既减少了授权管理的复杂性，又实现了访问控制的一致性。目前RBAC的经典模型RBAC96模型包括RBAC0、RBAC1、RBAC2、RBAC3四个模型，RBAC0模型定义了用户、角色、权限，RBAC1将角色划分不同的层次，实现角色之间的继承关系，RBAC2增加了对角色、权限等的限制，RBAC3包含了前三个所有模型[7]-[9]。

3. 设计思想

RBAC模型主要有3个实体构造关联关系：用户(User)、角色(Role)和权限(Privilege)，用户与角色是多对多的关系，角色与权限也是多对多的关系；如图1所示。

访问控制主要包括角色管理、权限管理、角色授权、权限授权、用户权限验证。角色管理是对角色的增删改，角色授权是为不同的用户赋予不同的角色，权限管理是对权限的增删改，权限授权是对不同的角色授予不同的权限。用户权限验证是对用户主体的所有权限操作的验证，用户主体通过角色中介来查找自己的系统资源的访问权限[10]。

在系统使用过程中，系统中的资源都以URL的方式向外界展现。默认状态下，所有的资源或操作是可以被URL访问的。权限管理就是使一部分人拥有一部分资源的使用权限，当用户使用浏览器访问特定

Figure 1. The RBAC entity relationship
图 1. RBAC 实体关系

网址时，系统根据用户信息或传入的参数值确定如何处理数据、返回哪些数据等。

在实际应用中会遇到相同角色不同用户使用不同的权限的需求，在这种情况下仅仅通过粗粒度的角色划分无法实现系统资源的细粒度访问控制的需求，可以模拟现实中的个岗位管理制。机构和角色形成岗位，模块与操作结合形成权限[11]-[14]。

4. 实现原理

B/S 结构的系统，业务逻辑的处理和数据的操作都是在服务器端执行的。权限管理可以作如下处理，服务器端接收到浏览器的访问，在业务逻辑执行前，通过过滤该 URL，并读取当前用户的访问权限决定执行或不执行该操作并返回不同的视图。

对于浏览器端的用户，获得不同的视图展现。在互联网中，每个资源的地址都是公开的，不同的就是根据用户的不同返回不同的视图。权限控制即在服务端构造不同的 URL 访问视图，并将该视图返回。

如何区分访问同一个 URL 的不同用户呢？用户与 URL 之间是一个多对多的关系，建立关系数据表：用户、角色、操作(URL)，该关系类似于 RBAC 模型的实体关系，如图 2 所示。

对于角色和权限的设计通常有两种方式，一种是把不同的角色划分等级，高等级的角色拥有低等级角色的所有权限；例如高级管理员拥有管理员的所有权限，管理员拥有普通用户的所有权限。另一种是所有的角色是平等的；例如，超级管理员与管理员平等，超级管理员不一定能执行管理员可执行的操作。由于 URL 资源的唯一性的特性，把不同的可访问资源视为平等的；把各种角色视为平等的要比把角色规划为不同等级的更简单有效率。

用户管理，即用户的增加、删除、修改等操作，例如在增加用户时，选择用户具有系统管理员的角色；权限分配，是指把 URL 的访问权限分配给特定的角色；系统登录，是指用户使用系统用户登录系统，并在系统 session 中记录用户信息；日志，是指系统记录已登录用户的操作。

权限处理流程，如图 3 所示。

权限控制算法，对于每一种角色设定一个角色编码。编码值依次为 2^0，2^1，2^2，2^3，…；如果用户属于某种角色，则在“角色值”中累加该种角色的“编码”。例如：角色 A、B、C、D 分别对应的编码值为 2^2、2^3、2^4、2^5，某用户属于(或说拥有) A、B、C 三种角色，则该用户的角色值是 $2^2 + 2^3 + 2^4 = 28$；当判断该用户是否拥有 B 角色时，使用二进制进行“与”运算，如图 4 所示。即 U&B == B，那么该用户是否拥有 D 角色呢？如图 5 所示。

即，使用用户的“权限值”与角色“编码”取“与”运算，运算结果等于该“编码”说明当前用户属于该角色。对于操作使用同样的方式处理，在操作中设置“角色值”属性，属性值是可执行此操作的角色“编码”的加和。

用户执行操作前，获取当前用户的全部角色和可执行该操作的全部角色，对比两个角色的集合，如果有交集说明有执行权限，反之无执行权限。

5. 实现方式

对应于数据库实体关系模型，在数据库中建立如下数据表结构，如表 1~3 所示。

Figure 2. The database entity relation
图 2. 数据库实体关系

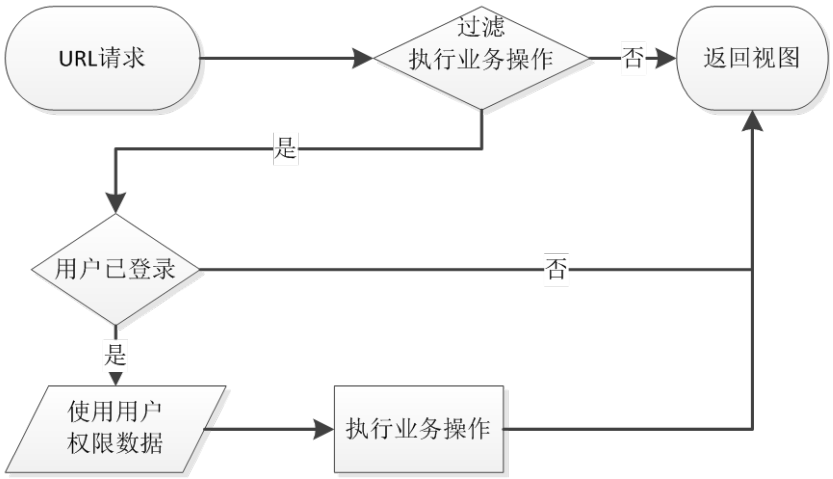

Figure 3. Permission process
图 3. 权限处理流程

```
  11100
&01000
------------------
   01000 == 8(十进制)  ==  2^3
```

Figure 4. U&B == B
图 4. U&B == B

```
  011100
&100000
------------------
   00000 == 0(十进制)  !=  2^5
```

Figure 5. U&D! = B
图 5. U&D! = B

Table 1. User table
表 1. 用户表

列名	类型	非空	自增	Flags	说明
id	INT(10)	√	√	UNSIGNED	主键
name	VARCHAR(45)	√		BINARY	用户名
password	VARCHAR(45)	√		BINARY	密码
email	VARCHAR(90)	√		BINARY	邮箱
role_sum	BIGINT(20)	√		UNSIGNED	角色编码的和

Table 2. Role table
表 2. 角色表

列名	类型	非空	自增	Flags	说明
id	INT(10)	√	√	UNSIGNED	主键
name	VARCHAR(45)	√		BINARY	角色名称
number	INT(10)	√		UNSIGNED	角色编码
description	VARCHAR(36)	√		BINARY	角色描述

Table 3. Authorization table
表 3. 权限表

列名	类型	非空	自增	Flags	说明
id	INT(10)	√	√	UNSIGNED	主键
url	VARCHAR(180)	√		BINARY	URL 片段
description	VARCHAR(360)	√		BINARY	操作描述
role_sum	BIGINT(20)	√		UNSIGNED	角色编码的和

如数据库表中所示，“用户表”与“操作表”中都有“角色编码的和”的属性。当过滤器拦截到 URL 请求时，在数据库“操作表”中查找匹配该 URL 的记录，获取到全部的可操作角色。根据当前用户的登录状态查找当前用户所具有的角色，根据“用户角色列表”匹配“可操作角色列表”确认是返回“数据视图”或“无操作权限视图”。请求流程如图 6 所示。

6. 程序示例

该程序示例使用Java实现，省略数据库访问等代码，只用于描述处理流程和配置。

1) 程序代码

```
public class PrivilegeFilter extends HttpServlet implements Filter {
@Override
public void doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain chain) throws IOException, ServletException {
HttpServletRequest request = (HttpServletRequest) servletRequest;
HttpServletResponse response= (HttpServletResponse) servletResponse;
HttpSession session = request.getSession(false);
String appPath = request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+request.getContextPath();
    String actionPath = request.getRequestURL().toString().replace(appPath, "");
    Integer userId = Integer.valueOf(session.getAttribute("userId").toString());
//根据 actionPath 查询角色列表，省略数据库访问代码
List roleA = getRoleList(actionPath);
                ...
//根据 userId 查询角色列表，省略数据库访问代码
List roleB = getRoleList(userId);
                ...
```

```
//验证两个 list 是否存在交集
boolean cross = isCrossing(roleA,roleB);
if(cross){
    chain.doFilter(request, response);
}else{
//导向无操作权限页面
}
}
@Override
public void init(FilterConfig config) throws ServletException {
    }
}
```

2) web.xml 配置

```
<filter>
        <filter-name>privilegeFilter</filter-name>
        <filter-class>com.glin.sys.PrivilegeFilter</filter-class>
    </filter>
    <filter-mapping>
        <filter-name>privilegeFilter</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>
```

7. 扩展及说明

应用系统的权限管理模块，根据需求的不同可以有不同的设计。例如，把访问权限控制在模块级别可以使用模块URL匹配的方式。如果用户的角色是动态可变的，可以使用规则引擎，按照规则动态分配角色。如果要把访问权限控制在数据级别就需要在业务逻辑中详细控制读取哪些数据。当然，也可以用分层的思想进行权限的控制，使一部分用户拥有某一类“方法”的使用权限，Spring AOP就是这种方式很好的一个实现。

本文所介绍的实现方式可能会有角色数量的限制，如果角色数量庞大需要注意两个问题：

1) 数据库类型(或程序类型)值不足以表示角色编码值，角色编码值越界；因为角色编码值都是以2^N

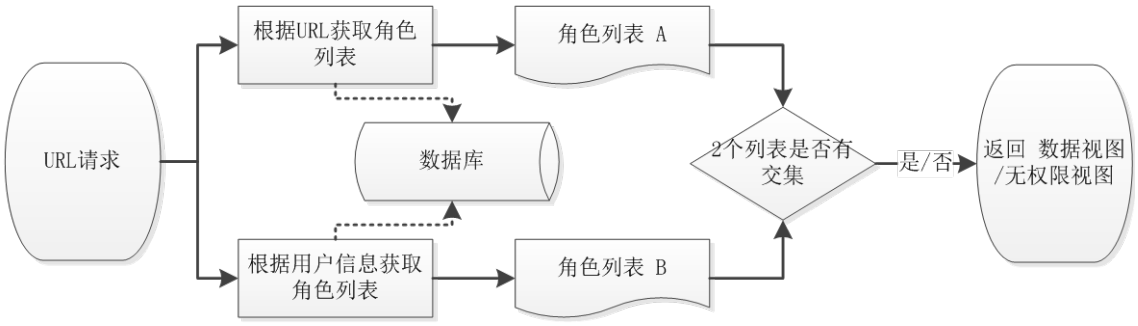

Figure 6. URL requests and data query
图 6. URL 请求与数据查询

形式表示的，值的大小指数型增长，在Java中Long的最大值为2^{63}-1，如果在程序中以这种类型进行运算就角色就不能超过63种。

2) 根据用户获取角色集合和根据访问路径获取角色集合，集合中元素的增多会影响检查交集的运算次数。

8. 结束语

权限管理在应用系统中使用广泛，本文介绍的是一种依赖于关系数据库的实现方式，通过使用关系数据库表之间的关系和权限控制算法，结合在实际应用中B/S访问与操作的流程，和互联网资源标示唯一的特点，为解决资源权限管理问题进行了设计和实现。本文所述的权限访问控制理论已经在管储网管道智能巡检系统中得到应用，多个部门，多用户同时在线时，能够更好地实现模块权限访问控制和数据权限访问控制。本文通过统一访问控制实现和方法，消除角色冗余，保证用户数据的访问控制，提出了解决用户权限细粒度的控制方案。对B/S结构的访问控制研究具有一定的指导意义。

基金项目

本文系教育部人文社会科学研究规划基金项目资助“关联数据环境下安全机制与数据溯源的研究”(基金编号：12YJA870014)。

参考文献 (References)

[1] 肖锋 (2010) 角色权限控制在图纸管理系统中的应用. *现代电子技术*, **18**, 104-109.

[2] 贺德富, 苏喜生, 胡安胜, 康勇 (2009) 一种RBAC改进模型及其在军事Web信息系统中的应用. *计算机与数字工程*, **7**, 99-102.

[3] 储晨曦, 王纯, 李炜 (2012) 基于 LAMP 架构的 Web 权限控制组件的设计与实现. *电信工程技术与标准化*, **9**, 82-85.

[4] 王伟全, 张学平 (2012) 基于岗位抽象的角色权限控制模型设计与实现. *软件导刊*, **1**, 107-108.

[5] 杨骏, 李中华, 倪明涛 (2014) 车辆监控中多粒度 RBAC 改进模型. *计算机应用与软件*, **2**, 109-111.

[6] 江伟欢, 张作萍 (2014) 多层次复用系统访问权限控制方法设计于实现. *计算机工程与设计*, **7**, 2617-2621.

[7] 吴娇 (2014) 基于 RBAC 的权限管理研究与设计. *计算机应用*, **1**, 91-94.

[8] 汤才友 (2014) 基于 RBAC 模型的档案管理系统权限控制体系应用. *档案时空*, **6**, 11-13.

[9] 韩江 (2014) 基于 SSH 架构的 Java Web 通用权限管理的设计. *智能计算机与应用*, **2**, 114-116.

[10] 田传凤, 刘培鹤 (2014) 基于角色的访问控制在研究生校园网中的应用研究. *北京电子科技学院学报*, **2**, 81-86.

[11] 庞希愚, 王成, 仝春玲 (2014) 基于角色－功能的 Web 应用系统访问控制方法. *计算机工程*, **5**, 144-148.

[12] 赵秀霞, 付秀丽 (2014) 基于组件和拦截器的 Web 系统权限设计与实现. *现代电子技术*, **8**, 105-107.

[13] 王倩, 包岩 (2014) 一种面向数字化校园的扩展权限控制模型的研究与实现. *电子测试*, **8**, 75-78.

[14] 岳康军, 周红专 (2014) 一种新的细粒度 URL 权限管理方法. *湖南邮电职业技术学院学报*, **2**, 52-57.