第 36 卷 第 1 期 计 算 机 工 程 2010 年 1 月
Vol.36 No.1 Computer Engineering January 2010
·安全技术· 文章编号：1000—3428(2010)01—0143—03 文献标识码：A 中图分类号：TP311

ASP.net 中基于 RBAC 的通用权限管理系统

范明虎[1]，樊 红[1]，伍孝金[2]
(1. 武汉大学测绘遥感信息工程国家重点实验室，武汉 430079；2. 荆楚理工学院教育技术中心，荆门 448000)

摘 要：资源的差异性和权限管理的复杂性导致不同应用系统中的权限管理子系统难以通用。针对上述问题，通过扩展 ASP.net 中已部分实现的基于角色的访问控制并扩展资源管理部分，设计一个可以在中小型 Web 应用系统中通用的权限管理子系统。从访问控制的粒度出发，以实例阐述其实现的关键技术，证明该系统可以简化权限管理的设计与实现，有效降低 Web 应用系统开发的工作量。

关键词：角色；访问控制；权限管理；资源管理；通用性

Universal Privilege Management System Based on RBAC in ASP.net

FAN Ming-hu[1], FAN Hong[1], WU Xiao-jin[2]
(1. State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan 430079;
2. Education Technology Center, Jingchu University of Technology, Jingmen 448000)

【Abstract】Privilege management subsystems are difficult to be reused in different application systems because of the differences in resources and complexity of privilege management. This paper designs a universal privilege management system that can be reused in small-and-medium-sized Web application system by expanding Role-Based Access Control(RBAC) partly implemented in ASP.net and complementing resources management. Its key technologies are expatiated with some examples from the granularity of access control, and it is proved that the system can simplify the design and implementation of privilege management and reduce the workload of Web application system development.

【Key words】role; access control; privilege management; resource management; universality

在 Web 应用系统中，权限管理是最重要的组成部分之一，担负着用户分级分类管理、系统和数据的访问控制等重要职责。但是，由于权限管理实现的复杂性，设计一个好的权限管理子系统并非易事。同时，由于不同系统之间资源的差异性大，为一个系统设计的权限管理子系统往往不能在其他系统中通用，而需要重新专门设计，浪费人力物力，因此一个良好的通用权限管理子系统对 Web 应用系统的设计和实现具有重要意义和应用价值。基于角色的访问控制(Role-Based Access Control, RBAC)是目前比较成熟、应用比较广泛的统一资源访问控制方法，其核心内容包括用户、角色和资源三部分。ASP.net 提供了对用户和角色进行管理的良好实现。如果能添加对资源管理的良好实现，就能以其为基础，设计一个更为完善的权限管理子系统。

1 需求分析

为了便于分析和设计，将权限管理所涉及的内容划分为 5 个部分：用户，角色，文件，菜单和权限。菜单与应用程序中具有独立功能的模块(在 Web 应用程序中指一个或多个网页)相对应，应用程序通常用菜单来管理它们。权限指模块实现的基本功能，如读和写。下面以图 1 所示的 RBAC96 模型[1-3]为依据进行需求分析，该模型的基本理论见文献[1-5]。

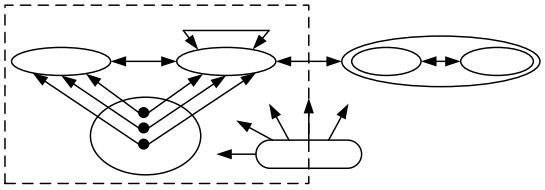

图 1 RBAC96 模型

ASP.net 中的实现部分包含在图 1 的虚线框内，主要包括用户管理、角色管理以及与其相关的会话管理。但这些都只实现了基本功能，应用中常需要对它们进行扩展。

在前台，ASP.net 提供了与用户管理相关的各种控件，由此实现了诸如用户注册、密码修改、密码找回、用户状态检测等常用的基本功能[1]。可对它们进行修改和扩展。在后台，ASP.net 提供了实现用户和角色管理所必需的数据库和表，它们由系统自动创建，并通过相关对象(User, Roles, Membership 和 MembershipUser)自动管理，实现了用户和角色的创建、编辑、删除、获取等功能[1]。可对这些表进行修改和扩充，对这些对象进行扩展。此外，ASP.net 还可通过 Web.config 文件配置文件和目录的访问授权[6]。相应的访问控制由 ASP.net 框架自动完成。

总之，ASP.net 中的 RBAC 已经实现了对用户、角色和文件的基本管理，在应用中只要根据需求加以修改和扩展即可。对于菜单和权限的管理，则必须由程序员实现，这是通用权限管理子系统实现的重点，主要的实现有权限(操作)、菜单和页面(对象)及它们之间的关系(操作与对象间的联系)。

2 基于RBAC的通用权限管理系统设计

(1)用户管理

一个比较完备的用户管理应包括以下功能：注册，登录，密码修改，密码找回，用户信息获取，当前用户状态获取，用户的添加、删除、资料修改等。前 6 项功能可以由 ASP.net

作者简介：范明虎(1974－)，男，博士研究生，主研方向：网络信息系统；樊 红，教授、博士；伍孝金，副教授

收稿日期：2009-05-04 **E-mail：**fanminghu@21cn.com

提供的控件实现，其他功能则需要由程序员实现。另外，还可以实现基于组的用户管理功能，以方便权限分配与回收。Membership 对象可完成用户的添加和删除，但如果要增加用户详细资料信息，则需要对其进行扩展。

(2)角色管理

主要实现角色的添加、删除、修改和分配，均可通过 Roles 对象完成。还可以实现基于用户组的角色分配功能，以方便权限分配与回收，但需要对 Roles 对象进行扩展。

(3)资源管理

主要实现 2 项功能：控制对菜单(通常对应模块的首页)的访问(粗粒度)，控制角色对某个网页的访问权限(细粒度)。实现这些功能必须自定义对象。

(4)数据库和表

对于 ASP.net 自动创建和维护的库和表，除非对它们进行扩展，否则无须关心。需要做的是添加一些对用户资料、用户组和资源进行管理的表，如用户资料表、用户组表、资源表、权限表。

(5)系统架构

系统框架为 3 层设计，见图 2，最下层是数据库；最上层是页面；中间层是对象，对下负责数据库操作，对上完成页面数据的存取。各层相对独立，方便系统的修改和扩充。

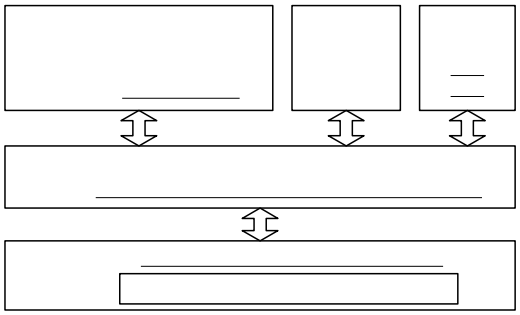

图 2　系统框架

图 2 中的用户与角色指 ASP.net 自动创建的 11 个表。加下划线的部分必须由程序员设计和实现，而其他部分可以借助 ASP.net 提供的对象和表来实现。因为文件和目录的权限管理可以直接通过 ASP.net 框架完成，所以未在图 2 中标出。

(6)权限表示

权限最简单的表示方法是使用 0(无权)和 1(有权)。它们可以是一个二进制位，也可以是一个字符。使用二进制位的优点是运行速度极快，解读方便，但表示的权限数有限；使用字符的优点是可以表示的权限数很大，但解读相对较慢。一种资源的权限通过多个 0 和 1 来表示，每个代表一种权限。

32 位系统中的一个整数最多能表示 64 种权限，因为一般的应用程序不会用到这么多种权限，所以这种方法是可取的，在一般的应用中是合适且足够的。

3　关键实现与实例

用户资料修改、用户组、角色按组分配等功能的实现比较容易，仅作简要描述。用户资料是指用户的详细情况，一般只作显示用，只要增加相应的表即可。组的实现与 Windows 中的组类似，每个用户都属于某个或某些组，每个组具有某种或某些角色。使用组的好处是可以大大降低为很多用户分配多种角色的工作量，简化操作的同时降低了出错机率，增强了用户体验。用户、组和角色之间的关系为：以用户为主，用户属于→用户组←角色分配给。

文件和目录的权限管理只需手工配置 Web.config 文件即可，也可以由程序员设计相应的管理程序。

下面给出资源管理的实现及其实例。

(1)与资源管理相关的表

资源表、权限表和操作表如表 1~表 3 所示，其中仅列出关键字段，且都基于 SQL Server 2005。

表 1　资源表

字段名	数据类型	主键	含义
ApplicationId	uniqueidentifier	是	应用程序 ID
ResourceId	uniqueidentifier	是	资源 ID
ParentId	uniqueidentifier	否	父资源 ID
ResourceName	nvarchar(50)	否	资源名
ResourceURL	nvarchar(50)	否	资源地址
IsMenu	bit	否	是否为菜单
IsGroup	bit	否	是否为组
Privilege	bigint	否	权限

表 2　权限表

字段名	数据类型	主键	含义
ApplicationId	uniqueidentifier	是	应用程序 ID
RoleId	uniqueidentifier	是	角色 ID
ResourceId	uniqueidentifier	是	资源 ID
Privilege	bigint	否	权限

表 3　操作表

字段名	数据类型	主键	含义
ApplicationId	uniqueidentifier	是	应用程序 ID
OpertionId	uniqueidentifier	是	操作 ID
OperationName	nvarchar(50)	否	操作名
OperationMask	bigint	否	操作掩码

以上 3 张表中都有 ApplicationId 字段，ASP.net 通过此字段区分同一服务器中不同 ASP.net 应用程序的 RBAC 数据。

系统中的 2 个权限概念如下：(1)基本权限：模块的某种基本功能(如添加、删除)。(2)访问权限：赋予用户的对基本权限的使用许可。2 种权限在系统中使用统一的名称、编码和解读方式，以简化管理。表 1 中的权限指某个资源的基本权限，表 2 中的权限指角色被授予的对某个资源的访问权限。表 3 用于定义和解读权限，操作名面向用户时即为权限名。

(2)权限的实现

权限的实现采用基于整数二进制位的方法，示例如下：用户定义如表 4 所示的权限时，系统将自动按从低位到高位的顺序生成掩码。之后添加、删除、修改模块基本权限和用户的访问权限以及解读权限时，均严格与表 4 一一对应。整个过程对用户完全透明。权限解读非常简单高效，将某角色对某资源的访问权限值与操作掩码进行与运算，结果与掩码相等则具有该掩码对应的权限，反之则没有。

表 4　权限实现实例

操作/权限	查看	执行	新增	修改	删除	审核	打印
掩码	1	2	4	8	16	32	64

(3)粗粒度访问控制

粗粒度访问控制的功能是对用户能够访问的系统模块进行控制。只有被授权的模块，用户才能访问，反之则不能。

资源采用统一编码，每个页面对应一个编码，每个页面都根据其实现的功能被赋予一定数量的基本权限。其中，“查看”作为一种最基本的权限而存在，默认每个页面都具有这种权限。如果某用户被授予某页的“查看”权限，则该用户可以访问该页，反之则不能。

菜单项默认代表系统中某模块的首页。未获得菜单项“查

用户管理　角色管理　资源管理
注册、登录、密码修改、密码找回、用户信息、用户状态、添加、删除、修改、分配、菜单、页面
添加、删除、资料修改、分配
对象
User, Roles, Membership, MembershipUser
自定义的用户对象、角色对象、资源对象、权限对象

看”权限的用户将无法在菜单中看到该项。菜单组(仅用作显示)也作为一种资源予以编码。未获得菜单组“查看”权限的用户将无法在菜单中看到该组及该组下的所有项。当用户以直接 HTTP 方式在浏览器中请求执行某个页面时，系统将同时检测该用户对该页和该页所在组的“查看”权限，如果有一项未被授予，则禁止访问。总之，由于网页加载时会检测当前用户的访问权限，因此未被授权的用户无论通过何种方式均不能访问，只有被授权的用户，才能通过检查。

图 3 是对资源进行综合管理的页面截图，为节省篇幅，进行了简化。由于使用了资源树管理系统中的菜单和页面资源，因此自定义了 ResourceTree 对象来实现资源的添加、删除、修改、读取、绑定等功能。图中，“权限菜单”默认出现在权限管理子系统的菜单中；“权限其他”可以通过设置“是否菜单”决定是否出现在菜单中，默认通过页面链接或直接 HTTP 方式访问。

图 3　资源综合管理页面

(4)细粒度访问控制

细粒度访问控制的功能是对用户使用页面内的功能进行控制。只有被授权的功能，用户才能使用，反之则不能。

权限管理系统中对页面定义了 3 种存在状态：(1)未被纳入资源表的管理，此状态页面默认可以被任何用户访问；(2)已经纳入资源表的管理，但如果其访问权限未被分配给某个角色，即权限表中没有记录，此状态页面默认只能被查看；(3)已经被分配给某个角色，即资源表和权限表中都有记录，此状态页面只能被授权角色访问。为管理访问权限，自定义了 PrivilegeData 对象来实现权限的分配、获取、解读等功能。当页面加载时，通过 PrivilegeData 对象从权限表中获取当前用户对当前页的访问权限，根据返回结果，禁用当前页中未被授权的功能，从而完成对页面内功能使用的控制。

图 4 是为角色分配访问权限的页面截图，为节省篇幅，进行了简化。其中的实例用统一的方法实现了为任意角色分配权限。

图 4　权限分配页面

4　通用性

总结上文，系统的具体实现如下：

(1)操作(权限)由用户定义，权限处理则完全由程序自动进行。

(2)通过资源综合管理将页面纳入权限管理子系统，并为角色分配对页面的访问权限。

(3)在菜单显示或页面加载时，从权限管理子系统获取当前用户对当前页的访问权限，并完成对页面的访问控制。

如果将该系统应用到 Web 应用程序开发中，程序员只需要做如下工作：

(1)定义角色和用户组。

(2)为用户组分配角色。

(3)将用户分配到用户组。

(4)定义操作(权限)。

(5)通过资源综合管理纳入系统资源。

(6)通过权限分配管理为角色分配各种资源的访问权限。

(7)应用程序加载时调用 ResourceTree 对象绑定(显示)菜单，或者通过 ResourceTree 对象获取菜单资源自行处理。

(8)页面加载时调用 PrivilegeData 对象获取当前用户对当前页的访问权限并完成访问控制。另外，可通过配置 Web.config 控制角色对文件和目录的访问。

可以看出，权限管理实现的整个过程不再需要设计新的页面，编程也只有简单的对象调用。因此，该系统的独立性比较强，与应用程序的配合却很好，具有良好的通用性。

5　结束语

本文在充分利用 ASP.net 中已经实现的部分 RBAC 特性的基础上，结合实际应用，设计并实现了一个具有良好通用性的基于 RBAC 的权限管理系统。实践证明，该系统能够较好地实现基于 RBAC 的权限管理，在中小型 Web 应用程序的设计与开发中通用，具有较高的应用价值。

该系统的局限性表现在仅实现了 RBAC96 基本模型 RBAC0[1-4]。如果需要实现更复杂、功能更强的权限管理，则需要对 ASP.net 中 RBAC 的实现做进一步的扩展。本系统的实现已经有了一个良好的开端，可以在此基础上扩展对其他模型的支持，最终实现一个比较完善的基于 RBAC 的通用权限管理系统。

参考文献

[1] 戴莹莹. B/S 结构的 OA 系统中基于角色访问控制模型研究与实现[D]. 武汉：武汉理工大学, 2006.

[2] 王　鑫. 基于角色的权限系统开发与应用[D]. 成都：西南交通大学, 2005.

[3] 李志英, 黄　强, 楼新远, 等. RBAC 模型研究、改进与实现[J]. 计算机应用, 2006, 26(12): 2945-2947.

[4] 鲁　柯, 周保群, 王惠芳. 基于带时间特性 RBAC 的使用控制模型及其管理[J]. 计算机工程, 2008, 34(6): 170-172.

[5] 高正宪, 李中学. Web 环境下基于角色的访问控制策略及实现[J]. 计算机工程, 2004, 30(8): 133-135.

[6] Walther S. ASP.net 2.0 揭秘(卷 2)[M]. 谭振林, 黎　志, 译. 北京：人民邮电出版社, 2007.

编辑　张　帆