一种扩展的 RBAC 模型——ERBAC①

An Extended RBAC Model——ERBAC

谭　振　杨贯中　曾　熠　(湖南大学 软件学院 湖南 长沙 410082)

摘　要： 针对在拥有多个 RBAC 应用系统的企业中，系统管理困难的问题，建立一个扩展的 RBAC 模型—ERBAC (extended role based access control)。该模型把应用系统分为组织结构层次和系统层次；组织结构层次是对企业组织机构进行映射，系统层次只对各个应用系统的角色和权限进行管理。当企业的组织机构或系统发生改变时，只需对相应的层次部分进行修改即可，从而简化了系统的管理。

关键词： RBAC 访问控制策略 岗位 组织

1　引言

随着计算机技术和网络技术的发展，以及信息系统的普及，基于角色的访问控制系统越来越流行。尽管 RBAC 是现在应用最为广泛的访问控制方式，但是它并没有涉及到在多个 RBAC 应用系统中的管理问题。事实指出在一个大型企业中，基于 RBAC 的应用系统将变的特别庞大(包含数以百计的角色)。而在企业组织机构发生变动，或者系统发生改变时，管理这个系统是一件非常困难的工作，尤其是在多个系统并存的情况下。

例如，某企业拥有多个基于 RBAC 的应用系统，而每个系统拥有不同的权限和角色。从而，企业中的某一成员可能在不同的系统中拥有不同的角色。在这种情况下，如果人员发生调动，或者说权限发生改变，就会引发一系列问题：由谁来进行这些改变？哪些系统需要改变？系统如何进行改变？

本文主要研究的是，建立一个基于 RBAC 的扩展模型—ERBAC(extended role based access control)。通过该模型，管理员可以方便的对系统进行管理。当人员、组织机构或者系统发生改变时，能够快捷简单的通过更改相关设置来符合企业实际的需求。

2　RBAC基本模型

RBAC 的核心思想就是：在用户和权限之间引入角色的概念，将用户和角色联系起来，通过对角色的授权来控制用户对系统资源的访问。一个用户可以被赋予多个角色，一个角色可以对应多个用户，这些角色是根据组织内为完成各种不同的任务需要而设置的；同样，一个角色可以拥有多个权限，一个权限可以被多个角色所拥有。这样通过应用 RBAC 可将安全性放在一个接近组织结构的自然层面上进行管理。

现在有许多不同的 RBAC 模型，在这里我们仅限于讨论一般层次的 RBAC 模型[1]，如图 1 所示，它是基本的 RBAC 模型标准。

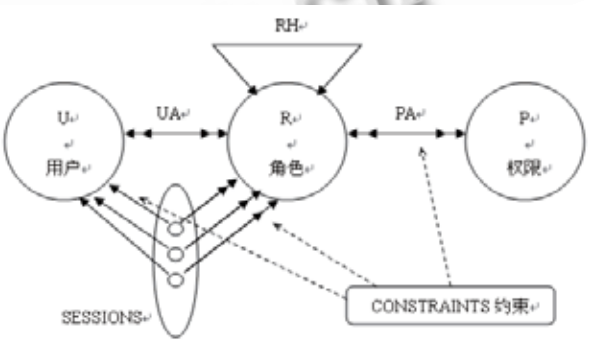

图 1　RBAC 基本模型

RBAC 基本概念：

- U、R 和 P(用户集合、角色集合和权限集合)
- $PA \subseteq P \times R$ (PA 是角色和权限的分配关系)
- $UA \subseteq U \times R$ (UA 是用户和角色的分配关系)

① 收稿时间:2009-02-25

• $RH \subseteq R \times R$ (RH 是角色上的一种偏序关系，表示角色间的继承关系)

示例：如图 2，存在：

$U = \{U_a, U_b, U_c\}$　　$R = \{R_1, R_2, R_3, R_4\}$

$P = \{P_1, P_2, P_3, P_4, P_5, P_6\}$

其中，用户 U_a 的角色为 R_1，角色 R_1 具有权限 P_1，同时 R_1 又继承了角色 R_4，R_4 具有权限 P_2 和 P_3。所以用户 U_a 具有的权限为 P_1、P_2 和 P_3。同理，用户 U_b 具有权限 P_4、P_5 和 P_6，用户 U_C 具有权限 P_6。

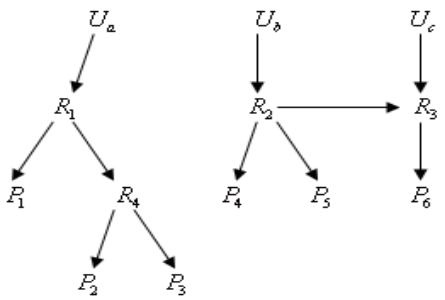

图 2　RBAC 简单示例

3　扩展的模型—ERBAC

在当今的现实中，伴随着企业组织机构的日益庞大，也必然导致企业子系统数目的增加。所以，基于 RBAC 模型的系统在遇到组织机构或者系统变化时，它的管理和维护将会变成一件非常复杂的工作。因此，我们提出了这个扩展的 RBAC 模型—ERBAC，如图 3 所示，基于此模型的系统可以只通过少量变动来适应多变的企业需要。

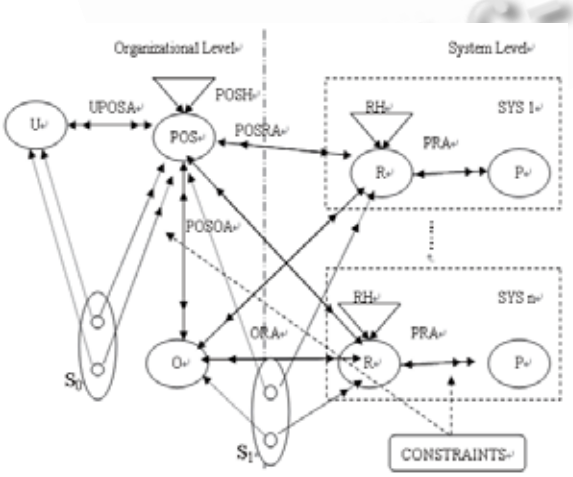

图 3　ERBAC 模型

在这个模型中，我们把整个企业的系统分为 2 个层次[2]：组织结构层次和系统层次。

在组织结构层次 :是对现实的组织机构进行映射，其中元素包括：用户 U，岗位 POS，组织 O。

系统层次：是对各个应用系统的权限、角色进行描述，包括多个子系统的权限 P，角色 R。

约束 Constraints：支持静态职责分离和动态职责分离。静态职责分离即通过定义相关约束，允许在对用户进行岗位、角色分配的时候，一个用户最多只能被指派到一组互斥角色中的一个角色，特定的权限不能同时被指派到互斥的角色或岗位。动态职责分离即允许同一个用户拥有多个互斥的角色或岗位，但在用户登陆时不可同时激活这些互斥的角色或岗位。

根据上图，我们给出以下定义[3-5]：

在组织结构层次中，集合：U、S、O、POS 分别代表用户、会话、组织、岗位。

$\mathrm{UPOSA} \subseteq \mathrm{U} \times \mathrm{POS}$，用户和岗位之间的一对多的关系，一个用户可以分配给多个岗位，一个岗位只能分配一个用户。

$\mathrm{POSOA} \subseteq \mathrm{POS} \times \mathrm{O}$，岗位和组织之间的多对多的关系，一个组织可以拥有多个岗位，一个岗位也可以属于多个组织。

$\mathrm{POSH} \subseteq POS \times POS$，岗位继承，是在岗位 POS 上的一种偏序关系。表示成$\geq$。

在系统层次中，集合：R，P 分别代表角色，权限。

$\mathrm{PRA} \subseteq \mathrm{P} \times \mathrm{R}$，权限和角色之间多对多的分配关系，一个权限可以分配给多个角色，一个角色能拥有多个权限。

$RH \subseteq R \times R$，角色继承，是在角色 R 上的一种偏序关系。表示成≥。

$\mathrm{POSRA} \subseteq \mathrm{POS} \times \mathrm{R}$，角色和岗位之间多对多的分配关系，一个角色可以分配给多个岗位，一个岗位拥有多个角色。

$\mathrm{ORA} \subseteq \mathrm{O} \times \mathrm{R}$，组织和角色之间的多对多的分配关系，一个组织可以有多个角色，一个角色可以分配给多个组织。

$u : S \to U$，每一个会话 S 到唯一相对应用户 U 的映射函数。

$permissions\_r(r) : R \to 2^P$，角色 R 到权限 P 幂集的映射函数，结果为该角色所拥有的权限集合。角色 r 拥有的权限为：

$permissions\_r(r)=\{p\,|(\exists r'\le r)\wedge(p,r')\in PRA\}$

$o\_r(o):O\to 2^R$，组织 O 到角色 R 幂集的映射函数，结果为该组织所拥有的角色集合。一个组织 o 拥有的角色为 $o\_r(o)=\{r|(o,r)\in ORA\}$ 。

$pos\_o(pos):POS\to 2^O$，岗位 POS 到组织 O 幂集的映射函数，结果为该岗位所属于的组织集合。一个岗位 pos 属于的组织为：

$$pos\_o(pos)=\{o|(pos,o)\in POSOA\}$$

$pos\_r(pos):POS\to 2^R$，岗位 POS 到角色 R 幂集的映射函数，结果为该岗位直接拥有的和通过组织或者其他岗位继承得到的所有角色集合。一个岗位 pos 拥有的角色为：

$$pos\_r'(pos)=\{r|[(\exists pos'\le pos)\wedge(pos',r)\in POSRA]\vee[(\exists o\in pos\_o(pos))\wedge(o,r)\in ORA]\}$$ ，

$pos\_r(pos)=\bigcup_{r'\in pos\_r'(pos)}\{r|r\le r'\}$。

$u\_pos(u):U\to 2^{POS}$，用户 U 到岗位 POS 幂集的映射函数，结果为该用户所属的岗位集合。一个用户 u 属于的岗位为：

$u\_pos(u)=\{pos|(\exists pos'\ge pos)\wedge(u,pos')\in UPOSA\}$。

通过以上表达式，我们就能建立起这个 ERBAC 模型[6,7]。通过用户—岗位—组织—角色—权限这样的关系，最终可以找出一个用户的权限。

例子：假设某企业现有 2 个系统(暂不考虑职责分离)S1 和 S2，存在 4 个岗位：POS1，POS2，POS3，POS4；两个组织 O1，O2，组织 O1 由 POS2 和 POS3 组成，组织 O2 由 POS1 和 POS4 组成；2 个系统共有 6 种角色，分别为 R1，R2，R3，R4，R5，R6；2 个系统共有 8 个权限，分别为 P1，P2，P3，P4，P5，P6，P7，P8。人员、岗位、组织、角色、权限的分配如图 4 所示：

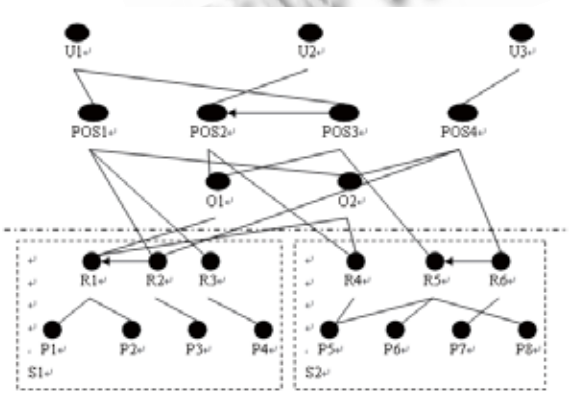

图 4　系统人员、组织权限分配

从上图可以看出，用户 U1 的岗位为 POS1 和 POS3，而 POS3 继承了岗位 POS2，即拥有岗位 POS3 的用户也同时拥有了岗位 POS2；因此，当 U1 登陆后，则可以同时激活 POS1、POS3 和 POS2 这 3 个岗位。POS1 直接对应的角色为系统 S1 中的 R2 和 R3，而 R2 又继承 R1；另外，POS1 从属于组织 O2，O2 对应的角色为系统 S1 中的 R1 和系统 S2 中的 R4，故 POS1 对应的角色集为{R1，R2，R3，R4}，POS1 拥有的权限为这 4 个角色拥有权限的并集{P1，P2，P3，P4，P5}；同理，POS2 直接对应的角色为系统 S2 中的 R4，POS2 从属于组织 O1，O1 对应的角色为系统 S1 中的 R1，因此，POS2 对应的角色集为{R1，R4}；POS3 直接对应的角色为 R5，也从属于组织 O1，从而 POS3 对应的角色集为{R1，R5}。综上，用户 U1 拥有的角色集为{R1，R2，R3，R4，R5}，从而拥有的权限集为{P1，P2，P3，P4，P5，P6，P8}。同理可得，用户 U2 的角色集为{R1，R4}，权限集为{P1，P2，P5}。用户 U3 的角色集为{R1，R2，R4，R5，R6}，权限集为{P1，P2，P3，P5，P6，P7，P8}。

假设企业组织结构和人事发生变动，提出如下需求：

①．将用户 U1 从岗位 POS1 调整到岗位 POS2；

②．独立岗位 POS2，即让 POS2 跟 POS3 之间无继承关系；

③．为系统 S1 中的角色 R2 增加权限 P4；

④．企业需要再增加一个系统 S3，S3 中有 3 个权限 P9、P10 和 P11，1 个角色 R7。S3 系统是专门给组织 O2 用的。

如图 5 所示，我们通过简单调整，便可完成此需求。

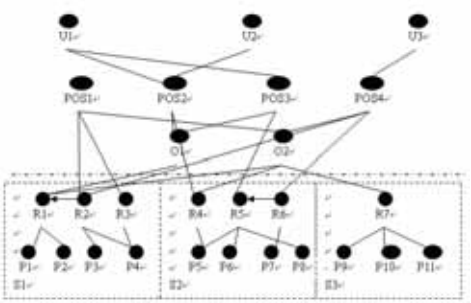

图 5　调整后的人员、组织权限分配

4　小结

本文在基本的 RBAC 模型中引入了岗位(POS)和组

(下转第 83 页)

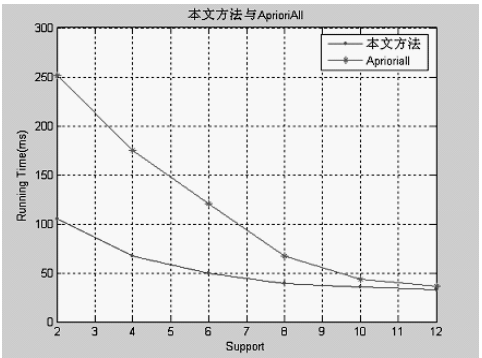

图 2 本文方法与 AprioriAll 算法在不同支持度情况下耗时

相同的规则集，而运行效率优于 AprioriAll 方法，满足金融时间序列预测的实时性要求，下一步的研究工作应该考虑对多个时间序列进行分析[4,8-10]，发现不同时间序列间可能存在的关联关系，发现这种多时间序列中的频繁结构模式对于人们更彻底的认识各个时间序列的相互影响并据此做出合理的决策具有重要的参考价值。

参考文献

1 Agrawal R, Ramakrishnan S. Fast algorithms for mining association rules in large databases. Proceedings of the Twentieth International Conference on Very Large Databases, Santiago: ACM Press, 1994. 487 - 499.

2 Manila H, Toivonen H, Verkamo AI. Discovery frequent episodes in sequences. Proc. of KDD,95.

3 Das G, Lin K, Mannila H, et al. Rule discovery from time series. Proceedings of Fourth Annual Conference on Knowledge Discovery and Data Mining. New York: AAAI Press,1998.16 - 22.Montreal: AAAI Press, 1995. 210 - 215.

4 朱冲,朱贤贵,张向利.金融时间序列挖掘综合模型.计算机系统应用, 2009,18(2):46 - 48.

5 Kwok CO, Etzioni O,Weld DS. Scaling question answering to the Web. ACM Trans. Information Systems, 2001,19(3):242 - 262.

6 史忠植.知识发现.北京:清华大学出版社, 2002.

7 Whitehead SD. Auto-FAQ: An experiment in cyberspace leveraging. Proc. of the Second International WWW Conference.1995.25 - 38.

8 王晓晔.时间序列数据挖掘中相似性和趋势预测的研究[博士学位论文].天津:天津大学, 2003.

9 黄河,黄轲,杭小树,熊范纶.时间序列中快速模式发现算法的研究.计算机工程与应用, 2003,39(21):192 - 194.

10 Oyama S, Kokubo T, Ishida T. Domain-Specific Web search with keyword spices. IEEE Trans. Knowledge and Data Eng., 2004,16(1):17 - 27.

(上接第 86 页)

织(O)的概念，从而建立了一个扩展 RBAC 模型—ERBAC。该模型解决的主要问题是：面对企业频繁复杂的需求变动，可以方便进行系统的管理，方便确定哪些子系统的访问策略需要变更，使得现实企业中的组织机构和职责权限较真实的映射到了计算机系统中。

参考文献

1 Standhu RS, Coyne EJ, Feinstein HL. Role-Based Access Control Models. IEEE Computer, 1996,29 (2):38 - 47.

2 Park JS, Costello KP, Neven TM, Diosomito JA. A composite rbac approach for large, complex organizations. SACMAT'04. 2004.163 - 172.

3 Li N, Mao ZQ. Administration in role-based access control. ASIACCS'07. 2007.127 - 138.

4 Haidar DA, Cuppens-Boulahia N, Cuppens F. An Extended RBAC Profile of XACML. Sws'06. 2006.13 - 21.

5 MAC D, Crampton J, Etalle S. RBAC Administration in Distributed Systems. SACMAT'08. 2008.93 - 101.

6 胡金柱,陈娟娟. RBAC 模型中角色的继承与互斥问题的研究.计算机科学, 2003,30(11):160 - 163.

7 张方舟,王东安.采用 J2EE 安全机制支持 RBAC 模型的研究与实现.计算机工程, 2006,32(13):125 - 127.