基于多亲树的RBAC角色可视化管理

封孝生 黎湘运 孙 扬 张维明
(国防科技大学C4ISR技术国防科技重点实验室 长沙 410073)

摘 要 基于角色的访问控制(RBAC)被广泛地应用于各类复杂信息系统中,通过对用户指派角色进行授权以访问系统中的特定数据或资源。一些问题已在应用过程中逐渐暴露:如何较好地展现角色层次关系、用户角色指派和角色指派中约束如何体现、冗余的角色授权如何检测与解除等。从可视化的角度采用层次信息可视化技术来辅助RBAC中的角色管理。首先阐述了所研究的问题,并定义了可视化过程中使用的多亲树结构;然后给出一个多亲树规范化过程,以建立一个符合可视化要求的标准角色层次;随后提出一种双层可视化范例来展示角色管理过程,其中下层用于展示角色层次和权限,上层用于配置用户节点;此外,针对所述问题给出若干交互方法,以可视地辅助解决角色管理中的约束和冗余问题。

关键词 基于角色的访问控制,角色管理,多亲树,角色层次,启发式布局
中图法分类号 TP393.08 **文献标识码** A

Facilitating Role Management in RBAC: Using Multi-parents Tree

FENG Xiao-sheng LI Xiang-yun SUN Yang ZHANG Wei-ming
(C4ISR Technology National Defense Science and Technology Key LAB, National Univ. of Defense Technology, Changsha 410073, China)

Abstract Role-Based Access Control(RBAC) has been widely applied to authorize certain users to access certain data or resources within complex information systems. Several problems are coming about during the application of RBAC models, which include well-representing the role hierarchy, following the constraints applied in user-role assignments and role-role relations, revoking redundant roles and assignments, etc. This paper addressed these problems from the perspective of information visualization to facilitate role management in RBAC, particularly leveraging the experience of tree(s) visualization. A detailed problem statement was made first, and the data structure of multi-parents tree was defined. Then a multi-parents tree normalization process was proposed to construct a refined role hierarchy for elegant representation. Subsequently, a two-layered paradigm, the nether for displaying role hierarchy and permissions, and the upper for placing users, was presented for the visualization of role management in RBAC. Additionally, some specific interaction techniques were put forward to visually aid in solving the constraint and redundancy problems.

Keywords RBAC, Role management, Multi-parents tree, Role hierarchy, Heuristic layout

1 引言

基于角色的访问控制作为复杂信息系统中安全问题的一种解决方案,能够保证只有被授权的用户才能访问特定的系统资源或数据。具有代表性的是Sandhu等人提出的RBAC96及其补充模型[1,2]、ARBAC97[3]、NIST RBAC建议标准[4]、角色图模型[5]和ARBAC02[6]。在这些模型中,权限是关于某主体对某客体所被授权而能进行的操作的描述;它与一个组织之中的不同职能相关联,后者被称为角色;角色被指派给适当的用户,使得用户获得相应角色被授予的权限[4]。

RBAC中的角色管理是多方面的,具体可分为:角色定义、用户角色指派(URA)、角色权限指派(PRA)和角色之间的指派(RRA,定义角色层次)[7]。角色层次是角色之间的一种偏序关系,一个上级(父)角色包含下级(子)角色所包含的权限。若某父角色被指派给用户,则该用户就具有该父角色的全部子角色及其所包含的权限。通过角色工程的方法可以建立系统的一个角色和角色层次[8-10],这通常也是开发RBAC系统的第一个阶段。

在实际应用中,角色的数量通常是成百上千,而用户和权限则更是成千上万。方便管理这些角色、权限、用户以及它们之间的关系,是一项十分困难的任务。目前,多数角色管理的对话界面仍停留在线性的缩进列表形式,如图1所示。系统安全管理员通过勾选给角色授权权限,对话过程中无法获得关于角色层次整体结构和其他授权情况的信息。显然,这种角色管理模式中存在着一些不足:

(1) 表现问题。角色层次通常得不到良好的展示,内部的联系难于理解和遵守。类似Windows资源管理器的可视化方式虽然得到广泛认可,但它具有缺少上下文信息等显而

到稿日期:2010-01-12 返修日期:2010-03-31 本文受国家自然科学基金(60903225)和(70971134)资助。
封孝生(1971—),男,副教授,主要研究方向为信息安全、信息管理与智能决策等,E-mail:fxs365@sina.com;黎湘运 硕士生;孙 扬 博士生;张维明 博士,教授,博士生导师。

易见的缺点。而且，角色层次实质上并不是一棵通常意义上的标准树，因而使用缩进方式来展示整体结构是不确切的，一些结构信息无法得以表达，如用户难于发现或者理解角色层次中一个子角色从属于多个父角色之类的信息。

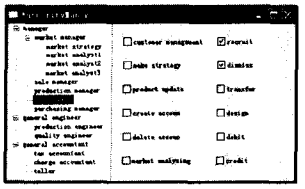
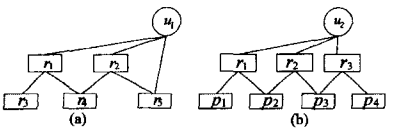


图 1　用于角色-权限指派的常见人机对话界面

(2) 约束问题。RBAC 角色管理中用户角色和角色中的各种约束常常无法得到满足。如角色互斥约束、会计和出纳这两个互斥角色不能被同时赋予同一个父角色或者同一个用户。

(3) 冗余问题。冗余问题包括两类：一类是角色冗余，将一个父角色和其若干个子角色同时指派给一个用户，或不同的父角色指派给一个用户导致同一子角色多次指派给同一用户；另一类是权限冗余，即将某权限多次授权给一个用户。例如，图 2(a)中角色 r_5 作为 r_2 的子角色被冗余指派给用户 u_1。同时，由于 r_1 和 r_2 指派给用户 u_1 而导致 r_4 被多次指派给用户 u_1；图 2(b)中权限 p_2 和 p_3 两次指派给了用户 u_2。适度的冗余不可避免，但过多的冗余将导致系统维护困难。图 2(a)中撤销 u_1 到 r_5 的指派时，须分别撤销 u_1 到 r_2 和 u_1 到 r_5 的指派。



(a)中 r_4 和 r_5 均被重复指派给用户 u_1；
(b)中用户被重复授权许可 p_2 和 p_3。

图 2　角色冗余和权限冗余的例子

上述 3 个问题中，表现问题是其中最基本的问题。若能很好地展示角色层次，以准确且可视的方式辅助系统安全管理员进行角色可视化管理，将有助于解决其他两个问题。因此，我们考虑利用信息可视化的相关技术来解决上述问题。

在角色层次中，一个子角色可以同时隶属于多个父角色，一个权限也可以隶属于多个角色，这也是角色层次与单树的本质区别。我们把角色的这种层次结构称之为多亲树(Multi-parents Tree)。本文从信息可视化的角度定义了适用于角色层次的多亲树结构，并基于多亲树设计了一个可视化交互系统，用于辅助信息系统安全管理员进行 RBAC 中角色的可视化管理。针对问题(1)和部分问题(2)，提出了一种双层可视化结构来表现 RBAC 中的角色层次结构和权限指派；通过改进和实现一系列操作性较强的交互方法来可视地辅助解决角色管理中的约束问题和冗余问题。

本文第 2 节介绍角色图模型以及层次可视化的相关工作；第 3 节中给出文章的记号和假设并介绍一个多亲树规范化算法；第 4 节详细阐释角色管理系统的相关设计；第 5 节展示一个实验原型系统，包括可视化及交互；最后进行总结和给出未来工作展望。

2　相关工作

2.1　角色图模型

角色图模型由 Matunda Nyanchama 和 Sylvia Osborn 提出[5,11]，用于可视化角色与角色之间的关系。RBAC 模型中权限到用户的授权管理涉及 3 个层面(planes)，如图 3 所示[11]。用户层面(Users plane)和权限层面(Privileges plane)分别用来对用户和用户组、权限和权限间的蕴涵关系进行建模，角色层面(Roles plane)通过角色图模型表示角色-角色关系。

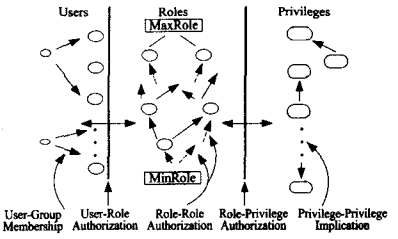


图 3　三类授权

角色图是一个无环的有向图，其中节点表示系统中的角色，边描述了角色之间的层次关系，如 $r_1 \rightarrow r_2$ 表明 r_1 就是 r_2 的子角色。每个角色图有一个最大角色(MaxRole)和最小角色(MinRole)。最大角色表示角色图中所有角色的权限集的并，最小角色表示所有角色都有的最小权限集。模型中区分直接权限(Direct Privileges)和有效权限(Effective Privileges)，前者是直接指派给角色的权限，而后者包含前者以及由子角色继承过来的权限，即每一个角色的有效权限是指该角色的直接权限以及该角色所有子角色的有效权限。

角色图有如下属性：①有且仅有一个最大角色；②有且仅有一个最小角色；③角色图没有环路；④最小角色到任一角色都存在一条路径；⑤任一角色到最大角色都存在一条路径；⑥对于两个角色 r_1，r_2，若角色 r_1 的有效权限集是角色 r_2 的子集，则必定存在一条从 r_1 到 r_2 的路径；⑦通常从下到上绘制角色图，并删除冗余边。

角色图模型还包含了一些算法，用以操作角色图，使其支持管理功能。但是角色图模型对角色的管理还只是一个初步的探索，没能很好地可视化展示整个角色系统的结构。他们后来对该角色图模型进行了补充[11]，考虑了权限-权限冲突和角色-角色冲突关系，但也只是一些形式化的定义和分析，在可视化表现方面仍没有改进。

2.2　层次信息可视化

层次信息可视化一直是信息可视化领域中的一个热点，最早的相关文献可参见文献[12]。在信息可视化文献中，研究者习惯于交替地使用“层次”和“树”二词，认为研究层次即是研究树。这里，我们顺着从单树到多树的主线给出一些层次可视化的相关背景，为下述的多亲树可视化做准备。

单树可视化的方法主要有 3 类。其一是节点连接法，即使用各式不同形状的节点表达数据，连边表达数据之间的关系，从而展示节点的层次结构。空间树(Space Tree)[14]、圆锥树(Cone Tree)[15]、放射树(Radial Tree)[16]等都属于这一类，它们擅长给用户传达树的整体结构。其二是空间填充法，即利用不同外围形状的包围框表达数据，包围框之间或嵌套或邻接的关系表达数据之间的关系。这方面的著名例子是树图[17]，它能接近 100%地高效利用显示空间。最后一种是混合方法，它们集成多种先前的可视化方法。由于先前的可视化方法都存在着优点和缺点，近年的部分工作倾向于组合前人的工作以充分发挥其优势。一些成功的例子包括弹性层次

(Elastic Hierarchy)[18]、层次网络(Hierarchical Net)[19,20]和包围+连接(Enclosure+Linking)[20]等。表1从2D和3D两个维度简要分类和概括了这些技术,并总结了它们的主要特点。

表1 单树可视化分类总结

	节点连接法	空间填充法	混合方法
2D	空间树、双曲树、放射树	树图、圆形封装、放射性填充	弹性层次、CHEOPS、包围+连接
3D	圆锥树、Magic Eye View、轴环柱形树	信息金字塔、信息立方体	植物可视化、层次网、文件系统导航器
特点	较好地传达层次的结构信息	高效地利用显示空间	集成若干种方法的优点

除了单树可视化,多树可视化也在过去的几十年里得到关注和研究,尤其是交叠多分类的可视化。如文献[21]中所总结,分配、动画和聚集3种形式被广泛地用于解决多树可视化问题。分配式,即将可用的屏幕空间分割为若干块区域,在每个区域中绘制一棵单树。Graham等人[22]的工作即属于这一种形式。动画式显示结构或视点的变化而导致的多树在表现上的渐变,常被用于展示树在时间上的变化。其应用实例有可视化网络访问信息的TreeViewer[23]等。在视觉表现上将多树的结构聚集在一起,使对应的节点相互重叠来可视化具有相似结构的多树,即为聚集式。这种情况下,多树中的对应节点在视觉表现上相互覆盖,从而一个子节点可能从属于多个父节点。Furnas和Zack[24]将聚集式运用于实际的多树可视化中。表2给出了多树可视化基数的简要分类总结,包括应用实例及其主要特征。

表2 多树可视化分类总结

	分配式	动画式	聚集式
实例	自然历史收藏可视化,网络生态进化可视化	动画式树图,可视化网络信息访问	Multitrees,基于图的重叠分类可视化
特征	多树的数量不大时效果良好,但扩展性不好	适于显示一系列树的渐变,但难于完全重现	有效利用屏幕空间,但认知上难于解读接合在一起的层次结构

3 角色多亲树及其规范化

3.1 角色多亲树

定义1(RBAC模型元素) USERS,ROLES,OPS,OBS,SESSIONS分别代表用户集、角色集、操作集、对象集、会话集。

$PRMS \subseteq 2^{(OPS \times OBS)}$,表示权限集。

$UA \subseteq USERS \times ROLES$,一个多对多的用户和角色的映射关系。

$PA \subseteq PRMS \times ROLES$,一个多对多的权限和角色的映射关系。

$RH \subseteq ROLES \times ROLES$,一个称为继承的角色之间的偏序关系,记作$\geqslant$。如果$r_1 \geqslant r_2$,则角色$r_1$拥有$r_2$的权限,称$r_1$是$r_2$的父角色。

定义2(角色多亲树,Role Multi-parents Tree,RMT) 即由ROLES和RH构成的有向无环图,其中节点为角色,边为角色之间的偏序关系,简称多亲树。RMT可能有多个根。

定义3(角色规范多亲树,Normalized Role Multi-parents Tree,NRMT) 亦即每个角色只与其相邻层次上的角色相连的RMT。

3.2 角色多亲树规范化

在RMT中,一个角色节点可以有多个父角色节点,这些父角色节点可能不在同一角色层次上。可视化时,若该角色节点直接与处在不同层次的父角色以线相连,则可能会导致很多的边交叉问题。角色节点的子角色节点也存在同样的问题。若将角色层次事先进行规范化处理,使得每个角色节点只与其相邻层次上的角色节点相连,那就能减少可视化时的边交叉的几率。因而在可视化前需要对角色多亲树RMT进行角色层次规范化,以得到角色规范多亲树NRMT。

角色多亲树规范化算法引入哑角色节点,以减少边交叉问题,同时能保持角色层次的特征。算法的输出是一棵角色规范多亲树,也是随后的可视化阶段的输入数据结构。规范化算法的伪代码示于表3。

表3 角色多亲树规范化算法

For each layer of the role hierarchy from top to down
  Given a role set $rs=\emptyset$
  For each role node r in certain layer
  If r is real role and $r \supset (r_{child1} \cup r_{child2} \cup \cdots \cup r_{childk})$, where k is the number of r's child roles
    Compute the r child dummy role-node $r_{dummy} = r - (r_{child1} \cup r_{child2} \cup \cdots \cup r_{childk})$, which may be at the next layer in the hierarchy
  If $r_{dummy} \subset a$, where $a \in rs$
    Add r_{dummy} into rs and $a = a - r_{dummy}$, construct the parent-child relationship between a's parent and r_{dummy}
  Else if $r_{dummy} \supset a$, where $a \in rs$
    Add $r_{dummy} - a$ into rs, construct the parent-child relationship between the parent of r_{dummy} and a
    Else if $r_{dummy} = a$, where $a \in rs$
      Construct the parent-child relationship between the parent of r and a
    End if
  Else
    Add r_{dummy} into rs
  End if
  Else if r is dummy role
    Add dummy role node $r_{dummy} = r$ at the next layer in the hierarchy and add r_{dummy} into rs
  End for
End for

经规范化处理后,每个角色只与其相邻层次上的角色相连。图4(b)是图4(a)进行角色多亲树规范化后的结果。

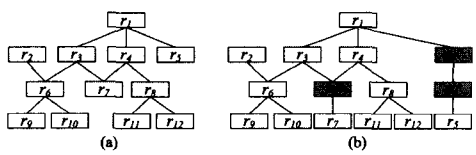


图4 角色多亲树规范化

整个规范化过程可用图5中的例子加以直观说明。若一个上级父节点的权限完全等同于其所有子节点的权限,则不需添加任何哑节点,如图5(a)所示;否则,需要添加一个哑节点,以构建一个"完整"的层次结构,如图5(b)所示,其中$r_{2childdummy} = r_2 - r_{2child1} - r_{2child2}$,$r_i$表示角色$i$所持有的权限。然而,若后加入的哑节点包含了(或被包含于)同层中先前加入的某哑节点,则需对权限集合较大的哑节点进行分割,如图5(c)中对角色节点r_3和r_4的处理,其中$r_{34childdummy} = r_4 - r_{4child1}$,$r_{3childdummy} = r_3 - r_{3child1} - r_{34childdummy}$;若一个潜在哑节点的上级父节点同样是哑节点,则只需直接添加一个权限集合相同的下级哑节点即可,如图5(d)所示。

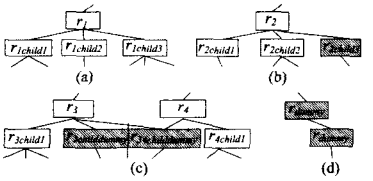


图 5　规范化应用过程示例

Graham 等人在文献[21]中提出了一种哑节点聚类的方法。将同一父节点之下的多条到达底层叶节点的哑节点通路合并，使层之间的节点连边尽量少地交叉，以构建更加美观的节点连接布局。虽然这种方法的可视化效果良好，但是我们并不在规范化过程中采纳该种聚类方法，因为上述的多亲树规范化算法的输出结果中不可能会产生文献[21]中的同父节点多通路情况，而至多只有一条全哑节点的通路。至此，角色可视化的前期工作就全部完成了。

4　角色可视化管理模型的设计

由于实际的 RBAC 角色管理中有大量的角色-权限和用户-角色关系需要管理，若将所有的关系配置在一个图层中，将导致整个视图的混乱。因此，考虑到角色-权限、用户-角色以及角色-角色关系之间的结构特点，提出双层可视化布局：下层用于展示角色层次结构和权限，其中权限作为角色层次的叶节点配置在层次结构的底层；上层用于排布用户节点。所有的关系用节点之间的连边表达，且使用图层和用户-角色连边的透明处理以减少用户在使用初期信息过载的几率。

4.1　角色层次布局

角色层次布局的重点是清晰地展示整个多亲树层次结构以及其中包含的互斥约束等。为了方便约束关系的处理，假设互斥关系只存在于同层的角色之间，且只存在于两个角色之间而非多个。互斥关系能够在层之间传递。如果两个父角色分别包含了一个互斥关系集合中的一个且互不相同的子角色，则两个父角色也互斥。这种互斥关系的传递也适用于哑角色之间或者哑角色与真实角色之间。

角色节点的布局主要使用了一种改进的 DAG 启发式绘制算法，因为角色层次亦可看作是 DAG 的一个特例。当前，可用的 DAG 启发式算法很多，其中最常用的一种是 Eades 和 Wormald 所设计[25]并使用 Barth 等人的方法[26]计算边交叉数量。综合考虑互斥信息，我们将问题节点（包含互斥子角色或者权限的父角色）配置在最左边，成为孤立节点。首先，待布局的角色层次应该具有清晰的层次结构和层内排序。在本应用中即角色节点按照层次结构有序地安排在各层之上，且依据角色隶属于组织结构的关系按一定顺序自左向右排列。

从自上而下第二层开始，该层的每一个节点找到与自己直接相连的上层邻居节点，计算这些节点位置的中位数值作为配置该当前节点的位置；该层的全部计算结束后，将排序过程中检测到的问题节点单独放置在左边，距离层次结构最左边节点 L 长度的位置，从而依据互斥信息改变了之前的排序结果。选择中位数启发式算法的原因在于，这种算法与重心启发式算法相比对节点的初始分布不太敏感。将上述的排序过程自上而下地运行。然后又反向地自下而上运行，循环往复，直到终止条件（如一定的循环次数、边交叉达到一定阈值）。至此，所有问题节点就被排开在层次结构的左边，形成易于被用户感知和辨认的一群孤立节点。

图 6(a)中 r_2 是一个拥有两个互斥角色节点 r_5 和 r_7 的问题角色节点。按照上面的算法，经过处理后可获得如图 6(b)所示的合理布局的部分角色层次结构。

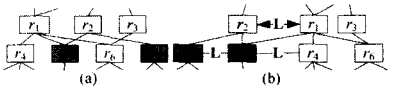


图 6　算法处理的效果图

4.2　用户节点布局

双层可视化布局的上层中使用圆形节点代表用户，节点旁边附带用户的名称，并引入力导引布局方式将其组织在一起。鉴于 Fruchterman 等人设计的算法[27]具有较好的效果和较高的效率，我们直接使用该算法计算所有节点的位置。布局过程中，如果两个用户节点拥有相同的权限，则在布局中认为该两个节点是相连的，即有力的作用，但这种联系并不在可视化视图中展示。引力存在于相连的节点之间，而斥力存在于所有的节点之间；并约定若两个用户节点共享的权限越多，则它们之间的斥力越大，节点之间越靠近。这样，拥有相同权限的节点就会自动靠近在一起，甚至形成聚类结构。此外，增加用户节点和角色节点之间的引力作用，使用户节点的布局亦接近一准层次结构。

5　实验原型系统

针对之前设计的角色可视化管理模型，我们开发了一个实验原型系统。提出一种双层可视化范例，以较好地展示角色管理过程，其中下层用于展示角色层次和权限，上层用于配置用户节点。此外，针对所述问题给出若干交互方法，以可视地辅助解决角色管理中的约束和冗余问题。可视化系统的整体设计如图 7 所示。图中左边使用传统的缩进列表，以给用户提供一个熟悉的角色层次可视化结果；右边使用双层可视化布局展示角色-权限、角色-角色以及用户-角色之间的关系，顶部配置一个工具栏（DM 是互斥检测按钮；DP 是冗余检测按钮）；两个视图协同动作；另外，菜单栏还附带了所有的系统功能选项。

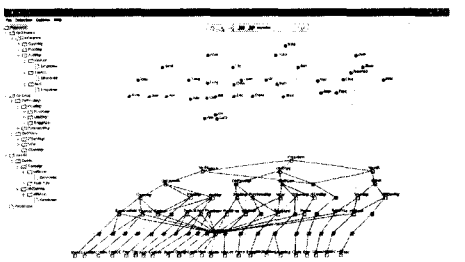


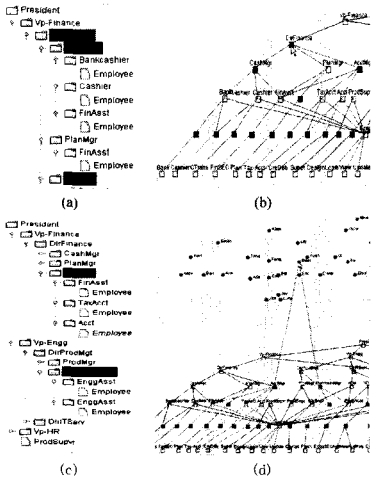
图 7　角色管理原型系统

该角色管理系统已经赋予了一些基本操作以满足管理的需要，如缩放、基于关键词的动态查询、缩进列表，以帮助用户快速领会系统的使用。缩进列表视图和双层布局视图这两个视图能够同步进行操作和显示。接下来，设计针对角色管理应用需要的交互技术，以进一步解决问题(2)和(3)。

5.1　互斥检测

RBAC 角色管理应用中至少存在两种类型的互斥：一是包含互斥权限或者子角色的角色，二是包含互斥角色的用户。利用高亮显示通知用户检测到互斥的存在：对于前者，使用红色高亮，存在互斥关系的子角色则使用蓝色高亮。若鼠标放

置在角色节点上方，以该节点为根节点构建的多亲树会同时被高亮显示；对于后者，使用红色高亮显示处在上层面的问题用户节点。当鼠标放置在上层问题用户节点上方时，用户和授权的角色以及其下构建的多亲树被同时高亮显示，互斥的角色节点蓝色高亮。通过这些可视化的辅助，用户能便捷地根据当前视图的提示并综合组织的实际情况进行角色授权的优化。互斥检测截图如图 8 所示。



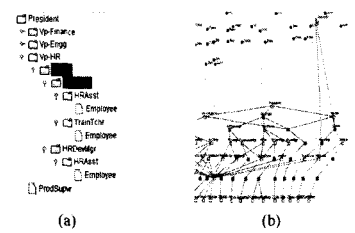
(a)、(b)中问题角色 DirFinance 被红色高亮显示，包含两个互斥的子角色 CashMgr 和 AcctMgr；(c)、(d)中问题用户 Evan 同样被红色高亮显示，包含两个互斥的角色 AcctMgr 和 PurchaseMgr。

图 8　互斥检测截图

5.2　冗余检测

通过检查用户-角色关系，用户能够发现当前角色管理中的冗余问题。在用户层中，当鼠标放置在某节点之上时，用户节点及对其授权的角色之间的连边和这些角色下构建的多亲树均会高亮显示，直观地展示当前用户-角色和用户-权限之间的授权情况。

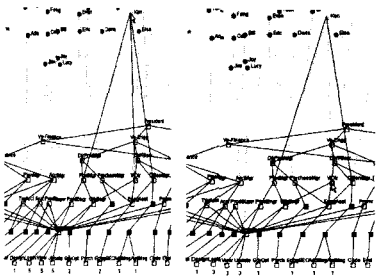
若一个用户被授权了冗余的角色，用户节点将闪烁，以通知用户检测到冗余的存在。当鼠标放置在该节点之上时，高亮显示使相关的用户-角色结构突出显示。冗余的角色节点呈紫色，因而用户能够容易地辨认所有的关键位置，并通过删除连边来解除对应的授权。



用户 Amanda 由于被赋予两个重复的角色 DirHR 和 TrainMgr 而在冗余检测中呈现闪烁提示。

图 9　冗余授权检测

在角色节点闪烁的同时，权限下方将显示一系列数字，提示用户当前该权限被授权于用户的次数。用户可通过将鼠标放置在待删除的角色节点上来查看若取消该角色授权后用户拥有的权限情况，确认删除该角色后用户的权限是否发生变化。若暂时取消授权后，权限下方出现数字 0，则对应的权限将变灰，提示当前操作用户将不再拥有该权限。用户在交互中完成了对角色管理中的冗余问题的处理和优化。图 9、图 10 展示了上述系统冗余问题检测的可视化效果。



(a) 原先的授权方案　(b) 新的授权方案

权限下方的数字显示了该权限被指派的次数。

图 10　优化用户-角色授权过程

结束语　针对 RBAC 角色管理中存在的问题，本文从信息可视化的角度，提出一个双层可视化布局来方便角色管理，其中下层排布角色层次和权限，上层配置用户节点。根据角色管理的需要，设计多项交互技术辅助用户浏览和管理，这些技术包括问题节点高亮和闪烁、解除和优化操作等。

在下一步工作中将考虑约束关系中的其他约束，如前提约束和势约束，以更好地进行角色管理。另外，将考虑增加引进 focus+context [28] 等技术，以设计更加得当的交互浏览操作，并进行正式严格的用户评测实验。

参 考 文 献

[1] Sandhu R S. Role-based Access Controll[J]. The Engineering of Large Systems, 1998, 46: 237

[2] Sandhu R S, Coyne E J, Feinstein H L, et al. Role-based access control models[J]. IEEE Computer, 1996, 29(2): 38-47

[3] Sandhu R, Bhamidipati V, Munawer Q. The ARBAC97 model for role-based administration of roles[J]. ACM Transactions on Information and System Security(TISSEC), 1999, 2(1): 105-135

[4] Ferraiolo D F, Sandhu R, Gavrila S, et al. Proposed NIST standard for role-based access control [J]. ACM Trans. Inf. Syst. Secur, 2001, 4(3): 224-274

[5] Nyanchama M, Osborn S. Access Rights Administration in Role-based Security Systems[C]// Proceedings of the IFIP WG11. 3 Working Conference on Database Security VII. 1994: 37-56

[6] Oh S, Sandhu R. A model for role administration using organization structure[C]// ACM. 2002

[7] Sandhu R, Munawer Q. The ARBAC99 model for administration of roles[C]// Citeseer. 1999

[8] Guo Qi, Vaidya J, Atluri V. The Role Hierarchy Mining Problem: Discovery of Optimal Role Hierarchies[C]// Proceedings of the 24th Annual Computer Security Applications Conference (ACSAC). 2008

[9] Colantonio A, Di Pietro R, Ocello A, et al. A formal framework to elicit roles with business meaning in RBAC systems[C]// ACM. New York. NY, USA, 2009

[10] Molloy I, Li N, Li T, et al. Evaluating role mining algorithms[C]// ACM. New York. NY, USA, 2009

[11] Nyanchama M, Osborn S. The role graph model and conflict of interest [J]. ACM Trans. Inf. Syst. Secur, 1999, 2(1): 3-33

[12] Reingold E M, Tilford J S. Tidier drawings of trees [J]. IEEE Transactions on Software Engineering, 1981: 223-228

[13] Beebe N. A Bibliography of Tree Drawing Algorithms[M]. Salt Lake City: Department of Mathematics, University of Utah, 2006: 22

[14] Plaisant C, Grosjean J, Bederson B B. SpaceTree: Supporting explora-tion in large node link tree, design evolution and empirical evaluation[J]. The Craft of Information Visualization: Readings

and Reflections, 2003: 287

[15] Robertson G G, Mackinlay J D, Card S K. Cone trees: animated 3D visualizations of hierarchical information[C]// ACM. New York. NY, USA, 1991

[16] Yee K P, Fisher D, Dhamija R, et al. Animated exploration of dynamic graphs with radial layout[C]// Citeseer. 2001

[17] Johnson B, Shneiderman B. Tree-maps: A space-filling approach to the visualization of hierarchical information structures[C]// VIS'91 Proceedings of the 2nd Conference on Visualization'91. Los Alamitos, CA, USA: IEEE Computer Society Press, 1991

[18] Zhao S, Mcguffin M J, Chignell M H. Elastic hierarchies: Combining treemaps and node-link diagrams[C]// Citeseer. 2005

[19] Balzer M, Deussen O. Hierarchy Based 3D Visualization of Large Software Structures [C]// Proceedings of the Conference on Visualization '04. IEEE Computer Society, 2004

[20] Nguyen Q V, Huang M L. EncCon: an approach to constructing interactive visualization of large hierarchical data[J]. Information Visualization, 2005, 4(1): 1-21

[21] Kennedy J. Exploring Multiple Trees Through DAG Representations[J]. IEEE Transactions on Visualization and Computer Graphics, 2007, 13(6): 1294-1301

[22] Graham M, Kennedy J, Downey L. Visual comparison and exploration of natural history collections [C]// Proceedings of the Working Conference on Advanced Visual Interfaces. ACM, 2006

[23] Sigman E, Wittenburg K. Visual Focusing and Transition Techniques in a Treeviewer for Web Information Access[C]// IEEE Symposium on Visual Languages(VL'97). 1997

[24] Furnas G W, Zacks J. Multitrees: enriching and reusing hierarchical structure[C]// ACM. New York. NY, USA, 1994

[25] Eades P, Wormald N C. Edge crossings in drawings of bipartite graphs[J]. Algorithmica, 1994, 11(4): 379-403

[26] Barth W, Junger M, Mutzel P. Simple and efficient bilayer cross counting[J]. Lecture Notes in Computer Science, 2002, 2528: 130-141

[27] Fruchterman T, Reingold E M. Graph drawing by force-directed placement[J]. Software-Practice and Experience, 1991, 21(11): 1129-1164

[28] Card S K, Mackinlay J D, Shneiderman B. Information visualization: Using vision to think[M]. San Francisco: Morgan Kauffman Publisher, 1999

(上接第 29 页)

目增加一倍，3 个类别的延迟比预设为 1∶2∶4。另外将系统服务线程的总数设置为 120 个。根据式(8)，为向 3 个服务类别提供比例延迟保证，需要使用两个控制器，构建两个控制回路。第一个控制器作用于 A、B 两个相邻类别；第二个控制器作用于 B、C 两个相邻类别。在每个采样时刻可以根据式(8)计算 A、B、C 3 个服务类获取服务线程的比例。由仿真结果(见图 7)可知，在两个控制器的作用下，Apache 服务器较好地为 3 个服务类别提供了比例延迟保证，达到了预期的目标。

表 2　系统仿真配置 2

客户机	服务类别	客户数目	运行时段/秒
1	A 类(高)	50	0～1200
2	B 类(中)	50	0～1200
3	C 类(低)	50	0～1200
4	B 类(中)	100	300～750

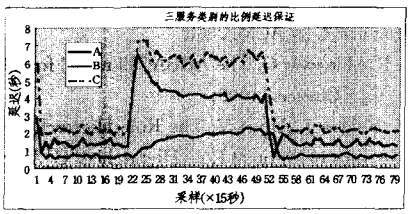


图 7　3 个服务类别的比例延迟保证

结束语　本文首先分析了 Apache 服务器的几种等效模型及其在服务器性能调节与改进方面的应用。队列模型适合于对服务器进行分析和对其各种指标做出预测，而控制模型则能实现服务器端参数的动态设置，并提供特定的区分服务和性能保证等。接着本文在 Window 平台上通过系统辨识和控制器设计，实现了闭环系统中 Apache 服务器上的面向多个服务类别的比例延迟保证。仿真表明系统辨识获得的 LTI 模型以及设计的反馈控制器都是有效的。如何选择合适的系统指标，建立更加精确的等效模型，实现服务器集群的性能改进，是下一步研究的重点。另外如何将队列的预测机制与控制方法相结合，实现基于会话或请求的准入控制，也是需要解决的问题。

参 考 文 献

[1] The netcraft web server survey [OL]. http://www.netcraft.com

[2] Voigt V, Tewari R, Freimuth D, et al. Kernel Mechanisms for Service Differentiation in Overloaded Web Servers[C]// Usenix Annual Techanical Conference. 2001

[3] Abdelzaher T F, Stankovic J A, Lu Chengyang, et al. Feedback Performance Control in Software Services [J]. IEEE Control Systems, 2003, 23(3)

[4] Lu Chengyang, Abdelzaher T F, Stankovic J A, et al. A Feedback Control Approach for Guaranteeing relative Delays in Web Servers[C]// Proceedings of IEEE Real-Time Technology and Applications Symposium. TaiPei, Taiwan, June 2001

[5] Bloom R B. Apache Server 2. 0-The Complete Reference[M]. America: McGraw-Hill, Inc, 2002

[6] Gandhi N, Parekh S, Tilbury D, et al. Using MIMO feedback control to enforce policies for interrelated metrics with application to the Apache web server[C]// Proc. of the Network Operations and Management Symposium. 2002

[7] Cao Jianhua, Andersson M, Nyberg C, et al. Web Server Performance Modeling Using an M/G/1/K * PS Queue[C]// 10th International Conference on Telecommunication. Papeete, Tahati, 2003

[8] 郭齐胜，等. 系统建模原理与方法[M]. 长沙：国防科技大学出版社，2002

[9] The Apache Software Foundation[OL]. http://www.apache.org

[10] Gandhi N, Parekh S, Hellerstein J, et al. Feedback Control of a Lotus Notes Server: Modeling and Control Design[C]// American Control Conference. 2001

[11] Barford P, Crovella M. Generating Representative Web Workloads for Network and Server Performance Evaluation [J]. Measurement and Modeling of Computer Systems, 1998: 151-160

[12] Liu Zhen, Nicolas N, Villanueva J. Traffic Model and Performance Evaluation of Web Servers[J]. Performance Evaluation, 2001: 77-100