计算机研究与发展
Journal of Computer Research and Development
ISSN 1000-1239/CN 11-1777/TP
44(12):2020~2027, 2007

一种基于层次命名空间的 RBAC 管理模型

夏鲁宁　　荆继武
(中国科学院研究生院信息安全国家重点实验室　北京　100049)
(halk@lois.cn)

An Administrative Model for Role-Based Access Control Using Hierarchical Namespace

Xia Luning and Jing Jiwu
(*State Key Laboratory of Information Security*, *Graduate University of Chinese Academy of Sciences*, *Beijing* 100049)

Abstract　Access control is an important information security mechanism. Role-based access control is a famous access control approach with good flexibility and expandability. The classical RBAC models are RBAC96 and ARBAC97. The ARBAC97 model is an administrative model with the idea of "using RBAC to administrate RBAC". It facilitates decentralized administration of RBAC through three assignment models: URA97, PRA97 and RRA97. Though ARBAC97 works well in traditional RBAC applications, it has some shortcomings if employed in a large organization composed of many autonomous subsidiaries. The member of an administrative role can operate directly in the role range of a junior administrative role, which violates the autonomy of subsidiaries. The authorization relationship is rather complex. And the names of the roles have to be globally unique. A new administrative model named N-RBAC is proposed to overcome these weaknesses. In N-RBAC, all resources (including users and roles) are arranged into a hierarchical namespace structure. Thus the role hierarchy is constructed in a local space instead of in a global space. The administrative role hierarchy is obsolete, and a unique administrative role is assigned to each namespace instead. Experimental results show that the N-RBAC model is more suitable to autonomous distributed role administration than the ARBAC97 model.

Key words　RBAC; RBAC96; ARBAC97; N-RBAC; namespace

摘　要　访问控制是一种重要的信息安全机制. 基于角色访问控制(RBAC)提供了一种策略中立、具有强扩展性的框架,使访问控制机制具备了相当的灵活性. RBAC96 和 ARBAC97 模型是基于角色访问控制的经典模型,其中 ARBAC97 定义了一系列的角色管理模型,实现了在 RBAC 模型基础上的分布式管理. 但对于由多个自治的分支机构组成的大规模组织,ARBAC97 模型存在授权关系复杂、允许越级操作以及角色名称必须全局惟一等不足. 为了解决这些问题,提出了一种基于层次命名空间的 RBAC 模型——N-RBAC,使用命名空间来组织角色和资源,各命名空间之间的资源相互不可见. 命名空间结构提供了良好的分布式 RBAC 管理能力,简化了角色继承结构的复杂性,并对局部自治的 RBAC 管理提供有力支持.

关键词　RBAC; RBAC96; ARBAC97; N-RBAC; 命名空间

中图法分类号　TP393.08

收稿日期:2006-10-16;**修回日期**:2007-08-27
基金项目:国家"八六三"高技术研究发展计划基金项目(2003AA144050)

“访问控制(access control)”是一种保护资源免遭未授权访问的安全机制. 在基于角色的访问控制模型(role-based access control,RBAC)出现之前,自主访问控制(DAC)[1]和强制访问控制(MAC)[2]已经提出并在诸多应用领域取得了巨大的成功. 然而随着计算机网络的快速发展和应用系统规模的不断扩大,这两种传统的访问控制模型已经无法适应新的应用环境:它们都无法提供一种策略中立、具有强扩展性的访问控制框架.

RBAC 模型就是在这种背景下被提出来的. 它实际上是一种强制访问控制模型,即用户不能进行自主授权和权限转移;但是它没有如 MAC 中那样限制信息的流向,而是引入了一种抽象的中介元素——角色,来传递授权信息,从而提供了足够的灵活性和扩展性[3].

1 RBAC 与 ARBAC 模型

RBAC 最初的形式化定义来源于文献[4],文中第1次引入了角色的概念并给出其基本语义. Sandhu 于 1996 年提出了著名的 RBAC96 模型[5],将传统的 RBAC 模型根据不同需要拆分成 4 种嵌套的模型并给出形式化定义. 1997 年他们提出了一种分布式 RBAC 管理模型 ARBAC97,实现了在 RBAC 模型基础上的分布式管理[6]. 这两个模型清晰地表征了 RBAC 概念,成为 RBAC 的经典模型. 大多数基于角色访问控制的研究都以这两个模型为出发点[7].

1.1 RBAC96 模型

RBAC96 模型分 4 个层次,分别称为 RBAC0,RBAC1,RBAC2 和 RBAC3. 其中 RBAC0 是基本模型,定义了实体集 U(用户集)、R(角色集)、P(权限集)、S(会话集),用户角色指派关系 $UA\subseteq U\times R$,权限角色指派关系 $PA\subseteq P\times R$,会话到用户的映射 $user:S\rightarrow U$,以及会话到角色的映射 $role:S\rightarrow 2^R$. RBAC1 模型包含 RBAC0 并定义了角色继承结构 $RH\subseteq R\times R$,并将会话到角色的映射关系修改为 $role:S\subseteq\{r\mid(\exists r'\geqslant r)[user(s),r'\in UA]\}$. RBAC2模型同样包含 RBAC0,但是定义了约束. RBAC3 包含 RBAC1 和 RBAC2,自然也包含 RBAC0. 这是完整的 RBAC96 模型,包含一切模型元素. 图 1 给出了 RBAC3 模型的图示.

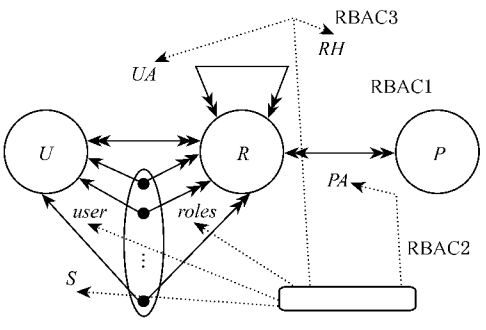

Fig. 1 RBAC3: The complete RBAC96 model.

图 1 RBAC3:完整的 RBAC96 模型

1.2 角色管理模型 ARBAC97

RBAC96 模型没有详细谈到具体如何进行角色管理. Sandhu 等人在 1997 年提出了分布式角色管理模型 ARBAC97[6],从理论上给出了 RBAC 模型中角色管理的办法.

ARBAC97 模型的基本思想是利用 RBAC 模型本身来进行 RBAC 模型的管理. 模型的管理员称做管理员角色,管理员用户通过拥有管理员角色得到管理权. 管理员角色继承结构是一个单独的继承关系,该继承关系上的每个管理员角色对应非管理员的常规角色继承结构上的一部分管理区域,称做角色区间. 角色区间的引入使得对 RBAC 体系实现了分工明确的分布式角色管理. 管理员角色继承结构上每个管理员角色都有自己的角色区间,在自己的角色区间内可以进行用户角色指派(URA)、权限角色指派(PRA)以及角色关系管理(RRA). 高级别管理员继承低级别管理员的管理能力,其角色区间也包含了低级别管理员的角色区间. ARBAC97 模型分为 3 个部分:用户角色指派管理(URA97[8])、权限角色指派管理(PRA97[9])以及角色继承结构管理(RRA97). 有关这 3 个模型的细节内容可以参见文献[6].

2 RBAC 管理中的自治问题

随着社会的发展,商业组织的规模不断扩大. 这些较大规模的组织常常采用比较松散的自治管理形式,其每个分支机构从事不同的业务,并且在人事、部门设置和经营管理上拥有自主权,上级组织不干涉分支机构的自主权. 我们将这些分支机构称为“自治分支机构”. 在这样的组织部署典型的 RBAC96 和 ARBAC97 访问控制体系会产生与分支机构自治权的冲突.

假想一个新闻杂志 VeryNews,由社会栏目(society column)、娱乐栏目(entertainment column)和军事栏目(military column)组成. 每个栏目是一个自治分支,由一个独立编辑团体负责维护. 这些栏目拥有

各自不同的栏目(column)结构和文章审核的工作流程(workflow)等,如图 2 所示:

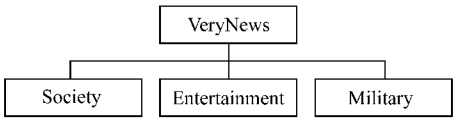

Fig. 2 The organizational structure of VeryNews.

图 2 VeryNews 的组织结构

按照 RBAC96 和 ARBAC97 模型,我们分别给出 VeryNews 的常规角色继承结构和管理员角色继承结构,如图 3 和图 4 所示.

Fig. 3 The role hierarchy of VeryNews.

图 3 VeryNews 常规角色继承结构

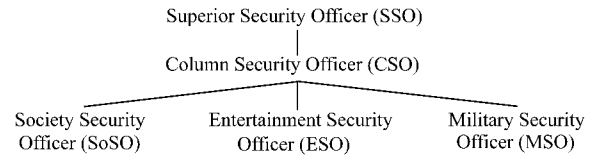

Fig. 4 The administrative role hierarchy of VeryNews.

图 4 VeryNews 管理员角色继承结构

URA97 的 *can_assign* 关系则在表 1 中给出.

Table 1 The *can_assign* Relation

表 1 *can_assign* 关系

Administrative Role	Prerequisite	Role Range
SoSO	ED	[SE,SE]
SoSO	$SE \wedge \overline{SWE}$	[SAE,SAE]
CSO	$ED \wedge \overline{ME}$	[SE,SE]
CSO	ED	(ED,DIR)
SSO	EP	[ED,ED]
SSO	ED	(ED,DIR]

分析图 3、图 4 和表 1,可以看出 RBAC96 和 ARBAC97 应用到 VeryNews 时,存在以下问题:

问题 1. 角色指派关系复杂.

假设 SoSO 试图将新用户 John 加入到 SAE 角色中,那么根据表 1 的先决条件,John 应该首先成为 SE 角色的成员;为了成为 SE 角色的成员,他又必须是 ED 角色的成员;同样地,如果想将 John 加入 ED 角色,他必须是 EP 角色的成员. 所以对 John 的角色指派过程如下所示:

SSO 将 John 指派到 EP,SSO 将 John 指派到 ED,SSO 或 CSO 将 John 指派给 SE,SoSO 将 John 指派到 SAE.

从这个例子可以看出,按照 URA97 的用户角色指派规则,可能需要多步指派才能达到目的. 在角色继承结构中处在越高位置的角色,需要的指派步骤也可能越多. 这导致一次用户角色指派要涉及多个不同级别的安全管理员. 另外在用户角色指派中可能存在一些冗余. 假如 John 已经是 SCL 角色的成员,那么由于角色继承关系,他已经隐式成为 SAE 角色的成员;如果随后 John 又由 SSO,CSO 或者 SoSO 指派到 SAE 角色,那么这个指派将毫无效果. 文献[10]中详细讨论了多步指派和冗余指派的问题.

问题 2. 越级管理.

根据图 4 的继承结构,高级别的管理员角色继承了它下面所有管理员角色的权限. 这意味着高级别管理员角色的成员可以直接在被它所继承的管理员角色的角色区间内进行操作. 例如 CSO 的成员可以将 ED 角色的成员 John 直接指派到 SAE 角色. 而在表 1 所示的 *can_assign* (SoSO, $SE \wedge \overline{SWE}$, [SAE,SAE])关系中,对于 SoSO 来说,只有已经是 SE 的成员,并且不是 SWE 的成员的用户,才可以被指派给 SAE. 上述 CSO 所进行的指派操作把不符合这个先决条件的用户 John 指派给了 SAE,实际上是绕过社会栏目的安全管理员 SoSO 而直接对社会栏目的管理结构进行干涉. 如前所述,社会栏目是 VeryNews 的一个自治分支,作为高级别安全管理员的 CSO 不允许干涉该栏目内部管理,所以 CSO 的上述操作与该栏目的自治权产生冲突.

问题 3. 角色命名问题.

对于大规模的组织,角色设置相当多,这意味着角色继承结构上节点众多. 显而易见,这些节点是不能重名的,例如图 3 中的 SAE 和 MAE 尽管都表示文章编辑(Article Editor)的角色,但必须加上"society"或"military"的栏目名作为前缀才能区分二者,如果都命名为 Article Editor,将无法知道是哪个

栏目的文章编辑. 如果要在 SAE 之下再增加一个角色,那么必须命名为类似“Society ××× Article Editor”这样的长名字. 角色继承关系越复杂,每个角色的命名也越冗长.

关于问题 1,文献[10]中提出了一种解决方案,使用组织结构(organization structure)而不是角色继承结构来表达先决条件. 关于问题 2 和问题 3,ARBAC97 以及文献[10]都没有提出有效的解决方案.

本文提出了一种使用层次命名空间对 RBAC 进行管理的框架. 在此框架中,我们将文献[10]中的“组织结构”进一步抽象成一个层次化的命名空间结构(namespace hierarchy),角色不再存在于全局统一的角色继承结构中,而是分散到不同层次的命名空间里. 各命名空间中的角色彼此不可见,相互之间也没有继承关系,从而有效解决了越级管理和名字重用问题. 我们将其称做“基于命名空间的 RBAC(namespace-based RBAC,N-RBAC)”.

3 N-RBAC 模型

3.1 命名空间

“命名空间”在计算机科学领域的许多地方都被用到. 参照文献[11],给出如下定义:

定义 1. 命名空间(namespace)是一个限定的范围,用来对名字集合做划分. 在同一个命名空间内定义的名字是惟一的,并且这些名字的视野被限定在本命名空间范围内.

这个命名空间的定义类似于 X.500[12] 目录中的“自治管理域(autonomous administrative area,AAA)”的概念,其中的“名字”,指的是 RBAC 中所有被定义的符号,可以是用户名、角色名、权限名以及其他一切资源的名称. 如果在同一个命名空间内有多个“Article Editor(AE)”的角色,那么只能定义成诸如 SAE,MAE,AE1,AE2 等等;但是不同命名空间之间的名字相互不可见,例如空间 Society 内的 AE 和空间 Military 内的 AE 相互不可见,因而都可以使用 AE 作为名字. 在每个空间内部执行操作时,“AE”这个名字仍是惟一;如果需要在多个命名空间之间执行操作,那么可以以命名空间的名字作为前缀,例如 Society.AE,Military.AE. 这是一种分段命名的方式 在局部范围使用时可以用较短的名称来表示一个角色. 在一个构造合理的命名空间结构中,每个空间内的角色继承结构会非常简洁,定义的符号也不会太多,因而这种分段式的角色命名有效缓解了问题 3. 事实上这种分段式命名类似于 X.500 目录中的“相对识别名(relative distinguish name,RDN)”的概念. 因而在 N-RBAC 模型工程实现时,使用 X.500 目录来构造信息的存储结构或检索信息,比使用传统的关系数据库更易于理解.

命名空间是有树状层次结构的,一个命名空间可以有子命名空间;但是各个命名空间都是自治的,任何两个命名空间,即使是直接上下级的命名空间,它们之间的资源也都相互不可见.

3.2 资源与操作

在 RBAC96 以及 ARBAC97 模型中,“权限”是作为单一概念描述的. 事实上在访问控制中,“权限”不是一个原子的概念,可以进一步分解为对某客体执行的操作. 客体可以是物理或逻辑的实体,而操作则是施加在客体上的某种动作,如“修改”、“删除”等.

N-RBAC 将访问控制客体称为“资源”,因而权限集 P 被分解成资源集 RS 和操作集 O,$P \subseteq RS \times O$. 资源集 RS 表示访问控制中所有可以被用户访问的资源的集合,如文件、数据、参数等. N-RBAC 中,这些资源分布在命名空间层次结构中,不同命名空间内的资源相互不可见. 操作集 O 表示可以对一个资源进行的操作的集合,例如添加、删除、修改等. 这些操作对所有的资源都是统一的,因而与命名空间结构无关.

N-RBAC 中,“用户”、“角色”和“命名空间”也是资源的一种,对它们的添加、删除等操作也形成权限.这些权限只能由管理员角色所拥有. 在第 3.3 节中将详细叙述管理员角色. 需要注意的是:命名空间内的“命名空间”资源,指的是本空间直接下级的子命名空间.

引入命名空间、资源和操作的概念后,我们对 RBAC0 模型做相应修改,得到如下的 N-RBAC0 模型.

定义 2. N-RBAC0 模型包含如下元素:

1) 若干实体集 U(用户集)、R(角色集)、RS(资源集)、O(操作集)、S(会话集)、N(命名空间集);

2) $UA \subseteq N \times U \times R$,为多对多的用户角色指派关系,$\forall n \in N, .n.UA \subseteq U \times n.R$. 其中 N 表示命名空间集合,n 表示 N 一个成员,即一个具体的命名空间;

3) $PA \subseteq N \times RS \times O \times R$,为多对多的权限角色指派关系,$\forall n \in N, .n.PA \subseteq n.RS \times O \times n.R$.

其中 N 表示命名空间集合,n 表示 N 一个成员,即一个具体的命名空间;

4) $user: S \rightarrow U$,映射每个会话到一个用户;

5) $role: S \rightarrow 2^R$,映射每个会话到一组角色,这些角色分布在多个不同级别的命名空间中. $roles(s) \subseteq \{r \mid (user(s), r) \in UA\}$,并且会话 s 拥有权限 $\bigcup_{r \in roles(s)} \{(rs, o) \mid (rs, o, r) \in PA\}$.

从以上定义可以看出,N-RBAC0 与 RBAC0 模型的显著不同在于:角色和其他用户资源不再是全局定义的,而是封闭在不同层次的命名空间内. 注意到上述定义中的用户集 U 并没有分布到各个命名空间,而是专属于根命名空间. 这意味着用户被统一管理,而不是由各自治的命名空间自行管理. 这样做的理由是,多数的组织,即便由多个自治机构组成,其人事权也常常是统一管理的,而不能自主决定. 因而用户资源放在整个组织高层统一管理比较适当. 但这并非强制性的,也可以将用户集分散到各个命名空间进行分散管理.

类似地,可以定义 N-RBAC1 模型,给 N-RBAC 加入角色继承关系. 根据上述描述,不同命名空间内的角色不可见,也不存在继承关系,所以这种继承关系只能存在于一个命名空间内部.

定义 3. N-RBAC1 模型包含如下元素:

1) $U, R, RS, O, S, N, UA, PA, user$ 与 N-RBAC0 一致;

2) $\forall n \in N, n.RH \subseteq n.R \times n.R$ 是 $n.R$ 上的偏序关系,记为 $\geqslant$,称做角色继承. 其中 n 表示命名空间集 N 的一个成员,即一个具体的命名空间;

3) $role: S \rightarrow 2^R$ 修改为 $role: S \subseteq \{r \mid (\exists r' \geqslant r)[user(s), r' \in UA]\}$,同时会话 s 拥有权限 $\bigcup_{r \in roles(s)} \{(rs, o) \mid (\exists r'' \leqslant r)[(rs, o, r'') \in PA]\}$.

有关 N-RBAC2 和 N-RBAC3,较之 RBAC96 模型没有变化,不再赘述.

3.3 管理员角色

RBAC 中的管理员角色指的是那些能够对"用户"、"角色"和"命名空间"资源进行操作的角色. 在 ARBAC97 中,管理员角色是一个如图 4 所示的单独的继承关系;而在 N-RBAC 中由于引入了"资源"和"操作"两个概念来替代原来单一的"权限"概念,"用户"、"角色"和"命名空间"也都成为资源之一,所以管理员角色与非管理员角色在表达方式上统一由角色集合 R 统一表达.

原则上可以在单个命名空间范围内实现管理员角色的继承结构,其实现方式与 ARBAC97 相同. 但是回顾 ARBAC97 模型,管理员角色继承结构的初衷是便于分布式管理,将各级角色区间的权限交由各级管理员来控制. 在 N-RBAC 中,分布式管理已经完全体现在了命名空间结构,在各个命名空间内部对用户、角色或命名空间资源进行操作的权限不宜再扩散,因而 N-RBAC 不再定义管理员角色继承结构,而在每个命名空间内定义惟一的管理员,并定义如下两个基本约束:

约束 1. 在 N-RBAC 中,每个命名空间中有且只有一个管理员角色.

约束 2. 在 N-RBAC 中,对"用户"、"角色"和"命名空间"3 种资源的操作权限只能指派给各命名空间管理员角色.

定义了这两项约束后,命名空间的管理员角色将是惟一能对本空间内角色继承关系进行管理的角色,同时它们也有权新建、编辑或删除下一级的命名空间. 对用户进行操作的权限则由根命名空间的管理员惟一拥有.

同非管理员角色一样,上级命名空间的管理员不会继承下级命名空间管理员的权限,因而也无法直接更改下级命名空间的 URA,PRA 和 RRA 关系,这些操作只能由该空间的管理员来做. 这样,问题 2 得到解决.

3.4 命名空间和管理员角色的创建

根据第 3.3 节中的描述,每个命名空间只有惟一的管理员角色,不同级别之间的管理员没有继承关系. 那么,这些管理员角色将由谁添加、删除和编辑呢?命名空间本身又如何产生的?

N-RBAC 中采用了简单的方式来处理这个问题."命名空间"作为资源的一种,由上级命名空间的管理员创建并负责指派给用户;当每个命名空间产生时,该空间的管理员伴随而产生,管理员的生命期与命名空间相同. 显然,管理员角色是不可编辑的——因为没有别的角色能够控制它. 因而,必须在管理员角色被创建的同时就为其赋予权限,并且在其整个生命期中权限不再改变.

N-RBAC 并不限定管理员角色应具有哪些权限.作为命名空间的惟一管理者,它可以具备对本空间内所有资源的全部操作权限. 但为了胜任管理职能,管理员角色需要具备一个最小权限集合,定义如下:

定义 4. 命名空间管理员角色的最小权限集合,包括对"用户"、"角色"和"命名空间"3 种资源的全

部操作.

虽然管理员可以拥有命名空间内的所有权限,但仅指派给它最小权限是有积极意义的:这体现了"权责分离"的思想,管理员的工作只是管理 N-RBAC 体系结构,而无权过问用户资源.

3.5 N-RBAC 中的 URA,PRA 和 RRA

N-RBAC 与 RBAC96 模型相比最大的变化是增加了命名空间的概念,从而使得角色继承关系不再是全局的,而是分布在各个命名空间中. 与 ARBAC97 模型相比,除增添了命名空间结构带来的变化外,还摈弃了管理员角色继承结构,而以每个命名空间惟一的、不可编辑的管理员角色来代替. 这样大大简化了 URA,PRA 和 RRA.

回忆 ARBAC97 中的 $can\_assign(x,y,[a,c])$ 关系,在 N-RBAC 中,每个命名空间只有惟一的管理员,因而在空间内部 x 取值为常数;由于管理员角色继承结构的消失,"角色区间"的概念已经没有用处了:每个管理员的角色区间都覆盖所在命名空间的整个常规角色继承结构;在命名空间的范围之内,管理员角色可以将任何常规角色直接指派给用户,而没有必要先指派低级别角色,再指派高级别角色. 这大大简化了角色指派步骤,从而使得命名空间内部任何的角色指派都只需要一步,从而有效改善问题 1 中的困难. 在 N-RBAC 中,can_assign 关系事实上已经不存在,这是因为在一个构建合理的命名空间结构中,分布式管理的复杂性已经全部由命名空间的层次结构来承担,而无须再去定义复杂的 URA 模型. 同样的原因,PRA 的 $can\_assignp$ 关系也不再存在.

RRA 的 can_modify 关系,也因管理员角色的惟一和管理员角色继承结构的消失而没必要存在了;但是在命名空间内部的常规角色继承结构中,添加、删除角色和修改角色的继承关系仍旧可能带来一致性的歧义,所以必要时仍可引入 ARBAC97 中对于 can_modify 关系的各种一致性约束,可参阅文献[6].

4 N-RBAC 的实现示例

作为一个演示,我们按照图 2 中的组织结构实现了上述的 VeryNews 的 N-RBAC 体系. 按照自治关系,VeryNews 的 N-RBAC 体系是一个两层的命名空间结构,根空间为 VeryNews,3 个子空间分别是 Society,Entertainment 和 Military 三个栏目. 资源集 RS 定义为 $RS=\{Article,Column,Template,Workflow,Role,User\}$. 其中 $Article$ 包括已发布和未发布的文章;$Column$ 表示栏目,在本示例中等价于命名空间;$Template$ 是文章的布局模板,不同栏

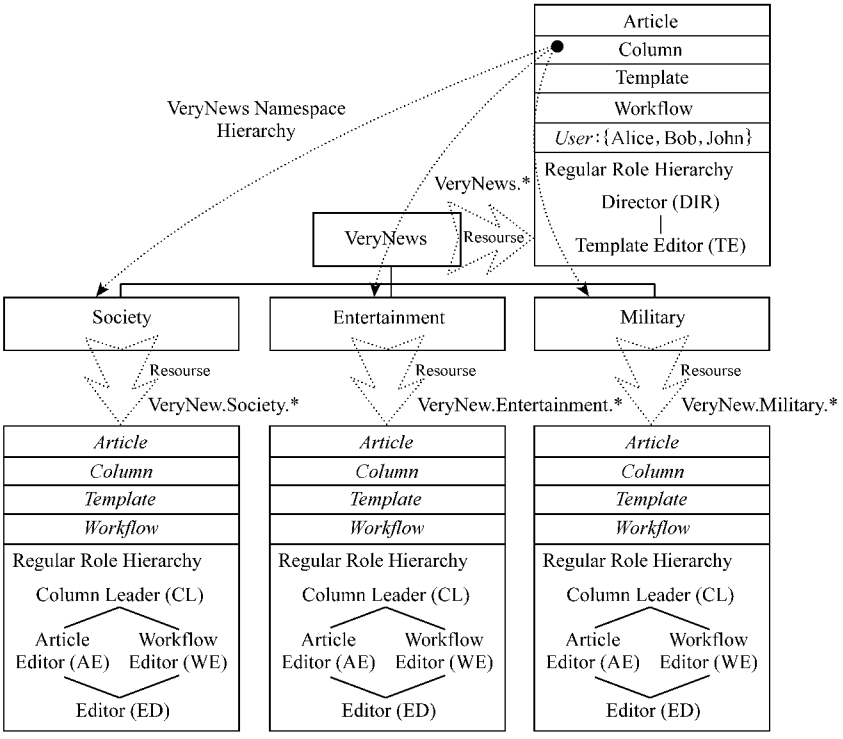

Fig. 5 The namespace hierarchy of VeryNews.

图 5 VeryNews 命名空间结构

目可以使用不同风格的文章模版;*Workflow* 表示工作流,在某个栏目内,创建、编辑、审核、发布一篇文章所要遵循的流程;*Role* 为常规角色和管理员角色;*User* 为用户. 对于任意一种资源的操作集合 $O=\{Create, Modify, Enable, Disable, Delete\}$.

整体命名空间层次结构以及各级空间中的资源如图 5 所示.

简单起见,这里只实现了 2 层的命名空间结构. 但它可以容易地扩展为更多层次. 如上所述,在 VeryNews 的 N-RBAC 体系中,"栏目"就是命名空间,Sociey,Entertainment 和 Military 三个栏目都可以进一步划分更低层次的子栏目. 每个栏目都是一个自治单元,当某栏目的管理员新建子栏目时,新的下级命名空间就产生了,其管理员角色也伴随而产生. 例如 VeryNews.Society.SO 可以创建一个"热点聚焦(Focus)"子栏目,这就是一个新的子命名空间,其管理员角色 VeryNews.Society.Focus.SO 也随之而创建.

5 结 论

RBAC 模型的引入为访问控制提供了足够的灵活性和扩展性. RBAC96 模型是对 RBAC 的经典诠释,ARBAC97 模型为分布式 RBAC 管理提供了很好的框架和思路.

应用在分支机构自治管理方式的组织中,RBAC96 和 ARBAC97 模型存在如下不足:问题 1:角色指派关系复杂;问题 2:越级管理;问题 3:角色命名问题. 本文提出了基于层次命名空间进行 RBAC 管理的思想,并给出 N-RBAC 模型,将角色继承关系和各种资源限制在树状层次结构的多个命名空间内,不同命名空间内的资源相互不可见. N-RBAC 中不再使用管理员角色继承结构,而代之以每个命名空间惟一的管理员角色,和固定不变的权限设置. 命名空间内的用户角色指派不必逐级指定,管理员可以直接将需要的角色指派给用户,从而简化了用户角色指派的步骤,有效解决问题 1. 不同的命名空间的管理员角色之间没有继承关系,因而高层次命名空间的管理员不能直接管理低层次命名空间内的各种资源,从而解决问题 2. 各命名空间之间的角色以及其他资源都相互不可见,因而可以使用同样的名称,而以命名空间的不同来区分他们,从而解决问题 3. N-RBAC 有良好的扩展性和兼容性,根据需要,在某个命名空间内部仍可以无差别地实现 RBAC96 模型和 ARBAC97 模型,从而为已有的访问控制体系升级到 N-RBAC 提供便利.

参 考 文 献

[1] B Lampson. Protection [J]. In: Proc the 5th Annual Princeton Conf on Information Sciences and Systems. Princeton, New Jersey: Princeton University Press, 1974. 437-443

[2] R S Sandhu. Lattice-based access control models [J]. IEEE Computer, 1993, 26(11): 9-19

[3] Yu Shipeng. Research on theory and application of role-based access control: [Master dissertation] [D]. Beijing: School of Mathematical Sciences, Peking University, 2003 (in Chinese)
(俞诗鹏. 基于角色访问控制的理论和应用研究:[硕士论文] [D]. 北京:北京大学数学学院, 2003)

[4] D Ferraiolo, R Kuhn. Role based access controls [C]. The 15th NIST-NCSC National Computer Security Conference, Baltimore, MaryLand, 1992

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

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

[7] Long Qin, Liu Peng, Pan Aimin. Research and implementation of an extended administrative role-based access control model [J]. Journal of Computer Research and Development, 2005, 42(5): 868-876 (in Chinese)
(龙勤, 刘鹏, 潘爱民. 基于角色的扩展可管理访问控制模型研究与实现 [J]. 计算机研究与发展, 2005, 42(5): 868-876)

[8] R S Sandhu, V Bhamidipati. Role-based administration of user-role assignment: The URA97 model and its Oracle implementation [J]. Journal of Computer Security, 1999, 7(4): 317-323

[9] R S Sandhu, V Bhamidipati. An Oracle implementation of the PRA97 model for permission-role assignment [C]. In: Proc of the 3rd ACM Workshop on Role-Based Access Control (RBAC'98). New York: ACM Press, 1998. 13-21

[10] S Oh, R Sandhu. A model for role administration using organization structure [C]. In: Sandhu R, Bertino E, eds. Proc of the 6th ACM Symp on Access Control Models and Technologies (SACMAT 2002). New York: ACM Press, 2002. 155-162

[11] Oswego Suny, *et al*. WISR 1993 design-for-reuse working group report [OL]. http://gee.cs.oswego.edu/dl/WISR93WG/WISR93WG/WISR93WG.html, 1993-11-03/2006-03-20

[12] D W Chadwick. Understanding X.500: The Directory [M]. London, UK: Chapman & Hall, 1994

Xia Luning, born in 1977. Ph. D. candidtate. His main research interests include access control, data mining technique, *etc*.

夏鲁宁,1977 年生,博士研究生,主要研究方向为信息与网络安全、访问控制、数据挖掘等.

Jing Jiwu, born in 1964. Professor and Ph. D. supervisor. His main research interests include PKI technology, network security, *etc*.

荆继武,1964 年生,教授,博士生导师,主要研究方向为 PKI 技术、网络安全等.

Research Background

Role-based access control is a focus topic in recent access control researches. Though ARBAC97 enabled decentralized administration for RBAC96 model, it is not suitable to a large organization composed of some autonomous subsidiaries. The main limitation can be concluded as 3 questions. (1) The member of an administrative role can operate directly in the role range of a junior administrative role, which violates the autonomy of subsidiaries. (2) The authorization relationship is too complex. (3) The names of the roles have to be globally unique. These shortcomings reveal the need of distributed and autonomous administrative model. The research of this paper is under this condition. Through the introduction of a hierarchical namespace structure, a new administrative model is designed for autonomous RBAC administration. Experimental results show that this model is very suitable for the role based access control systems in large organizations composed of many autonomous subsidiaries.

第 3 届中国可信计算与信息安全学术会议
征文通知

由解放军密码管理局和中国计算机学会容错专业委员会主办,中国人民解放军信息工程大学电子技术学院承办的"第 3 届中国可信计算与信息安全学术会议"将拟于 2008 年 10 月 25～28 日在河南郑州举行. 录用的英文稿件将在《武汉大学学报(英文版)》(EI 刊源)上发表,录用的中文稿件在核心期刊《武汉大学学报》(正刊)发表,欢迎大家积极投稿.

论文征集范围

会议重点征集可信计算与信息安全理论和技术方面的研究论文. 具体包括(但不限于):

·可信计算体系结构　　·可信软件

·可信硬件　　·网络与通信安全

·密码学　　·信息隐藏

·信息安全应用

征文要求

论文必须为未公开发表且未向学术刊物和其他学术会议投稿的最新研究成果,文稿使用中文或英文书写,字数一般不超过 6000.

重要日期

征文截止日期:2008 年 4 月 30 日

录用通知日期:2008 年 6 月 1 日

投稿方式

本次会议投稿一律通过会议网站 www.tc2008.org 的投稿系统进行.

会议通讯地址

河南郑州商城东路 12 号信息工程大学电子技术学院信息安全研究所　邮编　450004

联系人:李立新　周雁舟

联系电话:0371-63538081;0371-66094401;0431-38081(军)

Email:tc2008_zz@163.com