RBAC 与 MAC 在多级关系数据库中的综合模型

李 斓, 冯登国, 徐 震
(中国科学院软件研究所信息安全国家重点实验室, 北京 100080)

摘 要: 多级安全数据库的安全策略需要各种模型来表达, 访问控制模型是其中之一. 强制访问控制(MAC)模型保证多级数据库中的信息流动符合系统的安全策略. 利用基于角色的访问控制(RBAC)来实现 MAC 能方便多级安全数据库的权限管理. 提出了一种 MAC 与 RBAC 的综合模型, 定义了多级角色与内部角色的概念, 并给出了综合模型中经过修改后的操作, 使得系统能自动地完成符合强制访问控制策略的用户权限的管理. 该模型方便了管理员的权限管理, 适合用户较多, 安全层次比较复杂的多级关系数据库系统. 最后给出了模型的部分实现机制.

关键词: 数据库安全; 访问控制; 综合模型; 基于角色的访问控制; 强制访问控制
中图分类号: TP309 文献标识码: A 文章编号: 0372-2112 (2004) 10-1635-05

A Integrated Model of RBAC and MAC in Multilevel Relation Database System

LI Lan, FENG Deng-guo, XU Zhen
(State Key Laboratory of Information Security, Institute of Software of Chinese Academy of Sciences, Beijing 100080, China)

Abstract: Security policy of multilevel security DBMS is expressed by many models. One of them is access control model. Mandatory Access Control (MAC) model guarantees that information flow in multilevel databases is inaccordance with security policy. Role-Based Access Control can simplify administration of privileges in multilevel databases. This paper proposes an integrated model of RBAC and MAC. Multilevel role and internal role are defined, and some modified operations in the model are presented. Administration of privileges under MAC policy is done by system automatically. The integrated model can simplify administration of privileges, and is appropriate for multilevel relation database system which has many users or complex security levels. Some implementing mechanisms of the model are provided.

Key words: database security; access control; integrated model; RBAC; MAC

1 引言

安全数据库一直以来都受到研究者的广泛关注, 其中有很多的工作都是在多级安全数据库上进行的. 多级安全数据库的访问控制包括自主访问控制(DAC)和强制访问控制(MAC), 它们共同完成数据库中的权限分配. 基于角色的访问控制(RBAC)的出现方便了系统的权限管理[4], 并即将形成一个标准[5]. 许多研究者讨论和研究了如何将 RBAC 应用于多级安全系统, 包括如何使用 RBAC 来模拟 DAC 与 MAC[6]; 如何在不改变多级系统的核心模块情况下使用 RBAC[7]; RBAC 的角色图与 MAC 层次结构的关系[8] 等等. 国内也有关于 RBAC 与传统访问控制之间关系的研究, 包括讨论 RBAC 和 DAC、MAC 模型相比的优越性[10]; 用 RBAC 实现多级关系中向上读, 向下写的安全策略[11]; 或者提出用 RBAC 实现 DAC 和 MAC 的一种方法[12], 作者也曾经研究了多级安全数据库中的 RBAC 模型[13].

利用 RBAC 实现多级关系中向上读, 向下写的安全策略[11]; 或者提出用 RBAC 实现 DAC 和 MAC 的一种方法[12], 作者也曾经研究了多级安全数据库中的 RBAC 模型[13].

以上大部分研究者在使用 RBAC 模拟 MAC 的时候, 都给角色定义了读写的级别, 然后根据这些级别规定角色与权限、用户之间的分配关系, 如果系统的级别比较复杂, 用户也比较多, 工作性质类似而级别不同的用户所能拥有的角色是不一样的, 于是系统中维护的角色数量也将会很大, 这又造成了权限管理的不便. 为了减少系统管理员的工作量, 可以让那些工作性质类似而级别不同的用户共享相同的角色, 这时不能给角色规定级别. 本文定义了多级角色, 使得这些角色可以拥有任何级别对象的权限, 也能分配给任何级别的用户. 同时又定义了内部角色, 并给出内部角色与多级角色之间的关系. 通过激活不同的内部角色, 具有相同的多级角色而级别不同的用户获取的权限是不同的. 内部角色完全由系统自动处理, 管理者处理的角色数量将大大减少. 在本文提出的 RBAC 和 MAC 的综合模型中, 多级角色处于模型的外层, 用于管理者管理权限, 而内部角色由系统自动处理, 是用来实现强制访问控制的基本元素.

2 模型简介

本节将简单介绍强制访问控制模型、RBAC 模型与多级关系数据库模型.

2.1 严格的 BLP 模型

Bell-La Padula 模型[1] 是多级系统中最常用的强制访问控制模型, 我们把系统中的对象称为客体(Object), 用户称为主

收稿日期: 2003-05-08; 修回日期: 2003-11-17
基金项目: 国家 973 项目(No. G1999035802); 国家自然科学基金(No. 60025205, 60273027); 国家 863 项目(No. 2002AA141080)

体(Subject), K(o), K(s)分别代表客体和主体的标签,标签包含两个元素,密级 C(class)和范畴 G(category),因此它是一个二元组(C, G),其中密级 C 是可以比较大小的序列,而范畴 G 则是一个位的集合.标签之间的关系有两种:支配与不可比.一个级别 K_1 支配另一个级别 K_2 当且仅当 $C_1 E C_2$ 并且 $G_1 B G_2$,记为 $K_1 E K_2$(或者 $K_2 F K_1$).如果两个级别不存在支配关系,则它们不可比.模型的规则如下:

Ó(简单安全特性)主体 s 能够读客体 o 必须满足 K(o) F K(s);

Ó(* 特性)主体 s 能够写客体 o 必须满足 K(o) E K(s).

简单安全特性保证用户/不能上读0,而* 特性保证用户/不可下写0.这两个特性控制信息只能在同级之间或从低级向高级流动.为了保证安全数据库的数据完整,我们需要严格的 BLP 模型,采用的是严格的* 特性:

Ó(严格的* 特性)主体 s 能够写客体 o 必须满足 K(o) = K(s).

212 基于角色的访问控制(RBAC)

因为具有很多显著的优点,最近 RBAC 一直受到很大关注.国内外的研究者提出了很多 RBAC 的模型并且对它们进行了深入的研究分析.Ferraiolo 和 Kuhn 第一次提出了 RBAC 的概念和术语[2],此后 Sandhu 等人提出了比较成熟的 RBAC96 模型[4],并对这个模型做了大量的后续研究.美国 NIST 将根据 Ferraiolo 等提出的建议标准[5]推出 RBAC 的标准,下面简单介绍这个标准提出的 RBAC 建议模型中的主体部分.

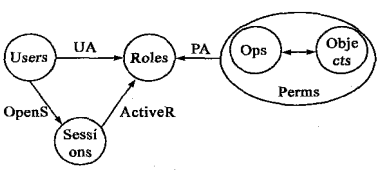

图 1 RBAC 的主体结构

如图 1, RBAC 模型可表示为下面的形式[3]:

3 Users, Roles, Perms, Sessions, Objects, Ops4,

其中:

(1) Users 是用户的集合,每个用户都可以用三元组表示 U= 3 uname, urset, asset4,其中 uname 是用户名, urset 是用户可担当的角色集, asset 是用户打开的会话集.

(2) Roles 是角色的集合,每个角色表示为一个二元组 R = 3 rname, rpset4,其中 rname 是角色的名称, rpset 是角色的权限集合.

(3) Perms 是权限的集合,每个权限是一个二元组, P = 3 obj, opset4,其中 obj 是客体集 Objects 的成员, opset 是访问方法集 Ops 的子集,可能包含不止一种访问方法.

(4) Sessions 是会话的集合,每个用户在登录系统后都会打开一个会话,通过会话激活角色,继而获得角色的权限.每个会话可以表示为二元组 S= 3 sname, arset4,其中 sname 是会话的标识, arset 是会话激活的角色集合.

RBAC 中各组件及其关系共同表示权限的管理,四个操作用来改变组件之间的关系:

(1) 权限分配(PA):

(a)把权限 P 授予角色 R: R. rpset= R. rpset G {P}.

(b)收回角色 R 的权限 P: R. rpset= R. rpset- {P}.

(2)用户分配(UA):

(a)使用户 U 可以担当角色 R: U. urset= U. urset G {R}.

(b)收回用户 U 担当的角色 R: U. urset= U. urset- {R}.

(3)打开会话(OpenS):

(a)用户 U 登录系统,打开一个会话 S: U. asset= U. asset G {S}.

(b)用户 U 退出会话 S: U. asset= U. asset- {S}.

(4)激活角色(ActiveR),只有在会话中激活了角色,用户才真正获取了角色的权限:

(a)用户 U 在某会话 S(S I U. asset)中激活可担当的角色 R: S. arset= S. arset G {R}.

(b)用户 U 在某会话 S(S I U. asset)中去活已激活的角色 R: S. arset= S. arset- {R}.

213 多级安全数据库模型[9, 14]

数据库管理系统的核心是数据,而数据是以记录的形式存放在关系中的.为了方便讨论,我们采用一种比较简单的安全数据库模型:假设安全数据库中的客体只有数据库(D)、关系(T)和记录(E),其中数据库是关系的容器,关系又是记录的容器: E I T, T I D.

多级安全数据库的强制访问控制粒度为记录级,也就是说具有标签的客体的最小单位是记录.用 K(D), K(T)和 K(E)分别表示数据库、关系和记录的标签,这些标签必须满足下面的规则:

规则 1 客体的标签一定要支配其容器的标签.

表示为: P E I T, K(E) E K(T); P T I D, K(T) E K(D).

根据严格的 BLP 模型,系统中的用户与客体之间如果有读写关系,那么它们应该符合下面的规则:

规则 2 用户的标签必须支配它能访问的客体标签.

规则 3 用户创建的客体的标签等于用户的标签.

根据规则 2,如果一个用户能连接数据库 D,则 K(D) F K(U);如果用户能访问数据库 D 中的关系 T,则 K(T) F K(U);如果用户能看到关系 T 中的记录 R,必然有 K(R) F K(U).根据规则 3,一个用户 U 创建的数据库 D 满足 K(D) = K(U),在数据库 D 中创建的关系 T 满足 K(T) = K(U),在关系中生成的记录 R 满足 K(R) = K(U).

3 综合的访问控制模型

安全数据库的权限管理比较复杂,特别是引入了 BLP 模型后,为了保证信息流符合强制访问控制的要求,需要对权限进行更严格的管理.基于角色的访问控制模型(RBAC)能简化系统的权限管理,因此我们希望将 RBAC 和强制访问控制结合在一起,形成一个综合的访问控制模型. RBAC 本身既没有自主特性,也没有强制特性,可以用来模拟自主和强制访问控制.角色是权限的集合,而权限是客体的操作集合,系统给权限对应的客体分配了标签,使得权限也有了标签,这时权限与用户之间的联系将受到标签关系的限制.因此我们需要一个特殊的综合模型,既可以使用 RBAC 模型的功能,又实现了强制访问控制策略.

311 多级角色与内部角色

其它的研究者在利用 RBAC 模型模拟强制访问控制的时候,一般都根据角色的权限给角色设定了读级别(r2level) 和写级别(w2level) ,根据角色的读写级别与用户的标签来决定角色能否授予某个用户. 然而这并不利于权限的管理,例如企业中某部门的高级职员与普通职员分别属于两个级别,他们的工作性质相同但是工作的数据不同,如果按照前面的授权原则,角色管理者必须定义两个角色,这两个角色分别包含不同级别的权限,然后把低级别的角色授予普通职员,把高级别的角色授予高级职员. 如果需要实现的系统级别比较多,权限比较复杂,角色管理员的工作将会变得非常繁杂. 而且我们无法定义角色的写级别: 如果用户可以写一个关系,这并不意味着用户的标签被关系的标签所支配,因为用户可以在低级的关系中插入高级的记录. 因此我们需要引入多级角色的概念.

数据库管理系统的权限一般可以分为三类: 读(P_r) 、写(P_w) 和修改属性(P_u) . 我们曾经讨论过这些权限能否授予某个用户的必要条件[12]: 根据严格的 BLP 模型,如果用户的标签支配客体,则他可以被授予该客体的读权限; 因为权限是针对数据库与关系这样的容器,写客体的过程就是处理更小粒度客体的过程,例如写数据库是生成或删除与用户同级的关系,写关系就是生成、修改或删除与用户同级的记录,所以如果用户的标签支配客体,那么他就可以被授予客体的写权限; 对于修改属性的权限,因为客体的属性对于所有能观察到客体的用户都是可见的,为了保证信息的流动符合 BLP 模型的要求,只允许能观察到客体的最低级用户来执行,也就是与客体同级的用户. 这就产生了两个规则:

规则 4　当用户的标签支配客体的标签,客体的读写权限可以授予该用户.

规则 5　当用户的标签等于客体的标签,客体的修改属性权限可以授予该用户.

为了实现多级角色,我们必须先定义内部角色的概念. 顾名思义,内部角色是只用于系统内部的角色,不为权限管理者和用户所见.

定义 1　内部角色是指系统内部使用的包含某个特定级别客体权限的角色,可以分为两类: 读写角色与修改属性角色. 读写角色(R_{rw}) 只包含某个特定级别客体的读写权限,而修改属性角色只包含某个特定级别客体的修改属性权限(R_u) . 表示如下:

读写角色 R_{rw} = 3 rname, rpset, K_{rw}4, 其中

P P I R_{rw}. rpset 都有 K(P . O) = K_{rw};

修改属性角色 R_u = 3 rname, rpset, K_u4, 其中

P P I R_u. rpset 都有 K(P . O) = K_u.

每个内部角色都是一个三元组: 名称、权限集合与标签. 权限集合中所有权限对应的客体标签都应该等于这个内部角色的标签.

定义 2　多级角色(R_m) 是一种通用的角色,它可以拥有任何级别的权限,也可以授给任何级别的用户. 多级角色没有标签,但是它由具有标签的内部角色组成: R_m = 3 rname, rrset4, 其中

R_m. rrset = {{R^i_{rw}} G {R^j_u}}, 并且

P R^i_{rw}, R^j_{rw} I R_m. rrset, K^i_{rw} X K^j_{rw};

P R^i_u, R^j_u I R_m. rrset, K^i_u X K^j_u.

多级角色是一个二元组: 名称与内部角色集. 它的权限分布在所有隶属于它的内部角色中. 在某个级别上,某种类型的内部角色最多只有一个,而且一个内部角色只能为某一个多级角色服务.

多级角色没有标签,所以系统管理者根据角色的职责,可以把任何级别的客体上的权限授予多级角色,也可以让任何级别的用户来担当这个多级角色. 执行管理操作时不用考虑角色的标签与用户之间的标签存在什么样的关系. 但是为了保证系统的 MAC 特性,用户在激活多级角色时却不能获取所有的权限,只有那些符合 MAC 规则的权限才能被用户使用,而这是用户在某个会话中激活多级角色时系统自动完成的,不需要管理员来干预,所以并没有增加管理员的工作量. 下面我们将讨论引入了多级角色和内部角色后综合模型的结构以及发生变化后的操作.

3.2　综合模型的结构

新的模型可以表示为一个七元组:

3 Users, MRoles, NRoles, Perms, Sessions, Objects, Ops4

(1) Users、Perms、Object 和 Ops 与传统的 RBAC 模型一样.

(2) MRoles 是系统中多级角色的集合,每个多级角色如定义 2 所说的,可以表示为一个二元组, R_m = 3 rname, rrset4, 其中 rname 是角色的名称, rrset 是角色的内部角色集.

(3) NRoles 是系统中内部角色的集合,每个内部角色如定义 1 所描述的,可以表示为一个三元组, R_n = 3 rname, rpset, K_r4. 内部角色集包括两类内部角色: 读写角色和修改属性角色.

(4) Sessions 是系统中会话的集合,与传统的 RBAC 不同,综合模型中每个会话可以表示为三元组, S = 3 uname, amrset, anrset4, 其中 uname 是会话的名称, amrset 是会话激活的多级角色集, anrset 是会话激活的内部角色集.

增加了内部角色与多级角色的概念后,传统的 RBAC 被扩展为图 2 中的结构. 与传统的 RBAC 相比,新模型的角色集合被分成了两个子集))) 多级角色集和内部角色集,其中多级角色集代替了原来角色集的位置,与其它组件的关系仍然与传统的 RBAC 一样. 内部角色对管理员和用户是透明的,它们由系统根据需要创建或删除. 内部角色集与多级角色集、会话和权限也存在着联系,但是这些联系不需要操作直接干预或修改,而是在传统的四种操作执行过程,由系统根据一定的规则自动更改,所以这些联系在图 2 中用的是虚线来表示. 对于管理员来说,综合模型的管理与传统的 RBAC 几乎是一样的. 增加内部角色后,MAC 规则的执行将由系统自动完成,管理员在分配权限和用户的时候不用考虑他们标签之间的关系,也不会增加需要管理的角色数目. 下面我们将分别讨论综合模型中的四种操作以及它们引发的内部变化.

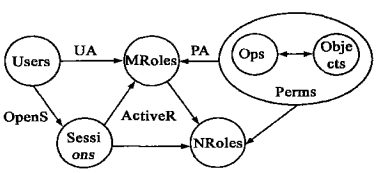

图 2　综合模型的结构

3.3 权限分配

权限分配包括给角色授予权限与收回角色的权限. 管理员在分配权限的时候, 只针对多级角色进行. 同时, 系统自动地将权限分配给某个对应的内部角色, 并完成多级角色与内部角色之间关系的调整. 这样的调整将通过下面两个步骤来完成.

如果管理员给多级角色 R_m 授予了权限 P(O, M), 系统将自动执行下面的步骤:

Ó 获取 P(O, M)的标签, 也就是客体的标签: K(O);

Ó 根据权限 P(O, M)的类型, 获取 R_m 的某个内部角色, 这个内部角色的类型与权限 P(O, M)的类型相同, 而且标签与权限的客体标签相同:

寻找 R_n, 满足 $R_{rw} \in R_m.rrset \cap K(R_{rw}) = K(O)$,

如果 $P(O, M) \in P_{rw}$, 那么 R_n 是读写角色, 否则 R_n 是修改属性角色.

Ó 如果找到了这样的内部角色, 则把该权限授予这个内部角色:

$R_n.rpset \cup \{P\} \rightarrow R_n.rpset$;

Ó 如果没有找到, 则创建这样的内部角色, 并把权限授予这个角色, 同时把这个内部角色添加到多级角色的成员中:

创建 R_n, 且 R_n= 3rname, {P(O, M)}, K_n=4,

$K_n = K(O)$,

并执行: $R_m.rrset \cup \{R_n\} \rightarrow R_m.rrset$.

如果管理员收回多级角色 R_m 的某个权限 P(O, M), 系统将执行下面的步骤:

Ó 获取 P(O, M)的标签, 也就是客体的标签 K(O);

Ó 根据权限 P(O, M)的类型, 获取 R_m 的某个内部角色, 这个内部角色的类型与权限 P(O, M)的类型相同, 而且标签与权限对应的客体标签相同, 如果 R_m 确实拥有这个权限, 那么前面授权机制将保证我们能找到这样的内部角色:

设该内部角色为 R_n, 满足条件 $P \in R_n.rpset$, 此时, $K(R_u) = K(O)$;

Ó 收回找到的内部角色的权限:

$R_n.rpset - \{P\} \rightarrow R_n.rpset$,

Ó 如果该内部角色的权限集成为空集, 从多级角色中去掉这个内部角色, 并在系统中删除它:

如果 $R_n.rpset = \Phi$, 执行

$R_m.rrset - \{R_n\} \rightarrow R_m.rrset$, 并删除 R_n.

3.4 用户分配

多级角色可以让任意级别的用户来担当, 所以修改后的用户分配操作与传统的用户分配类似, 唯一的不同是用多级角色代替原来的角色:

Ó 让用户 U 可以担当角色 R_m: $U.urset = U.urset \cup \{R_m\}$;

Ó 收回用户 U 担当的角色 R_m: $U.urset = U.urset - \{R_m\}$.

3.5 打开会话

用户在登录系统后也会打开会话, 退出登录后会话也会结束, 因此该操作与传统的 RBAC 一样. 但是在强制访问控制策略下, 用户可以在打开会话时选择使用某个标签, 只要这个标签低于管理员分配给用户的标签. 用户选择的标签就是会话的标签:

Ó 用户 U 登录系统时将打开一个会话 S: $U.asset = U.asset \cup \{S\}$,

如果 U 选定的标签为 K, 那么它就是该会话的标签, 而且必须满足 $K \leq K(U)$.

用户结束工作后, 需要退出会话. 因为用户可以同时打开多个会话, 而且这些会话之间是独立的, 所以用户退出某个会话时不会影响其他会话.

Ó 用户 U 退出会话 S: $U.asset = U.asset - \{S\}$.

3.6 激活角色

为了实施强制访问控制策略, 虽然激活了同一个多级角色, 但是不同级别的用户所得到的权限不一样. 管理员只需要将多级角色授予用户, 当用户激活该角色时, 系统会自动地将多级角色中符合 MAC 规则的内部角色授予用户.

当用户 U 在会话 S 中激活多级角色 R_m 时, 系统执行的步骤如下:

Ó 检查管理员是否将多级角色 R_m 授予了用户 U;

Ó 激活多级角色 R_m:

$S.amrset \cup \{R_m\} \rightarrow S.amrset$;

Ó 获取会话的级别 K(S);

Ó 根据规则 4, 只能把客体的读写权限授予那些标签能支配客体标签的主体, 所以用户只能激活那些标签被 K(S)支配的读写角色:

$P \in R_{rw} \cap R_m.rrset$, 如果 $K_{rw} \leq K(S)$, 则执行

$S.anrset \cup \{R_{rw}\} \rightarrow S.anrset$;

Ó 根据规则 5, 只能把客体的修改属性权限授予标签与客体标签相同的主体, 所以用户只能激活那些标签等于 K(S)的修改属性角色:

$P \in R_u \cap R_m.rrset$, 如果 $K_u = K(S)$, 则执行.

$S.anrset \cup \{R_u\} \rightarrow S.anrset$.

激活角色执行成功后, 用户实际上只获得了多级角色的一部分权限. 因为在激活内部角色时需要比较用户与内部角色之间的标签, 所以尽管不同级别的用户都被授予了某个多级角色, 但是他们激活这个多级角色时能获取的权限是不一样的. 系统自动地完成强制访问控制策略的执行, 而不需要管理员专门去处理.

为了降低风险, 如果任务完成, 用户可以在会话中去活某个已经激活的多级角色. 一旦去活了某个角色, 用户在这个会话中就不能再使用该角色的权限. 去活某个多级角色会引起系统自动地执行去活内部角色的操作, 所以用户 U 在会话 S 中去活多级角色 R_m 的执行步骤为:

Ó 去活多级角色 R_m:

$S.amrset - \{R_m\} \rightarrow S.amrset$;

Ó 去活多级角色 R_m 的所有读写类型的内部角色:

如果 $R_{rw} \in R_m.rrset \cap R_{rw} = S.unrset$, 则执行

$S.unrset \cup \{R_{rw}\} \rightarrow S.unrset$;

Ó 去活多级角色 R_m 的所有读写类型的内部角色:

如果 $R_u \in R_m.rrset \cap R_u \in S.unrset$, 则执行

$S.unrset \cup \{R_u\} \rightarrow S.unrset$;

3.7　综合模型实现机制

在多级关系数据库中, 主体与客体的安全信息都可以存放在系统关系中. 系统关系可以与普通关系一样被查询和更新, 但与普通关系不同的是, 系统关系存放的信息用来维护整个数据库的正常运行, 它们只能被管理员访问, 或者在执行某些命令的过程中由系统自动更新. 我们的访问控制模型同样也需要维护着某些信息, 例如角色的权限, 用户可以担当的角色, 多级角色的内部角色集等等. 这些信息也可以存放在系统关系中, 便于系统在进行访问控制时使用. 例如, 可以创建一个系统关系记录内部角色与权限的关系))) NRolePerms, 表 1 是 NRolePerms 的字段及其含义:

表 1　系统关系 NRolePerms 的字段

字段名称	含　　义
RoleName	内部角色名称
ObjName	内部角色拥有的权限对应的对象名称
Operators	内部角色拥有的权限对应的操作集合

关系 NRolePerms 存放了所有内部角色的权限, 系统可以在这里查询到用户激活的内部角色的权限. 如果内部角色比较多, 角色的权限也比较多, 那么这个关系的记录数目可能会非常多, 为了提高查询效率, 系统可以为这个关系建立定义在 RoleName 上的索引, 这样就能加快查询的速度. 综合模型需要的某些安全信息可能是在系统运行过程中创建的, 例如会话激活的角色集合. 这些信息可以存放在内存中, 也可以存放在临时的系统关系中. 如果存放在临时的系统关系中, 系统也可以采用访问关系的机制来处理这些信息. 所以在实现模型的过程中, 我们尽量把安全信息存放在固定或临时的系统关系中, 并使用数据库中访问关系的机制来处理这些安全信息. 在已经完成的 B1 级安全数据库))) LOIS 安全数据库中, 我们利用上面描述的机制实现了第五节所描述的综合模型, 并达到了方便权限管理的目的.

4　结束语

基于角色的访问控制模型方便了系统的权限管理, 但是在利用 RBAC 模拟 MAC 的时候, 如果为角色定义标签并根据标签来决定角色与用户之间分配关系, 那么一旦系统中的标签层次或用户数目比较多, 势必造成角色数量的大大增加, 从而引起管理的不方便. 本文提出的综合访问控制模型引入了多级角色和内部角色的概念, 多级角色可以与任何级别的用户和权限发生联系, 而内部角色则有固定的标签, 只接收与其标签相同的对象上的权限. 但是系统管理者只需管理多级角色与权限、用户之间的关系, 不用考虑它们之间的标签是否符合强制安全策略, 系统会自动完成内部角色与权限、用户的关系. 本文描述了综合模型的操作, 包括修改后的权限分配、用户分配、打开会话以及激活角色. 这些操作使得激活同一个多级角色的不同级别用户只能获得那些符合强制访问控制策略的权限. 这样, 管理者能方便而有效地管理多级安全数据库系统中权限的分配, 被管理的角色数量也不会因为标签层次或用户数目的增加而变得非常巨大. 最后, 我们给出了模型的一些实现机制, 并在安全数据库实例中实现了该模型.

参考文献:

[1] D Elliott Bell, Leonard J LaPadula. Bell2LaPadula Model For Secure Computer Systems[R]. The MITRE Corporation, March 1976.

[2] D F Ferraiolo, D R Kuhn. Role2based access control[A]. In Proc. of 15th National Computer Security Conference[C]. October, 1992. 554-563.

[3] Nyanchama M, Osborn S L. Information flow analysis in role2based security system[J]. Journal of Computing and Information, 1994, 1(1): 1368- 1384.

[4] Ravi Sandhu, Edward J. Coyne, Hal L. Feinstein, Charles E. Youman. Role2based access control models[J]. IEEE Computer, February 1996, 29(2): 38- 47.

[5] D Ferraiolo, R Sandhu, S Gavrila, D Kuhn, R Chandramouli. Proposed NIST standard for role2based access control[A]. ACM TISSEC[C]. Volume 4, Issue 3, August 2001, 4(3): 224- 274.

[6] S Osborn, R Sandhu, Q Munawer. Configuring role2based access control to enforce mandatory and discretionary access control policies[A]. ACM TISSEC[C]. Volume 3, Issue 2, May 2000. 85- 106.

[7] D Richard Kuhn. Role based access control on MLS systems without kernel changes[A]. In Proc. of the third ACM Workshop on Role2Based Access Control[C]. Fairfax, Virginia, United States, October 222 23, 1998. 25- 32.

[8] Sylvia Osborn. Mandatory access control and role2based access control revisited[A]. In Proc. of the Second ACM Workshop on Role2Based Access Control[C]. Fairfax, Virginia, United States, November 0607, 1997. 31- 40.

[9] Ravi Sandhu. Design and implementation of multilevel databases[A]. In Proc. of 6th RADC Workshop on Multilevel Database Security[C]. Southwest Harbor, Maine, June 1994.

[10] 吴承荣, 张世永等. 数据库访问控制模型分析[J]. 计算机工程与应用, 2002, 38(13): 183- 185, 194.

[11] 李沛武, 周铭. 集成 RBAC 到多级关系模型[J]. 计算机工程与科学, 2002, 24(1): 20- 23.

[12] 刘琼波, 施军. 用 RBAC 实现 DAC 和 MAC 的一种方法[J]. 计算机工程. 2000, 26(10): 62- 64.

[13] 李斓, 冯登国. 多级关系数据库中的 RBAC[A]. 第三届中国信息与通信安全学术会议论文集[C]. 2003. 329- 333.

[14] 徐震, 冯登国. SKLOIS 多级安全数据库管理系统的体系结构[A]. 第三届中国信息与通信安全学术会议论文集[C]. 2003. 334- 328.

作者简介:

李　斓　男, 1977 年 6 月生于江西省新干县, 2001 年毕业于武汉大学计算机学院, 获工学硕士学位, 现正在中国科学院软件研究所攻读计算机应用技术专业博士, 主要研究方向为系统与网络安全. E2mail: lilan@is. iscas. ac. cn

冯登国　男, 1965 年生于陕西靖边, 通信与信息系统专业博士, 现任中国科学院软件所研究员、博士生导师, 信息安全国家重点实验室主任, 主要研究方向为信息安全.