“长城”安全政策的扩充研究及其实现

赵庆松, 孙玉芳, 梁洪亮, 张相锋, 孙 波
(中国科学院软件研究所, 北京 100080)

摘 要: “长城”安全政策(Chinese Wall Security Policy, CWSP)是商业信息领域中重要的安全政策之一. 但是 Brewer-Nash 提出的 CWSP 并不能很好地满足实际的需要. 基于角色的访问控制(Role-Based Access Control, RBAC)模型是一种“政策中性(Policy Neutral)”的模型, 被看作是最有可能替代传统的自主和强制访问控制模型的一种全新的模型, 正越来越被信息安全领域所重视. 本文首先介绍了 RBAC 和“长城”安全政策, 然后根据实际应用对 CWSP 作了系统的扩充, 最后本文系统地论述了基于 RBAC 的扩充 CWSP 的实现方法.

关键词: 信息安全; 角色; 基于角色的访问控制; 扩充的“长城”安全政策
中图分类号: TP309 文献标识码: A 文章编号: 0372-2112 (2002) 11-1658-05

Research and Enforcement of Enhanced Chinese Wall Security Policy

ZHAO Qing-song, SUN Yu-fang, LIANG Hong-liang, ZHANG Xiang-feng, SUN Bo
(Institute of Software, Chinese Academy of Sciences, Beijing 100080, China)

Abstract: Chinese Wall security policy (CWSP) is one of the most important security policies in commercial information area. But the CWSP proposed by Brewer and Nash can't fully meet the practical requirement. The role-based access control (RBAC), a policy neutral model, has recently received considerable attention as a most promising alternative to traditional discretionary access control (DAC) and mandatory access control (MAC) models. RBAC and the Chinese Wall security policies are given, and expanded due to the practical application. The RBAC-based method to expand CWSP is systematically discussed. Thus, the enhanced CWSP (ECWSP) is presented firstly. And then the method of configuring RBAC to enforce the ECWSP is systematically studied.

Key words: information security; role; role-based access control; enhanced Chinese Wall security policy

1 引言

作为政策中性的模型, RBAC 被看作是替代传统的 DAC 和 MAC 的一种全新的模型[1~3]. 在 RBAC 中, 权限(Permissions)被指派给角色, 用户作为角色的成员而享有角色所具有的权限, 这就极大地简化了对权限的管理. 在实际应用中, 依据机构中各种工作的功能及其所需要的权限, 抽象成不同的角色; 而根据用户在现实中的职责和资格, 授予他不同的角色. RBAC 可以灵活的实现系统的管理, 一方面, 新的权限可以指派给角色, 角色已经具有的权限也可以被撤消; 另一方面, 也可以授予用户新的角色或者撤消他已经具有的角色. RBAC 受到普遍重视的原因正是在于它的“政策中性”的特点[1,2], 即 RBAC 不是为专门的安全政策而设计的, 而可以用来实现多种不同的安全政策, 如用来实现 DAC[4] 和 MAC[5,6] 等, 本文首次基于 RBAC 实现了扩充的 CWSP. CWSP 最初起源于商业咨询领域中, 是由 Brewer 和 Nash[7] 首次提出并命名的. 在商业咨询领域中, 咨询顾问向作为他们的客户的其他公司提供咨询服务, 因此, 咨询顾问就不可避免的需要访问(读或者写)客户公司的机密信息. CWSP 的目的在于防止可能引起客户利益冲突的信息流发生[9]. 例如, 不能让一个咨询顾问同时能够访问两家或者两家以上的保险公司的机密信息, 当然, 也不能让他能够同时访问两家 PC 制造公司的机密信息. 因为同时掌握多家具有利益冲突的公司的机密信息, 一方面会影响咨询顾问的分析结果的客观性, 另一方面会损害这些公司的商业利益. 但是, 当咨询顾问还没有访问过某一竞争领域的任何一家公司的机密信息时, 他就具有选择任何一家公司作为其咨询客户的自主权; 一旦他访问过某一家公司的机密信息后, 在他和其他与该公司具有利益冲突的公司之间就会形成一道“墙”, 防止他再去访问墙外面的信息. CWSP 正是将这种自主性和咨询顾问必须遵守的强制性结合在一起[7]. 在许多商业领域中, 商业活动都要求遵守 CWSP. 就像 Bell-LaPadula 的保密性原则在军事领域中起到重要作用一样[11], CWSP 对商业领域中的信息安全是至关重要的.

尽管用 Brewer-Nash 模型(Brewer-Nash model)[7]、基于“格”的访问控制(Lattice-Based Access Control, LBAC)模型[8,9]、基于踪迹的访问控制(Trace-Based Access Control, TBAC)模型[10] 等

收稿日期: 2001-08-06; 修回日期: 2002-04-08
基金项目: 国家 863 高科技项目(No. 863-306-ZD12-14-2); 国家自然科学(No. 60073022); 中国科学院知识创新工程项目(No. KGCX1-09)

许多模型可以实现 CWSP, 但这些实现都存在缺陷和不足. 通过合理的构造 RBAC 组件, 并配置各个组件之间的关系, RBAC 就可以成功的实现 DAC 和 MAC. 同样, 正是 RBAC 的这种灵活性, 我们可以用它来实现扩充的 CWSP[5]. 当前, 学术界比较认同的是 Sandhu 提出的 RBAC96 模型[2], 本文将基于此模型来实现扩充的 CWSP.

2　基于角色的访问控制(RBAC96)

RBAC96 是由 Sandhu 研究、定义及总结的一组模型[2], 它的结构如图 1 所示. RBAC96 模型的框架定义如下: (1) U: 用户集合; R 和 AR: 角色和授权管理角色, $R \cap AR = \varnothing$; P 和 AP: 权限和授权管理权限, $P \cap AP = \varnothing$; S: 会话集合. (2) $UA \subseteq U \times R$: 用户到角色的指派关系; $AUA \subseteq U \times AR$: 用户到授权管理角色的指派关系. (3) $PA \subseteq P \times R$: 权限到角色的指派关系; $APA \subseteq AP \times AR$: 授权管理权限到授权管理角色的指派关系. (4) $RH \subseteq R \times R$: 角色与角色之间的继承关系; $ARH \subseteq AR \times AR$: 授权管理角色与授权管理角色之间的继承关系. (5) user: $S \rightarrow U$: 会话和用户之间的映射关系; roles: $S \rightarrow 2^{R \cup AR}$ 会话到角色和授权管理角色之间的映射关系; roles$(s_i) \subseteq \{r' \mid (\exists r' \geqslant r)[\text{user}(s_i), r' \in UA \cup AUA]\}$. (6) constrains: 在 RBAC 的各种关系和各个组件中起作用的一组约束.

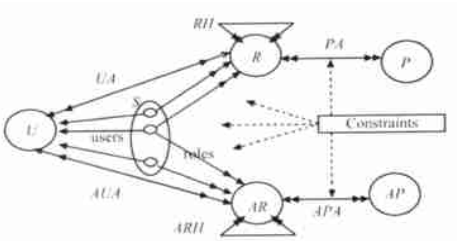

图 1　RBAC96 模型结构

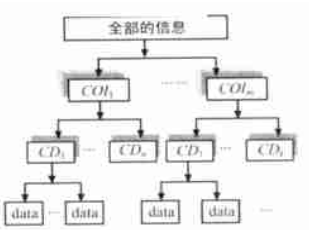

图 2　CWSP 中的客体结构图

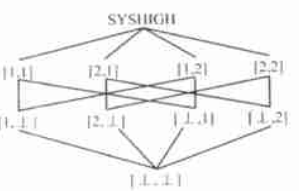

图 3　CWSP 标签及其组成的“格”

3　“长城”安全政策(CWSP)

CWSP 是一种实际的商业政策, 在实现该政策时, 可以首先形式化为这种政策建立模型. 图 2 是 Brewer Nash 采用的 CWSP 客体结构图. 图中所有的公司信息都分成三个层次存储, 底层是单个公司的数据项(Data Item), 所有属于一个公司的数据项组合成该公司的数据集(Company Dataset, CD); 高层是相互竞争的公司组成的利益冲突类(Conflict of Interest Class, COI). 在 Brewer Nash 的 CWSP 实现过程中, 每一公司仅属于一个 COI 类. 最初, 咨询顾问有完全自由选择为哪个公司提供咨询, 一旦该咨询顾问选择了一家公司, CWSP 就在他和该公司所在的 COI 类中的其他公司之间建起了一道“墙”. 尽管如此, 该咨询顾问仍具有选择其他 COI 类中的任何一家公司的权利. 不过, 他对新公司的信息的后续访问会重新改变已经存在的“墙”的形状, 对该咨询顾问形成新的限制. CWSP 的目的就是拒绝一个咨询顾问访问多家处于同一 COI 类中的公司信息.

4　各种实现方式的分析

4.1　基于 Brewer Nash 模型的 CWSP 实现

Brewer Nash 给出的 CWSP 读写规则如下: (1) Brewer Nash 读访问规则: 对于客体 $o \in O$ 和主体 $s \in S$, 仅当 s 已经读过 o (o 在“墙”内)或者 o 属于一个 s 从未从其中读过任何客体的 COI 类时, s 可以读访问 o. (2) Brewer Nash 写访问规则: 对于客体 $o \in O$ 和主体 $s \in S$, 仅当 s 按照 Brewer-Nash 读访问规则可以读 o, 且任何与 o 不在同一 CD 集合中的客体都不能够被 s 读访问到时, s 可以写访问 o.

从上述规则中, 我们可以得出 Brewer-Nash 模型的如下缺陷: (1) 在该模型中, Brewer Nash 假设每一个公司的数据集 CD 属于且仅属于一个 COI 类, 这在现实的商业活动中是不合实际的. 比如: 一家银行, 不仅要与其他的银行发生竞争, 同时也可能会与其他的投资公司发生竞争, 即这家银行应该属于多个 COI 类. (2) 每一公司都具有大量的信息, 但不是所有的这些信息都会与其他同类公司的信息发生冲突. 所以不能如 Brewer Nash 那样将公司所有的信息都放入一个 CD 中, 而应该从更细的粒度上对这些信息加以处理. (3) Brewer Nash 的写规则能够防止从一个公司的 CD 内读取信息, 然后写入其他的公司的 CD 内的泄密行为. 但实际上, 该规则有如下的含义:

(1) 系统中的一个访问过两个或两个以上的客体的主体不能再进行任何写操作;

(2) 系统中的一个仅仅访问过一个的客体的主体可以对该客体进行写操作.

很显然, 这些限制在实际中是不可接受的.

4.2　基于“格”的访问控制模型的 CWSP 实现

Ravi S Sandhu 曾经提出了一种基于“格”的实现方式[8]. 在这种实现方式中, Sandhu 对 COI 和 CD 定义和 Brewer Nash 的定义基本相似, 且对 COI 和 CD 是静态定义的, 同时, 他也是假设一个公司只能属于一个 COI. LBAC 中, 标识主体和客体的标签(Label)的维数与系统中的 COI 数目相同. 图 3 是一组标签及其所组成的“格”, SYSHIGH 是为了“格”的完整性而设立的一个虚拟的标签, 在实际中不会赋予任何客体和主体. 图中处于高层的标签“大于(Dominate)”处于底层的标签, 记作“$\geqslant$”. CWSP 的读写规则完全按照 Bell-Lapadula 模型的简单规则(simple-security)和 * 规则(*-properties), 即“不往上读”和“不往下写”的原则.

在基于 LBAC 的 CWSP 实现中, Sandhu 同样假设一个公司仅属于一个 COI 类, 这样就存在着与 Brewer-Nash 模型同样的缺陷. 再者, 对主客体的标签, Sandhu 是静态定义的, 因此, 从

系统中删除或者往系统内增加 COI, 系统中所有的主客体的标签都要随之改变, 系统的扩充性差.

4.3 基于"踪迹"的访问控制的 CWSP 实现

Kelley Sobel 等于 1999 年提出了基于"踪迹"访问控制模型(TBAC), 并基于 TBAC 实现了 CWSP. 在 Kelley Sobel 的实现过程中,"踪迹"(trace)就是客体被访问的历史记录, 系统中的每一客体都有自己的"踪迹", 所有踪迹的集合记录了系统中主体的活动. 图 4 是客体 $Object_i$ 的踪迹示例.

图 4 客体 $Object_i$ 的踪迹

在基于 TBAC 的 CWSP 实现中, Kelley Sobel 将系统中的活动主体统一抽象为"个体"(Individual), 且不再限制一个公司仅属于 COI, 公司的信息不再是一个整体, 而是可以分成多个不同的客体(Objects), TBAC 就是在此客体粒度上对公司的信息进行操作. 当一个个体对某客体发出读请求时, 首先要检查系统中所有与该客体属于同一个 COI 的客体的 trace, 看是否该个体已经访问过这些客体, 如果没有, 则允许此次请求, 否则拒绝该请求; 对于写请求, Kelley Sobel 认为是一种特殊的创建新客体的操作, 因此, 他将写请求和创建新客体的请求统一按创建新客体请求处理. 这种实现方式的不足是明显的, 系统中的读操作和创建新客体的操作毕竟不同, 不应该二者看作是一样的操作而统一处理; 再者, 每创建一个新客体, 系统都要创建一个新的 COI 类放置此客体, 但实际上整个系统并没有引入具有新的利益冲突的一类公司, 所以这种处理结果破坏了 COI 本来的含义.

5 扩充的"长城"安全政策(Enhanced Chinese Wall Security Policy, ECWSP)

基于上一节的分析, 我们从三个方面对 CWSP 进行扩充, 以求使之更加符合实际. (1)一个公司可以属于多个 COI, CDI (Company Date Item) 是公司信息的最小单位, 同一个 CDI 可以属于不同的 COI. 允许系统中存在复合客体(Compositive Objects CO), 即其所包含的信息来自两个或者两个以上的 CDI 或者其他的 CO. (2) 允许公共信息的存在. 一个实际的系统中, 并不是所有的信息都是公司的机密信息, 访问这些信息不会影响继续访问其他的信息, 也不会产生利益冲突. 在扩充后的 CWSP 中, 这种信息称作公共信息(Public Information, PI). (3) 允许读、写和创建新客体三种操作. 在扩充后的 CWSP 中, 对写操作和创建新客体的操作区别对待: 写操作的规则与 Brewer Nash 模型中的写操作的规则一样, 即只有主体能够读且仅仅读过某个 CDI 时, 主体对此 CDI 的写操作才会被允许; 我们允许主体创建新的客体, 新客体作为复合客体(CO)在系统中存在, 我们对后续的读写该复合客体的操作加以限制, 使之复合 CWSP 原则, 且不会泄密. 考虑到系统的一致性, 我们把创建新客体的操作看作是写一个不存在的 co. 这样, 当主体对一个 co 写操作时, 如果该 co 已经存在, 就执行正常的写操作, 否则, 首先创建新的 co, 然后, 再对该 co 执行写操作. 图 5 是 ECWSP 的数据结构图.

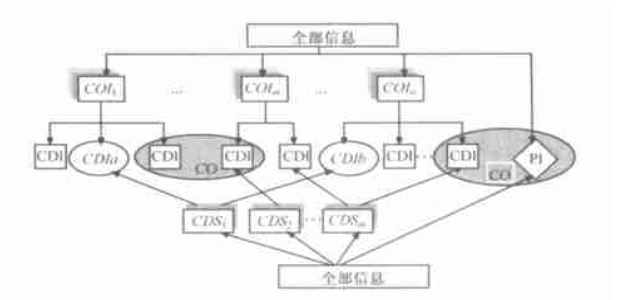

图 5 ECWSP 的客体层次结构图

6 基于角色的访问控制(Role-based Access Control)模型的 ECWSP 实现

6.1 权限和角色

如上一节所述, 在 ECWSP 中, 读写操作是两种不同的操作, 因此, 在基于 RBAC96 而构造的模型中, 我们指定读和写两种权限, 分别记作(o, r)和(o, w). 针对这两种权限, 我们建立两类角色, 一类是读角色, 另一类是写角色. 在我们的构造中, 规定权限和角色之间是一一对应的关系, 即一个权限只能赋予一个角色, 且一个角色只能具有一个权限. 表 1 是关于角色的几个定义.

6.2 层次关系

在我们构造的模型中, 有三种层次关系: 客体对客体的层次关系、读权限对读权限的层次关系和读角色对读角色的层次关系. 它们分别由图 6(a)、6(b)和 6(c)表示. 图 6(a)是一个客体层次关系示例, 图 6(b)和图 6(c)分别是基于图 6(a)的层次关系的读权限和读角色的层次关系. 图 6(c)中 $R\_r\_CDI_1$ 表示具有权限(CDI_1, r)的角色.

在 ECWSP 中, 所有的 CDI 及其包含 CDI 的 CO 都是机密的信息. 因此在如图 6(a)所示的层次关系图中, 尽管 CDI_1 和 CO_3 之间具有层次关系, 但写权限(CDI_1, w)和写权限(CO_3, w)之间并不存在层次关系, 即: 对 CDI_1 有写的许可并不意味着对 CO_3 就有写的许可, 反之亦然. 具有写权限的写角色之间以及读角色与写角色之间也不存在层次关系. 角色之间的层次关系定义如下:

$(junior, R\_r\_O_i, R\_r\_O_j)$: 当层次关系树中存在从 $R\_r\_O_j$ 到 $R\_r\_O_i$ 的路径时, 关系$(junior, R_r_O_i, R_r_O_j)$为真, 表示角色 $R\_r\_O_i$ 比角色 $R\_r\_O_j$ 低级, 即意味着角色 $R\_r\_O_j$ 的权限包含了角色 $R\_r\_O_i$ 的权限.

关系$(junior, R_r_O_i, R_r_O_j)$是一种自反的、传递的关系.

$Junior\_roleset(R\_r\_O_i) = \{R\_r\_O_j \mid (junior, R\_r\_O_j, R\_r\_O_i)\}$: 所有比角色 $R\_r\_O_i$ 低级的角色集合

表 1 符号及其含义

符 号	含 义
R	系统中所有角色的集合.
R_R	系统中所有读角色的集合.
R_W	系统中所有写角色的集合.
P	系统中所有权限的集合.
$Roles(u)$	用户 u 所具有的角色的集合.

构成了 $Junior\_roleset(R\_r\_O_i)$. 例如, 在图 6(c) 中有 $Junior\_roleset(R\_r\_CO_5)=\{R\_r\_CDI_1, R\_r\_CDI_2, R\_r\_CO_3, R\_r\_CDI_4\}$.

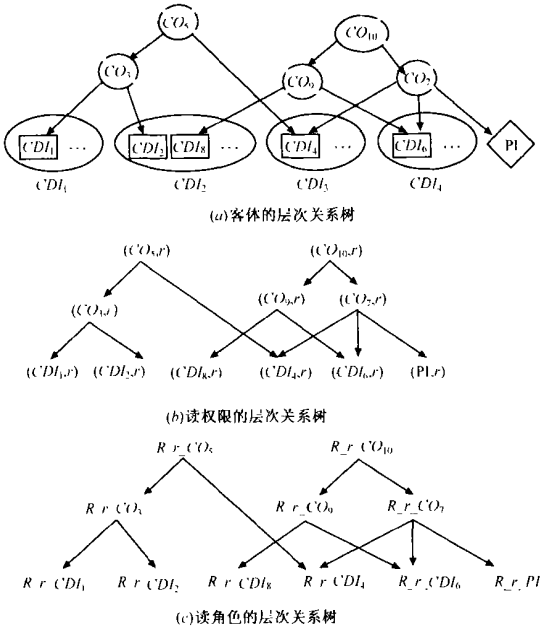

图6　ECWSP 的各种层次关系

6.3　冲突关系

在RBAC96中, 具有冲突的两个或者多个角色不能同时赋予一个用户. 例如在我们构造的模型中, 就不能将角色 $R\_r\_CDI_2$ 和角色 $R\_r\_CDI_8$ 同时赋予一个用户, 因为 CDI_2 和 CDI_8 同属于 COI_2, 但是, 我们可以将属于不同的COI的 $R\_r\_CDI_1$ 和 $R\_r\_CDI_2$ 同时赋予一个用户, 而不会违背 CWSP 原则. 冲突不但存在于读角色和读角色之间, 而且存在于读角色和写角色之间. 例如我们不能将角色 $R\_r\_CO_3$ 和角色 $R\_w\_CDI_1$ 同时赋予一个用户, 因为这样有可能造成 CDI_2 中的信息违背 CWSP 规则的流向 CDI_1. 单独的写角色和写角色之间不存在冲突, 因为只有写角色不会构成违反 CWSP 规则的信息流.

$(conflict\_rr, R\_r\_O_i, R\_r\_O_j): (\forall r_1 \in Junior\_roleset(R\_r\_O_i), \forall r_2 \in Junior\_roleset(R\_r\_O_j)$, 如果 r_1 和 r_2 可以读访问到同一 COI 中的不同 CDI, 则 $(conflict\_rr, R\_r\_O_i, R\_r\_O_j)$ 为真, 称 $R\_r\_O_i$ 和 $R\_r\_O_j$ 之间具有冲突关系.

$conflict\_rr\_roleset(R\_r\_O_i)=\{R\_r\_O_j \mid (conflict, R\_r\_O_i, R\_r\_O_j)\}$: 系统中所有的与 $R\_r\_O_i$ 具有冲突关系的角色构成 $R\_r\_O_i$ 的角色冲突集.

$(conflict\_rw, R\_r\_O_i, R\_w\_O_j)$: 当且仅当关系 $(junior, R\_r\_O_j, R\_r\_O_i)$ 为真时, 关系 $(conflict\_rw, R\_r\_O_i, R\_w\_O_j)$ 为真. 即从一个客体中读信息的读角色与将读到的信息写往比该客体低级的客体中的写角色之间具有冲突关系.

关系 $(conflict\_rr, R\_r\_O_i, R\_r\_O_j)$ 和 $(conflict\_rw, R\_r\_O_i, R\_w\_O_j)$ 是对称但非传递、非自反的关系.

$conflict\_rw\_roleset(R\_r\_O_i)=\{R\_w\_O_j \mid (conflict\_rw, R\_r\_O_i, R\_w\_O_j)\}$: 所有与角色 $R\_r\_O_i$ 具有冲突关系的写角色构成 $R\_r\_O_i$ 的冲突角色集.

$conflict\_rw\_roleset(R\_w\_O_j)=\{R\_r\_O_i \mid (conflict\_rw, R\_r\_O_i, R\_w\_O_j)\}$: 所有与角色 $R\_w\_O_i$ 具有冲突关系的读角色构成 $R\_w\_O_i$ 的冲突角色集.

$conflict\_roleset(R\_r\_O_i)=conflict\_rr\_roleset(R\_r\_O_i) \cup conflict\_rw\_roleset(R\_r\_O_i)$.

$conflict\_roleset(R\_w\_O_j)=conflict\_rw\_roleset(R\_w\_O_j)$.

作为特例, PI 的冲突角色集如下: $conflict\_roleset(R\_r\_PI)=\phi$; $conflict\_roleset(R\_w\_PI)=R\_R-R\_r\_PI$.

6.4　实现过程

构造 **1**

(1) $R=\{R\_r\_o_i \mid o_i \in O\} \cup \{R\_w\_o_i \mid o_i \in O\}$;

(2) RH:

$o_i \in O, o_j \in O, (junior, R\_r\_O_i, R\_r\_O_j)=T \Rightarrow$ 角色 $R\_r\_O_i$ 比角色 $R\_r\_O_j$ 低级.

$o_i \in O, o_j \in O \Rightarrow (junior, R\_r\_O_i, R\_w\_O_j)=F, (junior, R\_w\_O_i, R\_w\_O_j)=F$.

(3) $P=\{(r, o_i) \mid o_i \in O\} \cup \{(w, o_i) \mid o_i \in O\}$;

(4) 对 URA 的限制: $\forall r' \in roles(u), r \notin conflict\_roleset(r') \Rightarrow can\_assign(r, u)$;

表 2　$(conflict\_rr, R\_r\_O_i, R\_r\_O_j)$

roles	$R\_r\_CDI_1$	$R\_r\_CDI_2$	$R\_r\_CO_3$	$R\_r\_CDI_4$	$R\_r\_CO_5$	$R\_r\_CDI_6$	$R\_r\_CO_7$	$R\_r\_CDI_8$	$R\_r\_CO_9$	$R\_r\_CO_{10}$	R_r_PI
$R\_r\_CDI_1$								T	T	T	
$R\_r\_CDI_2$								T	T	T	
$R\_r\_CO_3$								T	T	T	
$R\_r\_CDI_4$											
$R\_r\_CO_5$								T	T	T	
$R\_r\_CDI_6$											
$R\_r\_CO_7$											
$R\_r\_CDI_8$		T	T		T						
$R\_r\_CO_9$		T	T		T						
$R\_r\_CO_{10}$		T	T		T						
R_r_PI											

表 3 (*conflict_ rw, R_ r_ O_i, R_ w_ O_j*)

roles	$R\_r\_CDI_1$	$R\_r\_CDI_2$	$R\_r\_CO_3$	$R\_r\_CDI_4$	$R\_r\_CO_5$	$R\_r\_CDI_6$	$R\_r\_CO_7$	$R\_r\_CDI_8$	$R\_r\_CO_9$	$R\_r\_CO_{10}$	R_r_PI
$R\_w\_CDI_1$		T	T	T	T	T	T	T	T	T	
$R\_w\_CDI_2$	T		T	T	T	T	T	T	T	T	
$R\_w\_CO_3$				T	T	T	T	T	T	T	
$R\_w\_CDI_4$	T	T	T		T	T	T	T	T	T	
$R\_w\_CO_5$						T	T	T	T	T	
$R\_w\_CDI_6$	T	T	T	T	T		T	T	T	T	
$R\_w\_CO_7$	T	T	T		T			T	T	T	
$R\_w\_CDI_8$	T	T	T	T	T	T	T		T	T	
$R\_w\_CO_9$	T	T	T	T	T		T			T	
$R\_w\_CO_{10}$	T	T	T	T	T						
R_w_PI	T	T	T	T	T	T	T	T	T	T	

(5) 对 session 的限制: 无.

(6) 对 PRA 的限制: 一个权限只能赋予一个角色, 一个角色只能具有一个权限. 角色和权限之间是 1:1 的关系, 即: $\forall r \in R, p \in P, p \Leftrightarrow r$.

定理 按照上述构造 1 建立的 RBAC 模型满足 ECWSP.

证明 一方面, 对 URA 的限制意味着要赋予用户的角色不能与该用户已经具有的任何角色之间具有冲突关系, 因此用户不可能同时读到来自一个 COI 中的多个 CDI 的信息; 再者, *can_ assign(r, u)* 同样隐含如下的含义: 用户读操作所能访问到的客体的信息不会多于用户写操作能访问到的客体的信息, 这样就保证了 CDI 信息的机密性, 不会造成泄密.

结论 按照上述构造 1 建立的 RBAC 模型满足 ECWSP.

7 总结及展望

本文首先简要的介绍了 RBAC96 和 CWSP, 在分析各种实现 CWSP 的方法的基础上, 提出了更加切合实际 ECWSP, 克服了 CWSP 的固有缺陷: 允许一个公司属于多于一个 COI; 将公司的信息分为多个 CDI, 在更细的粒度上对数据操作; 允许公共信息的存在; 允许创建新的复合客体的操作与写系统原有的 CDI 的操作共存等. 然后在基于 RBAC96 实现 ECWSP 的过程中, 我们构造的模型完全实现了 ECWSP.

一个完善的系统应该支持诸如系统中机密信息的解密、访问过机密信息的用户的脱密、用户所具有的角色的安全撤消等功能, 我们将在今后的工作中实现这些功能; 再者, 仅仅支持 ECWSP 的系统不会完全满足实际应用的, 所以, 如何基于 RBAC96 同时实现 EWSP、MAC、DAC 等多政策(multi-policies) 也是我们今后工作的一个方面.

参考文献:

[1] Ravi Sandhu. Role-based access control[J]. Advances in Computers, 1998, 46: 237-286.

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

[3] L Giuri, P Iglio. A formal model for RBAC with constraints[A]. Proc of the CSFW[C]. Luigi Giuri and Pietro Iglio: IEEE Press, 1996. 136-145.

[4] R Sandhu, Q Munawer. How to do discretionary access using roles[A]. Proc of the Third ACM Workshop on Role Based Access Control[C]. Barkley, Cincotta: 1998. 47-54.

[5] Matunda Nyanchama, Sylvia Osborn. Modeling MAC in role-based security systems[A]. Proc of the IFIP WG 11.3 Ninth Annual Working Conference on Database Security[C]. New York, USA: 1995. 129-144.

[6] S Osborn. Mandatory access control and role-based access control revisited[A]. Proc of the Second ACM Workshop on Role Based Access Control[C]. Fairfax, Virginia, USA: ACM Press, Nov 1997. 31-40.

[7] Brewer D, Nash, M. The Chinese Wall security policy[A]. Proc of the 1989 IEEE Symposium on Security and Privacy[C]. IEEE Computer Society Press, 1989. 206-214.

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

[9] Ravi Sandhu. Lattice-based enforcement of Chinese Walls[J]. Computers and Security, December 1992, 11(8): 753-763.

[10] Ann E Kelley Sobel, Jim Alves-Foss. A trace-based model of the Chinese Wall security policy[A]. Proc of 22nd National Information Systems Security Conference [C]. Arlington, Va: Oct 1999.

[11] Bell D E, LaPadula L. Secure Computer Systems: Mathematical Foundations and Model[R]. M74-244, MITRE Corp Bedford, MA, 1973.

作者简介:

赵庆松 男, 1973 年 9 月生于山东, 1997 年毕业于山东工业大学计算机系, 获学士学位, 2000 年于山东工业大学获硕士学位, 现为中科院软件所 2000 级博士生, 主要研究方向为操作系统、信息安全.