云计算环境中基于用户访问需求的角色查找算法

杨柳，唐卓，李仁发，张宗礼
（湖南大学 嵌入式系统与网络实验室，湖南 长沙 410082）

摘 要：提出了一种云计算环境中基于角色的访问控制模型 CARBAC，将角色分为用户角色和资源拥有者管理角色。针对管理角色对用户访问的角色指派，提出了在混杂角色层次关系中基于用户权限的角色查找算法。对于一组给定的授权，该算法能在云计算系统的角色中选择一组数量最少的角色指派给用户。仿真实验表明，针对云计算环境中的海量用户访问，本算法能显著减少系统中角色的数量，缩短用户授权时间，提高系统运行效率。

关键词：云计算；角色查询；访问控制
中图分类号：TP309.2 **文献标识码**：A **文章编号**：1000-436X(2011)07-0169-07

Roles query algorithm in cloud computing environment based on user require

YANG Liu, TANG Zhuo, LI Ren-fa, ZHANG Zong-li
(Embedded Systems and Networking Laboratory, Hunan University, Changsha 410082, China)

Abstract: Roles based access control model CARBAC for cloud computing environment was proposed. Roles in the model include user roles and resources owner roles, the latter were called administrator roles. For the resources owner roles assigning the roles to the users, roles query algorithm in the hybrid hierarchies based on users' access require was proposed. Through this algorithm, for the certain privileges set, this model could choose the least roles set to be assigned to the users. The simulation experiment indicates that for the mass user access, this algorithm can reduce the roles quantity, shorten the authorization procedure, and advance the system efficiency.

Key words: cloud computing; roles query; access control

1 引言

云计算环境中由服务提供商的数据中心负责存储过去一直保存在最终用户个人计算机上或企业自己数据中心的信息，安全和效用已经成为云计算领域中最受关注的 2 个问题[1]，关注度分别达到了 74.6%和 63.1%[2]。云安全联盟多次发布报告，建议和督促企业采取安全措施，增强云环境中服务和用户数据的安全[3]。

云计算环境中包含 3 类实体：最终用户(end user)、服务提供商(service provider)以及数据拥有者(data owner)。用户发起对云环境资源的访问请求，服务提供商所提供的资源可能来自不同的安全域并且这些资源可能属于不同的数据拥有者。典型的云环境下，最终用户对数据的访问权限一般由数据拥有者指派，其基本表现形式：数据拥有者直接对用户指派对资源的操作权限；或者通过角色将权限指派给用户。基于角色的访问控制(RBAC)模型[4]

收稿日期：2011-01-19；修回日期：2011-07-04
基金项目：国家自然科学基金资助项目（60873074）；中国博士后科学基金资助项目（20100480936）
Foundation Items: The National Natural Science Foundation of China (60873074); China Postdoctoral Science Foundation (20100480936)

是一种广泛应用的访问控制机制，在适合云计算环境的访问控制模型中，基于角色的授权是数据拥有者对访问者权限控制的最为普遍的方式。

本文在 RBAC 模型的基础上，给出了一种满足计算及资源节点动态需求的访问控制管理模型：CARBAC (cloud administrate based on RBAC)。本模型中的角色包括用户角色和数据拥有者角色，充分考虑了数据拥有者这一实体在云环境访问控制中的作用。当用户需要获得某一数据块的访问权限时，可以通过用户角色从数据拥有者角色获得相应的访问证书。

用户的权限主要是通过对其指派的角色来获得的。在云计算环境尤其是公有云环境中，有着海量的资源、用户和角色，对于用户的访问控制权限，往往存在多种不同的角色指派方案。因为角色的权限的不同，使得数据拥有者可以将许多种不同的角色组合指派给数据访问者来实现同样的授权目的。云计算环境中拥有海量的用户及资源，在实际授权过程中会产生数量巨大的角色，将会消耗大量的计算和存储资源去维持其角色与用户间的指派关系。有必要研究一种基于权限集的角色查找算法，使得在云环境中可以根据用户的实际需求，选择出针对该用户会话中所访问资源的最小角色集合，指派给用户，以达到节省系统资源的目的。

本文在 CARBAC 模型的基础上建立一种基于用户会话中资源请求的角色查找方法。章节组织如下：第 2 节介绍相关工作和进展；第 3 节提出了一种云计算环境中基于角色的访问控制管理模型；第 4 节介绍如何根据用户的当前权限去查找合适的角色；第 5 节对文中算法进行了仿真实验；第 6 节是结束语。

2 相关工作

为了适应角色的动态变化以及更广泛的时间限制，Joshi 在 RBAC 模型的基础上提出了一个广义的时间限制的访问控制模型(GTRBAC)[5]。GTRBAC 对于指定细粒度语义允许的时间约束定义了一整套规范，允许表达角色层次和职责分离原则[6]的限制。GTRBAC 模型的角色具有 3 种状态：就绪、挂起和激活，通过定期和持续时间来完成对角色，用户角色指派、角色-权限指派的时态约束。GTRBAC 的时间约束并不包括对于数据拥有者的约束。

Jason Crampton[7]和 Koch[8]在角色层次中引入了管理模型并制定了一系列角色层次管理规则。Youngmin Jung[9]基于 RBAC 模型提出了云环境的自适应安全模型来描述云环境中的角色转换，当服务请求成本比预算限制要低时角色保持不变，但如果高于或者等于预算，角色就会转换为更高的版本。这种基于角色变换模型可以解决云计算等分布式系统环境变量的动态变化问题。Weichao Wang[10]等人讨论了云计算中数据分组的安全性。该研究讨论了数据报文的安全及数据的高效访问问题，建议对用户访问的数据分组提供灵活有效的细粒度访问控制机制，由此提出了一种比较全面的机制来处理用户权限的动态变化。

JOSHI 等人对 TRBAC 模型[11]进行了扩展，其提出的GTRBAC[5]模型解决了RBAC模型中用户有可能只在一个特定的时间段内才能指派给一个角色、一个角色可能只能临时依赖于另一个特定的角色的问题。此后又提出了有关混杂角色层次中的权限查询等问题[12]，提出了唯一激活集(UAS, uniquely activable set)[13]以及求解算法。UAS 能在既有角色继承，又含有角色激活关系的混杂角色层次中，针对一定的权限，给出相应的唯一角色集合。

以上访问控制模型均不能区分普通用户和资源拥有者管理角色，而且缺乏处理云计算环境中海量用户及高并发访问的能力。本文首先提出云计算角色管理模型 CARBAC，区分主体角色及数据拥有者角色的职责。针对该模型中用户–角色指派数量巨大的问题，提出最小唯一角色 (MUR, minimizing uniquely roles) 集及其求解算法，在集合的求解步骤和求解效率上对 UAS 算法作出了优化，以减少系统中角色的数量，缩短用户的授权时间。

3 CARBAC 模型

定义 1 主体角色(*UR*)。主体角色即传统 RBAC 中的用户角色，原则上所有可以用来区分主体角色的特征都可以用来区分 *UR*。主体的特性包括身份、在一个组织中位置及安全层次。

主体角色集合记为 *UR*，*UR* 的结构为 *UR*=<*IDU*，*PO*，*SL*>，其中，*IDU* 表示角色身份，*PO* 是角色位置，而 *SL* 代表角色的安全层次。由主体角色 *UR* 发送给数据拥有者的访问请求包含用户角色的信息，数据拥有者角色 *OR* 受到访问请求也会得到发送用户的信息。

定义 2　数据拥有者角色(*OR*)。数据拥有者角色包含数据拥有者所用数据的信息，并且可以及时更新存储在服务端的资源。数据拥有者角色管理新用户的注册，同时注销不再对资源和服务进行访问的用户。

拥有者角色记为 *OR*，*OR*=<*IDO*，*DT*，*Date*，*Cert*>，其中，*IDO* 是指拥有者角色的身份，*DT* 是数据拥有者所具有数据的数据类型，*Date* 是数据更新数据的时间，*Cert* 是用户角色要获得的认证证书。

定义 3　访问请求(*Request*)。用户端发送给数据拥有的访问请求记为 *Req*, *Req*=<*UR*, *OR*, *request index*, *data block index*, *op*>，其中，*request index* 表示用户的访问请求，*data block index* 表示 *UR* 请求访问的数据块，*op* 表示用户请求的对资源的操作。

因为用户和用户所指派的角色经常发生变化，数据拥有者需要区分哪些用户角色一直在对系统进行访问，哪些角色是第一次发送访问请求。因此数据拥有者必须记录每个用户角色的信息，能够准确判断用户的身份，做出正确的响应。

3.1　模型的相关概念

本模型中各实体及实体间的关系如下：

1) *U*，*R*，*P*，*S* 和 *SP* 分别代表用户，角色，访问规则，会话和服务提供者；

2) $PA \subseteq P \times R$，访问规则到角色的多对多指派关系；

3) $RA \subseteq R \times R$，角色到角色的多到多指派关系；

4) $UA \subseteq U \times R$，用户到角色的多到多指派关系；

5) *UR* 和 *OR* 分别代表用户角色集合和拥有者角色集合；

6) *SP* 指服务提供者。

3.2　CARBAC 角色层次

一个角色层次可以表示为一个有向图，图中的节点代表角色，每一条边代表两角色之间的继承关系。例如，角色 *A* 是角色 *B* 的父角色当且仅当有一条有 *A* 指向 *B* 的边，*B* 也叫做 *A* 的子节点。*D* 被称为 *C* 的管理节点当且仅当 *D* 和 *C* 之间有多条路径相通。

定义一个偏序集合<*R*，≤>，其中 *R* 是角色集合。本文定义节点 *x* 覆盖节点 *y* 为 $y \leqslant x$，如果在 *x* 和 *y* 之间有一条边，则角色 *x* 成为角色 *y* 的父节点，*x* 可以继承 *y* 的所有访问规则。

参照 GTRBAC 模型中的定义，CARBAC 模型中的角色具有就绪–失活–激活 3 种状态[5]，并可根据用户的访问需要相互转换。同时具有权限继承和激活关系的角色层次被称为混杂角色层次(hybrid hierarchy)。模型如图 1 所示，用虚线表示角色间的激活关系(a-hierarchy)，实线表示角色间的权限继承关系(i-hierarchy)。同时 CARBAC 模型中还引入了管理范围[7]概念，当一个角色被添加到角色层或从角色层删除时管理角色改变其数据拥有者角色的管理范围。

数据拥有者角色需要针对用户的访问，对访问者指派相应的用户角色。云计算环境中存在海量的数据资源和角色，如何根据用户的访问需要，在系统的用户角色中选取权限合适，同时角色数量最少的角色集合指派给用户。针对该问题，本文提出了一中基于用户权限的角色查找算法。

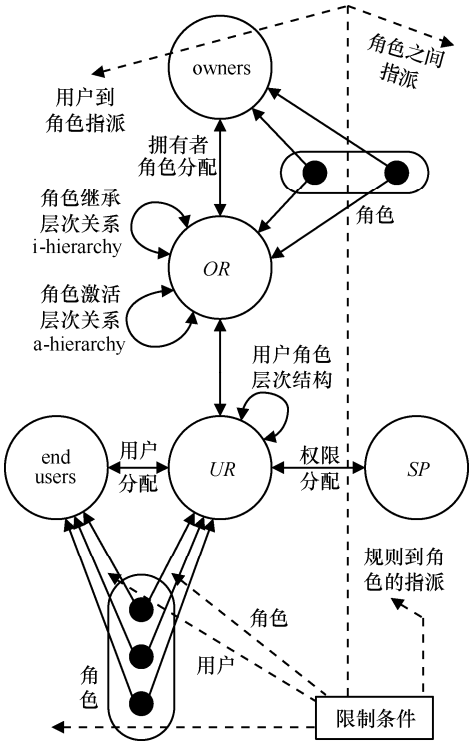

图 1　CARBAC 模型（黑色节点代表用户指派的角色）

4　基于用户权限的角色查找

4.1　角色查找算法

定义 4　*MUR* 集合　定义 $H=(R,F)$是 CARBAC 模型中的角色层次树。其中，*R* 是角色层次关系中的角色集合，*F* 是节点间的关系集合，包括继承关系和激活关系。$MUR(H)=\{R_1,R_2,\cdots,R_m\}$，其中，$\varnothing \subset R_i \subseteq R$，$i \in \{1,2,\cdots,m\}$，在 *H* 的所有权限集合中，$R_i$ 是对应权限集合的最小且唯一的角色组合，当且仅当：

1) $\forall R_i$，$R_j \in MUR(H)$，$PS(R_i) \neq PS(R_j)$，其中，i，$j \in Z$，$i \neq j$；

2) $\forall Z \subseteq R$ s.t.$Z \notin MUR(H)$，若存在 $R' \in MUR(H)$，$PS(R')=PS(Z)$，则有$|R'|<|Z|$成立；其中，$PS(R')$表示角色集合 R'的权限，$|R'|$为集合 R'中元素的个数。

MUR 是文献[13]中所提出的 UAS 的一个扩展，对于一组给定的授权，该算法能在云计算系统的角色中选择一组数量最少的角色指派给用户。

下面给出在角色激活和权限继承关系共存的混杂层次关系[5]中计算 *MUR* 集合的具体算法。本算法中的所输入的角色树均具有根节点。定义运算符如下：

$N_1 \otimes N_2=\{\{x_1 \cup x_2\} | x_1 \in N_1,\ x_2 \in N_2\}$

$N_1 \otimes N_2 \otimes \cdots \otimes N_m=$

$\{\{x_1 \cup x_2 \cup \cdots \cup x_m\} | x_1 \in N_1, x_2 \in N_2, \cdots, x_m \in N_m\}$

假设 $M=\{N_1, N_2, \cdots, N_m\}$，有：

$\Theta M=N_1 \otimes N_2 \otimes \cdots \otimes N_m$

算法 1 *MURGeneration* (*H*)
Input: *H* 一种混杂角色层次 *H*
Output: 角色层次 *H* 的 *MUR* 集合
1) Initialize *MUR*=∅
2) *IH*← *I-hieGen* (*H*) //*IH*={$IH_1, IH_2, \cdots, IH_m$}，algorithm 2
3) *IH*'← *No-InGen* (*IH*) //algorithm 3
4) Foreach *IH*''∈$2^{IH'}$ Do
5) If (|*IH*''|==1) then
6) foreach *r*'∈*IH*'' Do
7) *MUR*=*MUR* ⋃ {*r*'}
8) else
9) *MUR*=*MUR* ⋃Θ *IH*''
10) Return *MUR*
算法 2 *I-hieGen* (*H*)
Input: *H* 一种混杂角色层次 *H*
Output: *IH* 角色层次 *H* 的所有 *I-hierarchies* 子树
1) Initialize *TempR*=∅ //所有被搜寻到的角色的临时集合
2) Initialize *IH*=∅
3) *R*=*Roles*(*H*) //*R* 是角色层次关系 *H* 中的所有角色集
4) While *R*≠∅ Do
5) From *r* start *DFS* Search under *I-hierarchy* //深度优先遍历 *I-hierarchies* 子树，遍历完的节点加入集合 *TempR* 中
6) *DFS* 遍历完成后，产生子树 *sub-I-hierachy*: IH_i
7) *IH*=*IH* ⋃ IH_i
8) *R*=*R*−*TempR*
9) End While
10) Return *IH*
算法 3 *No-InGen* (*IH*) //求 *IH* 子树中不含继承关系的所有角色集
Input: *IH* 任意 *IH* 子树
Output: *IH*' *IH* 子树中不含继承关系的所有角色集
1) 将 *IH*'初始化成 *IH* 集合的幂集
2) Foreach *R*'∈*IH*' Do
3) if 集合 *R*'中的角色间存在继承路径 then
4) *IH*'= *IH*'−*R*'
5) Return *IH*'
End

图 2 计算 *MUR* 算法

在图 2 所示的算法中，子算法 *I-hieGen* (*H*)首先将角色激活关系断开（如图 3 中的虚线），将一个角色树分解为相互独立、只含继承关系（*I-hierarchy*，如图 3 中的实线）的子树。

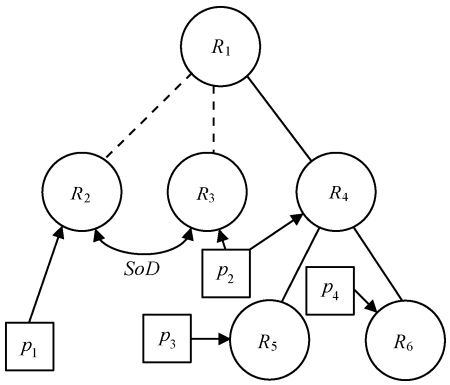

图 3 CARBAC 中的角色层次树

图 3 中的角色层次树 *H* 被分解为子树：IH_1、IH_2、IH_3，如图 4 所示。

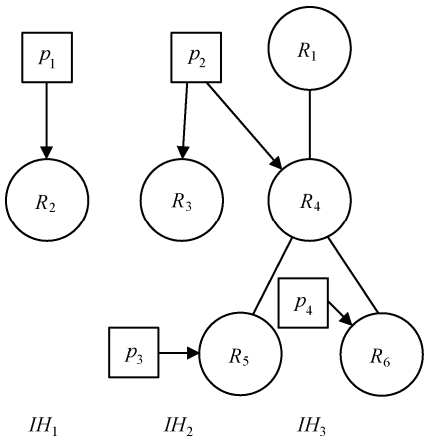

图 4 CARBAC 中角色层次树的分解

子算法 *No-InGen*(*IH*)用于寻找子树中不含继承关系的角色，子树 IH_3 中相互间没有权限继承关系的角色组合为：{{R_1}，{R_4}，{R_5}，{R_6}，{R_5，R_6}}。返回 *MURGeneration*，计算每个子树中不含继承关系的所有角色集的任意组合，如对角色 R_2，可完成的组合有：{R_2},{R_2, R_3},{R_2, R_1},{R_2, R_4},{R_2, R_5},{R_2, R_6},{R_2, R_5, R_6}。所有的角色组合构成的集合就是 *H* 的 *MUR* 集合。定理 1 证明了 *MURGeneration* 算法的结果满足 *MUR* 集合的定义。

定理 1 *MURGeneration* 算法中产生的角色集的集合是角色集 *R* 及其 *R* 上的关系 *F* 的 *MUR* 集合。

证明 令 *H*=(*R*, *F*)，算法 *MURGeneration* (*H*) 首先基于以下考虑：在严格的角色层次关系中，不同的角色权限不相等。否则，该系统的角色配置是冗余的。对于一定的权限集合，算法 *MURGeneration*

只能求解到最小的角色集，这种情况下该角色集并不是唯一的。

同理，在严格的角色层次关系中，不同角色的权限之间也不应该包含相互包含的关系，相互包含的关系可以通过上级角色对下级角色的继承来实现，所以这也是冗余的。所有一般性的角色层次关系，都可以经过适当的调整，建立这种严格的关系。

所以，*MURGeneration* 在分解后计算每个子树中互不包含继承关系的所有角色组合，这些角色组合的权限是相互不相等，互不包含的，满足定义 4 中的条件 1)。

在算法 *No-InGen* 中，集合 R'中的角色都属于不同的子树 IH_i，*MUR* 本质上是角色集的集合，算法 *MURGeneration* 将 R'的所有子集构成 *MUR* 中的角色集，其权限即是相互不相等，互不包含的。否则，假设集合 $R_i=\{r_i, r_j\}\in MUR$，若存在 $PS(R_i)=PS(R_j)$，$|R_i|>|R_j|$，不妨设 $R_j=\{r_j\}$，有 $PS(r_i)\subseteq PS(r_j)$，与假设矛盾。不失一般性，混杂角色树分解后，若几个角色属于不同的子树，或者是同一子树的角色之间无任何权限继承路径(如 IH_3 中的 R_5 和 R_6)，而角色指派的权限遵从严格的角色层次关系，没有冗余，则这几个角色所指派的权限集合是互不相等，互不包含的。满足定义 4 中的条件 2)。

以上分析说明了，算法 *MURGeneration* (*H*)中所产生的集合满足 *MUR* 集合定义中的 2 个必要条件，证毕。

定理 2　给定一组权限集合 P，若角色层次树 $H=(R,F)$中的存在角色集合的权限指派等于集合 P，即$\exists R^+\subseteq R$，$PS(R^+)=P$，则有：

若 $R^+\notin MUR(H)$且 $PS(R^+)=P$，则$\exists R'\in MUR(H)$，$PS(R')=P$，且 $R'\subseteq R^+$。

定理 2 说明：权限集合等于 P 的最小的角色集一定是该角色树 H 的 *MUR* 中的集合。

证明　给定一组权限集合 P，若角色层次树 $H=(R,F)$中的某个角色集合的权限指派等于集合 P，根据定义 4，$\exists R'\in MUR(H)$，$PS(R')=P$。若$\exists R^+\subseteq R$，$R^+\notin MUR(H)$，令 $R^+=\{r_1,r_2,\cdots,r_k\}$，则 R^+中存在权限继承（*I-hierarchy*）路径 $r_{i1},r_{i2},\cdots,r_{in}$，不妨令 r_{i1} 为 $r_{i1},r_{i2},\cdots,r_{in}$ 的顶层节点，继承角色 $r_{i2},\cdots,r_{in}$ 的权限。则可将 $r_{i1},r_{i2},\cdots,r_{in}$ 用 r_{i1} 代换掉，将 R^+集合中所有包含权限继承关系的路径做相同处理，即得 $R^*\subseteq R^+$ 且 $R^*\in MUR(H)$，$PS(R^*)=PS(R^+)$。因$\exists R'\in MUR(H)$，$PS(R')=P$，根据 *MUR* 中唯一性定义，有 $R^*=R'$。故 $R'\subseteq R^+$。证毕。

将一组角色 R'指派给某个用户 u，若该组角色不是其所在角色关系中的 *MUR* 中的集合，其 *MUR* 中的某个角色集 R''一定包含于该组角色：$R''\subset R'$，有：$PS(R'')=PS(R')$。定理 2 说明：任何一个用户的角色指派关系 UA，*MUR* 集合中都存在一个角色集合，该角色集合的权限与用户通过原有角色指派得到的权限相同。且 *MUR* 中的集合中角色的个数要小于或等于原用户指派的集合。这将是云计算环境中用户–角色指派关系进行简化的理论基础。

4.2　算法分析

从算法 1 中可以看出，其时间复杂度主要由运算 Θ 决定：$\Theta M=N_1\otimes N_2\otimes\cdots\otimes N_m=\{\{x_1\bigcup x_2\bigcup\cdots\bigcup x_m\}|x_1\in N_1,x_2\in N_2,\cdots,x_m\in N_m\}$，其中，$x_i$ 是集合 N_i 的元素或者为空集。运算 Θ 实际上是在求集合的幂集，该集合的每一个元素分别属于集合 $N_1, N_2,\cdots, N_m$，故其算法的时间复杂度为 $O(2^m)$，其中，m 为算法 2 所拆分成的 *I-hierarchy* 子树的数量。

在最好的情况下，混合角色层次结构中不含任何激活关系。也就是说，本地域中的角色层次结构是 *I-hierarchy*，算法 2 所返回只有一棵子树。这种情况下的时间复杂度取决于对角色树的深度优先搜索算法，可以在多项式时间内完成：$O(n^2)$，其中，n 为角色数量。

最坏的情况为混杂角色层次结构中不含任何继承关系。此时算法 2 所返回的子树的数量就是所有的角色的个数。故其算法的时间复杂度为 $O(2^n)$，其中，n 为本地域中的角色数量。

目前云计算系统中普遍采用的方式都是以权限继承为主的角色关系，少量对用户访问过程中的时间约束以及和先驱角色约束要求严格的系统采用混杂的角色层次关系，几乎没有系统是不含任何继承关系。故最坏的情况应用的几率很少，大多数系统的时间复杂度为 $O(n^2)$或者 $O(2^m)$。因为激活关系在系统中所占比例很少，故 $m<<n$。本算法在实际运行系统中是可行的。

5　仿真实验

本文在自行开发的云计算环境访问控制原型系统上对基于用户权限的角色查找算法进行了仿真实验。实验中不断调整用户所请求的文件数，取得算法中针对用户请求所需的角色数量和角色指派完成时间，并在同一平台和数据集中与 UAS 集

合算法进行对比分析。

实验环境：IBM System x3650，处理器：Xeon X5450 3GHz，内存：2GB，运行环境：JDK1.6。

结果如图 5~图 10 所示。

其中图 5 给出的是系统所指派的角色数的关系与用户所请求的资源数的关系，因为实验中每次选取的文件都是随机的，例如，用户在随机选取的 700 个文件并不一定包含前面实验中用户随机选取的 600 文件。所以，并不是文件数越大，所生成的角色数量就越多，这也可以由实验结果得到证实。

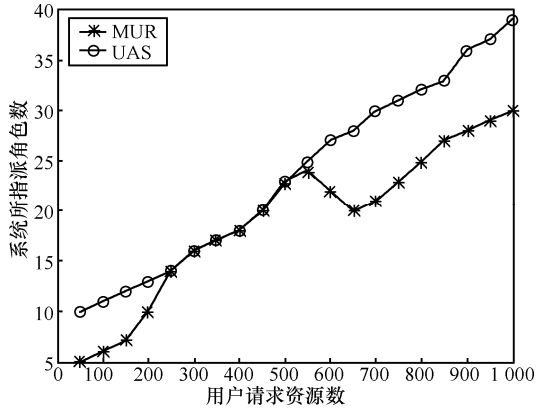

图 5 资源–角色指派数量关系

图 6 和图 7 给出的是当用户请求一定数量的资源时，系统中角色指派所完成的时间以及系统对角色表访问次数的关系。实验结果表明 *MUR* 集合算法在用户资源请求一定时，角色指派的效率要优于 UAS 集合。图 8 给出的是用户获得一定权限时，2 种算法下系统所指派的角色数量关系。虽然在权限数量 900～1 500 这个阶段两者差别不大，但 *MUR* 集合算法在用户指派权限一定的情况下，系统给用户指派的角色数要明显低于 UAS 集合。

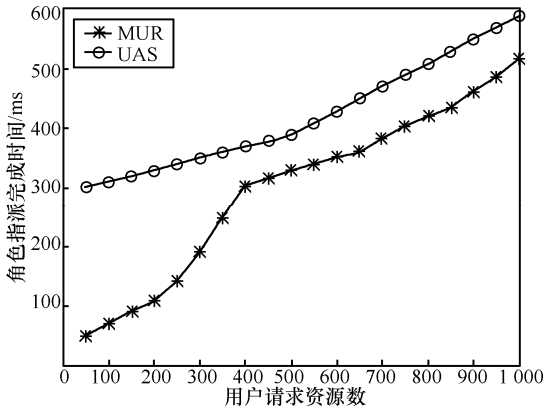

图 6 资源–角色指派完成时间关系

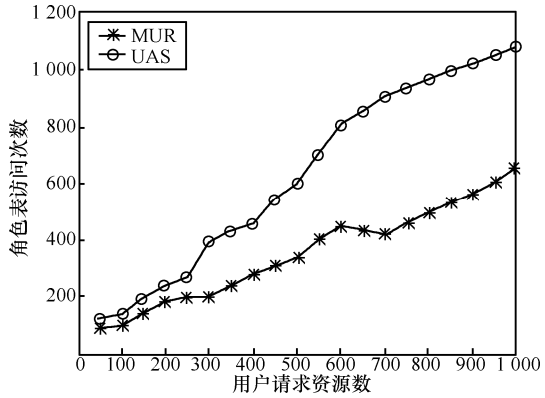

图 7 资源–角色表访问次数关系

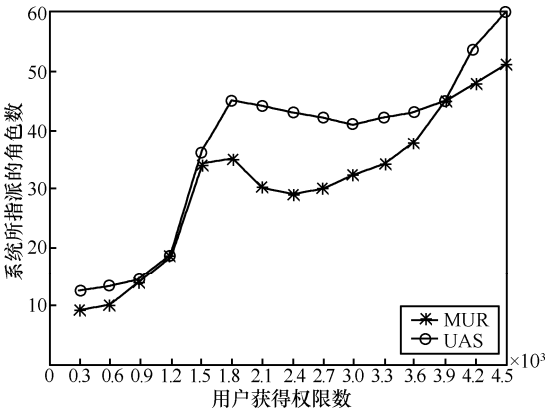

图 8 权限–角色指派数量关系

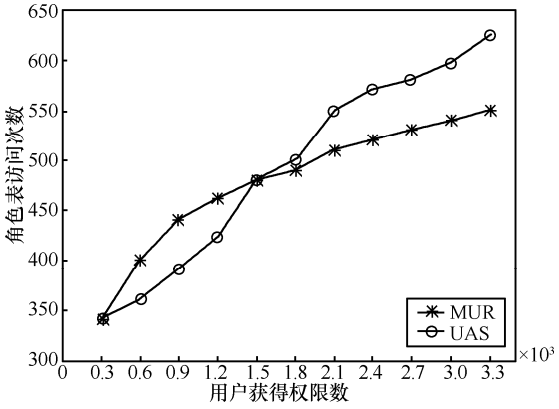

图 9 权限–角色表访问次数关系

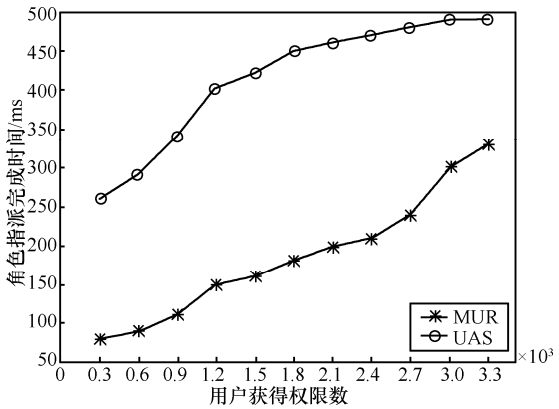

图 10 权限–角色指派完成时间关系

图 9 和图 10 给出了用户获得权限数与系统对角色表的访问次数以及角色指派所完成的时间，由仿真实验结果分析可知，*MUR* 集合及求解算法，针对相同的用户资源请求，相同的权限指派，其指派的角色集数量、角色表访问次数、以及角色指派完成时间均要优于 UAS 集合算法。

6　结束语

本文基于传统的 RBAC 模型提出了一种云计算环境中的访问控制模型 CARBAC。在本模型中，角色分为用户角色和数据拥有者角色，二者都具有相应的继承和激活的角色层次关系。

针对数据拥有者角色如何对用户指派用户角色的问题，本文提出了最小唯一角色集 *MUR* 及其求解算法。云环境中一个用户的任何角色指派，在 *MUR* 集合中都能找到一个角色集，该角色集合的权限与用户通过原有角色指派得到的权限相同，且 *MUR* 中的集合中角色的个数要小于或等于原用户指派的集合。仿真实验表明，该角色查找算法在云环境中的应用，可以使系统能根据用户资源访问需求，选择最小角色集合指派给用户，以达到缩短用户授权时间，节省系统资源的目的。

参考文献：

[1] ALMULLA S A, CHAN Y Y. Cloud computing security management[A]. Proceedings of the International Conference on Engineering Systems Management and Its Applications[C]. Sharjah, UAE, 2010.1-7.

[2] MELL P, GRANCE T. The NIST definition of cloud computing[J]. National Institute of Standards and Technology, 2009, 53(6): 50-57.

[3] 夏鲁宁, 荆继武. 一种基于层次命名空间的 RBAC 管理模型[J]. 计算机研究与发展, 2007, 44(12): 2020-2027.
XIA L N, JING J W. An administrative model for role-based access control using hierarchical namespace[J]. Journal of Computer Research and Development, 2007, 44(12): 2020-2027.

[4] JOSHI J B D, BERTINO E, GHAFOOR A. Hybrid role hierarchy for generalized temporal role based access control model[A]. Proceedings of the Annual International Computer Software and Applications Conference[C]. Oxford, England, 2002. 951-956.

[5] JOSHI J B D, BERTINO E, LATIF U, *et al*. A generalized temporal role-based access control model[J]. IEEE Transaction on Knowledge and Data Engineering,2005, 17(1): 4-23.

[6] LI N, TRIPUNITARA M V, BIZRI Z. On mutually exclusive roles and separation-of-duty[J]. ACM Transactions on Information and System Security, 2007, 10(2): 40-63.

[7] CRAMPTON J, LOIZOU G. Administrative scope: a foundation for role-based administrative models[J]. ACM Transactions on Information and System Security, 2003, 6(2): 201-231.

[8] KOCH M, MANCINI L V, PARISI-PRESICCE F. Administrative scope in the graph-based framework[A]. Proceedings of the ACM Symposium on Access Control Models and Technologies[C]. Yorktown Heights, USA, 2004. 97-104.

[9] JUNG Y, CHUNG M. Adaptive security management model in the cloud computing environment[A]. Proceedings of the International Conference on Advanced Communication Technology[C]. Washington D C, USA, 2010. 1664-1669.

[10] WANG W, LI Z, OWENS R, *et al*. Secure and efficient access to outsourced data[A]. Proceedings of the ACM Workshop on Cloud Computing Security[C]. Chicago, USA, 2009. 55-66.

[11] BERTINO E, BONATTI P A, FERRARI E. TRBAC: a temporal role-based access control model[J]. ACM Transactions on Information and Systems Security, 2001, 4(3): 191-223.

[12] CHANDRAN S M, JOSHI J B D. Towards administration of a hybrid role hierarchy[A]. Proceedings of the IEEE International Conference on Information Reuse and Integration[C]. Las Vegas, USA, 2005. 500-505.

[13] DU S, JOSHI J B D. Supporting authorization query and inter-domain role mapping in presence of hybrid role hierarchy[A]. Proceedings of the ACM Symposium on Access Control Models and Technologies[C]. Lake Tahoe, USA, 2006. 228-236.

作者简介：

杨柳（1983-），男，湖南长沙人，湖南大学博士生，主要研究方向为分布式系统。

唐卓（1981-），男，湖南湘潭人，博士，湖南大学讲师，主要研究方向为分布式系统和分布式系统安全。

李仁发（1957-），男，湖南郴州人，博士，湖南大学教授、博士生导师，主要研究方向为嵌入式计算、无线网络、网络与数字媒体。

张宗礼（1983-），男，山东临沂人，湖南大学硕士生，主要研究方向为分布式系统和分布式系统安全。