Computer Science and Application 计算机科学与应用, 2013, 3, 173-179
http://dx.doi.org/10.12677/csa.2013.33030 Published Online June 2013 (http://www.hanspub.org/journal/csa.html)

A Tenant-Based Access Control Model T-Arbac

Jin Cao[1,2], Peifeng Li[1,2], Qiaoming Zhu[1,2], Peide Qian[1,2]

[1]School of Computer Science and Technology, Soochow University, Suzhou
[2]Jiangsu Provincial Key Lab of Computer Information Processing Technology, Suzhou
Email: caojinhonghu007@163.com

Received: Feb. 23rd, 2013; revised: Mar. 9th, 2013; accepted: Mar. 24th, 2013

Abstract: SaaS service model not only greatly reduces the cost of enterprises, improves business efficiency, but also improves the management level of enterprises and accelerates the pace of innovation. However, it is an unavoidable problem that how to protect data for tenants when the data stored in the SP (service provider) data storage platform. And as the big amount of users in the platform, we also have to take management of user into consideration. This paper designed a new access control model T-ARBAC (Tenant-Administrative Role Based Access Control), combined with role-based access control (RBAC) model, to support multi-tenant access control and facilitate the management of SaaS platform, meeting the requirements of diversity of tenant roles and independent access of coexisting data.

Keywords: Multi-Tenant; Access Control, T-ARBAC

基于租户的访问控制模型 T-ARBAC

曹　进[1,2]，李培峰[1,2]，朱巧明[1,2]，钱培德[1,2]

[1]苏州大学计算机科学与技术学院，苏州
[2]江苏省计算机信息处理技术重点实验室，苏州
Email: caojinhonghu007@163.com

收稿日期：2013 年 2 月 23 日；修回日期：2013 年 3 月 9 日；录用日期：2013 年 3 月 24 日

摘　要：SaaS 服务模式极大地降低了企业成本、提高了企业效率，同时也提高了企业的管理水平及加快了其创新步伐。但由于租户数据存储于服务提供商数据存储平台，如何保障租户的数据安全是一个无法回避的问题，并且平台用户数量巨大，如何有效地对用户进行管理也是一个值得探索的问题。本文结合基于角色的访问控制模型，设计出一个支持多租户、方便租户权限控制与管理的 SaaS 平台访问控制模型 T-ARBAC (Tenant-Administrative Role Based Access Control)，满足了租户访问控制策略多样性与安全、独立访问共存数据的要求。

关键词：多租户；访问控制；T-ARBAC

1. 引言

SaaS (Software as a Service)是一种新的软件应用模式,它极大地减少了企业在信息基础设施上的投入。目前，SaaS 模式大概可以分为四级成熟度[1]，当 SaaS 达到第三级成熟度即“单实例，多租户”模式时，则要求 SaaS 能够满足租户各自配置基础数据并能够隔离租户间的数据，以此来使得每个租户的数据足够安全，此时，该模式是否足够安全并具有相当的可管理性是亟待解决的问题。“单实例，多租户”模式使得处于同一实例的租户数据有被其他租户非法访问的风险。企业用户通过访问该共享的平台使用其服务，但并不希望自己的数据被公司其他部门或其他租户访问，因而如何实现多租户安全的访问控制以及如何

简洁，高效地管理控制策略是急需解决的问题。

在目前存在的访问控制方法中，主流使用的有：自主访问控制(DAC)[2]、强制访问控制(MAC)[3]、基于角色的访问控制(RBAC)[4]。但对于 SaaS 模式来说，不管是安全性还是可管理性，已有的访问控制方法都不能够很好地适用,所以需要根据 SaaS 模型的特点提出新的访问控制模型。目前，国内外学者对此问题已经做出了相关研究，Jing Xu 等人通过禁用 RBAC 模型角色间的继承关系来防止角色间的权限继承，从而防止权限扩散的危险，达到安全的访问控制[5]。Yuri Demchenko 通过安全标签和令牌设计出一种同一的安全访问控制架构[6]。邓集波等人则专注于解决工作流管理中的问题，并提出了 TBAC(基于任务的访问控制模型)访问控制模型，解决了角色权限的动态分配问题[7]。夏鲁宁等则提出了一种基于命名空间访问控制模型 N-RBAC，在此模型中，角色和资源都通过各自不同的命名空间来进行管理，对系统角色及资源管理进行了简化，极大减轻了平台管理人员的工作[8]。马立林等进行了在 SaaS 模式下，在租户数据等方面的访问控制研究，但是并没有给出清晰权限模型的具体定义[9]。

目前的研究中，大多都是基于传统 RBAC 模型的，但是目前研究中依然存在着以下问题：

1) 不能满足租户访问控制策略的多样性要求

传统情况下，企业应用的部署是分开实施的，并且根据企业的不同需求来分配不同的资源，制定不同的角色层次；而在多租户环境下，传统的访问控制模型的使用会使得整个系统资源的分配、角色等级都是全局性的，不能满足租户的个性化需求。

2) 不能满足系统用户类型的增加的要求

传统的基于角色的访问控制模型中，根据 ARBAC97[10]，系统角色又可分为规则角色与管理角色，其中规则角色用来执行系统的业务功能，而管理角色则用来管理系统中角色的创建、权限的分配等工作。而在多租户环境下，每个租户都有各自的规则角色，管理角色，租户间的规则、管理角色是分开的，同时应用的提供者具有更高层次的平台规则角色、平台管理角色，平台规则角色用来对所有的租户进行管理，如资源分配，费用收取等，而这种管理角色在传统 ARBAC97 中是没有的。

3) 不能解决平台管理员与租户管理员之间的权限继承问题

由于平台管理员可以管理租户管理员，所以平台管理员就具有的租户管理员的部分权限，在将 ARBAC97 应用到多租户环境时，平台管理员可以通过部分继承租户管理员的权限来达到管理的目的，但是平台管理员是属于 SaaS 服务提供商的，如果其对租户管理员的权限过度继承，则会出现平台管理员干预租户业务的情况，这种情况是极不安全的。

4) 不能满足租户资源的定制的要求

由于租户需求不同，其可能仅需使用到系统部分功能，而如果不将系统资源分类，分别授予租户，实现差异化订制，则租户就需要为不需要的功能买单。

2. 基于租户的访问控制模型 T-ARBAC

针对以上问题，本文提出面向租户的访问控制模型 T-ARBAC，将传统的 ARBAC 访问控制模型加以改进，应用于多租户架构，以提供更灵活的访问控制。在 ARBAC97 模型中引入租户概念，实现租户的有效隔离。

本文的创新点主要集中在以下几个方面：

1) 根据租户需求，实现租户访问控制策略的定制，满足租户访问控制策略的多样性。

2) 在模型中引入系统管理用户类型，租户管理用户类型。实现系统管理、租户管理的职能分离。同时由于这种职能的分离，取消平台管理员和租户管理员权限的继承关系，而是将其作为两个独立层次，从而实现平台安全管理。

3) 为解决系统资源的按需授予，本文将系统资源按租户需求从总资源池中映射出子资源池给租户，简单且便于租户租费的计算和收取，资源包括各种基础数据以及各种系统功能。

2.1. SaaS 平台用户类型

在 SaaS 平台中，根据用户职能的不同，又可以将用户划分为以下几种类型：

1) 平台管理用户：负责管理平台的规则用户，其管理对象是平台规则用户，包括对该部分用户的授权管理等工作，其工作与租户完全没有交集。

2) 平台规则用户：即前文提到的平台管理员，负

责平台日常维护，如对租户的账号进行审核、对租户进行收费管理以及租户的权限管理等等。但是，平台管理员没有任何权限干涉租户的具体业务，一般提供该服务的企业才会是具有平台管理员。

3) 租户管理用户：访问 SaaS 平台的企业用户，在 SaaS 平台中各租户之间信息是独立的。租户信息包括租户的名称等企业相关信息，主要用来区别各租户，并且由平台管理员对租户账号状态进行管理。各租户可根据需要自行选择 SaaS 平台功能模块。

4) 租户规则用户：一般意义上他们是从属与租户的职员，根据租户管理员分配的权限进行相关的业务。各租户用户只能访问该租户选择的 SaaS 平台的功能模块。

2.2. 形式化描述

由于 SaaS 租户概念的引入，租户间的数据必须保持独立，一般情况下不允许互相访问资源，并且各用户只能在自己所属租户范围内进行访问，因此需要对传统 ARBAC97 管理模型进行扩展，以适应 SaaS 平台权限管理的需要，如图 1 所示。

SaaS 平台权限管理包括一下几个基本元素：

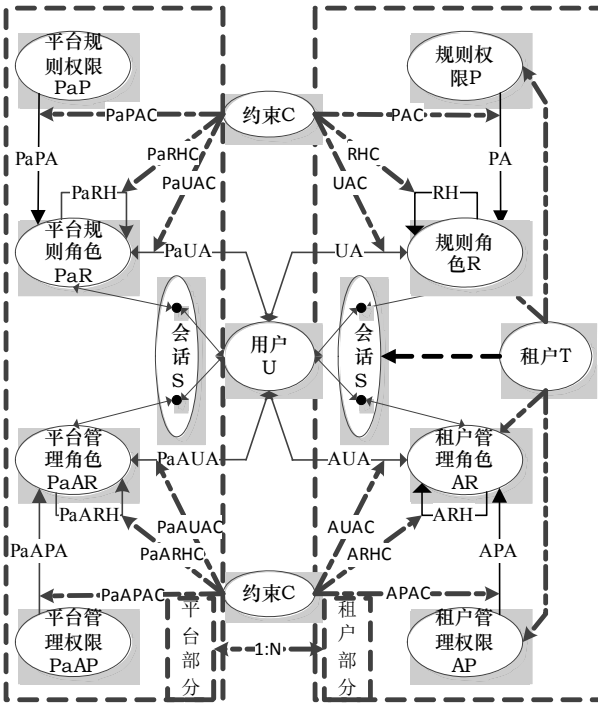

Figure 1. T-arbac access model
图 1. 访问控制模型 T-arbac

1) 租户：SaaS 平台的使用企业，各用户只能在租户许可的范围内使用系统，记作 $T=\{t_1,t_2,\cdots,t_n\}$，表示所有租户的集合。

2) 用户：可以独立访问系统中的资源的主体，记作 $U=\{u_1,u_2,\cdots,u_n\}$，表示所有用户的集合，在 SaaS 平台中，租户 t 的用户集为 $U(t)$，而 SaaS 平台管理员为 $U(Pa)$；

3) 角色：指组织或任务中的工作或岗位，在 SaaS 平台中角色包括规则角色 R，租户管理角色 AR，SaaS 平台规则角色 PaR 和 SaaS 平台管理角色 $PaAR$。其中 $R(t)$、$AR(t)$分别表示租户 t 的规则角色和管理角色。

4) 访问权限：表示允许对资源进行的各项操作，访问权限又分为：

租户规则权限： $P=\{p_1,p_2,\cdots,p_n\}$

租户管理权限： $AP=\{ap_1,ap_2,\cdots,ap_n\}$

平台规则权限： $PaP=\{pap_1,pap_2,\cdots,pap_n\}$

平台管理权限：$PaAP=\{paap_1,paap_2,\cdots,paap_n\}$

用 $P(t)$、$AP(t)$分别表示租户 t 的规则权限集和管理权限集。

5) 资源：所有需要设置权限的资源通称，如某部分数据，记作 $Res=\{res_1,res_2,\cdots,res_n\}$。其中 $Res(t)$ 表示租户 t 的资源集合。

6) 操作：对资源的操作，比如删除、增加等，记作 $Opera=\{opera_1,opera_2,\cdots,opera_n\}$，表示所有操作集，如读，写，执行。其中 $Opera(t)$表示租户 t 的操作集。

PaR/PaP (Platform Role/Platform Permission)：平台规则角色/权限，这些角色使用这些权限来负责平台日常维护，包括租户账号审核、租户状态管理、租户费用管理，租户权限的管理。但是平台管理员没有任何权限干涉租户的具体业务，一般部署该服务的企业才会是平台管理员。

PaAR/PaAP (Platform Administrative Role/Platform Administrative Permission)：平台管理角色/权限，这些角色使用这些权限来维护平台用户，规则角色等，一般整个 SaaS 平台中也只有较少几个这样的角色。

PaRH/PaARH (Platform Role Hierarchy/Platform Administrative Role Hierarchy)：平台规则角色层次/平台管理角色层次。

PaPAC/PaAPAC：平台规则权限/平台管理权限分

配约束，定义平台规则权限/平台管理权限在分配给规则角色/管理角色时的相关约束条件。

PaRHC/PaARHC：平台规则角色/平台管理角色层次约束，定义平台规则角色/平台管理角色间继承时的相关约束条件。

PaUAC/PaAUAC：平台规则角色－用户/平台管理角色－用户分配约束，定义了将平台规则角色/平台管理角色分配给用户时的相关约束条件。

T(Tenant)：租户。租户与相关部分用虚线连接，表示从属关系，单个租户包含多个规则角色、规则权限、管理角色、管理权限；而会话集、约束集中为所有租户各自的会话集的合集。并且由于平台的管理角色与租户没有关系，平台对租户的管理通过平台规则角色来完成。

其中平台部分与租户部分的 1:N 关系，表示该模型的平台控制结构只有一个，而可以有多个租户控制结构，由各租户自行实现。

与传统 ARBAC97 不同的是，由于租户概念的引入，在系统中分别增加了租户－用户管理，租户－角色管理，租户－权限管理，通过这些管理模型，可以有效地对同一系统中的不同租户的数据进行隔离管理，并且将 ARBAC97 的单层管理模型扩展到两层管理模型(平台层与租户层)，使得平台与租户管理模型之间得以部分分开，以实现平台管理者对租户的不可干预，有效保障各租户的隐私。以下是具体的相关定义：

租户规则部分定义：

定义 1：$xT \subseteq x \times T$，其中 $x \in \{R, U, P\}$，分别表示从角色集/用户集/权限集/到租户集的多对一映射。

定义 2：$RxT \subseteq RT \times xT$，其中 $x \in \{\mathrm{U}, P, R\}$，从角色集到/用户集/权限集/角色集的多对多映射。

租户管理部分定义：

定义 3：$xT \subseteq x \times T$，其中 $x \in \{AR, AP\}$，表示从管理角色集/管理权限集到租户集的多对一映射。

定义 4：$ARxT \subseteq ART \times xT$，其中 $x \in \{U, AP, AR\}$，从管理角色集到用户/管理权限集/管理角色集的多对多映射。

平台规则部分定义：

定义 5：$PaRx \subseteq PaR \times x$，其中 $x \in \{PaP, U, PaR\}$，表示某 SaaS 应用中，从 SaaS 平台规则角色集到 SaaS 平台规则权限集/用户/平台规则角色集的多对多映射。

平台管理部分定义：

定义 6：$PaARx \subseteq PaAR \times x$，其中 $x \in \{PaAP, U, PaAR\}$，表示某 SaaS 应用中，从 SaaS 平台管理角色集到 SaaS 平台管理权限集/用户/平台管理角色集的多对多映射。

在整个 SaaS 应用中，还有如下定义：

定义 7：$\mathrm{Permission} : \mathrm{Opera} -> \mathrm{Res}$，表示操作到资源的映射关系，用(opera, res)二元组表示。如 $(read, res1) \in$ Permission 表示对 res1 具有读取权限。

定义 8：$Users : S -> U$，表示会话到用户的映射关系，用(session, user)表示。如 $(\mathrm{session1}, user1) \in Users$ 表示 session1 是属于用户 user1 的会话。

定义 9：$Roles : S -> 2^{RT \cup ART \cup PaR \cup PaAR}$，表示会话到角色集的映射关系，用二元组(session, roleset)表示。

如 $(\mathrm{session1}, \mathrm{roleset1}) \in Roles$ 表示 session1 所具有的相应角色集，其中，roleset 表示一组角色的集合。并且有：(在实现时，可以将服务提供商视为一个特别的租户)

$$Roles(s_i) \subseteq \{r \mid (\exists r' \geq r)[(r', user(s_i), t) \in RUT \cup ARUT \cup PaRU \cup PaARU]\}$$

其中，会话权限如下：

$$\mathrm{Permission}(s_i) \subseteq \bigcup_{r \in Roles(s_i)} \{p \mid (\exists r'' \leq r)[(r'', p, t) \in RPT \cup ARPT \cup PaRPaP \cup PaARPaAP]\}$$

由于管理权限只能赋与管理角色，规则权限只能赋给规则角色，所以租户规则许可、租户管理许可、平台规则许可、平台管理许可两两无交集，即 P、AP、PaP、$PaAP$ 两两交集为空。

在将用户分为四种后，普通租户的用户－角色、权限－角色的分配关系也较传统 ARBAC97 发生了变化。

2.3. 用户－角色分配与撤销

定义 10：假设 CR 表示分配 R 中角色时的前提条件集合，用户 u 可以拥有角色 x 的先决条件是：

$\left(\left(\exists x' \geq x\right) \wedge \left(\left(x,t\right) \in RT\right)\right)\left(x,u,t\right) \in RUT$ 。

由于平台的复杂性以及安全的需要，通常用户角色的分配和撤销是有不同的管理员做的。其中，can-assign 和 can-revoke 分别表示分配和撤销规则集。

1) can-assign $\subseteq ART \times CR \times 2^{RT}$

如 can-assign$\left(x,y,\{a,b,c\},t\right)$，表示租户 t 管理角色 x，能够将{a, b, c}中的角色，分配给具备条件 y 的用户。

2) can-revoke $\subseteq ART \times 2^{RT}$

如，can-revoke$\left(x,Y,t\right)$，表示租户 t 管理角色能够撤销用户角色中的属于 Y 集合的角色。

2.4. 权限－角色分配与撤销

定义 11：假设 CP 表示分配 P 中权限的前提条件集合，权限 p 可以分配给角色 x 的先决条件是：

$$\left(\left(\exists x' \geq x\right) \wedge \left(\left(x,t\right) \in RT\right)\right)\left(p,x,t\right) \in PRT$$

1) can-assignp $\subseteq ART \times CP \times 2^{PT}$

如 can-assignp$\left(x,y,\{a,b,c\},t\right)$，表示租户 t 管理角色 x，能够将{a, b, c}中的权限，分配给具备条件 y 的角色。

2) can-revokep $\subseteq ART \times 2^{PT}$

如：can-revokep$\left(x,Y,t\right)$ 表示租户 t 管理角色 x 能够撤销用户角色中的属于 Y 集合的角色。

2.5. 角色－角色分配与撤销

定义 12：假设 CR 表示分配 R 中角色时的前提条件集合

1) can-assignr $\subseteq ART \times CR \times 2^{RT}$

如 can-assignr$\left\{x,y,\{a,b,c\},t\right\}$，表示租户管理角色 x，能够将{a, b, c}中的角色，分配给具备条件 y 的角色。

2) can-revoker $\subseteq ART \times 2^{RT}$

如 can-revoker$\left(x,Y,t\right)$，表示管理角色能够撤销用户角色中的属于 Y 集合的角色。

2.6. 模型在 SaaS 应用中解决的问题

1) 租户访问控制策略的多样性

如图 1 可知，模型的租户部分允许每个租户自己定义基于 RBAC 的访问控制策略，每个租户的用户、角色、权限都可以通过租户这一信息实现隔离，这样，每个租户都可以订制专属于自己的人事架构，权限系统，从而实现租户访问控制策略的多样性，满足各租户的不同需求。

2) 系统管理、租户管理的职能分离

SaaS 应用中，系统的许多维护皆由 SaaS 服务提供商进行，租户自己只需要管理与自己业务相关的事务。从图 1 中可以看出，通过将管理层次由以往的一层扩展为现在的两层(平台部分和租户部分)，实现了平台与租户的分开管理，从而实现系统管理、租户管理的职能分离。

3) 租户资源的按需授予

本文将系统资源按租户需求从总资源池中映射出子资源池给租户(资源包括各种系统功能及相应的数据)，而后租户的管理员在此资源的基础之上制定相应的访问权限。在系统权限建立初期，系统管理员根据租户的需求会得到相应的资源列表，这个列表由租户根据租约订制，而后在此基础上进行相应的权限建立。

3. T-ARBAC 模型的具体应用

大型企业级应用对 SaaS 服务模式的访问控制提出新的要求，本文以苏州押运管理系统为例，该系统主要实现银行钱箱押运过程的监管及相关业务，以服务平台的形式提供，并部署在云平台上，该云平台使用 VMware vSphere 搭建，实现了物理资源的逻辑抽象和统一表示，通过它可以提高资源利用率，并能够根据用户业务需求的变化，快速、灵活地进行资源部署[11]。业务流程可表示为：银行提交押运需求–>押运公司审核需求并生成押运任务–>押运中心派专人执行押运任务–>银行网点与押运人员交接钱箱。

3.1. 访问控制数据库表结构设计

如图 2 所示，为访问控制模型的数据库结构表示图，其中：

Tenant 表：用来记录租户的基本信息，在其他表中都有相应的租户 ID，以此来区分租户各个部分之间的信息。

User 表：用来记录所有的用户信息，包括平台用户以及各租户所属的用户；

Role 表：用来记录所有的角色信息，包括平台角

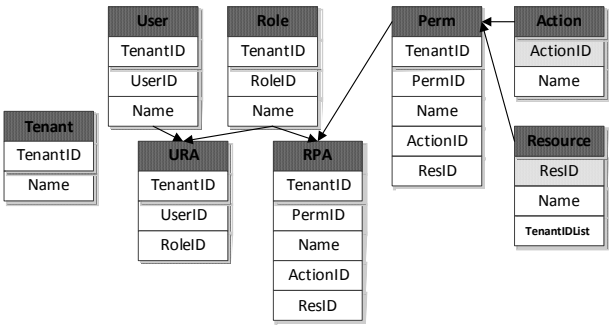

Figure 2. Design of access control database
图 2. 访问控制数据库设计图

色以及各租户所属的角色；

URA 表：记录角色到用户的指派关系；

Perm 表：用来记录所有的权限信息，包括平台权限以及各租户所属的权限；

RPA 表：记录角色到权限的指派关系；

Action 表：记录对平台资源的所有基本操作信息，如对资源的读、写、执行等；

Resource 表：记录平台资源的所有信息，并且在资源表 Resource 中，有 TenantIDList 列，该列用来记录可以使用该类资源的租户的 ID，从而实现资源的按需分配。

3.2. 访问控制主要流程

在该平台中，有以下三个重要的访问控制流程：

1) 押运公司注册流程。该流程主要对押运公司的注册及初始化进行控制。押运公司在使用该平台之前，需要进行注册，待服务提供商授权后，方能使用该平台，具体注册流程如图 3 所示。

首先是押运公司注册，包括企业名称等租约相关信息，并设置管理员账号，然后服务提供商对其进行审核，在审核通过后为其生成租户管理用户，租户在得到该管理用户后，可以自己根据实际情况进行角色划分，权限分配，系统资源选择等系统初始化工作。当租户系统订制完成以后，由服务提供商进行收费，并授权其使用系统，最后押运公司就可以进行押运业务。

该部分对 T-ARBAC 模型的平台管理部分进行了验证，实验很好地说明了平台管理与租户管理分开的优点及有效性。

2) 用户资源访问控制流程。该流程主要验证用户对相关资源的访问是否合法。在服务提供商授权该租户使用该平台后，押运公司的用户就可以使用其功能，在具体使用过程中，对资源的访问控制流程如图 4 所示。

用户访问相应资源时，平台根据其登录信息获取其相应的角色权限，并判定其是否有权限访问该资源，通过验证后，用户方可以对该资源进行访问。

该部分对 T-ARBAC 模型的租户规则访问控制部分进行了验证，实验很好地实现了租户数据的隔离，

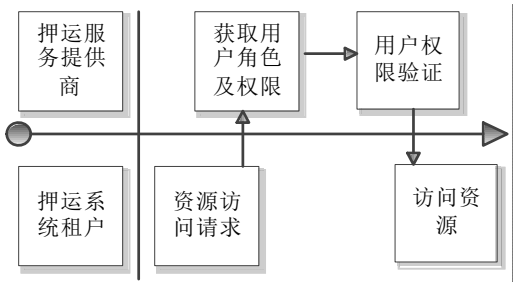

Figure 3. Procedure of application for service
图 3. 押运企业申请流程

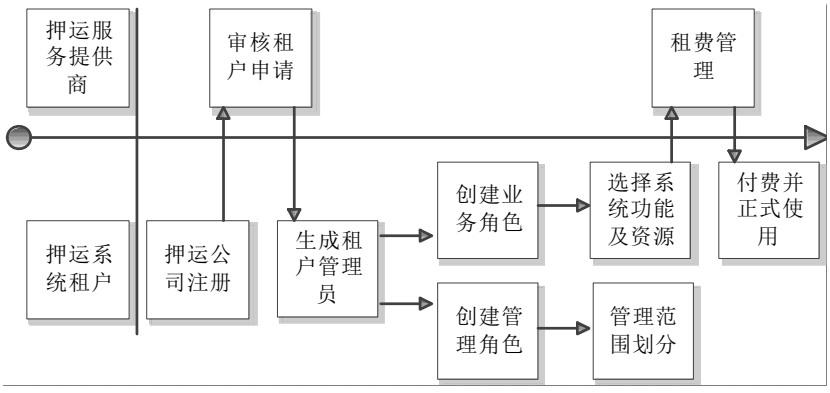

Figure 4. Procedure of access control of resource
图 4. 用户资源访问控制流程

有效保证了在同一实例下的每个租户数据安全。

3) 管理员管理控制流程。该流程主要说明了对管理员的管理行为的控制。与普通用户的资源访问控制相对应，租户管理员在进行管理时也要遵循相应的流程，如图 5 所示。

管理用户在提出管理请求后，平台根据其登陆信息获取其角色权限及管理范围，并验证其是否有权进行管理，在通过验证后，管理用户方可以对自己所在押运公司的相应信息进行管理。

该部分对 T-ARBAC 模型的租户管理访问控制部分进行了验证，实验很好地控制的租户管理者的管理范围，使得租户的管理行为都在自身的安全域内进行。

通过该 SaaS 平台，各押运公司可以很方便地进行押运系统的业务进行及内部管理，真正实现 SaaS 第三级成熟度模型，有效地对处于同一实例的租户的有效隔离，验证了 T-ARBAC 在这种模式下的有效性。

4. 结论

本章通过对基于角色的访问控制模型进行扩展，使其适用于“单实例，多租户”的 SaaS 的应用模式，比通过具体应用本文可以得出以下结论：

1) 实现租户访问控制策略的定制，满足租户访问控制策略的多样性。

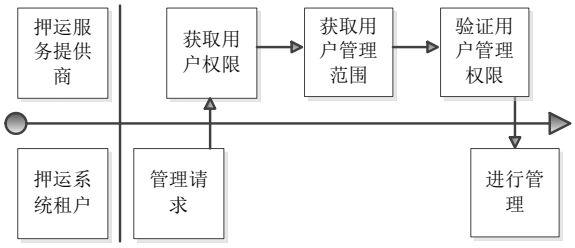

Figure 5. Procedure of access control of management
图 5. 管理员管理控制流程

2) 将平台管理与租户管理分离开来，实现两层管理模型，避免因平台管理角色与租户管理角色间权限的过度继承而对租户业务可能造成的伤害。

3) 提出租户资源池的概念，服务提供商更具租户的需求，从系统资源池中映射出子资源池分配给租户，从而避免了资源一次性分配带来的巨大开销，更加满足按需服务的要求。

该模型很好地适应 SaaS 应用的要求，并提供完善的访问控制管理模型，简化了租户的权限管理。

参考文献 (References)

[1] 许四平. SaaS 软件即服务模型研究[J]. 硅谷, 2009, 4.
[2] C. P. Pfleeger. Security in computing (2nd Edition). Upper Saddle River: Prentice Hall, Inc., 1997.
[3] R. S. Sandhu, P. Samarati. Access control: Principles and practice. IEEE Press, 1994, 32(9): 40-48.
[4] D. F Ferraiolo, D. R. Kuhn. Role-based access controls. Reprinted from 15th National Computer Security Conference, Baltimore, 13-16 October 1992: 554-563.
[5] J. Xu, J. L. Tang and D. J. He. Research and implementation on access control of management-type SaaS. The 2nd International Conference on Information Engineering and Computer Science, Chengdu, 16-18 April 2010: 388-392.
[6] Y. Demchenko. Access control infrastructure for on-demand provisioned virtualised infrastructure services. International Conference on Collaboration Technologies and Systems (CTS), Philadelphia, 23-27 May 2011: 466-475.
[7] 邓集波, 洪帆. 基于任务的访问控制模型[J]. 软件学报, 2003, 14(1): 76-82.
[8] 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.
[9] 马立林, 李红. 基于 RBAC 的 SaaS 系统的权限模型[J]. 计算机应用与软件, 2010, 4.
[10] R. Sandhu, Q. Munawer. The RRA97 model for role-based administration of role hierarchies. Proceedings of 14th Annual Computer Security Applications Conference, Phoenix, 7-11 December 1998: 39-49.
[11] 连鸿鹏. 云计算 VMware vSphere 虚拟化技术的架构分析[J]. 软件导刊, 2012, 8(11): 6-7.

