一种新的访问控制模型——TBPM-RBAC

王 瑜 卿斯汉
(中国科学院软件研究所信息安全技术工程研究中心 北京100080)

摘 要 基于角色的访问控制RBAC(role-based access control)能够降低访问控制管理工作的复杂性,但是要在RBAC中高效合理地为角色配置权限仍然具有一定的难度,而且RBAC不适合处理存在依赖和时序关系的访问控制。本文通过在RBAC中引入任务机制解决以上两点困难,即权限被授予任务,任务被授予角色,角色只能使用它正在执行的任务所允许的权限。提出了TBPM-RBAC(task-based permissions management in RBAC)模型,给出了模型的定义,对模型进行了分析并给出了模型的两个应用示例。

关键词 角色,RBAC,访问控制,任务

A New Access Control Model——TBPM-RBAC

WANG Yu QING Si-Han
(Engineering Research Center for Information Security Technology, Institute of Software, Chinese Academy of Sciences, Beijing 100080)

Abstract RBAC(role-based access control)can reduce the complexity of the management of access control, but it's still rather difficult to assign permissions to roles efficiently and reasonably, further more, RBAC is not fit to manage the access controls where exists dependency and sequence. This paper tries to solve the two difficulties mentioned above by embedding task mechanism in RBAC, that is, permissions are assigned to tasks, tasks are assigned to roles, and a role can only use the permissions that are allowed by the tasks it's executing. A model called TBPM-RBAC (task-based permissions management in RBAC)is proposed, then we present the definitions of the model, analyze the model and give two application examples of the model.

Keywords Role, RBAC, Access control, Task

1 引言

基于角色的访问控制RBAC(role-based access control)是近年来访问控制领域的一大热点,NIST(National Institute of Standards and Technology)于2001年制定了RBAC标准,根据该标准中的定义,角色是表示用户和权限之间多对多关系的一种方法,用来指明用户的职权和责任[1],这样访问权限不再直接授予用户而是授予角色,用户通过授予适当的角色获取相应的权限。在实际系统中,用户的权限会经常变化,而角色和权限之间的映射则相对稳定,所以将权限授予角色,然后再将角色分配给用户,可以大大简化安全管理工作。RBAC还支持角色层次关系和两个著名的安全原理:最小特权原理和职责隔离原理[2],所以RBAC能够很好地满足现实系统的安全需求。目前,RBAC机制已经成功地应用到许多系统的安全管理中,如Windows NT, Netware, Solaris, Oracle等。

RBAC虽然具有适应性强,管理方便,易于扩展,与策略无关等诸多优点,但在实际系统的开发和应用中,我们感到RBAC仍然有值得改进的地方,主要归纳为以下两点:

1. 角色与权限之间在抽象层次上的不同常常带来权限配置上的困难。角色在概念上对应一个职位,是较高层次上的概念,而权限则许可在数据上的操作,是较底层的概念。一个角色在工作中是否会用到某些数据,怎样配置访问权限才最适当,最能满足最小特权原理和职责隔离原理,现有的RBAC研究对此缺乏明确的标准和相应的指导。因此,在系统开发和应用的过程中,常常会出现"角色易设,权限难配"的情况。从RBAC96模型中开始出现的角色继承概念[3],可以在一定程度上缓解权限配置上的困难,但把解决问题的希望完全寄托在角色继承上是不现实的,因为困难产生的根源在于抽象层次上的差别,而且过度使用角色继承将给角色管理带来困难。我们认为应该在角色和权限之间引入一个适当的概念,弥补它们之间在抽象层次上的差别。

2. RBAC中没有依赖和时序的概念,这样就难以处理存在依赖关系的访问控制。例如系统中的一个操作序列包含顺序执行的三个任务,其中任一任务失败将导致整个操作序列取消,系统恢复到第一个任务开始之前的状态。某一角色只能在第二个任务中访问某项数据,在其它时间均不能访问该数据,这样的访问控制在RBAC中是难以实现的。在RBAC中,角色被授予权限后,将一直拥有该权限直到权限被收回,这样在RBAC中解决上面例子中的访问控制问题,就要频繁地对角色进行权限的授予和收回,不仅加重了管理负担,而且还会使同一角色的权限经常变化,出现概念上的不一致性。RBAC96模型中列举的约束关系主要包括角色互斥、角色最大成员数、前提角色和前提权限等[3],仅仅基于这些约束关系将难以实现上面例子中提出的访问控制问题。在RBAC中引入时间属性也不能有效地解决上面例子中的问题,因为依赖和时序问题使用的是相对时间,而RBAC中引入时间属性使用的是绝对时间,在上面的例子中,角色只在第二个任务中具有访问权限,而第二个任务在第一个任务完成之后才能开始,它的执行时间并不固定,这样的约束关系是无法用绝对时间来表达的。我们认为有必要在RBAC中引入新的机制实现存

在依赖和时序关系的访问控制。

为了有效地解决以上两个问题，我们在RBAC中引入任务机制进行权限管理，这一想法源自基于任务的访问控制TBAC(task-based access control)[4,5]。TBAC是近年来访问控制领域的又一热点，主要用于解决工作流中的访问控制问题。工作流是一类能够完全或者部分自动执行的过程，它根据一系列定义的过程规则，使文档、信息或任务在不同的执行者之间进行传递与执行。当数据在工作流中流动时，对其进行处理的用户不断变化，用户的权限也不断变化，因此其它一些访问控制模型，如DAC、MAC和RBAC等，在此并不适用。TBAC从基于任务的角度进行建模，在任务处理过程中对数据实施动态的访问控制，因此一个用户对数据是否具有访问权限还要看他是否正在执行相应的任务。在TBAC中，任务之间可能具有依赖关系[4,5]，这样任务的执行必须要依照给定的次序，满足限定的条件，存在时序关系。

已有的将RBAC和TBAC结合起来的有益尝试包括George Coulouris等人在PerDis群件平台中实现的访问控制[6]和Dirk Jonscher提出的应用于数据库设计和管理过程中的扩展访问控制[7]。George Coulouris等人的工作[6]主要针对分布式应用中对象的共享访问，试图解决群件应用中的数据共享问题。在PerDis平台中，任务特指存在数据共享的协作活动，用来定义一组角色对一组对象的操作权限。Dirk Jonscher的文章[7]中虽然有任务的概念，但并没有引入任务依赖关系，而且他的工作主要是对数据库设计和管理过程中的访问控制进行扩展，重点在于引入责任等概念对权限进行分类。

我们试图面向通用系统而不仅仅是分布式系统或者数据库应用进行建模，通过在RBAC中引入任务机制，解决前面提出的RBAC值得改进的两个问题。我们认为，在RBAC中引入任务机制，既能填补角色和权限之间在抽象层次上的差别，简化权限配置的工作，易于正确性验证，又能使RBAC适合处理存在依赖和时序关系的访问控制。

本文提出了一个基于任务进行RBAC权限管理的访问控制模型TBPM-RBAC(task-based permissions management in RBAC)，本文第2节对模型进行定义并分析模型的可行性与适用性，第3节给出模型应用的两个示例，最后是简短结论。

2 基于任务进行RBAC权限管理的访问控制模型TBPM-RBAC

基于任务进行RBAC权限管理的访问控制模型TBPM-RBAC如图1所示。与NIST标准中的RBAC模型[1]不同，在TBPM-RBAC模型中，访问权限不再是直接授予角色，而是先授予任务，然后再将任务授予角色，用户与角色之间的映射关系则没有变化。直观地说，在TBPM-RBAC模型中，用户要行使一定的访问权限，必须要满足两个条件：

a) 用户具有该访问权限，亦即，用户授予的某个角色要具有该访问权限。

b) 用户正在执行的任务允许使用该访问权限。

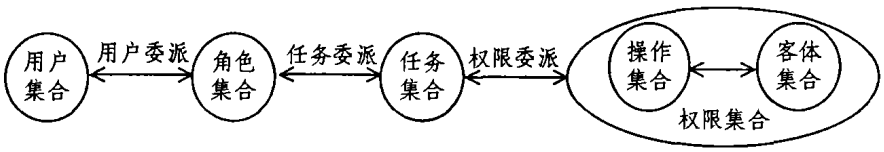

图1 基于任务进行RBAC权限管理的访问控制模型TBPM-RBAC

2.1 模型定义

用户：用户是一个可以自主访问系统中资源的主体，用户可以是人、网络中的计算机或者具有智能的自治软件，如agent，我们用*USERS*表示系统中的用户集合。

角色：角色是组织中的工作职能的语义表达，代表了被授予角色的用户的职权和责任，我们用*ROLES*表示系统中的角色集合。

操作：操作是对系统中的资源进行访问的动作，在文件系统中可能是读、写或执行，在数据库管理系统中则可能是插入、删除、添加或更新，我们用*OPS*表示系统中的操作集合。

客体：客体是系统中包含或接收信息的实体，它可以代表信息容器，如系统中的文件或目录，也可以代表可耗尽的系统资源，如打印机或磁盘空间等，我们用*OBS*表示系统中的客体集合。

任务：任务是系统执行过程中的一个逻辑单元，它由一系列授权操作组成，一个任务可以包含多个子任务，一个任务可以由一个用户完成，也可以由多个用户协作完成，我们用*TASKS*表示系统中的任务集合。

权限：权限是对一个或多个在系统中由TBPM-RBAC保护的客体进行操作的许可，我们用*PRMS*表示系统中的权限集合。

用户委派：用户委派表达系统中对用户授予角色的情况，我们用UA表示用户委派，$UA \subseteq USERS \times ROLES$，如果$(u, r) \in UA$，则用户$u$被授予了角色$r$。

任务委派：任务委派表达系统中角色承担任务的情况，我们用TA表示任务委派，$TA \subseteq ROLES \times TASKS$，如果$(r, t) \in TA$，则角色$r$承担了任务$t$。

权限委派：权限委派表达系统中任务执行时所具有的权限，我们用PA表示权限委派，$PA \subseteq TASKS \times PRMS$，如果$(t, p) \in PA$，则任务$t$执行时具有权限$p$。

依赖：依赖是指系统中各任务之间存在的关系，包括顺序依赖、失败依赖、撤销依赖、代理依赖和分权依赖。我们用*Dependency*表示依赖集合，用$DKind = \{$顺序依赖，失败依赖，撤销依赖，代理依赖，分权依赖$\}$表示依赖种类集合，则$Dependency \subseteq TASKS \times TASKS \times DKind$。如果$(t_1, t_2, d) \in Dependency$，则任务$t_2$对任务$t_1$有依赖关系$d$。

顺序依赖：$\forall t_1, t_2 \in TASKS$，如果$t_2$必须在$t_1$完成之后才能开始，则称$t_2$对$t_1$顺序依赖。

失败依赖：$\forall t_1, t_2 \in TASKS$，如果$t_2$必须在$t_1$失败之后才能开始，则称$t_2$对$t_1$失败依赖。

撤销依赖：$\forall t_1, t_2 \in TASKS$，如果$t_1$被撤销，那么$t_2$也被撤销，则称$t_2$对$t_1$撤销依赖。

代理依赖：$\forall t_1, t_2 \in TASKS$，如果$t_1$被撤销，那么$t_2$继承$t_1$的权限，则称$t_2$对$t_1$代理依赖。

分权依赖：$\forall t_1, t_2 \in TASKS$，如果$t_1$和$t_2$必须由不同的角色完成，则称$t_2$对$t_1$分权依赖。

任务状态：系统中的任务是动态变化的，在不同的时间会有不同的任务状态，包括睡眠状态、激活状态、有效状态、挂起状态、完成状态和失败状态。我们用 TS 表示任务状态集合，用 $SKind=$｛睡眠状态，激活状态，有效状态，挂起状态，完成状态，失败状态｝表示任务状态种类集合，则 $TS\subseteq TASKS\times Skind$。如果 $(t,s)\in TS$，则任务 t 的当前状态为 s。TBPM-RBAC 模型中任务状态的迁移如图2所示。

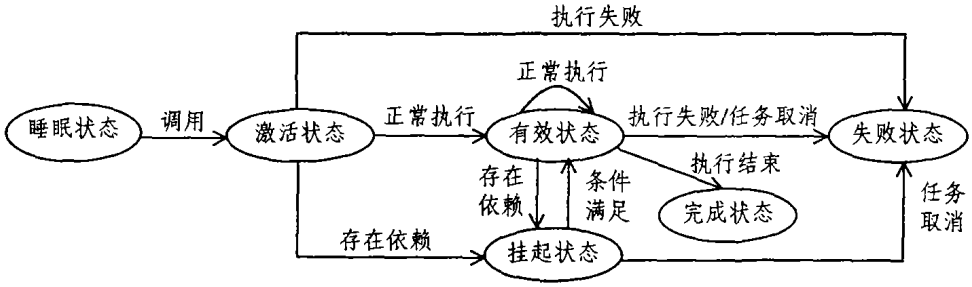

图2 TBPM-RBAC 模型的任务状态迁移图

睡眠状态：表示任务尚未生成。

激活状态：表示任务已经生成，开始执行。

有效状态：表示任务正在正常执行中。

挂起状态：表示任务因与其它任务之间存在的依赖条件不被满足而被挂起，进入等待。

完成状态：表示任务正常执行结束。

失败状态：表示任务执行出错导致不正常结束或者任务被用户取消。

角色具有的权限：角色被授予的任务所具有的权限集合，角色 r 具有的权限 $PRMS(r)=\{p\mid(\exists\ t)((r,t)\in TA\wedge(t,p)\in PA)\}$。

用户具有的权限：用户授予的角色所具有的权限集合，用户 u 具有的权限 $PRMS(u)=\{p\mid(\exists\ r)((u,r)\in UA\wedge p\in PRMS(r))\}$。

用户当前的权限：用户当前正在执行的任务所具有的权限集合，用户 u 当前的权限 $PRMS(u)=\{p\mid(\exists\ r,t)((u,r)\in UA\wedge(r,t)\in TA\wedge(t,p)\in PA\wedge((t,$激活状态$)\in TS\vee(t,$有效状态$)\in TS))\}$。

2.2 模型分析

TBPM-RBAC 模型采用角色和任务进行访问控制的权限管理，可以看成是将 TBAC 模型嵌入到 RBAC 模型中。因为 TBPM-RBAC 模型中保留了 RBAC 模型的全部元素，所以 TBPM-RBAC 模型可以用来描述 RBAC 模型，并且具有比 RBAC 模型更强的描述能力。应用 RBAC 模型建模的系统可以很方便地转化为应用 TBPM-RBAC 模型建模的系统。

TBPM-RBAC 模型很好地解决了在前文中提出的 RBAC 值得改进的两个地方：

1. 通过引入任务的概念来填补角色与权限之间在抽象层次上的差别，解决了权限配置的困难。给定一个角色，由它的职能和责任可以方便地配置它所能执行的任务，一般来说这是一个很直观的过程，所以配置角色所能执行的任务比配置角色所具有的权限要容易的多。另一方面，给定一个任务，由它的执行目标和执行过程可以方便地配置它所具有的权限，一般来说这也是一个很直观的过程，所以配置任务所具有的权限比配置角色所具有的权限要简单的多。不难看出，在引入任务概念之后，权限配置工作的整体难度降低了，更加直观，也易于保证其正确性。

2. 通过引入任务机制加强 RBAC 表达依赖和时序的能力，使之能够有效地处理存在依赖关系的访问控制。在 TBPM-RBAC 模型中，角色并不总能使用它所具有的权限，角色可以执行某些任务，任务又具有某些权限，角色只有在执行任务时才能使用该任务所具有的权限。任务之间具有依赖和时序的关系，当任务执行的条件不被满足时，任务会被挂起，条件满足以后才可以继续执行，或者由操作人员取消该任务的执行。容易看出，因为引入了任务机制，TBPM-RBAC 模型比 RBAC 模型具有更强的描述能力，能够处理更复杂的访问控制，而且也方便管理人员进行权限配置和管理。

TBPM-RBAC 模型能够很好地支持最小特权原理。在进行权限配置时，只授予用户所需的角色，只授予角色所需的任务，只授予任务执行时所需的权限，所以用户只有在执行任务时才能使用相应的权限，而在其他时间均不能使用相应的权限，这样可以有效地防止权限的滥用和误用。

TBPM-RBAC 模型还能够很好地支持职责隔离。在进行权限配置时，可以设置互斥角色、互斥任务和互斥权限。互斥角色必须由不同的用户担任，互斥任务必须由不同的角色执行，互斥权限必须被授予不同的任务。被授予互斥权限的任务之间是互斥的，执行互斥任务的角色之间是互斥的。把系统中的敏感工作设置为互斥的几部分，可以有效地防止因用户权力过大而导致的破坏行为。

3 TBPM-RBAC 模型的应用示例

在应用 TBPM-RBAC 模型开发系统时，一个关键问题是如何在系统中实现任务机制。我们认为，只要能体现 TBPM-RBAC 模型的思想的实现就是好的实现，具体实现方法则可以灵活多变。要标识一个任务的开始，可以在系统中显式地提供一个菜单项，单击该菜单项以后就开始某个任务，也可以通过识别用户的特征操作序列识别某个任务的开始。对于新生成的任务，应该在系统中开辟新的数据区域保存其状态信息；当任务在执行过程中状态发生改变时，系统中保存的该任务的状态信息应该及时修改；当任务执行成功或者被取消时，应该修改或从系统中删除该任务的状态信息。

下面给出两个 TBPM-RBAC 模型的应用示例，3.1节中的示例说明如何通过在 RBAC 中引入任务机制实现 TBPM-RBAC，3.2节中的示例说明如何应用 TBPM-RBAC 中任务之间的依赖关系实现职责隔离。

3.1 应用示例1——通过在 RBAC 中引入任务机制实现 TBPM-RBAC

本节以操作系统中的系统管理员角色为例，如果应用 RBAC 模型，该角色被授予的权限如表1所示，如果应用 TBPM-RBAC 模型，则该角色可执行的任务及各任务具有的

权限如表2所示。

表1 系统管理员角色权限配置表——RBAC模型

角色	权限
系统管理员	读审计报告、写审计分析、打印、删除打印任务、修改打印任务、查看打印任务、备份文件系统、改变磁盘分区大小、改变磁盘簇大小、添加目录、删除目录、修改目录、设置网络参数、启动网络服务、关闭网络服务、查看进程、改变进程优先级、杀死进程、写系统配置文件、添加文件、添加用户、删除用户、设置用户初始密码、为用户配置角色、添加角色、删除角色、为角色配置权限

表2 系统管理员角色任务权限配置表——TBPM-RBAC模型

角色	任务	权限
系统管理员	审计分析	读审计报告、写审计分析
	打印机管理	打印、删除打印任务、修改打印任务、查看打印任务
	文件系统管理	备份文件系统、改变磁盘分区大小、改变磁盘簇大小、添加目录、删除目录、修改目录
	网络管理	设置网络参数、启动网络服务、关闭网络服务
	进程管理	查看进程、改变进程优先级、杀死进程
	软件安装	写系统配置文件、添加目录、添加文件
	用户管理	添加用户、删除用户、设置用户初始密码、为用户配置角色
	角色管理	添加角色、删除角色、为角色配置任务
	任务管理	添加任务、删除任务、为任务配置权限

注意表2中"添加目录"这一权限出现了两次，分别出现在"文件系统管理"任务和"软件安装"任务之中。另外，因为引入了任务机制，所以表2中新增了"任务管理"任务、"添加任务"权限、"删除任务"权限和"为任务配置权限"权限；"角色管理"任务中的"为角色配置权限"权限也改为"为角色配置任务"权限。

不难看出，表1中系统管理员角色的权限众多，关系错综复杂，难以管理，难以保证配置结果的正确性，而在引入任务机制后，表2中根据系统管理员角色的职能创建了九个任务，然后根据任务的执行目标授予适当的权限，条理清晰，易于配置，也易于验证配置结果的正确性。

3.2 应用示例2——应用TBPM-RBAC中任务之间的依赖关系实现职责隔离

本节以系统中常见的新建用户的操作为例。一般来说，新建一个用户需要添加新用户的身份资料，然后授予权限，最后设定初始密码，从系统安全性考虑，以上三步应该分别由三个不同的用户完成，以实现职责隔离，防止系统中有人私自添加用户并授予非法权限导致对系统的破坏。

如图3所示，在应用TBPM-RBAC模型的系统中，可以把新建用户的操作定义为一个任务，这一任务包含了三个子任务：添加用户、授予角色和设置密码，三个子任务之间存在分权依赖的关系，亦即三个子任务是互斥的，必须由三个不同的角色承担，分别是系统管理员、权限管理员和安全管理员。

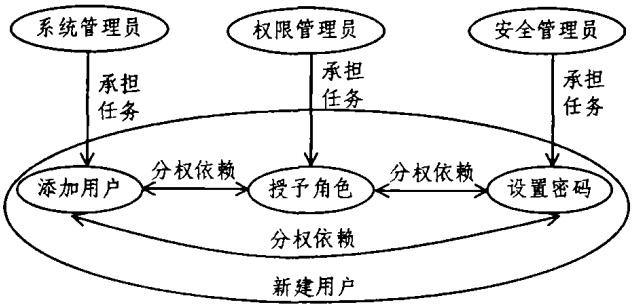

图3 应用TBPM-RBAC模型对新建用户的操作实现职责隔离

由2.2节中对模型的分析可知，图3中的系统管理员、权限管理员和安全管理员是三个互斥的角色，必须授予三个不同的用户。除了分权依赖以外，图3中的三个子任务之间还可能存在其他的依赖关系，比如撤销依赖和顺序依赖，限于篇幅在此不做进一步的讨论。

结论 RBAC和TBAC是当前访问控制领域的两大热点，本文基于在RBAC中引入任务机制的想法，提出了一种新模型——TBPM-RBAC模型，给出了模型的定义，分析了模型的可行性、适用性和安全性，并给出了模型的应用示例。TBPM-RBAC模型能够有效地降低访问控制中权限配置的难度，对存在依赖关系的访问控制也有很好的支持。今后的工作主要是对TBPM-RBAC模型继续细化，并应用TBPM-RBAC模型进行实际系统的开发和应用，在实践中进一步验证模型的有效性。

参考文献

1 Ferraiolo D F, et al. Proposed NIST Standard for Role-Based Access Control. ACM Transactions on Information and System Security, 2001, 4(3): 224～274

2 Sandhu R S, Samarati P. Access Control: Principles and Practice. IEEE Communications, 1994, 32(9): 40～48

3 Sandhu R S, Coyne E J, Feinstein H L, Youman C E. Role-Based Access Control Models. IEEE Computer, 1996, 29(2): 38～47

4 Thomas R K, Sandhu R S. Task-based Authorization Controls (TBAC): A Family of Models for Active and Enterprise-oriented Authorization Management. In: Proc. of the IFIP WG11.3 Workshop on Database Security. Lake Tahoe, California: Chapman & Hall, 1998. 166～181

5 Thomas R K, Sandhu R S. Conceptual Foundations for a Model of Task-based Authorizations. In: Proc. of the 7th IEEE Computer Security Foundations Workshop. Franconia, NH: IEEE Computer Society Press, 1994. 66～79

6 Coulouris G, Dollimore J, Roberts M. Role and Task-Based Access Control in the PerDiS Groupware Platform. In: Proc. of the Third ACM Workshop on Role-Based Access Control. New York, USA: ACM Press, 1998. 115～121

7 Jonscher D. Extending Access Control with Duties-Realized by Active Mechanisms. Database Security, VI: Status and Prospects. Amsterdam, The Netherlands: North-Holland, 1993. 91～111