

文章编号:1001-9081(2010)04-1045-03

## 基于扩展角色访问控制的普适计算访问控制模型

孙凌<sup>1</sup>, 辛艳<sup>2</sup>, 罗长远<sup>2</sup>

(1. 河南商业高等专科学校 计算机应用系, 郑州 450044; 2. 信息工程大学 电子技术学院, 郑州 450004)

(luocyzz@126.com)

**摘要:**针对普适计算访问控制对客体部分动态管理的需要,分析了现有扩展基于角色的访问控制(RBAC)的不足,提出一种新的扩展RBAC模型。模型引入客体与客体的关联,使得权限既可以通过角色也可以通过客体获得,并采用描述逻辑对模型访问控制过程进行了形式化描述。该模型能够实现细粒度的动态授权,解决了因决策的固有性导致角色数量过多、授权不灵活的问题。

**关键词:**普适计算;访问控制;基于角色的访问控制;动态描述逻辑

**中图分类号:** TP309 **文献标志码:** A

## Pervasive computing access control model based on extended RBAC

SUN Ling<sup>1</sup>, XIN Yan<sup>2</sup>, LUO Chang-yuan<sup>2</sup>

(1. Department of Computer Applications, Henan Business College, Zhengzhou Henan 450044, China;

2. Institute of Electronic Technology, Information Engineering University, Zhengzhou Henan 450004, China)

**Abstract:** Concerning the needs of dynamic management for object in pervasive computing access control and the shortages of the existing Role Based Access Control (RBAC), the paper presented an extended RBAC model. In the model an associated object was presented, so the permissions can be obtained through roles and objects. The access control processes were described with the description logic. The model considers the authorization from the point of view of object, and resolves the problems that the roles are too many and the authorization is not flexible caused by the inherence of decision.

**Key words:** pervasive computing; access control; Role Based Access Control (RBAC); dynamic description logic

### 0 引言

普适计算是一种新的计算模式,它使计算机融入人的生活空间,形成一个“无时不在、无处不在而又不可见”的计算环境。普适环境具有动态性、不确定性、上下文感知等特点,因此它的安全需求显得尤为重要<sup>[1-2]</sup>。访问控制正是保证其环境中设备和服务安全性的有效方法<sup>[3]</sup>。

基于角色的访问控制(Role Based Access Control, RBAC)系列模型相对而言比较适合普适计算的访问控制<sup>[4]</sup>。它引入了角色的概念,使得授权给用户的访问权限,由用户在一个组织中担当的角色来确定,从而减少了授权管理的复杂性,降低了管理开销。但是在普适计算访问控制实际运用中, RBAC在对客体的管理方面还不能完全适应。

1) 在RBAC中,决策是在请求时刻制定的,一旦制定在整个访问过程中不再改变和撤销。而在普适环境中存在很多长期访问过程和短暂访问过程,用户临时需要操作某一(或多个)客体,使用一段时间后释放,并且有可能还会使用,这就需要考虑决策的灵活性。

2) RBAC模型是从控制主体(用户)的角度出发的访问控制系统,其大多数结构都是针对访问控制主体,对访问控制客体涉及较少。

3) 有时会需要某些特定的授权机制,例如,如果要赋予用户操作某些客体的一些权限,但又不想只为这一个用户去创建一个新的角色。而根据RBAC模型,必须创建一个新的角色,使这个角色拥有给用户的权限,然后再将这个角色分配

给用户。这样,随着应用系统规模的逐渐扩大,就会造成角色数量过多,授权管理负担过重的问题。

目前许多研究者都从不同角度对RBAC模型进行了扩展,但是对授权管理负担过重的问题却较少考虑。如文献[5]提出一种基于任务和角色访问控制模型,但是仅仅从执行任务的角度考虑授权。文献[6]对权限进行了扩展,将权限分为操作和操作所针对的数据对象,提出了操作继承和数据对象继承概念,该方法在权限设置和管理过程中大大减少了系统安全管理人员的工作量;但是一旦角色分配给用户后,用户便得到该角色拥有的访问权限,可以对相应的客体进行操作,而对于授权后的客体使用情况却没有考虑。文献[7]在RBAC模型的基础上增加了用户—权限分配,采用双授权方式,对于临时需要拥有某些权限而现有角色均不合适的用户,无需增加新的角色,只需直接给该用户赋予所需权限,当不需要时再撤销该用户的此项授权即可;但是此方法没有完全实现用户和权限的逻辑分离,对系统的安全性不利,同时也使授权工作变得复杂。

因此针对普适计算访问控制的实际需求,本文打破RBAC中必须通过角色获得权限的规定,提出扩展的RBAC模型。模型中引入客体与客体的关联,使得权限而是既可以通过角色又可以通过客体获得,授权更加灵活。

### 1 基于扩展RBAC的普适计算访问控制模型

RBAC的访问控制过程分为权限与角色的关联以及角色与用户的关联。本文的模型在此基础上增加了客体与客体的

收稿日期:2009-10-09;修回日期:2009-12-23。

**作者简介:** 孙凌(1976-),女,河南郑州人,讲师,硕士,主要研究方向:网络安全; 辛艳(1983-),女,北京人,助理工程师,硕士研究生,主要研究方向:普适计算; 罗长远(1973-),男,河南信阳人,副教授,博士,主要研究方向:无线通信、网络安全。

关联。模型引入了主客体和相关客体的概念,其中主客体指的是比较稳定被分配给固定角色的客体,关联客体指的是不被长期使用的与主客体相关联的客体。用户可以通过主客体得到关联客体的访问权而不用创建新的角色,也不用改变用户角色而收回使用权。关联客体的授权更加简单灵活,并且只有当用户有需求时才分配,避免了用户只要拥有权限就可以长期占用并随时访问客体而导致安全隐患的缺点,当需求满足时则立即收回权限,符合“最小权限”的安全原则。

假设听众的操作对象是计算机,发言所必须的操作对象是投影仪。可以定义每一个操作计算机的用户作为推荐者,并允许其使用投影仪,这样不用创建发言者这个角色投影仪就可以动态地分配给用户,同样也可以在不改变用户角色的情况下收回。

### 1.1 模型的结构

模型可定义为一个五元组: {用户集, 角色集, 会话集, 权限集, 上下文信息}, 其中权限集为操作集与客体集的多对多映射, 客体集又分为主客体集和关联客体集。如图 1 所示。

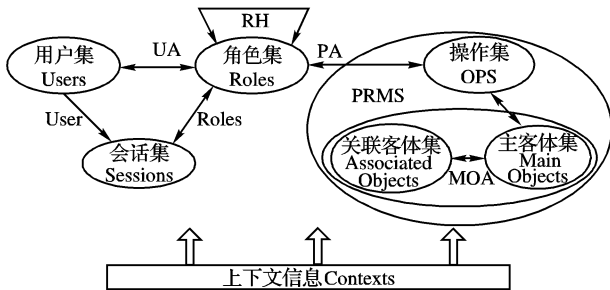


图 1 扩展 RBAC 模型

用户集  $U$  (Users): 访问系统中资源的实体集,  $Users = \{u_1, u_2, \dots, u_n\}$ 。

角色集  $R$  (Roles): 用户在组织内可执行的操作的集合,  $Roles = \{r_1, r_2, \dots, r_m\}$ 。

会话集  $S$  (Sessions): 一个用户对多个角色的映射, 即一个用户激活多个角色子集,  $Sessions = \{s_1, s_2, \dots, s_q\}$ 。

权限集  $PRMS$  (Permissions): 对一个或多个客体进行特定访问的操作,  $Permissions = \{p_1, p_2, \dots, p_d\}$ 。

主客体集  $MO$  (MainObjects): 比较稳定被分配给固定角色的客体集,  $MainObjects = \{mo_1, mo_2, \dots, mo_s\}$ 。

关联客体集  $AO$  (AssociatedObjects): 与主客体相关联的客体集,  $AssociatedObjects = \{ao_1, ao_2, \dots, ao_l\}$ 。

上下文信息  $C$  (Contexts): 时间、地点等环境信息以及用户的特性等, 是分配权限的重要依据。

模型中的对应关系:

$UA \subseteq Users \times Roles$ , 用户指派, 用户集到角色集的多对多映射, 表示用户被赋予某角色。

$PA \subseteq Roles \times Permissions$ , 权限指派, 角色集到权限集的多对多映射, 表示角色被赋予权限。

$MOA \subseteq MainObjects \times AssociatedObjects$  主客体集到关联客体集的多对多映射。

### 1.2 模型的访问控制过程

#### 1.2.1 单个用户对不同客体的访问控制过程

Alice 进入会议室, 对其认证后分配听众的角色, 她可以通过这个角色对计算机进行操作。当她需要发言时提出申请, 因为她可以使用计算机, 根据规则可以获得投影仪的使用权, 再根据上下文信息 (投影仪是否为空) 判断能否使用 (如图 2 所示)。发言完毕再将投影释放。当有新的用户 Bob 加

入时, 系统的安全策略不知道他的身份, 无法分配角色给他, 因此拒绝他的访问。但是如果他跟 Alice 熟识 (Alice 信任 Bob, 根据定义 Alice 可以作为推荐者), 就可以经 Alice 推荐以听众的身份加入会议。

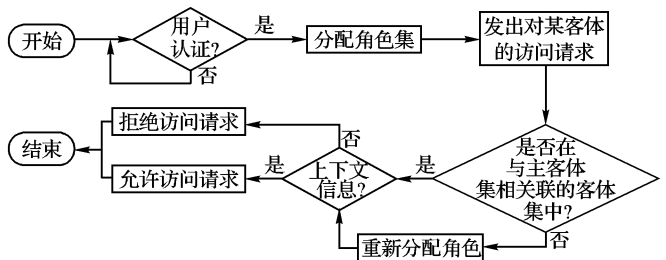


图 2 单个用户对不同客体的访问控制过程

#### 1.2.2 多个用户对同一客体的访问控制过程

以上是一个用户对客体的访问请求, 如果多个用户同时申请某一客体, 而该客体都在用户对应的关联客体集中, 则根据其角色集的重要程度 (即角色集权值) 和用户是否执行工作以及上下文信息授予权限, 即多个用户对同一客体的授权由三元组  $\{WR, T, C\}$  来决定。其中  $WR$  表示用户角色集的权值,  $T$  表示用户执行工作,  $C$  表示上下文信息。

设  $R$  为一个角色集合,  $R \subseteq Roles, R = \{r_1, r_2, \dots, r_m\}$ , 其中  $1 \leq m \leq Roles, W_1, W_2, \dots, W_m$  分别为  $m$  个角色的权限, 则用户角色集  $R$  的权值定义如下:  $WR = \sum_{j=1}^n W_j / \sum_{i=1}^m W_i, n \leq m, W_j$  为其中的  $n$  个最小权值。符合访问授权的最小权限原则, 即如果拥有权值最小的角色数量越多, 也就是说拥有权限数最少的角色数量越多, 那么就越有利于该用户对客体的访问。

$task(u, t)$  表示用户在时间  $t$  执行的工作。如果所执行的工作包括被请求客体, 则  $T$  为真。

在这里上下文信息  $C$  指的是被访问客体是否空闲。

当两个以上用户对同一客体进行访问时, 首先判断用户是否执行工作, 如果是则判断上下文信息, 最后根据用户角色集权值决定授权给哪一个用户 (如图 3 所示)。

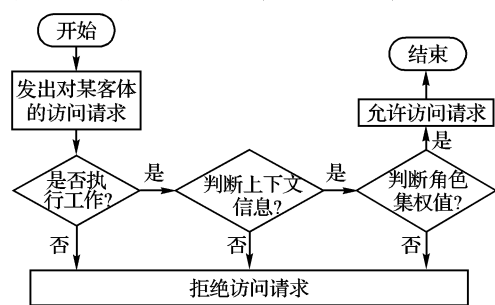


图 3 多个用户对同一客体的访问控制过程

## 2 访问控制形式化描述及可满足性证明

通过动态描述逻辑 (Dynamic Description Logic, DDL) 对模型访问控制过程进行形式化描述, 并通过 Tableau-算法给出可满足性的推理过程。

### 2.1 形式化描述

描述逻辑<sup>[5]</sup> (Description Logic, DL) 是一阶谓词逻辑的一个可判定子集, 具有较强的描述能力, 能够提供完备、高效的知识推理机制, 可以表达复杂安全策略并进行推理, 以验证其正确性。

动态描述逻辑 DDL 是对描述逻辑的推广, 它将静态和动态知识表示与推理有机地整合。它建立在概念和角色之上,

借助概念构子递归定义概念描述。概念  $C$  是表示一些个体的集合,角色  $R$  表示个体之间的二元关系,构子包括合取  $\wedge$ 、析取  $\vee$ 、否定  $\neg$ 、存在性限定  $\exists R.C$  和价值限定  $\forall R.C$  等。

DDL 的知识库是由两个部分组成的,即术语库 (TBox) 和断言库 (ABox)。TBox 存放本体中所有的公理,包括所有概念的声明和关系的定义;ABox 中包括所有实例的断言,用来描述实例的从属和实例间的关系。

DDL 的一个动作描述形如  $A(x_1, x_2, \dots, x_n) = (P_\alpha, E_\alpha)$ ,  $P_\alpha, E_\alpha$  分别表示执行该动作之前必须满足的前提条件以及执行动作后的效果,是由描述逻辑中的公式组成的集合。

本文模型的概念集为  $\{Users, Roles, Objects, Contexts, WR\}$ , 角色关系集为  $\{assignRole, authorize, operate, associate, enable, hold, execute, checkRight, hasWR\}$ , 描述如表 1 所示。

表 1 角色关系描述

名称	描述
$assignRole(u, r)$	根据规则某用户 $u$ 被分配了角色 $r$
$authorize(u, p)$	用户 $u$ 通过角色被授予权限 $p$
$canOperate(u, o)$	用户 $u$ 可以对客体 $o$ 进行操作
$associate(mo, ao)$	主客体 $mo$ 与关联客体 $ao$ 相关联
$enable(c, p)$	在上下文信息 $c$ 中, 权限 $p$ 可以被执行
$hold(u, c)$	用户 $u$ 具有上下文信息 $c$
$hasWR(r, n)$	角色 $r$ 具有权值 $n$

定义一些基本动作:

$SetAccess(u, o) = (\{User(u), Object(o)\}, \{canOperate(u, o)\})$

通过此动作可以设置用户  $u$  对客体  $o$  的访问。

$SetAssociate(mo, ao) = (\{mainobject(mo), associateobject(ao)\}, \{associate(mo, ao)\})$

通过此动作可以设置与主客体相关联的客体。

$GetWR(r, n) = (\{Roles(r)\}, \{WR(n), hasWR(r, n)\})$

通过此动作可以获得用户角色集的权值。

1) 单个用户对不同客体的访问控制执行过程描述如下,

听众 Alice 在要讲话时提出对客体 Projector 的使用请求:

a) 查看用户是否有权提出该请求:

$checkRight(Alice, pro) = (\{listener(Alice), Projector(pro)\}, \{conference(con), workIn(Alice, con), belongto(pro, con)\})$

b) 找出用户对应的关联客体集:

$getUAO(Alice, ao) = (\{assignRole(Alice, listener), canOperate(Alice, computer), associate(computer, ao)\}, \{hasAO(Alice, ao)\})$

c) 该客体是否在关联客体集中:

$\exists canOperate. (\exists associate. ao) \subseteq \exists authorize. ao$

d) 查看上下文信息决定是否接受请求:

$execute(Alice, pro) = \exists enable. p \wedge \exists associate. pro \wedge \forall checkRight. o$

2) 多个用户对同一客体的访问控制执行过程描述如下,

听众 Alice 和 Bob 同时对客体 Projector 提出使用请求:

a) 分别查看用户是否有权提出该请求;

b) 判断用户是否执行工作;

c) 查看上下文信息;

d) 根据用户角色集权值决定授权给哪一个用户:

$PermitAccess(u, ao) = \forall checkRight. o \wedge \exists task. t \wedge \exists enable. p \wedge WR(n)$

## 2.2 可满足性证明

Tableau-算法是通过构造概念表达式的模型来判定其可满足性。下面我们用它来证明概念表达式  $execute(u, o) = \exists enable. p \wedge \exists associate. ao \wedge \forall checkRight. o$  的可满足性。

算法构造的概念表达式的模型为一棵完整树,完整树的节点标注是一组概念集合  $L(x)$ , 边的标注是关系  $L(\langle x, y \rangle) = R$ 。只有当对形如  $\exists R.C$  的概念进行扩展时,才会在树上增加一条边,同时增加一个新节点。其余扩展都是在节点  $z$  的标注集合中增加新的概念。如果节点  $x$  和  $y$  由一条边  $\langle x, y \rangle$  连接,即  $L(\langle x, y \rangle) = R$ , 那么称  $y$  为  $x$  的  $R$ -后继。如果节点  $x$  中既包括概念  $D$  又包括该概念的否定式  $\neg D$ , 即有  $\{D, \neg D\} \subseteq L(x)$ , 则称  $L(x)$  中包含一个冲突。

算法的初始化树  $T$ , 仅由单节点  $x_0$  组成, 称  $x_0$  为树根。同时赋值  $L(x_0) = \{D\}$ ,  $D$  就是待证明的概念。通过不断应用算法规则(表 2), 对树  $T$  进行扩展。如果在完整树  $T$  上存在某个节点  $x$ ,  $L(x)$  中包含一个冲突, 或者对树  $T$  已没法实施任何算法规则了, 那么称树  $T$  是完全的, 即概念  $D$  是可满足的。

Tableau-算法规则如下:

$\wedge$ -规则 如果  $(1) C_1 \wedge C_2 \in L(x)$ ,  $(2) \{C_1, C_2\} \not\subseteq L(x)$ ; 那么  $L(x) \rightarrow L(x) \cup \{C_1, C_2\}$ 。

$\vee$ -规则 如果  $(1) C_1 \vee C_2 \in L(x)$ ,  $(2) \{C_1, C_2\} \cap L(x) = \emptyset$ ; 那么  $L(x) \rightarrow L(x) \cup \{C\}$ , 对于某个  $C \in \{C_1, C_2\}$ 。

$\exists$ -规则 如果  $(1) \exists S.C \in L(x)$ ,  $(2) x$  没有一个  $S$ -后续  $y$ , 使得  $C \in L(y)$ ; 那么新增一个节点  $y$ , 赋值  $L(\langle x, y \rangle) = S$  且  $L(y) = \{C\}$ 。

$\forall$ -规则 如果  $(1) \forall S.C \in L(x)$ ,  $(2) x$  有一个  $S$ -后续  $y$ , 但  $C \notin L(y)$ , 那么  $L(y) \rightarrow L(y) \cup \{C\}$ 。

对于单个用户对不同客体的访问控制, 待证可满足性的概念  $execute(u, o) = \exists enable. p \wedge \exists associate. ao \wedge \forall checkRight. o$

初始化完整树  $T$ , 树根  $x$  的节点标注为  $L(x) = \{execute(u, o)\}$ 。然后运用  $\wedge$ -规则在节点  $x$  的标注中新增概念  $L(x) = \{execute(u, o), \exists enable. p, \exists associate. ao, \forall checkRight. o\}$ 。再由  $\exists$ -规则分别得到  $x$  的两个  $R$ -后续  $y$  和  $z$ , 且有  $L(y) = \{p\}$  和  $L(z) = \{ao\}$ 。这时对  $x$  运用  $\forall$ -规则将扩充  $y$  和  $z$  的节点标注, 分别为  $L(y) = \{p, o\}$  和  $L(z) = \{ao, o\}$ 。由于已没有算法规则可应用, 得到一棵完全完整树(如图 4 所示), 而  $x, y, z$  三个节点标注中都没有包含冲突, 所以概念  $execute(u, o)$  是可满足的。

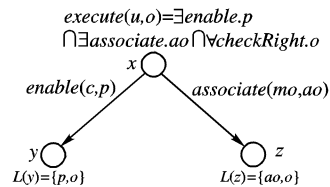


图 4 Tableau-算法举例

同样也可以用上述方法对表达式  $PermitAccess(u, ao)$  进行可满足性证明, 应用规则对其进行扩展, 得到一棵无冲突的完全完整树, 说明表达式是可满足的。

## 3 结语

针对传统 RBAC 模型不能从客体角度考虑授权、决策的固有性导致角色数量过多授权不灵活的问题, 本文对传统 RBAC 模型进行扩展, 引入主客体、关联客体的概念, 通过主

(下转第 1052 页)

一定差异(系统中当前运行的进程不同),受感染的进程也就不一样。

同时,BehaviorDetection 还识别出了 virus.win32.sality.aa 病毒在利用网络共享传播过程中,API 调用序列中反映出来的建立 IPCMYM 连接及获取远程主机共享资源的行为,而这些都是 Anbits 及 ThreatExpert 的代码分报告中没有的。

## 5.2 行为分析及危害评估的有效性测试

以常见的几款木马后门(Pcshare, Bitfrose, Hupigon),以及传播比较广泛的病毒 Sality, Sysanti 作为测试对象,将行为分析的结果同专家对该病毒的描述(参考了 Symantec 及 Kaspersky 的病毒描述)进行比较,其结果如表 2 所示。其中“√”表示识别出该攻击目标,“—”表示该攻击目标存在但未被识别。表中最后一行是各个程序的危害指数。测试中,对  $p_I$ 、 $p_D$ 、 $p_C$  的取值分别为 0.4、0.4、0.2。

表 2 行为分析及危害评估结果

行为	Pcshare	Bitfrose	Hupigon	Sality	Sysanti
感染本地文件		√	√	√	√
U 盘感染				√	√
局域网共享传播				√	
键盘记录	√	√			
修改系统配置文件				√	
修改安全设置				√	
修改 Internet 相关设置					√
删除/修改文件	√	√	√		√
禁用管理工具				√	
下载执行程序				—	
自启动	√	√	√		√
抵制杀毒软件				√	√
危害指数 $\bar{R}$	0.42	0.54	0.38	0.676	0.562

通过表 2 可以看出,实验得到的结果同专家对恶意程序的描述基本吻合的。需要说明的是,在行为检测过程中,检测器已准确识别出 Sality 具有网络行为,但由于目前的行为检测器并未对网络数据进行分析,导致了分析结果出现了疏漏。在后期的工作中,将进一步对网络数据流进行分析。

表 2 中最后一行的程序危害指数是利用 4.4 节中的式子计算出来的。根据表 2 可以看出,恶意行为较多的测试程序其危害指数较高。这正反映出程序的恶意行为越多,其对系统可能造成的破坏就越大。而几款木马相对其他恶意程序的危害指数偏低,这是由于木马或后门类程序,为了隐藏自身,相比病毒或蠕虫的恶意行为要少很多。它们通常只是写启动

项确保自身的运行,而很多正常的程序也会写注册表启动项或注册服务。在评估时对自身启动行为设置的  $r$  值并不大,这也就导致它们的危害指数不高。

## 6 结语

本文提出了一种针对恶意代码行为的层次化分析方法,首先获取程序动态执行时所产生的 API 调用序列中的行为信息,然后在此基础上分析行为的目的,评估程序的危害性。实验表明,该方法能够有效识别和分析程序运行时的各种行为信息。同时,考虑到程序行为对系统造成的危害程度可能因操作对象而异,采取不同的操作对象设定不同  $r$  值的方法,这样大大提高了对程序行为危害评估的准确性。下一步的工作将扩大程序运行信息的收集范围,不再局限于 native API 函数。同时,加强对程序数据流信息及网络数据的分析,进一步完善攻击树模型,丰富规则知识库中的信息。

### 参考文献:

- [1] 卡巴斯基安全公告:2008 相关统计数据 [EB/OL]. [2009-04-20]. <http://www.kaspersky.com.cn/KL-AboutUs/news2009/04n/090420a.htm>.
- [2] WILLEMS C, HOLZ T, FREILING F. Toward automated dynamic malware analysis using CWSandbox[J]. IEEE Security and Privacy, 2007, 5(2): 32-39.
- [3] BAYER U, KRUEGEL C, KIRDA E. TTAlyze: A tool for analyzing malware[C]// EICAR: Proceedings of the 15th Annual Conference of the European Institute for Computer Antivirus Research. Hamburg: [s. n.], 2006.
- [4] 张冲,吴灏.基于攻击树的脚本病毒样本分析方法[J]. 计算机应用研究, 2005, 22(6): 118-120.
- [5] 杨彦,黄浩.基于攻击树的木马监测方法[J]. 计算机工程与设计, 2008, 29(11): 2711-2714.
- [6] 许敏,赵天福.基于行为特征的恶意代码检测方法[J]. 网络与信息, 2009(6): 14-16.
- [7] CHRISTODORESCU M, JHA S, KRUEGEL C. Mining specifications of malicious behavior[C]// Proceedings of the the 6th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering. New York: ACM, 2008: 5-14.
- [8] SCHNEIER B. Attack trees: Modeling security threats [J]. Dr Dobbs's Journal, 1999, 24(12): 21-29.
- [9] 张健,梁宏,陈建民,等.计算机病毒危害评估[J]. 信息网络安全, 2005(1): 39-41.
- [10] Anubis[EB/OL]. [2009-06-01]. <http://anubis.isecclab.org/>.
- [11] ThreatExpert[EB/OL]. [2009-06-01]. <http://www.threatexpert.com/>.

(上接第 1047 页)

客体实现与固有角色的匹配,关联客体实现与短暂使用客体的匹配,用户只有在有需求时才分配给权限,具有灵活性、自适应性和安全性。

### 参考文献:

- [1] 徐光祐,史元春,谢伟凯.普适计算[J]. 计算机学报, 2003, 26(9): 1042-1050.
- [2] 郭亚军,洪帆,沈海波,等.普适计算面临的安全挑战[J]. 计算机科学, 2007, 34(6): 1-3.
- [3] ALMENAREZ F, MARIN A, CAMPO C, et al. TrustAC: trust-based access control for pervasive devices[C]// Proceedings of the Second International Conference on Security in Pervasive Computing. Brighton: Springer-Verlag, 2005: 225-238.

- [4] SANDHU R S, COYNEK E J, FEINSTEIN H L, et al. Role-based access control models[J]. IEEE Computer, 1996, 29(2): 38-47.
- [5] 龙涛,洪帆,刘铭.一种基于任务和角色和计算网络访问控制模型[J]. 计算机工程, 2008, 34(4): 176-178.
- [6] 叶春晓,符云清,吴中福. RBAC 中权限扩展的实现[J]. 计算机工程, 2005, 31(9): 141-142.
- [7] 王悦,高虎明.扩展式基于角色的访问控制模型的研究[J]. 计算机工程与设计, 2008, 29(2): 309-311.
- [8] BAADER F, CALVANESE D, McGUINNESS D L, et al. The description logic handbook: Theory, implementation and applications [M]. Cambridge: Cambridge University Press, 2003.