计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2014.20130973
51(11):2470-2482,2014

面向服务访问控制策略精化描述

吴迎红　黄　皓　曾庆凯
(计算机软件新技术国家重点实验室(南京大学)　南京　210046)
(wuyh@nju.edu.cn)

Description of Service Oriented Access Control Policy Refinement

Wu Yinghong, Huang Hao, and Zeng Qingkai
(*State Key Laboratory for Novel Software Technology* (*Nanjing University*), *Nanjing* 210046)

Abstract　Policy refinement is an important method to resolve the configuration complexity of access control policies for distributed applications. Although the current policy refinement techniques make it possible to describe the layered policies and refine the policies layer by layer, it is not easy of these methods to describe and analyze the associated attributes among different policies. The wide use of policy refinement is thus hindered. In this paper, new methods for the description of policies and relationships among them such as composition, mutual exclusion, refinement and path cooperation are given. A new algorithm for policies refinement with relationship description ability is proposed. A refine-tree construction method with the capability of describing the policies and the relationships among these policies is also proposed with the algorithm. This provides a basis for solving the issue of the associating attributes between policies in the policy refinement process. The policies refine-tree can also be used to demonstrate the SLA (service-level agreement) of access control.

Key words　model driven architecture; access control; policy description; policy refinement; policy conflict analysis; associated attribute

摘　要　策略精化是解决分布式应用访问控制策略配置复杂性的重要方法.现有的策略精化技术给出了分层策略描述和逐层精化的方法,但是描述和处理策略之间关联问题能力不足,影响策略精化应用.为此给出了策略和包括组合、互斥、精化、访问路径协同等策略之间关系的形式描述方法,提出了能够描述策略之间关联属性的精化算法和记录策略和策略之间这些关联属性的策略精化树构建方法,为策略精化中的策略关联问题处理提供基础.策略精化树还能直观呈现访问控制的服务品质协议(service-level agreement, SLA).

关键词　模型驱动架构;访问控制;策略描述;策略精化;策略冲突分析;关联属性

中图法分类号　TP309.2

面向服务架构(service-oriented architecture, SOA)和服务云(software as a service, SaaS)是分布式技术新发展方向,基于模型逐层细化的模型驱动架构(model driven architecture, MDA)方法[1]是其服务构建的重要方法.

策略是面向服务应用安全管理的关键[2].系统以不同抽象层次分层描述,以系统各层对象为策略描述对象,通过策略映射规则,将上层抽象的访问控制策略转化为下层具体的访问控制策略的过程称为策略精化[3-5].

收稿日期:2013-07-04;**修回日期**:2014-01-20
基金项目:国家"八六三"高技术研究发展计划基金项目(2011AA01A202)
通信作者:黄　皓(hhuang@nju.edu.cn)

策略精化需要满足:1)完全性,即每一个上层策略都有一个或一组下层策略支撑;2)一致性,即同层策略之间、下层策略与上层策略没有冲突.面向服务计算需要服务品质协议(service-level agreement, SLA)的呈现[6],访问控制抽象策略到配置策略之间的精化关系是访问控制的 SLA 呈现.

访问控制策略精化技术主要有基于策略模板的精化、基于角色访问控制(role-based access control, RBAC)对象结构关系的精化、基于系统和策略的分层模型及层间映射规则的精化.

推导并协同下层访问路径控制单元中一系列相关策略,控制访问过程中的不同侧面,实现上层访问控制策略纵深防御是衡量精化能力的重要方面.基于模板[7]与基于 RBAC 对象结构[8]的精化技术不具备描述和推导下层系统访问路径一系列相关策略的能力.网络及其安全监控技术由于多样性、变化性、复杂性,所以难于静态模式化描述,因此也难于发展这两种方法实现访问控制纵深防御.

Abadi 等人[9-10]提出了由抽象访问控制策略精化推导实际系统访问路径一系列相关策略,实现访问控制.Davy 等人[11]、Lück 和 Albuquerque 等人[12-13]在文献[9-10]研究基础上完善系统和访问控制策略分层描述,提出了基于系统和策略的分层模型及层间映射规则精化策略,设计冲突分析和消解方法以保证精化一致性和完全性.但是,由于策略冲突分析技术中策略关联属性描述、分析和修正能力不足,不能描述和处理策略之间组合和互斥关系,不能处理计算相关层访问路径相关策略和平台相关层多条访问路径相关策略的协同性,不能处理像 IPSec 协议实现的虚拟专用网络(Internet protocol security/virtual private network, IPSec/VPN)策略冲突,这类策略需要有序分析不同类型策略冲突[14],而且 Lück 和 Albuquerque 等人设计方法[12-13]的冲突消解为保证精化完全性和一致性,只能由上层策略取代下层策略,这在 IDS 等根据具体事件和状态实时决定应用策略的情况下是不适用的.

本文试图设计一种策略和策略关联属性描述方式,在消解冲突策略时,基于策略关联属性描述能够进一步分析修正策略关联属性,从而解决上述因策略关联属性分析修正能力不足造成的精化能力不足.策略描述方式从根本上影响策略冲突计算能力.从策略描述方式角度,策略冲突分析技术一般分为逻辑程序和图形规则两类.

逻辑程序策略描述和分析方法包括非经典逻辑和经典一阶逻辑.非经典逻辑策略冲突分析技术的主要理论基础有可废止逻辑[15]、博弈论[16]、多值逻辑[17]等.一阶逻辑策略冲突分析技术主要理论有分层模型[18]、稳定模型和良基模型[19].

精化计算中策略组合、互斥、精化、协同等关联属性,虽然可以用逻辑的“与”、“或”关系表示,但是逻辑程序的可计算性使得这些基于逻辑程序的策略冲突分析技术难于扩展处理这些包含“与”、“或”关系的策略关联属性;并且由于 Horn 子句的限制,策略之间关系信息在逻辑程序计算中丢失,不能由策略之间关系直接呈现访问控制 SLA.

一些学者采用图形规则描述和分析策略冲突.王雅哲等人[20]、Basile 等人[21]、姚键等人[22]设计的策略冲突分析技术,冲突计算基于主体之间和客体之间稳定的结构关系.分布式应用策略主客体并不一定具有稳定的、统一的结构关系,所以这些技术不适于分布式应用;另外这些技术也缺少策略之间关联属性描述和分析能力.

Jaeger 等人[23]设计的访问控制权限空间冲突分析技术,逐个计算主体或客体的所有权限,判断是否存在冲突,并能描述和分析策略互斥关系,可以用于分布式应用精化的冲突计算,但是缺少精化分析需要处理的其他策略关联属性描述和分析能力.

由于基于图形规则描述“与”、“或”逻辑关系是可行的,通常也是可计算的,并且图形规则描述、计算策略冲突与策略关联属性也有一定的基础,所以采用图形规则描述策略与策略关联属性,设计基于策略关联描述修正策略关联属性的策略冲突分析技术,解决因策略关联属性描述和分析能力不足导致的精化应用限制,是提高精化能力的可行路径.

本文试图基于图形方法,设计符合分布式策略对象特点、满足精化计算涉及策略关联属性分析需求的策略描述方式.

为此,本文形式描述精化计算的策略和策略之间的组合、互斥、精化、路径协同等关联属性,设计包含策略上述关联属性的精化算法,设计策略精化树及其构建方法,描述策略以及策略之间组合、互斥、精化、访问路径协同等关联属性,为精化计算中描述、分析和修正策略关联属性提供基础,策略精化树还可以直观呈现访问控制的 SLA.

1 策略精化形式规约及精化树描述

本文描述策略及策略关联属性是基于 OMG 组织

公布的 MDA 的 3 层系统模型及对象:1)计算无关模型(computational independent model, CIM)层;2)平台无关模型(platform independent model, PIM)层;3)平台相关模型(platform specific model, PSM)层.

精化计算原理基于文献[12-13].

1.1 原子策略与原子策略集合

约定 1. S 是主体的集合,O 是客体的集合,A 是权限(或约束)的集合.

根据 MDA 分层和策略精化研究,CIM,PIM,PSM 层策略主体、客体和权限与约束定义如下:

1) CIM 层

$S=\{s_1,\cdots,s_i,\cdots\}$,其中 s_i 为个体用户或任务.

$O=\{o_1,\cdots,o_i,\cdots\}$,其中 o_i 为资源个体实体(如文件、程序等).

$A1=\{$读,写,添加,执行,创建,通过认证,$\cdots\}$.

2) PIM 层

$S=\{s_1,\cdots,s_i,\cdots\}$,$s_i=$(个体用户或功能模块,访问类型等属性),$s_i$ 是以上层个体用户或服务具体化为特定访问方式的用户或功能模块,比如(用户 A,Web),(用户 B,FTP).

$O=\{o_1,\cdots,o_i,\cdots\}$,其中 o_i 为资源个体实体(如文件、程序等).

$A2=A1\cup\{$记录,备份,告警,$\cdots\}$,$A2$ 增加实现目标访问控制计算相关的主体到客体访问方式的认证、审计权限.

3) PSM 层

$S=\{s_1,\cdots,s_i,\cdots\}$,$s_i=$(个体用户或功能模块,访问类型等属性,IP 地址,端口),描述具体的行为主体及其平台相关的属性.

$O=\{o_1,\cdots,o_i,\cdots\}$,$o_i=$(文件或程序,IP 地址(有些程序还需要端口号,协议号)),描述程序、文件等客体及其平台相关的属性.

$A3=A2\cup\{$入,出,*IPSec AH*,*IPSec ESP*,$\cdots\}$,$A3$ 增加实现目标访问控制的主体到客体访问路径的通信以及通信机密性、完整性权限.

设 $s\in S,o\in O,a\in A3$,谓词 $p(s,o,a)$ 表示授予主体 s 对客体 o 的访问权限 a,谓词 $p(s,o,-a)$ 表示禁止主体 s 对客体 o 访问权限 a,$p(s,o,a)$ 和 $p(s,o,-a)$ 都称为原子策略.

在实际应用中,策略往往不是以一个主体对一个客体权限形式描述,而是描述一组主体对一组客体权限,我们用 $p(S,O,a)$ 来表示,称为原子策略集合,其中 $S=\bigcup_{i=1}^{n}\{s_i\}$,$O=\bigcup_{i=1}^{n}\{o_i\}$,$SO=\{(s_i,o_i)\mid i\in N\}$,则 $p(S,O,a)=\{p(s_i,o_i,a)\mid(s_i,o_i)\in SO\}$,$SO$ 是 $S\times O$ 的子集.

例 1. 假设一个常见企业信息服务系统中,CIM 层包括内部信息服务、邮件服务、销售服务、售后服务和财务服务等. CIM 层策略"销售和售后人员允许读技术资料". $S=\{u_1,u_2\}$,$S=$销售和售后人员,$u_1=$王开,$u_2=$周力,$O=\{z_1,z_2\}$,$O=$技术资料,$z_1=$资料 A,$z_2=$资料 B,$SO=\{(u_1,z_1),(u_1,z_2),(u_2,z_1),(u_2,z_2)\}$,$a=r$($r$ 表示读,如果访问路径依次有:访问控制单元 1,访问控制单元 2,$\cdots$,访问控制单元 n;各个控制单元读权限依次表示为 $r_1,r_2,\cdots,r_n$;写等其他权限权限也同样表示.);该策略原子策略集合形式为 $p(S,O,a)=\{p(s_i,o_i,r)\mid(s_i,o_i)\in SO\}$,以精化树的一个节点记录策略 $p(S,O,a)$.

1.2 组合策略

下面分别介绍两种类型的组合策略:

1) 假设有 m 个权限(o_i,a_i),$i=1,2,\cdots,m$,有 n 个主体 s_j,$j=1,2,\cdots,n$. 我们有一个 $n\times m$ 策略矩阵:

$$\begin{vmatrix} p(s_1,o_1,a_1) & \cdots & p(s_1,o_m,a_m) \\ \vdots & & \vdots \\ p(s_n,o_1,a_1) & \cdots & p(s_n,o_m,a_m) \end{vmatrix}.$$

对 n 个主体 s_j,$j=1,2,\cdots,n$ 而言,每个主体 s_j 在完成一项任务时,m 个策略 $p(s_j,o_i,a_i)$,$i=1,2,\cdots,m$,都是必须的,要么全部赋予,要么全部撤销,所以策略表示为$\{\bigwedge_{i=1}^{m}p_{ij}\mid j=1,2,\cdots,n\}$,其中 $p_{ij}=p(s_j,o_i,a_i)$,$i=1,2,\cdots,m$. 这个策略称为组合策略,其原子策略全体构成整个策略矩阵的元素的全体.

另一方面,从进一步精化计算的相关性出发,我们需将关于一个对象 o_i 的一组策略 $p_i=\{p(s_j,o_i,a_i)\mid j=1,2,\cdots,n\}$作为一个策略集合处理,其中 $i=1,2,\cdots,m$. 策略 p_i 的原子策略全体构成策略矩阵的第 i 列. m 个策略 p_i 的原子策略全体就是组合策略$\{\bigwedge_{i=1}^{m}p_{ij}\mid j=1,2,\cdots,n\}$的原子策略的全体. 因此,我们将该组合策略表示成 $\wedge\{p_1,\cdots,p_m\}$.

在精化树中,我们将组合策略$\{\bigwedge_{i=1}^{m}p_{ij}\mid j=1,2,\cdots,n\}=\wedge\{p_1,\cdots,p_m\}$表示为父节点,而 m 个策略 $p_i=\{p(s_j,o_i,a_i)\mid j=1,2,\cdots,n\}$,$i=1,2,\cdots,m$ 作为子节点. 这种类型的父节点类型(*Node_type*)为"*and*",原因是其他的各个子节点中的主体同为 s_j 的原子策略必须同时授权或都不授权,即原子策略

关系为 $p(s_j,o_1,a_1)\wedge\cdots\wedge p(s_j,o_m,a_m)$. 如果子节点 p_i 中某个原子策 $p(s_j,o_i,a_i)$不能授权,父节点策略$\{\overset{m}{\underset{i=1}{\wedge}}p_{ij}\mid j=1,2,\cdots,n\}$中相应的策略元素 $\overset{m}{\underset{i=1}{\wedge}}p_{ij}$ 就要撤销.

组合策略精化树表示如图 1 所示:

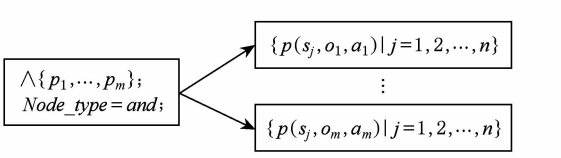

Fig. 1 The refinement tree representation of the policy $\wedge\{p_1,\cdots,p_m\}$.

图 1 策略$\wedge\{p_1,\cdots,p_m\}$的精化树表示

2) 有 m 个策略 $p_1=p(s_1,o_1,a_1),\cdots,p_m=p(s_m,o_m,a_m)$,它们必须同时授权或同时都不授权,我们也同样用$\wedge\{p_1,\cdots,p_m\}$表示这样的策略,称为组合策略,也就是$\wedge\{p_1,\cdots,p_m\}=p(s_1,o_1,a_1)\wedge\cdots\wedge p(s_m,o_m,a_m)$. 在精化树中,$\wedge\{p_1,\cdots,p_m\}$表示为父节点,$p(s_j,o_i,a_i)$为子节点,其中 $i=1,2,\cdots,m$.

例 2. PIM 层策略"售后人员通过 Web 写维修日志,必须同时写技术统计表."为简便起见,以 W 简化表示 Web. $S=\{(s_1,W),(s_2,W)\}$,$s_1=$李军,$s_2=$高明,$O=\{z_1,z_2\}$,$z_1=$维修日志,$z_2=$技术统计表,w 表述"写",$a_1=a_2=w$. 以$\wedge\{p_1,p_2\}$表示该策略,$p_1=p((s_j,W),z_1,a_1)$,$p_2=p((s_j,W),z_2,a_2)$,$j=1,2$. 节点类型为"*and*". 分别以节点表示策略 p_1 与 p_2,作为$\wedge\{p_1,p_2\}$的子节点. 如图 2 所示:

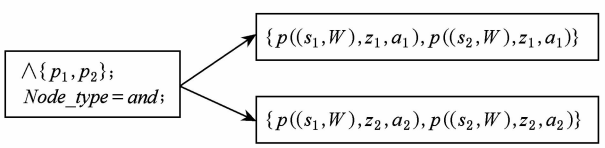

Fig. 2 The refinement tree representation of the policy $\wedge\{p_1,p_2\}$.

图 2 策略$\wedge\{p_1,p_2\}$的精化树表示

1.3 策略精化规则

1.3.1 CIM 层策略精化为 PIM 层策略

CIM 层策略描述主体对客体的访问权限,并没有确定主体用什么方式来访问客体,例如用 Web 的方式或 FTP 的方式等. 然而,访问计算方式不同涉及到的控制流程和数据对象就不同,则权限策略也会不同. 因此,当 CIM 层的访问细化为 PIM 层的一个或多个访问方式时,CIM 层的策略也必须根据其 PIM 层的一个或多个访问方式来细化.

CIM 层原子策略 $p(s_i,o_i,a)$在向 PIM 层精化过程中,主体 s_i 对 o_i 的访问根据 PIM 层计算方式不同(如 Web,FTP)细化成 PIM 层的 n 个主体$(s_i,c_1),\cdots,(s_i,c_n)$对 o_i 访问的集合,n 种计算方式中只要一种授予其访问的权限,s_i 就可以访问 o_i,所以 CIM 层策略对应 PIM 层 n 种访问方式策略的"或".

CIM 层原子策略 $p(s_i,o_i,a)$权限 a,在 PIM 层每种访问方式 c_j 下,$j\in\{1,2,\cdots,n\}$,可能涉及到多个(n_j)权限控制点,这些控制点上都授权才能实现该种方式的访问,即 CIM 层原子策略 $p(s_i,o_i,a)$在计算方式 c_j 下等价于 PIM 层的 $p((s_i,c_j),o_i,b_{j,1})\wedge\cdots\wedge p((s_i,c_j),o_i,b_{j,n_j})$,表现为"与"关系,所以 CIM 层策略 $p(s_i,o_i,a)$精化表示成:

$$p(s_i,o_i,a)\rightarrow\overset{n}{\underset{j=1}{\vee}}\overset{n_j}{\underset{k=1}{\wedge}}p((s_i,c_j),o_i,b_{j,k}).$$

对于 CIM 层的组合策略$\{\overset{m}{\underset{i=1}{\wedge}}p_{ij}\mid j=1,2,\cdots,n\}$,精化只对它的子节点进行. 子节点策略是 $p(S,O,a)=\{p(s_i,o_i,a)\mid(s_i,o_i)\in SO\}$的一种特例.

对于每一个二元组(s_i,o_i),都有 n 种计算方式 c_j,$j=1,2,\cdots,n$. 我们构造:$SC_jO=\{((s_i,c_j),o_i)\mid(s_i,o_i)\in SO\}$,$j=1,2,\cdots,n$. 策略 $p(S,O,a)$的精化表示成:$p(S,O,a)\rightarrow\{\overset{n}{\underset{j=1}{\vee}}\overset{n_j}{\underset{k=1}{\wedge}}p((s_i,c_j),o_i,b_{j,k})\mid(s_i,o_i)\in SO\}$.

算法 1 描述了 CIM 层策略精化计算.

算法 1. CIM 层策略精化为 PIM 层策略.

设 $p(S,O,a)=\{p(s_i,o_i,a)\mid(s_i,o_i)\in SO\}$,$SO=\{(s_i,o_i)\mid i=1,2,\cdots,m\}$.

1) 确定 PIM 层 s_i 访问 o_i 的 n 种计算方式:$c_1,c_2,\cdots,c_n$. 对于每个 $j\in\{1,2,\cdots,n\}$,构造 $SC_jO=\{((s_i,c_j),o_i)\mid(s_i,o_i)\in SO,j\in\{1,2,\cdots,n\}\}$.

2) 确定实现 CIM 层 s_i 访问o_i 的权限a 在 PIM 层 c_j 访问方式下对应的 n_j 个控制点的控制权限:$b_{j,1},b_{j,2},\cdots,b_{j,n_j}$,其中 $j=1,2,\cdots,n$.

3) 对于访问方式 c_j,构造策略:

$$\overset{n_j}{\underset{k=1}{\wedge}}p((s_i,c_j),o_i,b_{j,k}),j=1,2,\cdots,n.$$

4) 对于 $p(S,O,a)=\{p(s_i,o_i,a)\mid(s_i,o_i)\in SO\}$,构造策略集合:

$$\{\overset{n}{\underset{j=1}{\vee}}\overset{n_j}{\underset{k=1}{\wedge}}p((s_i,c_j),o_i,b_{j,k})\mid(s_i,o_i)\in SO\}.$$

以 n 个节点分别表示$\{\overset{n_j}{\underset{k=1}{\wedge}}p((s_i,c_j),o_i,b_{j,k})\mid(s_i,o_i)\in SO\}$,$j=1,2,\cdots,n$,作为 $p(S,O,a)$策略节点的儿子. $p(S,O,a)$节点类型为"*or*",表示父节点策略 $p(s_i,o_i,a)$等价于儿子节点策略$\overset{n_j}{\underset{k=1}{\wedge}}p((s_i,c_j),$

$o_i,b_{j,k}),j=1,2,\cdots,n$ 的“或”,即:

$$p(s_i,o_i,a)\rightarrow\bigvee_{j=1}^{n}\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,b_{j,k}),$$

对于父节点策略 $p(s_i,o_i,a)$ 来说,任意一个子节点策略 $\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,b_{j,k}),(s_i,o_i)\in SO,j\in\{1,2,\cdots,n\}$ 成立,父节点策略 $p(s_i,o_i,a)$ 就可以成立;只有当所有子节点策略 $\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,b_{j,k}),j=1,2,\cdots,n$ 均不成立时,父节点策略 $p(s_i,o_i,a)$ 才不成立.精化计算的精化树表示如图 3 所示:

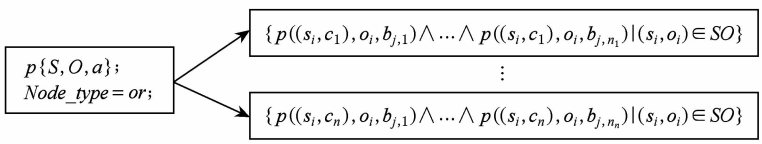

Fig. 3 The refinement tree representation of CIM layer to PIM layer policy refinement.

图 3 CIM 层到 PIM 层策略精化的精化树表示

例 3. 例 1 中的策略:$p(S,O,a)=\{p(u_1,z_1,r),p(u_1,z_2,r),p(u_2,z_1,r),p(u_2,z_2,r)\}$,在 PIM 层有 FTP 和 Web 两种计算方式,每种访问都需要经过系统 PKI 认证和客体所在的服务器的访问控制,为简便起见,F 表示 FTP,W 表示 Web,K 表示系统 PKI 认证,$auth$ 表示通过认证.由算法 1 得出:

$p(u_i,z_j,r)\rightarrow p((u_i,F),z_j,r)\wedge p((u_i,F),K,auth)\vee p((u_i,W),z_j,r)\wedge p((u_i,W),K,auth)$;

$p(S,O,a)\rightarrow\{p((u_i,F),z_j,r)\wedge p((u_i,F),K,auth)\vee p((u_i,W),z_j,r)\wedge p((u_i,W),K,auth)|(u_i,z_j)\in SO\}$.

以节点分别表示策略集合:

$\{p((u_i,F),z_j,r)\wedge p((u_i,F),K,auth)|(u_i,z_j)\in SO\}$,$\{p((u_i,W),z_j,r)\wedge p((u_i,W),K,auth)|(u_i,z_j)\in SO\}$,作为 $p(S,O,a)$ 的子节点,父节点类型为“*or*”,如 1.3.2 节图 6 中最小虚线框内部分所示.

1.3.2 PIM 层策略精化为 PSM 层策略

PIM 层策略描述主体以特定的计算方式访问客体的权限,主体是用户或功能模块与计算方式的二元组.PIM 层并没有确定主体或客体的物理位置,然而主体或客体的物理位置决定主体访问客体不同的网络路径,不同的路径上会有不同的访问控制单元,需要不同的访问控制策略控制安全访问.因此,PIM 层的策略应该针对 PSM 层不同的物理位置和网络路径进一步细化,形成 PSM 层的一个或多个网络路径相关的访问控制策略.

PIM 层的策略有两种情况:一种是从 CIM 层的策略精化来的,CIM 层策略精化成 PIM 层的策略可以表示为 $\{\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,a_{j,k})|(s_i,o_i)\in SO,j\in\{1,2,\cdots,n\}\}$;另一种是 PIM 层首次定义的策略,如果是组合策略 $\wedge\{p_1,\cdots,p_m\}$,如前所述它转化为若干子节点策略等待进一步精化,策略形式可以统一表示为 $p(S,O,a)=\{p((s_i,c_j),o_i,a)|(s_i,o_i)\in SO,j\in\{1,2,\cdots,n\}\}$,所以 PIM 层待精化策略可以表示成如下的统一形式:$\{\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,a_{j,k})|(s_i,o_i)\in SO,j\in\{1,2,\cdots,n\}\}$.

关于 PIM 层策略主体 (s_i,c_j) 对客体 o_i 访问,在 PSM 与位置相关的因素主要包括 IP 地址和端口,我们用 $ds=$(IP 地址,端口),$do=$(IP 地址,端口)分别表示主体 (s_i,c_j) 与客体 (o_i) 位置相关的 IP 地址和端口,所以在 PSM 层分别用 (s_i,c_j,ds_i) 和 (o_i,do_i) 来表示主体和客体.

PIM 层的集合 $SC_jO=\{((s_i,c_j),o_i)|(s_i,o_i)\in SO,j\in\{1,2,\cdots,n\}\}$,在 PSM 层转化成 $SC_jPO=\{((s_i,c_j,ds_i),(o_i,do_i))|(s_i,o_i)\in SO,j\in\{1,2,\cdots,n\}\}$.

如果 PIM 层的一组策略 $\{\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,a_{j,k})|(s_i,o_i)\in SO,j\in\{1,2,\cdots,n\}\}$ 在 PSM 层主体具有相同访问控制起始物理位置,并且客体也具有相同访问控制终止物理位置,那么对于给定的计算方式,这组主体和客体之间具有相同访问网络路径,并受到路径上一组相同控制点的控制.

我们根据是否具有相同访问控制起点 ds_i 和终点 do_i,将 SC_jPO 划分成 u 个子集:$SC_jPO=\bigcup_{t=1}^{u}SC_jPO_t$.PIM 层的策略相应地划分成 PSM 层几个策略子集:$\{\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,a_{j,k})|((s_i,c_j,ds_i),(o_i,do_i))\in SC_jPO_t\}$,$t=1,2,\cdots,u$.

以 $\{\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,a_{j,k})|(s_i,o_i)\in SO,j\in\{1,2,\cdots,n\}\}$ 为父节点,以 u 个节点分别表示 $\{\bigwedge_{k=1}^{n_j}p((s_i,$

$c_j),o_i,a_{j,k})|((s_i,c_j,ds_i),(o_i,do_i))\in SC_jPO_t\}$,$t=1,2,\cdots,u$,作为子节点.父节点类型为"*set*",表示父节点策略集合等于子节点策略集合的并集,即$\{\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,a_{j,k})|(s_i,o_i)\in SO,j\in\{1,2,\cdots,n\}\}\rightarrow\bigcup_{t=1}^{u}\{\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,a_{j,k})|((s_i,c_j,ds_i),(o_i,do_i))\in SC_jPO_t\}$.

父节点策略$\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,a_{j,k}),(s_i,o_i)\in SO$,$j\in\{1,2,\cdots,n\}$,存在相应子节点策略$\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,a_{j,k}),((s_i,c_j,ds_i),(o_i,do_i))\in SC_jPO_t,t\in\{1,2,\cdots,u\}$,子节点策略成立,则父节点策略成立;子节点策略失效则父节点策略失效.策略基于物理位置划分(精化)的精化树表示如图4长线虚线框所示:

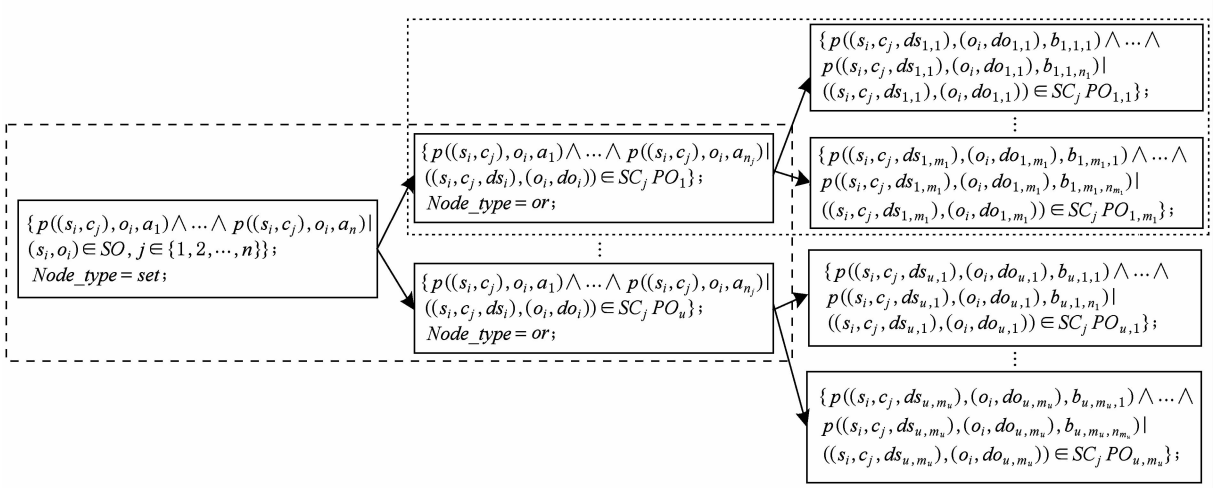

Fig. 4 The refinement tree representation of PIM layer to PSM layer policy refinement.

图4 PIM层到PSM层策略精化的精化树表示

每个SC_jPO_t中主客体具有相同网络访问路径,假设主客体之间存在m_t条网络访问路径,主体(s_i,c_j,ds_i)对(o_i,do_i)的访问根据细化成为m_t个网络路径上主客体$((s_i,c_j,ds_{t,z}),(o_i,do_{t,z}))$之间的访问,$z=1,2,\cdots,m_t$,只要其中的一条路径允许主体$(s_i,c_j,ds_{t,z})$访问客体$(o_i,do_{t,z})$,它就可以实现主体$(s_i,c_j,ds_i)$对客体$(o_i,do_i)$的访问,所以$m_t$条网络访问路径策略之间表示成"或"关系.

由SC_jPO_t构造$SC_jPO_{t,z}$:对于$z=1,\cdots,m_t$,如果$((s_i,c_j,ds_i),(o_i,do_i))\in SC_jPO_t$,那么$ds_{t,z}=ds_i,do_{t,z}=do_i,((s_i,c_j,ds_{t,z}),(o_i,do_{t,z}))\in SC_jPO_{t,z}$.

每个$SC_jPO_{t,z}$中主体访问客体的m_t条网络访问路径中任意一条网络路径z上,PIM层策略$\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,a_{j,k})$中$a_{j,1},\cdots,a_{j,n_j}$,对应$n_z$个访问监控以及相应$n_z$个权限或约束:$b_{t,z,1},\cdots,b_{t,z,n_z}$,这$n_z$个权限或约束要么同时授予、要么同时撤销,所以第$z$条路径$n_z$个策略表示为"与"关系.

策略$\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,a_{j,k})$精化表示成:$\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,a_{j,k})\rightarrow\bigvee_{z=1}^{m_t}\bigwedge_{k=1}^{n_z}p((s_i,c_j,ds_{t,z}),(o_i,do_{t,z}),b_{t,z,k}),\{\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,a_{j,k})|((s_i,c_j,ds_i),(o_i,do_i))\in SC_jOP_t\}\rightarrow\{\bigvee_{z=1}^{m_t}\bigwedge_{k=1}^{n_z}p((s_i,c_j,ds_{t,z}),(o_i,do_{t,z}),b_{t,z,k})|((s_i,c_j,ds_{t,z}),(o_i,do_{t,z}))\in SC_jOP_{t,z}\}$.

$\{\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,a_{j,k})|((s_i,c_j,ds_i),(o_i,do_i))\in SC_jOP_t\}$为父节点.

以m_t个节点分别表示$\{\bigwedge_{k=1}^{n_z}p((s_i,c_j,ds_{t,z}),(o_i,do_{t,z}),b_{t,z,k})|((s_i,c_j,ds_{t,z}),(o_i,do_{t,z}))\in SC_jOP_{t,z}\}$,$z=1,2,\cdots,m_t$,作为子节点.父节点类型为"*or*",表示父节点策略$\bigwedge_{k=1}^{n_j}p((s_i,c_j),o_i,a_{j,k})$为子节点策略$\bigwedge_{k=1}^{n_z}p((s_i,c_j,ds_{t,z}),(o_i,do_{t,z}),b_{t,z,k}),z=1,2,\cdots m_t$的"或",任何一条网络路径访问策略成立,则父节点策略成立,只有当所有网络路径访问策略均不能成立时,父节点策略失效.精化的策略精化树表述如图4点线框所示.

PIM层策略精化为PSM层策略流程如图5所示,算法2为PIM层策略精化为PSM层策略的算法描述.

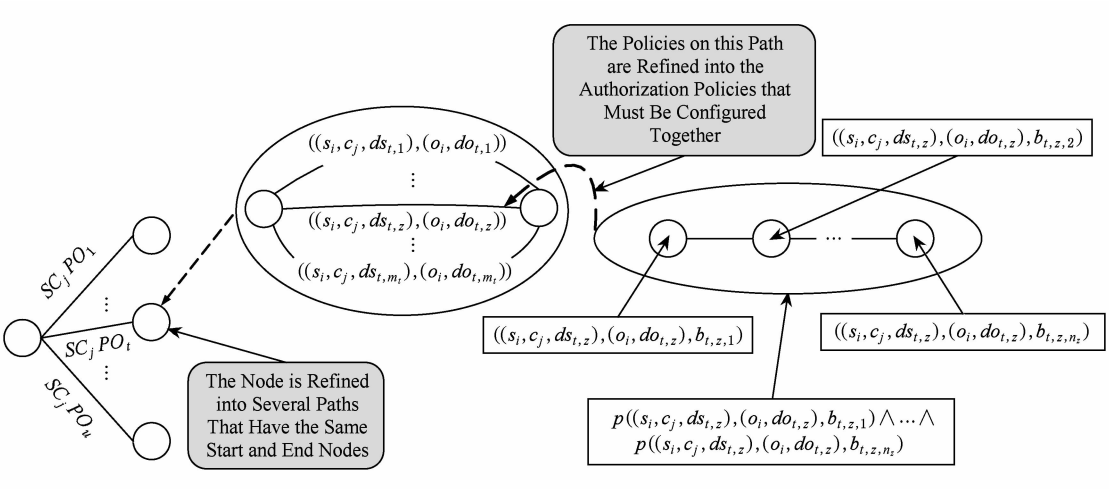

Fig. 5 The refinement process of PIM layer to PSM layer policies refinement.

图 5 PIM 层到 PSM 层策略精化的流程

算法 2. PIM 层策略精化为 PSM 层策略.

设 $j=1,2,\cdots,n$, $\{\bigwedge_{k=1}^{n_j} p((s_i,c_j),o_i,a_{j,k})\mid(s_i,o_i)\in SO, j\in\{1,2,\cdots,n\}\}$.

1) 对于 $j=1,2,\cdots,n$, $((s_i,c_j),o_i)\in SC_jO$, 确定 (s_i,c_j), o_i 对应的 PSM 物理位置 ds_i, do_i, 构造: $SC_jOP=\{((s_i,c_j,ds_i),(o_i,do_i))\mid((s_i,c_j),o_i)\in SC_jO\}$.

2) 构造具有相同访问控制起点 ds_i 和终点 do_i 的主客体子集 SC_jOP_t, $t=1,2,\cdots,u$, $\bigcup_{t=1}^{u} SC_jOP_t=SC_jOP$.

3) $j=1,2,\cdots,n$, 对 $\{\bigwedge_{k=1}^{n_j} p((s_i,c_j),o_i,a_{j,k})\mid(s_i,o_i)\in SO, j\in\{1,2,\cdots,n\}\}$, 构造: $\{\bigwedge_{k=1}^{n_j} p((s_i,c_j),o_i,a_{j,k})\mid((s_i,c_j,ds_i),(o_i,do_i))\in SC_jOP_t\}$, $t=1,2,\cdots,u$.

4) 对于 $j=1,2,\cdots,n$, $t=1,2,\cdots,u$, 确定 $\{\bigwedge_{k=1}^{n_j} p((s_i,c_j),o_i,a_j,k)\mid((s_i,c_j,ds_i),(o_i,do_i))\in SC_jOP_t\}$ 策略主体 (s_i,c_j,ds_i) 访问客体 (o_i,do_i) 的 m_t 条路径; 由 SC_jPO_t 构造 $SC_jPO_{t,z}$: 如果 $((s_i,c_j,ds_i),(o_i,do_i))\in SC_jPO_t$, 那么 $ds_{t,z}=ds_i$, $do_{t,z}=do_i$, $((s_i,c_j,ds_{t,z}),(o_i,do_{t,z}))\in SC_jPO_{t,z}$.

5) 对于 SC_jOP_t, $j=1,2,\cdots,n$, $t=1,2,\cdots,u$, 确定 PIM 层 $a_{j,1},\cdots,a_{j,n_j}$ 在 PSM 层任意网络访问路径 z 上相应的 $b_{t,z,1},\cdots,b_{t,z,n_z}$, $z=1,2,\cdots,m_t$.

6) 构造 PSM 策略: $\{\bigvee_{z=1}^{m_t}\bigwedge_{k=1}^{n_z} p((s_i,c_j,ds_{t,z}),(o_i,do_{t,z}),b_{t,z,k})\mid((s_i,c_j,ds_{t,z}),(o_i,do_{t,z}))\in SC_jPO_{t,z}\}$, $j=1,2,\cdots,n$, $t=1,2,\cdots,u$.

例 4. 假设例 3 的策略 PSM 层参数为: u_1 的物理位置为 ds_1, u_2 的物理位置为 ds_2, 资料 A 分布在服务器 1, 物理位置为 do_1, 资料 B 分布在服务器 2, 物理位置为 do_2, 所以 $\{p((u_i,F),z_j,r)\wedge p((u_i,F),K,auth)\mid(u_i,z_j)\in SO\}$ 的 SC_1OP 为

$SC_1OP=\{((u_1,F,ds_1),(z_1,do_1)),((u_2,F,ds_2),(z_1,do_1)),((u_1,F,ds_1),(z_2,do_2)),((u_2,F,ds_2),(z_2,do_2))\}$.

假设主体访问控制的入口点相同, 但是终止点不同, 分别为服务器 1 和服务器 2 的访问控制单元, 将 SC_1OP 划分为 2 个策略子集:

$SC_1OP_1=\{((u_1,F,ds_1),(z_1,do_1)),((u_2,F,ds_2),(z_1,do_1))\}$;

$SC_1OP_2=\{((u_1,F,ds_1),(z_2,do_2)),((u_2,F,ds_2),(z_2,do_2))\}$.

策略集合 $\{p((u_i,F),z_j,r)\wedge p((u_i,F),K,auth)\mid(u_i,z_j)\in SO\}$ 也根据 SC_1OP_1, SC_1OP_2 划分为 2 个子集: $\{p((u_i,F),z_j,r)\wedge p((u_i,F),K,auth)\mid(u_i,z_j)\in SO\}\rightarrow\bigcup_{t=1}^{2}\{p((u_i,F),z_j,r)\wedge p((u_i,F),K,auth)\mid((u_i,F,ds_i),(z_j,do_j))\in SC_1OP_t\}$.

分别以节点表示策略集合: $\{p((u_i,F),z_j,r)\wedge p((u_i,F),K,auth)\mid((u_i,F,ds_i),(z_j,do_j))\in SC_1OP_1\}$, $\{p((u_i,F),z_j,r)\wedge p((u_i,F),K,auth)\mid((u_i,F,ds_i),(z_j,do_j))\in SC_1OP_2\}$.

2 个节点作为节点 $\{p((u_i,F),z_j,r)\wedge p((u_i,F),K,auth)\mid(u_i,z_j)\in SO\}$ 的子节点.

同理: $\{p((u_i,W),z_j,r)\wedge p((u_i,W),K,auth)\mid(u_i,z_j)\in SO\}\rightarrow\bigcup_{t=1}^{2}\{p((u_i,W),z_j,r)p((u_i,W),K,auth)\mid((u_i,W,ds_i),(z_j,do_j))\in SC_2OP_t\}$. SC_2OP_1

$=\{((u_1,W,ds_1),(z_1,do_1)),((u_2,W,ds_2),(z_1,do_1))\}$;$SC_2OP_2=\{((u_1,W,ds_1),(z_2,do_2)),((u_2,W,ds_2),(z_2,do_2))\}$.

分别以节点表示:

$\{p((u_i,W),z_j,r)\wedge p((u_i,W),K,auth)\mid((u_i,W,ds_i),(z_j,do_j))\in SC_2OP_1\}$,$\{p((u_i,W),z_j,r)\wedge p((u_i,W),K,auth)\mid((u_i,W,ds_i),(z_j,do_j))\in SC_2OP_2\}$.

这2个节点作为节点$\{p((u_i,W),z_j,r)\wedge p((u_i,W),K,auth)\mid(u_i,z_j)\in SO\}$的子节点.

策略基于物理位置的精化如图6中号虚线框部分所示.

假设根据PSM层体系结构参数,销售和售后人员FTP和Web访问方式对资料A、资料B的访问各自存在两条物理拓扑路径,路径访问监控分别如下:

路径1. 系统有线http端口认证、防火墙1、服务器n的该种访问方式的访问控制单元1.

路径2. 系统WLAN端口认证、防火墙2、服务器n的该种访问方式的访问控制单元1.

$K1$为系统有线http端口认证,$K2$为系统WLAN端口认证.

对于策略$p((u_i,F),z_j,r)\wedge p((u_i,F),K,auth)$,对应PSM层权限分别如下.

路径1. 访问控制单元1的读权限$r1$、系统有线http端口认证$K1$权限、防火墙1的入权限$in1$、防火墙1的出权限$out1$.

路径2. 访问控制单元1的读权限$r1$、系统WLAN端口认证$K2$权限、防火墙2的入权限$in2$、防火墙2的出权限$out2$.

由算法2可知:

$SC_1OP_{1,1}=\{((u_1,F,ds_{1,1}),(z_1,do_{1,1})),((u_2,F,ds_{1,1}),(z_1,do_{1,1}))\}$,

$SC_1OP_{1,2}=\{((u_1,F,ds_{1,2}),(z_1,do_{1,2})),((u_2,F,ds_{1,2}),(z_1,do_{1,2}))\}$.

$\{p((u_i,F),z_j,r)\wedge p((u_i,F),K,auth)\mid((u_i,$

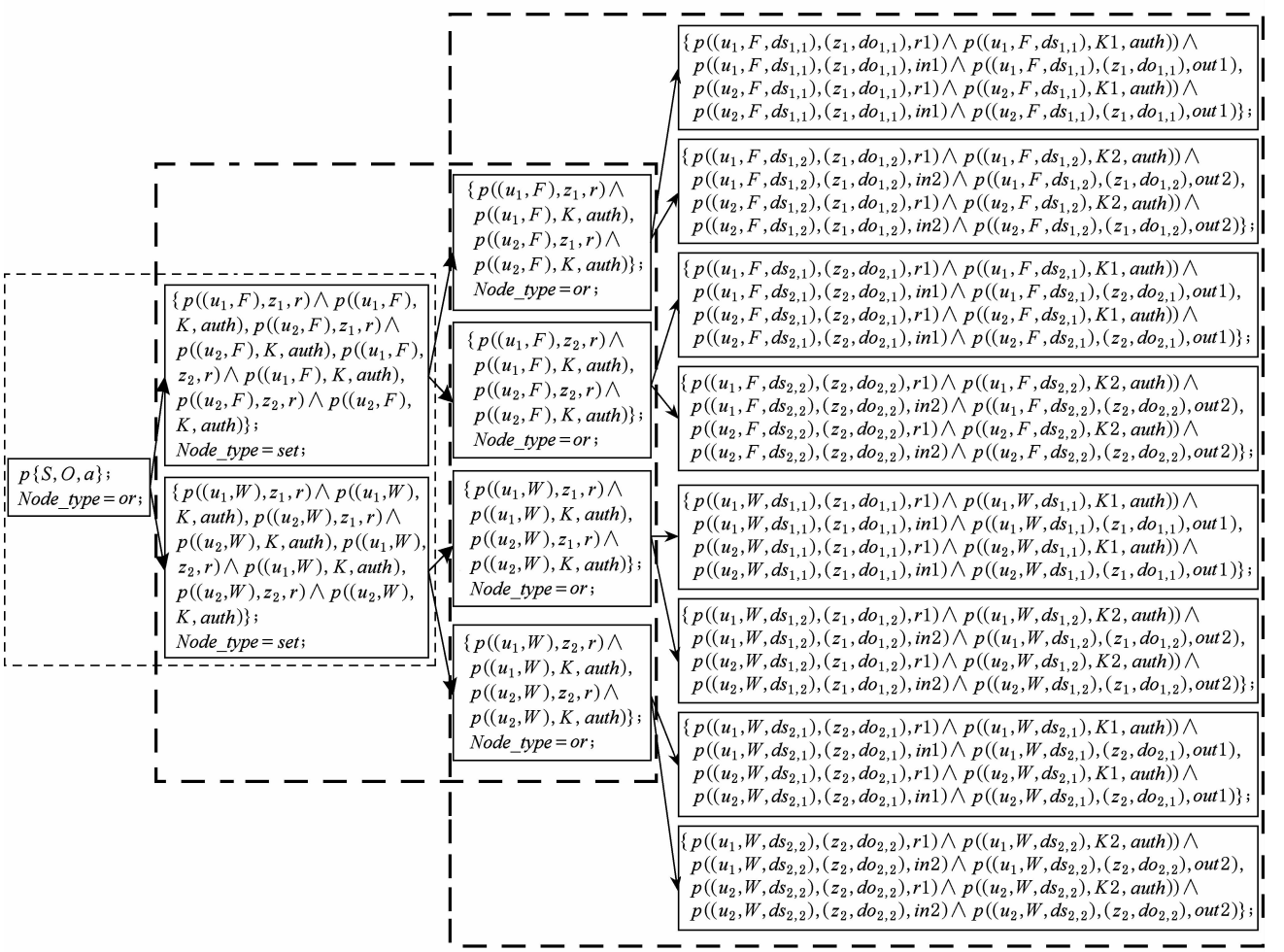

Fig. 6 The refinement tree of the example 1 policy $p(S,O,a)$.

图6 例1策略$p(S,O,a)$的精化树

$F,ds_i),(z_j,do_j))\in SC_1OP_1\}\rightarrow\{(p((u_i,F,ds_{1,1}),(z_j,do_{1,1}),r1)\wedge p((u_i,F,ds_{1,1}),K1,auth)\wedge p((u_i,F,ds_{1,1}),(z_j,do_{1,1}),in1)\wedge p((u_i,F,ds_{1,1}),(z_j,do_{1,1}),out1))\vee(p((u_i,F,ds_{1,2}),(z_j,do_{1,2}),r1)\wedge p((u_i,F,ds_{1,2}),K2,auth)\wedge p((u_i,F,ds_{1,2}),(z_j,do_{1,2}),in2)\wedge p((u_i,F,ds_{1,2}),(z_j,do_{1,2}),out2))\}$.

$\{p((u_i,F,ds_{1,1}),(z_j,do_{1,1}),r1)\wedge p((u_i,F,ds_{1,1}),K1,auth)\wedge p((u_i,F,ds_{1,1}),(z_j,do_{1,1}),in1)\wedge p((u_i,F,ds_{1,1}),(z_j,do_{1,1}),out1)\mid((u_i,F,ds_{1,1}),(z_j,do_{1,1}))\in SC_1PO_{1,1}\}$称为策略集合 1.

$\{p((u_i,F,ds_{1,2}),(z_j,do_{1,2}),r1)\wedge p((u_i,F,ds_{1,2}),K2,auth)\wedge p((u_i,F,ds_{1,2}),(z_j,do_{1,2}),in2)\wedge p((u_i,F,ds_{1,2}),(z_j,do_{1,2}),out2)\mid((u_i,F,ds_{1,2}),(z_j,do_{1,2}))\in SC_1OP_{1,2}\}$称为策略集合 2.

策略集合 1 与策略集合 2 分别作为$\{p((u_i,F),z_j,r)\wedge p((u_i,F),K,auth)\mid((u_i,F,ds_i),(z_j,do_j))\in SC_1OP_1\}$的儿子节点,父节点类型为"*or*".

策略节点$\{p((u_i,F),z_j,r)\wedge p((u_i,F),K,auth)\mid((u_i,F,ds_i),(z_j,do_j)\in SC_1OP_2\}$和策略节点$\{p((u_i,W),z_j,r)\wedge p((u_i,W),K,auth)\mid((u_i,W,ds_i),(z_j,do_j))\in SC_2OP_1\}$,$\{p((u_i,W),z_j,r)\wedge p((u_i,W),K,auth)\mid((u_i,W,ds_i),(z_j,do_j))\in SC_2OP_2\}$的精化计算同理.如图 6 最大虚线框内部分所示.

例 5. 例 2 PIM 策略"售后人员通过 Web 写维修日志,必须同时写技术统计表",表示为$\wedge\{p_1,p_2\}=\{p((s_j,W),z_1,a_1)\wedge p((s_j,W),z_2,a_2)\mid j=1,2\}$.子节点策略:$p_1=\{p((s_j,W),z_1,a_1)\mid j=1,2\}$,$p_2=\{p((s_j,W),z_2,a_2)\mid j=1,2\}$,子节点类型为"*set*".

p_1, p_2 的 SC_1OP 分别表示为 $SC_1OP_{(p_1)}$, $SC_1OP_{(p_2)}$.假设根据 PSM 层体系结构参数,主体$(s_1,W),(s_2,W)$物理位置分别为 ds_1,ds_2,在同一访问控制入口点;客体物理位置分别为 do_1,do_2,是在同一维修服务器上,那么:

$SC_1OP_{(p_1)}=\{((s_1,W,ds_1),(z_1,do_1)),((s_2,W,ds_2),(z_1,do_1))\}$,$SC_1OP_{(p_1)t}=SC_1OP_{(p_1)}$,$t=1$.

$p_1\rightarrow\{p((s_j,W),z_1,a_1)\mid((s_j,W,ds_j),(z_1,do_1))\in SC_1OP_{(p_1)1}\}$.

$\{p((s_j,W),z_1,a_1)\mid((s_j,W,ds_j),(z_1,do_1))\in SC_1OP_{(p_1)1}\}$作为 p_1 子节点.

同理:

$p_2\rightarrow\{p((s_j,W),z_2,a_2)\mid((s_j,W,ds_j),(z_2,do_2))\in SC_2OP_{(p_2)1}\}$.

$\{p((s_j,W),z_2,a_2)\mid((s_j,W,ds_j),(z_2,do_2))\in SC_2OP_{(p_2)1}\}$作为 p_2 子节点.

假设 PSM 层策略$\{p((s_j,W),z_1,a_1)\mid((s_j,W,ds_j),(z_1,do_1))\in SC_1OP_{(p_1)1}\}$与$\{p((s_j,W),z_2,a_2)\mid((s_j,W,ds_j),(z_2,do_2))\in SC_2OP_{(p_2)1}\}$均存在唯一网络路径,路径访问监控依次为:包过滤防火墙 1、系统 http 端口认证、维修服务器访问控制.

PIM 层策略 $p((s_i,W),z_1,a_1)$访问权限 a_1 对应 PSM 层权限:维修服务器访问控制的写权限 w、系统有线 http 端口认证 $K1$ 的权限、防火墙 1 的入权限 $in1$ 和防火墙 1 的出权限 $out1$,所以:

$p((s_j,W),z_1,a_1)\rightarrow p((s_j,W,ds_{1,1}),(z_1,do_{1,1}),w)\wedge p((s_j,W,ds_{1,1}),(z_1,do_{1,1}),in1)\wedge p((s_j,W,ds_{1,1}),(z_1,do_{1,1}),out1)\wedge p((s_j,W,ds_{1,1}),K1,auth)$;

$\{p((s_j,W),z_1,a_1)\mid((s_j,W,ds_j),(z_1,do_1))\in SC_1OP_{(p_1)1}\}\rightarrow\{p((s_j,W,ds_{1,1}),(z_1,do_{1,1}),w)\wedge p((s_j,W,ds_{1,1}),(z_1,do_{1,1}),in1)\wedge p((s_j,W,ds_{1,1}),(z_1,do_{1,1}),out1)\wedge p((s_j,W,ds_{1,1}),K1,auth)\}$.

将上式符号"→"右边策略集合节点作为左边策略集合节点的子节点,父节点类型为"*or*".

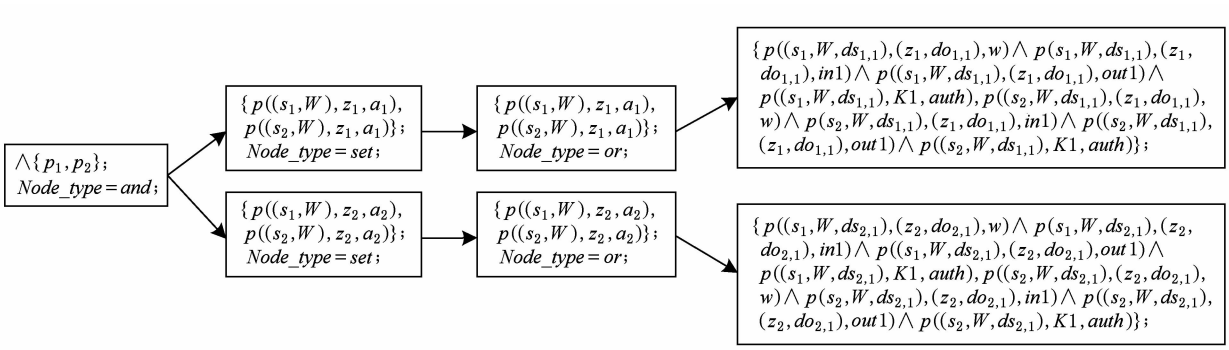

Fig. 7 The refinement tree of the policy $\wedge\{p_1,p_2\}$.

图 7 策略$\wedge\{p_1,p_2\}$精化树

同理:

$\{p((s_j,W),z_2,a_2)|((s_j,W,ds_i),(z_2,do_2))\in SC_2OP_{(p_2)1}\}\rightarrow\{p((s_j,W,ds_{2,1}),(z_2,do_{2,1}),w)\wedge p((s_j,W,ds_{2,1}),(z_2,do_{2,1}),in1)\wedge p((s_j,W,ds_{2,1}),(z_2,do_{2,1}),out1)\wedge p((s_j,W,ds_{2,1}),K1,auth)\}$.

将上面式子符号"→"右边策略集合节点作为左边策略集合节点的子节点,父节点类型为"*or*".策略精化树如图 7 所示.

1.3.3 系统策略精化树的类型

基于上述策略形式描述和精化计算中介绍的策略精化树构建方法,CIM,PIM 和 PSM 层策略精化可能形成的策略精化树形式如图 8～12 所示,其中的Ⅰ,Ⅱ,Ⅲ代表节点策略的形式,Ⅰ代表$\wedge\{p_1,\cdots,p_m\}$形式策略,Ⅱ代表 $p(S,O,a)$形式策略,Ⅲ代表 $p(s,o,a_1)\wedge\cdots\wedge p(s,o,a_n)$形式策略."*and*","*set*"和"*or*"为 *Node_type* 的值.

2 基于精化树策略关联属性的冲突分析

CIM,PIM,PSM 的各层策略经过精化,形成如图 8～12 类型的精化树.精化自上而下分层进行,逐层生长精化树.精化树不仅描述策略,还记录策略之间组合、精化、计算相关层与平台相关层访问协同等关联属性,利用这些关联属性可以方便地进行冲突分析,并修正由于策略冲突消解引起的策略关联属性问题.

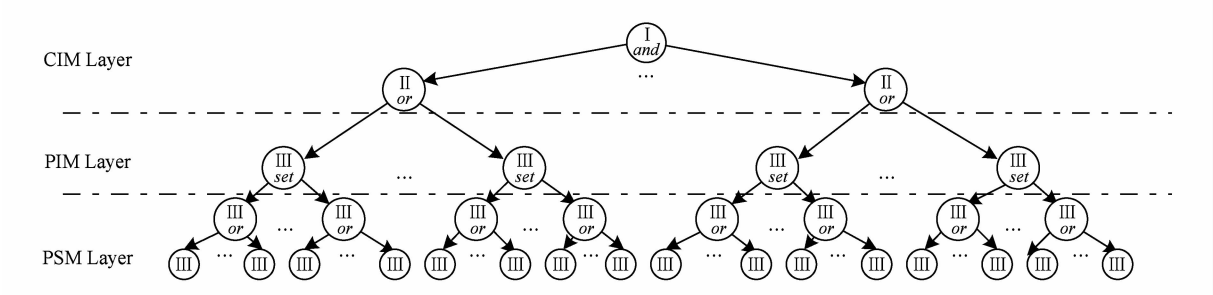

Fig. 8 The refinement tree representation of the policy $\wedge\{p_1,\cdots,p_m\}$ in CIM layer.

图 8 CIM 层$\wedge\{p_1,\cdots,p_m\}$策略精化树形式

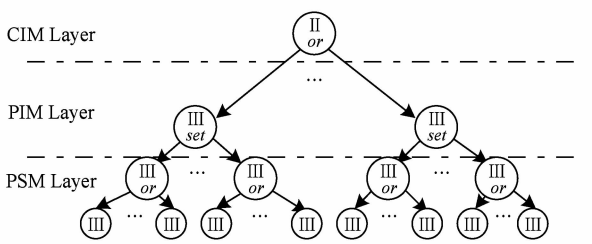

Fig. 9 The refinement tree representation of the policy $p(S,O,a)$ in CIM layer.

图 9 CIM 层 $p(S,O,a)$策略精化树形式

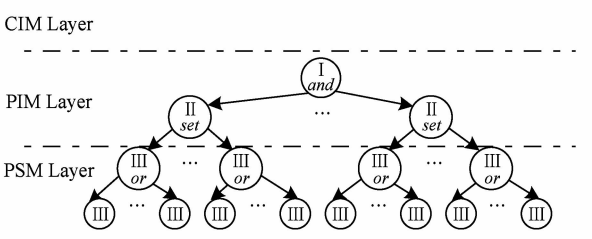

Fig. 10 The refinement tree representation of the policy $\wedge\{p_1,\cdots,p_m\}$ in PIM layer.

图 10 PIM 层$\wedge\{p_1,\cdots,p_m\}$策略精化树形式

CIM,PIM 各层策略精化之前,采用访问控制策略空间冲突分析技术或者其他合适的高效冲突分析技术,分析本层叶子节点策略集合中存在的策略冲突,根据冲突策略所在精化树记录的策略之间关联属性,分析由叶子节点冲突消解引起的策略关联问题并修正,分析修正沿精化树向上递归进行,直到精化树上所有策略符合精化树节点记录的策略之间关联属性约束.

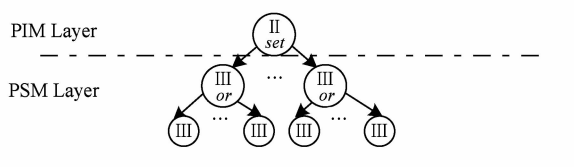

Fig. 11 The refinement tree representation of the policy $p(S,O,a)$ in PIM layer.

图 11 PIM 层 $p(S,O,a)$策略精化树形式

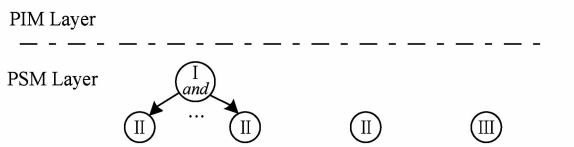

Fig. 12 The refinement tree representation of the policies that belongs to Ⅰ,Ⅱ,Ⅲ in PSM layer.

图 12 PSM 层Ⅰ,Ⅱ,Ⅲ策略精化树形式

本文通过例子来说明利用精化树上记录的关联属性,可以方便地分析修正策略之间关联问题.分析过程是一个复杂的过程,由于篇幅原因,具体算法及证明我们在另外的文章中详细阐述.

例 6. 假如 IDS 发现 http 协议 IP 地址为 202.119.28.211、端口为 80 的用户行为有恶意,设置策略禁止它的所有访问权限,记(202.119.28.211,80)=ds_1,策略为 $p((*,W,ds_{1,1}),*,\neg *)$,将它加入当前 PSM 层的策略集合中,表示为图 13 中 PSM 层黑粗线框策略节点,其中权限"$\neg *$"表示禁止任何行为.通过 PSM 层所有叶子节点策略集合的原子策略冲突分析,发现图 13 策略精化树上 PSM 层的虚线框叶子节点上原子策略 $p((s_1,W,ds_{1,1}),(z_1,do_{1,1}),w)$的主客体属于策略 $p((*,W,ds_{1,1}),*,\neg *)$主客体集合,权限 w 与 $\neg *$ 冲突,因此 2 个策略之间存在冲突.对该冲突策略进行消解并分析修正其关联属性的过程如下:

1) 假设策略冲突消解规则为否定优先,消解精化树叶子节点原子策略 $p((s_1,W,ds_{1,1}),(z_1,do_{1,1}),w)$.

2) 由节点策略描述可知:策略 $p((s_1,W,ds_{1,1}),(z_1,do_{1,1}),w)$存在访问路径协同策略:$p((s_1,W,ds_{1,1}),(z_1,do_{1,1}),w)\wedge p((s_1,W,ds_{1,1}),(z_1,do_{1,1}),in1)\wedge p((s_1,W,ds_{1,1}),(z_1,do_{1,1}),out1)\wedge p((s_1,W,ds_{1,1}),K1,auth)$.因为 $p((s_1,W,ds_{1,1}),(z_1,do_{1,1}),w)$被消解,这些具有"$\wedge$"关系的原子策略也必须同时取消,保证路径策略协同.

3) 被消解的路径协同策略:$p((s_1,W,ds_{1,1}),(z_1,do_{1,1}),w)\wedge p((s_1,W,ds_{1,1}),(z_1,do_{1,1}),in1)\wedge p((s_1,W,ds_{1,1}),(z_1,do_{1,1}),out1)\wedge p((s_1,W,ds_{1,1}),K1,auth)$,其父节点类型为"*or*",并且父节点只有一个儿子节点,父子节点策略关系为:

$p((s_1,W),z_1,a_1)\rightarrow p((s_1,W,ds_{1,1}),(z_1,do_{1,1}),w)\wedge p((s_1,W,ds_{1,1}),(z_1,do_{1,1}),in1)\wedge p((s_1,W,ds_{1,1}),(z_1,do_{1,1}),out1)\wedge p((s_1,W,ds_{1,1}),K1,auth)$.

子策略被取消,则父策略 $p((s_1,W),z_1,a_1)$失效,为保证父子精化关系成立也被取消.

4) PSM 层策略 $p((s_1,W),z_1,a_1)$的父节点的节点类型为"*set*",父子节点策略关系为 $p((s_1,W),z_1,a_1)\rightarrow p((s_1,W),z_1,a_1)$.

该式子左边 PIM 层的父节点策略因右边 PSM 层子节点策略取消而失效,同样为保证父子精化关系成立也被取消.

5) PIM 层策略 $p((s_1,W),z_1,a_1)$的父节点类型为"*and*",兄弟节点策略关系为:$p((s_1,W),z_1,a_1)\wedge p((s_1,W),z_2,a_2)$,$p((s_1,W),z_1,a_1)$失效,所以 $p((s_1,W),z_1,a_1)\wedge p((s_1,W),z_2,a_2)$失效.因此,也应取消 $p((s_1,W),z_2,a_2)$.取消 PIM 层策略 $p((s_1,W),z_2,a_2)$,由于该策略节点类型为"*set*",为保证父子精化关系成立也取消它的子节点 PSM 层策略 $p((s_1,W),z_2,a_2)$;PSM 层策略$p((s_1,W),z_2,$

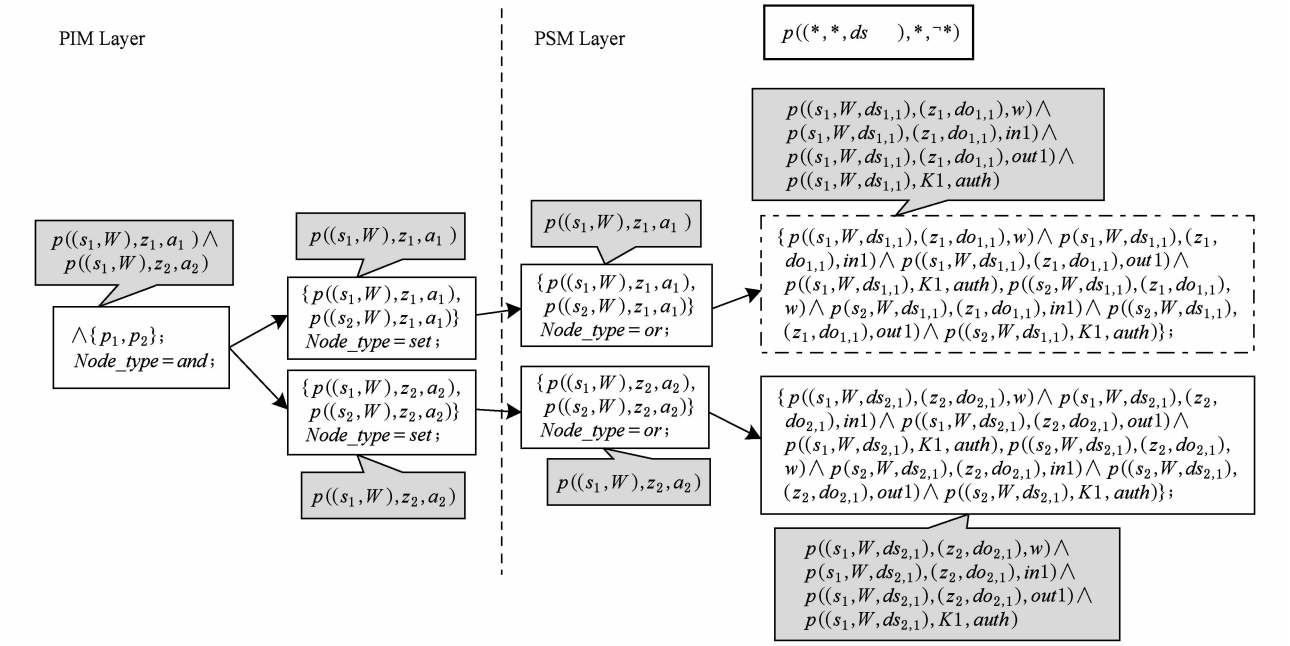

Fig. 13 Example of conflict resolution in policy refinement tree.

图 13 策略精化树冲突消解示例

a_2)节点类型"*or*",并且只有一个子节点,为保证父子精化关系成立也取消它的子节点策略$p((s_1,W,ds_{2,1}),(z_2,do_{2,1}),w)\wedge p((s_1,W,ds_{2,1}),(z_2,do_{2,1}),in1)\wedge p((s_1,W,ds_{2,1}),(z_2,do_{2,1}),out1)\wedge p((s_1,W,ds_{2,1}),K1,auth)$.

6) $\wedge\{p_1,p_2\}=\{p((s_1,W),z_1,a_1)\wedge p((s_1,W),z_2,a_2),p((s_2,W),z_1,a_1)\wedge p((s_2,W),z_2,a_2)\}$,节点类型为"*and*",父子节点之间存在关系:

$$p((s_1,W),z_1,a_1)\wedge p((s_1,W),z_2,a_2)\rightarrow p((s_1,W),z_1,a_1)\wedge p((s_1,W),z_2,a_2).$$

由于子节点策略 $p((s_1,W),z_1,a_1)$和 $p((s_1,W),z_2,a_2)$失效,所以为保证父子精化关系成立从$\wedge\{p_1,p_2\}$中删除 $p((s_1,W),z_1,a_1)\wedge p((s_1,W),z_2,a_2)$.

被消解删除的策略为图 13 灰框内的策略,灰框指向被删策略原先在精化树上的位置,经过基于精化树的分析修正,保留的策略符合精化树上记录的策略之间的关联属性约束.这样根据叶子节点原子策略冲突判断,基于精化树策略关联属性描述,实现有效分析、修正策略之间的组合、精化、路径协同关系.

互斥规则是一种重要和常用的策略关系,描述两个不能同时成立的策略.对于原子策略$p(s_1,o_1,\alpha)$与 $p(s_2,o_2,\beta)$,如果 $s_1=s_2,o_1\neq o_2$,$p(s_1,o_1,\alpha)$与$p(s_2,o_2,\beta)$不能同时被授权,称 $p(s_1,o_1,\alpha)$与 $p(s_2,o_2,\beta)$互斥,表示为 $p(s_1,o_1,\alpha)\oplus p(s_2,o_2,\beta)$,即$p(s_1,o_1,\alpha)\oplus p(s_2,o_2,\beta)=\neg((p(s_1,o_1,\alpha)\wedge p(s_2,o_2,\beta))$.

互斥关系判断与最终的策略集合相关,经过叶子节点原子策略冲突判断,基于策略精化树分析修正因叶子节点策略冲突消解造成的策略关联属性问题,再根据互斥规则确定叶子节点原子策略集合中违反互斥规则的原子策略,这些原子策略需要消解.消解这些原子策略,与上面消解叶子节点冲突策略相同,这样保证策略之间满足互斥规则.

本文设计策略和包括组合、互斥、精化、访问路径协同等策略之间关系的形式描述,设计具有策略之间关系描述的策略精化算法,设计策略精化树描述策略和策略之间的这些关系.结合策略冲突分析,以及基于策略关系描述,分析修正策略关联属性的技术,可以在以下 5 个方面提高策略精化的能力:

1) 精化计算不仅可以分析策略冲突,还能进一步基于精化树的策略关系描述,分析和修正由策略冲突引起的组合、互斥、精化完整性和一致性、各层访问路径协同策略一致性等关联属性问题.

2) 因为可以根据精化树自下而上修正策略关联属性,所以冲突解决规则的选择不再限制于由上层策略代替下层策略,可以根据应用需要自由选择,适用于根据具体事件和状态实时决定应用策略的情况.

3) 策略精化算法在 CIM 层策略向 PIM 层策略精化时,提供不同计算方式精化的精化关系描述,扩大了精化适用性.

4) 基于精化树的冲突消解和关联属性修正计算,对不同类型策略冲突是等价的,这样可以处理像 IPSec 这类技术需要有序分析多种类型策略冲突的情况.

5) 精化树描述的策略精化关系,呈现访问控制目标和实现之间的对应关系,是访问控制 SLA 的直接呈现,可以用于访问控制安全目标及其实现的审核.

3 结 论

由于现有策略精化技术策略之间的关系描述能力,以及基于关系描述分析、修正策略关联属性的能力不足,限制策略精化的实际应用.本文研究了策略冲突分析能力与策略描述方式之间的联系、形式规约、描述访问控制策略和策略关联属性,设计了具有策略之间组合、精化、访问路径协同等关系描述的策略精化算法,给出了策略精化树的构建方法.精化树不仅描述策略,还可以描述策略之间组合、精化、访问路径协同等关系,通过例子说明依据精化树描述的策略之间关系可以分析、修正策略组合、互斥、精化、路径协同等关联属性,有序处理不同类型策略冲突,根据应用需要自由选择冲突解决规则,从而提高策略精化能力,并且策略精化树能直观呈现面向服务访问控制 SLA.

参 考 文 献

[1] Sloman M. Policy driven management for distributed systems [J]. Journal of Network and Systems Management, 1994, 2 (4): 333-360

[2] Jason B. The SOA management landscape [R/OL]. 2006 [2010-05-12]. http://www.zapthink.com/2006/11/30/the-soa-management-landscape

[3] Maullo M J, Calo S B. Policy management: An architecture and approach [C] //Proc of the 1st IEEE Int Workshop on Systems Management. Piscataway, NJ: IEEE, 1993: 13-26

[4] Pieters W, Dimkov T, Pavlovic D. Security policy alignment: A formal approach [J]. IEEE Systems Journal, 2013, 7(2): 275-287

[5] Mont C M, Baldwin A, Goh C. POWER prototype: Towards integrated policy-based management [C/OL] // Network Operations and Management Symp. IEEE/IFIP, 2000 [2010-05-15]. http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=830429&queryText%3DPOWER+prototype%3A+Towards+integrated+policy-based+management

[6] The Open Group. SLA management handbook, Volum 4: Enterprise perspective [S/OL]. TeleManagementFORUM, (2004-10) [2010-05-15]. http://www.afutt.org/Qostic/qostic1/SLA-DI-USG-TMF-060091-SLA_TMForum.pdf

[7] Kumari P, Pretschner A. Deriving implementation-level policies for usage control enforcement [C] //Proc of the 2nd ACM Conf on Data and Application Security and Privacy. New York: ACM, 2012: 83-94

[8] Jayaraman K, Ganesh V, Tripunitara M, et al. Automatic error finding in access-control policies [C] //Proc of the 18th ACM conf on Computer and Communications Security. New York: ACM, 2011: 163-174

[9] Lampson B, Abadi M, Burrows M, et al. Authentication in distributed systems: Theory and pratice [J]. ACM Trans Computer Systems, 1992, 10(4): 265-310

[10] Abadi M, Burrows M, Lampson B, et al. A calculus for access control in distributed systems [J]. ACM Trans on Programming Languages and Systems, 1993, 15(4): 706-734

[11] Davy S, Jennings B, Strassner J. On harnessing information models and ontologies for policy conflict analysis [C/OL] //Int Symp on Integrated Network Management. IFIP/IEEE, 2009 [2010-06-18]. http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=5188889&queryText%3DOn+harnessing+information+models+and+ontologies+for+policy+con%EF%AC%82ict+analysis

[12] Lück I, Vögel S, Krumm H. Model-based configuration of VPNs [C/OL] //Network Operations and Management Symp. IEEE/IFIP, 2002 [2009-06-20]. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1015610&tag=1

[13] Albuquerque J P, Krumm H, Geus P L, et al. Formal validation of automated policy refinement in the management of network security systems [J]. Int Journal of Information Security, 2010, 9(2): 99-125

[14] Fu Z, Wu S F. Automatic generation of IPSec/VPN security policies in an intra-domain environment [C/OL] //Proc of the 12th Int Workshop on Distributed System Operation & Management. 2001 [2008-06-10]. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.12.789

[15] McDougall M, Alur R, Gunter C A. A model-based approach to integrating security policies for embedded devices [C] //Proc of the 4th ACM Int Conf on Embedded Software. New York: ACM, 2004: 211-219

[16] Boella G, Torre L. Security policies for sharing knowledge in virtual communities [J]. IEEE Trans on Syatems, Man, and Cybernetics, Part A: Systems and Humans, 2006, 36(3): 439-450

[17] Bruns G, Huth M. Access control via belnap logic: Intuitive, expressive, and analyzable policy composition [J]. ACM Trans on Information and System Security, 2011, 14(1): 9: 1-9: 27

[18] Jajodia S, Samarati P, Sapion M L, et al. Flexible support for multiple access control policies [J]. ACM Trans on Database Systems, 2001, 26(2): 214-260

[19] Bertino E, Buccafurri F, Ferrari E, et al. A logical framework for reasoning on data access control policies [C] // Proc of the 12th IEEE Computer Security Foundations Workshop. Piscataway, NJ: IEEE, 1999: 175-189

[20] Wang Yazhe, Feng Dengguo. A conflict and redundancy analysis method for XACML rules [J]. Chinese Journal of Computers, 2009, 32(3): 516-530 (in Chinese)
(王雅哲, 冯登国. 一种 XACML 规则冲突及冗余分析方法 [J]. 计算机学报, 2009, 32(3): 516-530)

[21] Basile C, Cappadonia A, Lioy A. Network-level access control policy analysis and transformation [J]. IEEE/ACM Trans on Networking, 2012, 20(4): 985-998

[22] Yao Jian, Mao Bing, Xie Li. A DAG-based security policy connicts detection method [J]. Journal of Computer Research and Development, 2005, 42(7): 1108-1114 (in Chinese)
(姚键, 茅兵, 谢立. 一种基于有向图模型的安全策略冲突检测方法[J]. 计算机研究与发展, 2005, 42(7): 1108-1114)

[23] Jaeger T, Zhang Xiaolan, Edwards A. Policy management using access control spaces [J]. ACM Trans on Information and System Security, 2003, 6(3): 327-364

Wu Yinghong, born in 1966. Master and senior engineer. Her main research interests include network security and cloud security.

Huang Hao, born in 1957. PhD and professor of Nanjing University. His main research interests include information security (hhuang@nju.edu.cn).

Zeng Qingkai, born in 1963. PhD and professor of Nanjing University. His main research interests include information security (zqk@nju.edu.cn).