# 分布式应用访问控制策略精化与冲突分析技术

吴迎红 黄 皓 曾庆凯
(南京大学计算机软件新技术国家重点实验室 南京 210023)

**摘 要** 策略精化是面向服务分布式应用访问控制策略配置的重要方法。分析了现有策略精化技术,包括系统和策略分层描述、推导下层协同控制策略的方法、策略精化中的策略冲突分析与消解方法、策略精化完全性和一致性、纵深防御协同策略一致性、应用策略组合与互斥约束等策略之间的关联属性分析方法。通过分析发现,策略关联属性分析能力不足是影响精化能力的关键问题。进一步分析了现有策略冲突分析技术的关联属性分析、分布式应用适用性和计算可扩展性。基于分析结果,提出了一些提高策略精化能力以适应面向服务的分布式应用的研究问题。

**关键词** 访问控制,分布式,模型驱动架构,策略精化,策略冲突分析
**中图法分类号** TP309.2 **文献标识码** A

## Techniques of Distributed Application Access Control Policy Refinement and Policy Conflict Analysis

WU Ying-hong HUANG Hao ZENG Qing-kai
(State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China)

**Abstract** Policy refinement is an important method to resolve the complexity of distribute access control policy configuration. This article analyzed the existing policy refinement techniques, including their system and policies hierarchical description, their methods to derive the lower levels coordination control policies, their ability of policy conflict analysis and dispel in policy refinement, and the associated attribute analysis methods about the completeness and consistency in policy refinement, the consistency among coordination control policies for defense in depth, the composing and mutual exclusion polices relationship of application. The analysis shows that the key problem that affects refinement ability is lack of policies associated attribute analysis ability. The article further analyzed policies associated attribute analysis of the existing policy conflict analysis techniques, usability of distributed application and calculation expansibility. Based on these analysis we pointed out some research problems which can improve policy refinement ability to adapt the service oriented distributed application.

**Keywords** Access control, Distributed, Model driven architecture, Policy refinement, Policy conflict analysis

## 1 引言

面向服务架构(SOA)、服务云(SaaS)是分布式技术的新发展,目标是解决系统实现的复杂性,实现商业应用需求到IT产品的自动构建和配置[1]。基于模型逐层细化的模型驱动架构(Model Driven Architecture, MDA)方法[2]是其中的重要方法,模型(Model)是MDA方法的核心。

策略是面向服务应用管理的关键[3]。以MDA方法的系统各层对象为策略描述对象,通过策略的映射规则,将上层抽象访问控制策略自动转化为下层实际访问路径监控的配置策略,这个过程称为策略精化[4-7]。

精化技术通常包括以下内容:(1)系统模型:各层系统对象类型、对象、对象之间关系的描述,不同层次对象集不同;(2)策略模型:各层系统对象权限描述形式;(3)具体系统对象权限,即安全策略模型实例化;(4)上下层映射规则(模板):系统映射规则提供上层系统对象和系统模型到下层系统对象和系统模型的映射;策略映射规则(模板)根据安全保障等级和系统参数,实现上层策略到下层策略的映射[8,9]。其中(1)—(3)体现精化技术的系统和策略分层描述能力,是精化计算的基础,影响精化技术的适用范围;(4)体现精化技术推导下层策略的能力。随着安全问题的日益严重,纵深防御是保证安全的重要手段,对于分布式应用访问控制而言,一个主体请求访问一个客体往往需要经过一个或多个访问控制单元进行权限检查,控制访问过程中的不同侧面。是否能够推导下层系统访问路径一系列控制单元的相应策略,是衡量精化能力的重要方面。

策略精化需要策略之间满足下列属性[4,10]:完全性,即所有上层策略在下层得到支撑;一致性,即同层策略之间、下层策略与上层策略之间没有冲突。面向服务的计算,需要服务品质协议(service-level agreement, SLA)的呈现[11],上层访问控制抽象目标、机制到下层实现技术和参数之间的精化关系是访问控制的SLA呈现。

访问控制策略之间有效的精化除了精化完全性和一致性外,还需要保证访问路径一系列协同策略一致性。一个主体

到稿日期:2013-05-29 返修日期:2013-09-02 本文受国家863项目:以支撑电子商务为主的网络操作系统研制(2011AA01A202)资助。
吴迎红(1966—),女,高级工程师,主要研究方向为信息安全;黄 皓(1957—),男,教授,主要研究方向为信息安全;曾庆凯(1963—),男,教授,主要研究方向为信息安全。

请求访问一个客体往往需要经过一个或多个访问控制单元进行权限检查，实现主体对客体的访问必须同时具有各个访问控制单元协同一致的权限，任何一个权限不成立，访问都不能实现。

还有应用有时需要策略之间保持组合与互斥等约束。应用系统有时要求一组不可分割的操作权限。比如要求一个售后人员维修设备时，必须登记维修日志，同时填写技术统计表，即必须同时具有写维修日志和技术统计表的权限，如果该售后人员离职，要同时取消上述2个写权限。同样，互斥约束在应用中也是常见的。比如财会人员职责分离，即财会人员访问出纳文件的权限和访问会计文件的权限互斥。

假设谓词 $p(s,o,a)$ 或 $p(s,o,-a)$ 表示访问控制策略，$p(s,o,a)$ 表示授予主体 $s$ 对客体 $o$ 的访问权限 $a$，$p(s,o,-a)$ 表示禁止主体 $s$ 对客体 $o$ 的访问权限 $a$；假设两个策略 $p_1=p(s_1,o_1,\alpha)$ 或 $p_2=p(s_2,o_2,\beta)$，$s_1=s_2$，$o_1=o_2$，如果 $\alpha=a$，$\beta=-a$，或者 $p_1$、$p_2$ 语义上不能同时成立，则 $p_1$、$p_2$ 冲突。策略冲突分析是策略之间关联属性分析的基础。根据冲突解决规则，其中一个策略被取消，被取消策略如果与其它策略之间存在上述精化完全性与一致性、访问路径协同一致性、应用组合与互斥等关联属性，则关联属性被破坏，需要同时取消与之关联的策略，否则会出现安全漏洞，精化就不具有实际运用意义。因此访问控制策略精化的冲突分析不仅需要分析策略冲突，还需要分析冲突消解后策略之间的关联属性是否正确。精化中策略冲突和策略关联属性的分析、修正能力决定精化技术的实际精化能力。

本文是关于分布式应用访问控制策略精化与冲突分析技术的综述。第2节分析现有策略精化技术，包括系统、策略分层描述，访问路径控制单元相关策略推导方法，策略冲突和关联属性分析方法，访问控制策略SLA呈现能力。基于分析结果，本文第3节进一步分析现有策略冲突分析技术的关联属性分析、分布式应用适用性和计算可扩展性。第4节基于上述分析结果，提出一些提高精化能力以适应面向服务分布式应用的研究问题。

## 2 访问控制策略精化技术

### 2.1 基于策略模板的精化

模板以固定的格式描述常见问题及其解决，基于策略模板的精化，上层抽象策略根据模板和上下文参数映射为下层配置策略。

POWER[12]是比较早的基于模板策略的精化技术，它基于环境和技术参数将抽象策略主体、客体、权限映射为实现层主体、客体和操作权限。比如图1抽象策略“工程师可以操作其组织内部信息”(c2)，上下文参数(c8)给出工程师名单和隶属组织目录，策略模板根据上下文参数将抽象策略转换为实现层策略“各个工程师对个人所在组织目录下的资源具有存取权限”(c3)。该技术提出了策略冲突分析需求，但是没有给出具体方法。

```
template(t3,
[[ c0, keywords, [$engineer$, $information$, $organisation$ ]],
[ c1, category, $Access to Information$],
[ c2, abstract, $All engineers can perform operations on information within their organisation$],
[ c3, description, $Users that are Engineers can perform operations on \r\n information that belong to\r\n the same Organisation they belong to]. $],
[ c4, expiration-date, $01/01/1999$],
[ c5, deployable, $deployable$],
[ c6, start, c7],
[ c7, sequence, [c8, c12, c13, c16, c18]],
[ c8, context, [internal: [and([belongsTo(information,orgUnit(U)),
    isMember(user(Un,Uid), engineer),
    isMember(user(Un,Uid), orgUnit(U))])],
    refinementBy: [[information,c10],[orgUnit(U),c10]]]],
[ c10, refinementDetails, [category: ism,
    Condition: [],
    refinementBy: [class]]],
[ c12, policyStatement, [category: deployable,
    internal: [and([canAccess(user(Un,UId), operation, information)])],
    Condition: [],
    refinementBy: [[user(Un,Uid),c10],[information,c10]]]],
[ c13, classRefinementChoice, [class: [orgUnit(U),c10]]],
[ c16, constraintChoice, [constraint:[and([about(information,user(Un,UId))])],
    Choices:[accept:c18, ignore:c18]]],
[ c18, end, []] ]).
```

图1 策略精化模板示例

基于模板精化技术的发展，细化策略和系统分层，丰富策略描述，提供基于状态和基于事件策略的精化，精化参数包括时序(until,after,…)、状态(isNotIn,isCombinedWith,isOnlyIn,…)、事件(attemCopy,never do,…)以及更多网络平台参数[13-20]。

Prachi Kumari等[18]将系统分成3个抽象层次：PIM层(platform-independent layer)，描述平台无关的系统计算；PSM层(platform-specific layer)，描述平台相关的系统计算，是平台状态和行为共性的描述，不是特定平台状态和行为描述；ISM层(implementation-specific layer)，描述在实际平台的系统计算。图2所示为一个访问控制策略映射模板。PIM层基于事件或基于状态的抽象策略，根据PSM层平台机制等参数，精化为PSM层相应对象的约束和操作，再根据ISM层特定平台技术参数，精化为ISM层特定对象的约束和操作。

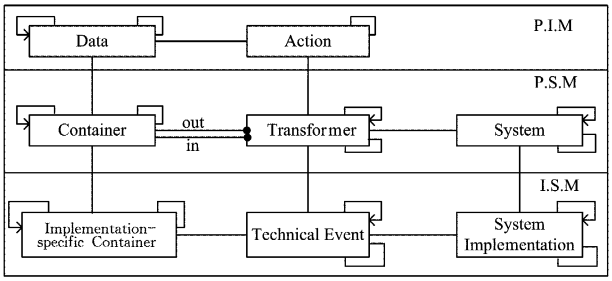



图2 各层策略模型和层间映射

图3是一个基于图2策略映射模板的策略精化实例：PIM层策略为：“禁止Song散发”；在PSM层Song对应操作系统文件song.mp3，散发对应由Socket输出，因此“禁止Song散发”精化为PSM层策略“禁止song.mp3由Socket输出”；在ISM层操作系统为OpenBSD，Socket名为Cnet，Socket输出对Cnet的Write系统调用，因此“禁止song.mp3由Socket输出”精化为ISM层策略“禁止通过Cnet对文件song.mp3的Write系统调用”。当任何事件通过Cnet试图触发对文件song.mp3的Write系统调用时，该策略禁止其执行。该技术也不具有策略冲突分析能力。

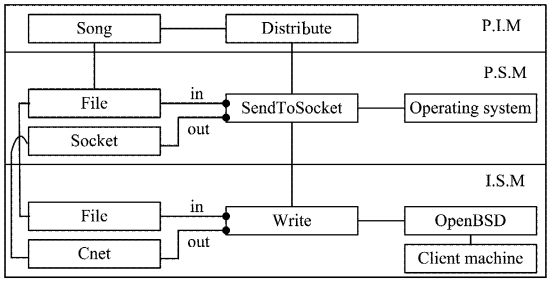



图 3　各层策略模型和层间映射实例

基于模板的精化自然呈现策略的 SLA 关系。但是随着安全问题日益严重，网络平台纵深防御必须考虑。由于网络和其中的安全监控技术具有多样性、变化性、复杂性，很难以固定的模式描述系统、对象的属性和结构，因此难以发展静态的基于模板方法，使其具备访问控制纵深防御能力。同时，目前基于模板的精化技术也缺少策略冲突和关联属性分析能力。

### 2.2　基于 RBAC 对象结构关系的精化

RBAC 模型中角色的隶属关系，也是推导隶属角色主体权限的规则[21,22]。

Karthick Jayaraman 等[22]以一阶逻辑描述 RBAC 模型策略，Users、UA、Roles 分别代表用户、用户隶属角色、角色，URA(见图 4)描述角色的约束规则，也是应用的元规则，其中 can_assign$\{\langle r_a, c, r_t\rangle, \cdots\}$表示如果用户属于角色 $r_a$ 且满足条件 $c$，则可以赋于目标角色 $r_t$，can_revoke$\{\langle r_a, r_t\rangle, \cdots\}$表示如果用户属于角色 $r_a$，则可以撤消其他用户的 $r_t$ 角色。

图 5 为一个应用的用户、用户隶属角色、can_assign($ca$)、can_revoke($cr$)规则，图 6 为该应用的角色隶属关系及精化计算优先顺序，精化计算依据图 5 所示规则和图 6 角色关系自上而下顺序计算用户权限，计算结果对比应用元规则，Refinement 2 计算发现 Finance 同时被赋予 Acct 和 Audit 角色(〈Admin，Acct∧ Audit，Finance〉，如表 1 中标有下划线部分所示)，与图 4 应用元规则 can_assign {〈Admin，Acct∧ ¬Audit，Finance〉}不符，需要修正。

can_assign{〈Admin，Finance，BudgetCommitee〉，
〈Admin，Acct∧ ；Audit，Finance〉，
〈Admin，TRUE，Accti，hAdmin，TURE，Audit〉，
〈Admin，TechSupport，IT〉，
〈Admin，TRUE，TechSupport〉}；
can_revoke{〈Admin，Accti，hAdmin，Audit〉，
〈Admin，TechSupport〉}；

图 4　系统策略约束规则

Roles BudgetCommitee Finance Acct Audit
TechSupport IT Admin；
Users Alice Bob；
UA〈Alice，Admin〉〈Bob，Acct〉〈Bob，Audit〉；
CR 〈Admin，Acct〉〈Admin，Audit〉
〈Admin，TechSupport〉；
CA 〈Admin，，Finance，BudgetCommittee〉
〈Admin，Acct&Audit，Finance〉〈Admin，TRUE，Acct〉〈Admin，TURE，Audit〉〈Admin，TechSupport，IT〉〈Admin，True，TechSupport〉；

图 5　系统策略赋值

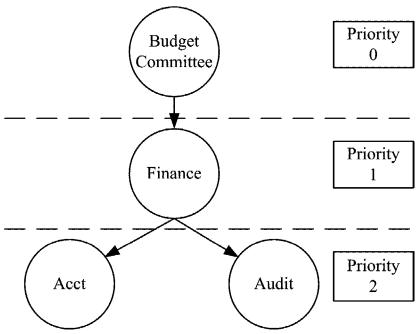



图 6　系统 RBAC 角色层次

表 1　策略精化及冲突分析数据

| Steps | Users | Roles | UA | can_assign | can_revoke | Result |
|---|---|---|---|---|---|---|
| Abstractionstep | Alice，Bob | BudgetCommittee，Admin | (Alice，Admin) | | | Nocounterexample |
| Refinement 1 | Alice，Bob | BudgetCommittee，Admin，Finance | (Alice，Admin) | 〈Admin，Finance，BudgetCommittee，〉 | | Nocounterexample |
| Refinement 2 | Alice，Bob | BudgetCommittee，Admin，Finance，Acct，Audit | (Alice，Admin)，(Bob，Acct)，(Bob，Audit) | 〈Admin，Finance，BudgetCommittee，〉〈Admin，Acct∧ Audit，Finance〉〈Admin，TRUE，Acct〉〈Admin，TRUE，Audit〉 | 〈Admin，Acct〉〈Admin，Audit〉 | Counterexample found |

该技术具有策略冲突分析能力，保证策略之间满足互斥约束，保证精化完全性和一致性，也可以呈现不同层次策略的 SLA 关系，但是不具备系统分层属性和结构描述能力，不能描述和推导网络平台访问路径安全监控相关策略。

### 2.3　基于系统和策略的分层模型及层间映射规则的精化

基于系统和策略分层模型及层间映射规则的精化包括：(1)各层的系统模型描述；(2)上层系统对象和系统模型到下层系统对象和系统模型的映射规则；(3)各层策略模型描述；(4)依据各层系统对象、模型及其映射规则，将上层策略精化为下层策略的规则。

Butler Lampson[23]分析分布式环境访问路径相关的认证、访问控制、通信等安全监控中的相应策略配置。在此基础之上，Maullo[24]设计一阶逻辑系统，根据策略、网络拓扑和安全监控参数，推导网络访问路径上相关的认证、访问控制、通信等安全监控中的相应配置策略。F. Barrère 等[25]将上述逻辑分析系统用于推导 RBAC 策略在网络访问路径安全监控中相应的配置策略，具有路径策略的角色冲突分析能力。比如：

$EF$ 表示网络接口集合，$U$ 表示用户集合；

$\forall ef \in EF, u \in U$，Is_Connected$(u, ef) \Rightarrow Role(u) = Role(ef)$；

$C$ 是数据通道集合，通道 $c \in C$，角色 $r \in R$ 对通道 $c$ 可信即 $trust(c, r)$，当且仅当 $\forall ef \in EF, active\_EF(ef) \wedge (r' = Role(ef)) \wedge$ is_connected$(ef, c) \wedge \neg(r$ is_in_conflict $r')$。

上述技术基于上下层系统的映射，可以推导网络路径访问控制单元相关策略，分析策略冲突，但是缺少针对实际网络

平台的系统、对象以及策略的分层描述与映射计算方法，没有策略之间的关联属性分析能力，也不具备呈现访问控制 SLA 的能力。

S. Davy 等[26-30]设计的精化技术，以本体描述系统各层结构、对象以及策略模型，精化基于事件的访问控制策略，策略由{事件(E)，条件(C)，行为(A)，主体(S)，客体(T)}描述。

增加或改变的策略触发策略精化与冲突分析。如图 7 所示，上层策略根据上下层主客体对象的映射，精化生成下层策略，精化计算并不推导下层访问路径安全监控中的相关策略。

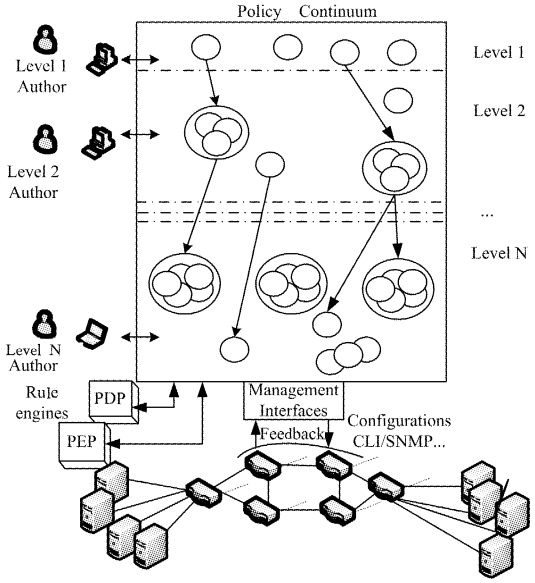



图 7 策略层间精化计算

冲突分析由增加或改变的策略依次与原先策略集合中的任意一条，分别计算两个策略主体、客体、事件、条件的相关性，看其是属于、包含、交叠还是无关，根据相关性判断这两个策略的行为是否冲突。

下层策略经过冲突判断和消解之后，保留的策略根据精化关系，搜索其上层对应策略，没有下层对应精化策略的上层策略被取消，这样保证精化计算的完全性。

经过精化计算得到的实现层策略，再根据策略的源和目的地址以及系统本体信息模型中的安全监控分布描述，推导实现层访问路径安全监控中的相关策略；接着根据路径相关策略之间的冲突类型，判断、消解路径相关策略之间的冲突，保证路径相关策略的一致性。

比如图 8 是一个网络信息系统。

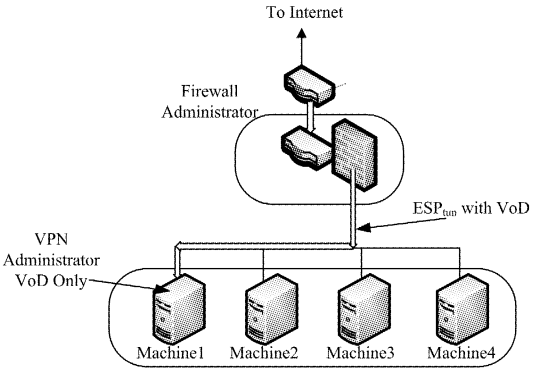



图 8 一个系统实现层网络拓扑图

实现层访问路径监控有防火墙和 IPsec 技术，该访问路径安全监控相关策略及关系信息记录如下：

policy⊆SecurityPolicy；

SecurityPolicy⊆IPsecPolicy⋃FirewallPolicy；

IPsecPolicy⊆IPsecTunPolicy⋃IPsecTraPolicy。

表 2、表 3 是其中具体的防火墙策略(FirewallPolicy)和 IPsec 策略(IPsec Policy)。

表 2 实现层防火墙策略

| ID | SrcIP | DestIP | DstPrt | Action |
|---|---|---|---|---|
| 1 | Machine1 | InternetIP1 | VoD | Drop |
| 2 | Machine2 | InternetIP2 | HTTP | Allow |
| 3 | Machine3 | InternetIP2 | IPsec | Allow |

表 3 实现层 IPSec 策略

| ID | SrcIP | DestIP | DstPrt | Action |
|---|---|---|---|---|
| 1 | Machine1 | InternetIP1 | * | ESPtun |
| 2 | Machine2 | InternetIP2 | * | AHtun |
| 3 | Machine3 | InternetIP2 | * | AHtra |

访问路径相关策略之间冲突定义有：

(1)disjointFrom(FirewallPolicy，IPsecTunPolicy)；

(2)disjointFrom($ESP_{Tun}$，Drop)；

(3)disjointFrom($ESPAH_{Tun}$，Drop)。

假设系统管理员要增加表 2 防火墙 ID 为 1 的策略，假设此时其它策略为系统原有策略，通过分析策略源和目的地址，发现该防火墙路径相关策略为表 3 中 ID 为 1 的 IPsec 策略，根据访问路径相关策略之间冲突定义(3)，判定这两个策略形成冲突，需要消解其中一个策略，保证路径策略一致性。

该技术提出了分析精化完整性、实现路径策略一致性的方法，采用本体信息模型呈现策略 SLA 关系，但是存在以下缺点：

(1)实际网络平台常常存在不止一条访问路径，一条路径不能通信和所有路径均不能通信对于策略精化的意义完全不同，前者是精化可以实现，而后者是精化失败，上层策略应该被取消。该技术实现层访问路径相关策略以集合形式描述，缺少策略之间路径关系的描述和计算，不能处理不止一条访问路径情况下的路径相关策略一致性。

(2)本文分析防火墙、IPSec/VPN 的路径相关策略一致性，而防火墙、IPSec/VPN 可能形成多种类型策略冲突，不同类型冲突分析与消解有顺序要求，例如先进行隧道冲突分析与消解，明确需要构建的隧道，才能进一步判断隧道构建是否与通信端口禁止加密数据通过策略冲突[31,32]，需要将策略集合整体按一定冲突类型顺序分析，才能形成稳定一致的结果。

该技术路径策略一致性分析不是以策略集合按冲突类型顺序分析，而是将变化或新增策略依次与原有策略集合中的一个策略相结合来分析策略冲突和路径相关策略一致性，一条策略分析完再与下一条策略分析，这样后继策略冲突分析可能改变先前分析确定的策略和其访问路径相关策略，比如改变先前隧道的配置策略，引发新的通信端口禁止加密数据通过的策略冲突，无法形成稳定的计算结果。

(3)只有实现层有访问路径策略一致性分析，事实上除实现层外，系统计算相关层也可能存在路径相关的安全约束，比如计算相关层的计算系统要求访问邮件服务器时首先需要通过 Web 认证，那么 Web 认证和邮件服务器认证策略需要保证一致性才能实现邮件的安全访问，此时访问路径策略一致性如果同样采用实现层访问路径策略一致性分析方法，则其

精化完全性又不能保证。

(4)不能描述应用策略之间组合、互斥约束,也不能保证策略经过精化后满足策略之间组合与互斥约束。

Ingo Lück 等[33,34]设计了服务层 RBAC 策略到实现层策略的精化技术,该技术系统分为 3 个层次描述,系统分层与 OMG 组织 MDA 模型[35]层次相对应:Roles&Object 层(RO 层)对应计算无关模型(Computational Independent Model, CIM),Subject&Resources 层(SR 层)对应平台无关模型(Platform Independent Model, PIM),Processes&Hosts 层(PH 层)对应平台相关模型(Platform Specific Model,PSM)。

该技术具有推导网络访问路径相关安全监控策略的能力。在此研究基础之上,为了适应大型网络的策略精化,减少计算复杂性,Jo~ao Porto de Albuquerque 等[10]将 PH 层再分为两子层:AS 层和 ES 层。AS 层主要描述原先 PH 层的网络拓扑结构,ES 层与 AS 层结点一一对应,描述 AS 层网络结点的具体参数,包括进程号、身份、结点地址、接口等。

表 4 是该技术系统各层策略对象、策略及其含义,同列上下层间具有映射关系。

表 4　系统各层策略、策略对象

| 层次 | | 策略模型 | | | 策略含义 |
|---|---|---|---|---|---|
| | | 主体 | 权限 | 客体 | |
| RO(CIM) | | Role | AccessMode | Object | Role 对 Object 拥有 AccessMode 权限 |
| SR(PIM) | | (User Subject) | Services;ServiceDependency | Resources | (User Subject)对 Resources 拥有访问路径(安全监控)Services(上的相应)权限,以及路径(相关安全监控)ServiceDependency(上的相应)权限。访问路径指访问计算的路径(比如 FTP 或 Web 等) |
| PH (PSM) | AS | Actor | Mediator | Target | Actor 对 Target 拥有物理层网络拓扑路径(必经安全监控)Mediator(上的相应)权限 |
| | ES | (UserCredential,Processes, Host,NetworkConnections) | (Processes,Hosts,Network-Connections) | (Processes/Objects,Hosts, NetworkConnections) | (UserCredential, Processes, Host, Network-Connections) 对 (Processes/Objects, Hosts, NetworkConnections)拥有物理层网络拓扑路径(必经安全技术监控)(Processes,Hosts, NetworkConnections)(上的相应)权限 |

图 9 给出一个应用系统分层模型及其层间映射关系。RO 层策略:“雇员(包括 Tom 和 Joe)允许访问(view)公司 Intranetweb”,RO 层 Intranetweb 服务对应 SR 层体 Intranetwebpages,SR 层雇员可以在办公室通过 Web 服务访问 Intranetwebpages,所以策略精化为“Tom、Joe 允许在办公室通过公司的 Web 服务访问 Intranetwebpages”;同样,SR 层体 Intranetwebpages 对应 PH 层 Server1 和 Server2 上的 InternalWebPages,根据安全等级访问 InternalWebPages 需要 VPN 安全服务,对 Server1 访问的路径 1,安全监控为路由器 R1 和 R2;对 Server2 访问的路径 2,安全监控为路由器 R1、R3,所以对应的 PH 层精化策略包括:“路由器 R1 与 R2 之间为代表 Tom、Joe 的(IP 地址、进程号、httpd、端口号)与代表 Server1 上文件 InternalWebPages 的(httpd,Interface,IP Address)的通信配置 VPN 安全服务”,以及“路由器 R1、R3 之间为代表 Tom、Joe 的(IP 地址、进程号、httpd、端口号)与代表 Server2 上文件 InternalWebPages 的(httpd,Interface,IP Address)的通信配置 VPN 服务”。

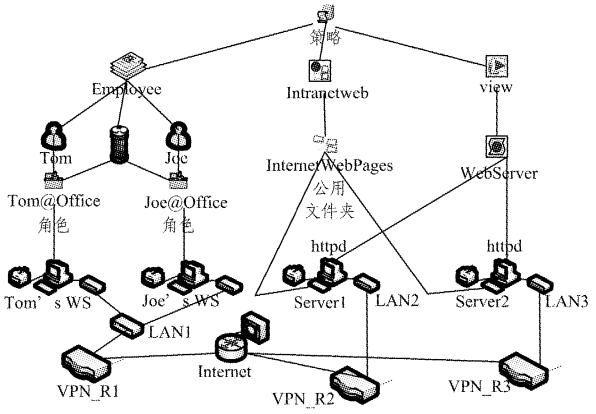



图 9　系统 MDA 结构和映射

该技术面向服务计算,具有清晰的系统、对象和策略分层描述和映射规则,可以灵活计算网络路径,推导路径安全监控相关策略,具有一定的策略冲突分析能力,但是存在以下不足:

(1)精化计算中如果出现策略冲突,冲突解决总是由上层精化而来的策略取代本层策略,以此保证精化的完全性和一致性,但是这样解决策略冲突有时是不合适的,因为访问控制决策基于对资源使用者的认识和了解,资源使用者安全相关行为和状态知识常常分散于下层系统功能单元中,上层决策者无法预知所有安全相关知识,所以不能保证制定的策略始终正确合理。比如,上层策略规定用户 A 允许写资源 B,而下层的 IDS 发现 A 的行为有危险性,提出禁止用户 A 读写资源 B,IDS 策略与上层精化策略冲突,因此对于实际情况,应根据动态知识采用 IDS 策略,而不是上层的策略。

(2)不具备描述应用策略组合、互斥约束能力,不能保证策略精化后满足组合、互斥安全约束,不能分析访问路径上相关策略协同一致性。

(3)没有排序处理不同类型策略冲突的能力,不能处理像 IPSec 这样需要有序处理不同类型冲突的情况。

(4)在 CIM 层策略向 PIM 层精化时,只能描述和计算单一访问计算方式,而实际应用系统有时存在多种访问计算方式。

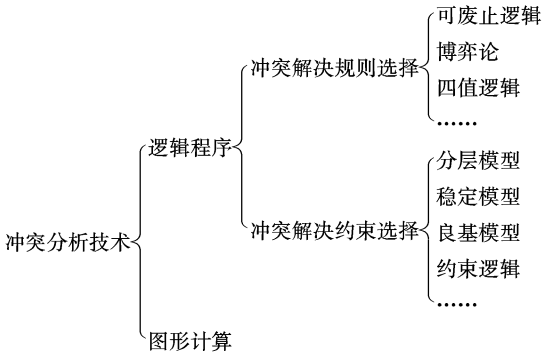
(5)不具备呈现抽象策略到相应配置策略的 SLA 对应关系能力。

上述精化技术有些具有明确的系统和策略分层描述,有些具有推导访问路径控制单元相关策略的能力,但是普遍存在策略冲突分析中策略关联分析能力不足的问题,不能根据策略冲突消解分析修正策略关联失效。而策略冲突与关联属性分析决定精化技术的实际能力。因此本文进一步分析现有策略冲突分析技术的冲突与关联属性分析能力以及它们的分

布式应用适用性和计算可扩展性，寻求可以提高策略精化的冲突与关联属性分析能力的方法。

## 3 访问控制策略冲突分析技术

策略冲突分析与策略描述关系密切，策略描述从根本上影响冲突分析计算，策略冲突分析技术根据策略描述一般分为逻辑程序计算方法和图形规则计算方法两类。




### 3.1 逻辑程序

逻辑程序描述主体对客体的权限，如果存在多个逻辑表达式对某一主体或客体的权限赋值，则可能形成策略冲突。根据规则参数或类型选择唯一规则，计算主体对客体的权限，称为冲突解决的规则选择方法。计算不同规则主体对客体的权限，对于冲突的权限，根据权限优先级确定主体对客体的唯一权限，称为冲突解决的约束选择方法。逻辑程序冲突分析技术根据冲突解决方法又可以分为规则选择和约束选择两类。

#### 3.1.1 冲突解决的规则选择

冲突解决的规则选择方法常见于非经典逻辑计算，有可废止逻辑、博弈论、多值逻辑等。

(1)可废止逻辑

一些学者提出利用可废止逻辑理论，将策略分为不同类型规则，为不同类型规则分配不同的优先级，通过规则优先级，选择策略计算权限，有效解决冲突[36-39]。一个可废止逻辑系统包括：事实(facts)、硬性规则(strict rules)、可废止规则(defeasible rules)、废止者(defeaters)和规则上的优先关系(superiority relation)。比如下述策略：

P3：Allow up to 3 purchases per day.

PE：Guarantee payment to emergency services twice.

Pcc：A cash card；spend no more than $500 total.

PN：No alcohol can be purchased.

Pt：Prevent purchases of prescription drugs which conflict with the anti-depressant Tofranil.

硬性规则与传统的规则意义相同，即只要前提是不可辩驳的，那么结论也是如此，例如 PE；可废止规则是指那些可以被反面证据废止的规则，例如 P3，意指一般情况下可以购买 3 次(药)，除非有证据表明其不可使用；废止者是指那些不能用于得出任何结论，只用于防止某些结论发生的规则，例如 Pcc；规则之间通过“优先关系”定义规则优先级，即某条规则可以推翻另一条规则的结论，优先关系必须是非循环的，例如规则 Pcc 与 P3 相互冲突，如果 Pcc≻P3，“≻”表示优先关系，那么，我们得出不可以购买的结论。

(2)博弈论

访问控制策略选择基于对资源使用者行为、状态的分析和认识，而这些信息常常是动态变化并分散于系统管理者、个体资源拥有者以及功能单元之中，难于有及时、集中的知识用于决策。为此 Guido Boella 等[40-43]设计分布式系统 agent 功能单元负责制定、选择、申请策略，agent 类型包括系统策略制定者 agent、资源提供者 agent、资源使用者 agent。

策略规则基于认知哲学研究，它将权限表示为关于信念(表示已知信息)、职责(表示依据信念、意图和愿望规则生成的执行准则)、意图(表示实现目标的行为)、愿望(表示长远目标)的规则(逻辑理论为缺省逻辑)。假设 $W$ 表示规则文字，$s$ 表示拒绝和否定行为，$x$ 是 $a$ 对 $b$ 的资源请求，$Y$ 表示 $c$ 对 $a$ 的知识信息；$Va(x)$表示 $a$ 的 $x$ 行为是违规行为；$Vb(Va(x))$ 表示如果 $b$ 认为 $a$ 的 $x$ 行为是违规行为，则 $b$ 的行为是违规行为；$MD(Dc)$表示 $c$ 的愿望，$MD(Gc)$表示 $c$ 的目标。权限规则表示如下：

①$W\wedge Y\wedge x\rightarrow\neg Va(x)\in MD(Dc)\cap MD(Gc)$；

②$W\wedge Y\wedge x\wedge Va(x)\rightarrow Vb(Va(x))\in MD(Dc)\cap MD(Gc)$；

③$T\rightarrow\neg Vb(Va(x))\in MD(Dc)$；

④$W\wedge Y\wedge Vb(Va(x))\rightarrow s\in MD(Dc)\cap MD(Gc)$；

⑤$W\wedge Y\rightarrow\sim s\in MD(Dc)$；

⑥$W\wedge Y\rightarrow\sim s\in MD(Db)$。

每个 agent 基于对策略主体的认知提出各自授权策略。各个 agent 将自己和其它 agent 的策略进行博弈计算，根据计算结果和策略优先级决定自己的实际授权策略。

比如主体 $a$ 通过自己的策略 agent 提出读 $b$ 的资源 $y$ 的请求，全局 agent 授予 $a$ 读 $b$ 的资源 $y$ 权限。$c$ 为资源提供者 $b$ 的策略 agent，$c$ 不仅根据全局策略制定者的策略决定是否授权，还根据自己的对资源使用者 $a$ 行为知识提出资源请求者的权限策略。$c$ 接收全局 agent 授权 $a$ 读 $b$ 的资源 $y$ 权限的策略，策略为$\neg Va(read(y))\in MD(Dc)\cap MD(Gc)$(规则 1)，此策略触发 $c$ 检查有关 $a$ 行为知识，经过检查认定 $a$ 存在可疑的黑客行为，授予 $a$ 读 $b$ 的资源 $y$ 权限会带来危险($T\rightarrow hack(a)\in B_p$)，所以 $c$ 提出策略：$hack(a)\rightarrow\neg Vb(Va(read(y)))$(规则 3)，因为规则 3 优先级高于规则 1，$c$ 选择策略 $\neg Vb(Va(read(y)))$，拒绝 $a$ 访问 $b$ 的资源 $y$。

(3)四值逻辑

四值逻辑为不含命题常元的(四值)命题语言，包含逻辑连接词$\neg$、$\vee$、$\wedge$、$\rightarrow$、$\supset$。Belnap 的四值逻辑[43-45]用 FOUR $=\{\cdot,t,f,\cdot\}$ 表示真值集，4 个真值也相应记为{(0,0),(1,0),(0,1),(1,1)}，分别表示未知、真、假、矛盾 4 种情形。四值逻辑运算规则：

$(x_1,y_1)\wedge(x_2,y_2)=_{def}(x_1\wedge x_2,y_1\vee y_2)$；

$(x_1,y_1)\vee(x_2,y_2)=_{def}(x_1\vee x_2,y_1\wedge y_2)$；

$(x_1,y_1)\supset(x_2,y_2)=_{def}(\neg x_1\vee x_2,x_1\wedge y_2)$；

$(x_1,y_1)\rightarrow(x_2,y_2)=_{def}\neg(x_1,y_1)\vee(x_2,y_2)$。

Glenn Bruns[46]运用四值逻辑命题演算规则描述、计算访问控制策略(RBAC 和防火墙策略)权限，策略基本形式为 $b$ if $rp$，$b\in\{f,t\}$，$rp$ 为包含参数(主体，客体，行为)的谓词，比如：

$t$ if $subject=Doctor \wedge action=read \wedge object=patient\text{-}record$。

四值逻辑可以直接计算冲突策略并加以取舍，也可以通过排序选择授权策略，从而确定主体权限，比如两个策略：

$P_{doc}(role:=Physician)$：

$t$ if $role=Physician \wedge operation=prescribe$；

$P_{doc}(role:=Surgeon)$：

$F$ if $role=Surgeon \wedge operation=prescribe \wedge object=coughMedicine$；

角色 $Surgeon$<角色 $Physician$，所以 $P_{doc}(role:=Surgeon)>P_{doc}(role:=Physician)=f>t=f$。

3.1.2 冲突解决的约束选择

冲突解决的约束选择，以逻辑规则表示约束的优先级参与权限计算，得到一致的结果。比如否定优先可以表示为：$P(s,o,+a) \wedge P(s,o,-a) \rightarrow P(s,o,-a)$。根据策略描述内容又可以分为两类：不带属性限定策略，其只描述主体、客体和权限；带属性限定策略，其描述包括主体、客体、权限以及时序、事件状态等的限定属性。

(1)不带属性限定策略

这类策略冲突分析主要基于一阶逻辑理论。一阶逻辑程序计算基于 Horn 子句：$A \leftarrow L_1, L_2, \cdots, L_n$，其中 $A$ 称为该规则的“头”，而诸文字 $L_i$ 合称为该规则的“体”。如果一个规则的规则体中不出现负文字，则称该规则为“正的”；如果一个程序的所有规则都是正的，则称该程序为“正的”。每个正逻辑程序有唯一的最小 Herbrand 模型 Mp[47] 作为解。一阶逻辑程序对负文字计算能力的拓展，也拓展了逻辑程序描述和分析策略的能力。含有负文字的逻辑程序计算，其计算结果与选择的解释模型有关，主要有分层模型、稳定模型和良基模型。

a)分层(或局部分层)模型

一个逻辑程序 $P$ 是分层的(或局部分层的)，如果可以在所有谓词符号集上得到一个划分，满足：

①子句的负前提具有高于子句头的优先级；

②子句的正前提的优先级不低于子句头。

满足分层(或局部分层)属性的逻辑程序，在多项式时间内有唯一的理想模型作为解[48,49]。

Sushil Jajodia 等[50]设计的基于分层逻辑理论的策略分析技术中，策略主体和客体分别具有树型结构关系，祖先结点是子孙结点的集合，应用赋予部分树结点权限，基于主体或客体树结构和冲突解决规则计算叶结点主客体个体最终权限。

b)稳定模型

稳定模型[51]是对逻辑程序的一种化简，它消去逻辑程序 $\Pi$ 的所有负文字，将其变换为一个正程序 $\Pi M$，使得 $\Pi$ 的 Herbrand 模型都是 $\Pi M$ 的 Herbrand 模型，由 $\Pi M$ 的唯一极小 Herbrand 模型得到了 $\Pi$ 的一个极小 Herbrand 模型。有些程序没有稳定模型；有些程序只有空稳定模型；有些程序有多个稳定模型。

Bertino 等[52]基于稳定模型理论分析策略冲突，计算主客体权限。与文献[49]相似，策略主体和客体分别具有树型结构关系，祖先结点是子孙结点的集合，应用赋予部分树结点权限，基于树结构推导叶结点主客体个体最终权限。不同之处在于稳定模型计算没有谓词计算顺序的限制，还可以用于(中国墙等)动态互斥选择的策略计算，比如互斥权限策略：要么只有 Ann 可以存取对象 $O$，要么只有 Bob 可以存取对象 $O$，不同策略选择可以生成不同的一致策略集合。

c)良基模型

良基模型[53]同样没有谓词计算顺序的限制。Bertino 等[52]设计的冲突计算也可以选择良基模型作为权限的解集。良基模型计算的意义在于：首先，有些逻辑程序不一定存在稳定模型，但是任意一个逻辑程序总存在唯一良基模型；其次，良基模型的解计算只是将已知为真的事实取为真，是关于可能世界的最保守的选择，对访问控制权限计算来说也是一种保守选择。

(2)带属性限定策略

XACML、Ponder[54]等策略描述语言，包含状态、事件等属性约束条件，策略分析也有所不同[55-57]。

a)约束逻辑

约束逻辑程序设计(Constraint Logic Programming，简称 CLP)，将 Herbrand 域上的约束求解扩充为多项式约束的论域上的求解，通过多项式缩减论域可能解范围，规则体中可以包含否定谓词。

Steve Barker 和 Peter J. Stuckey[55]采用约束逻辑分析 RBAC 模型策略，以时间作为论域的约束选择，时间约束计算包括：=、+、−、*、÷。比如，从 2001/03/01 到 2002/04/01，$r_2$ 在工作日(except Thursday)具有 write $o_1$ 的权限，表示为：

pra(write, $o_1$, $r_2$, $T$) ← 20010301 ≤ $T$, $T$ ≤ 20020401, month($T$,$M$), $M$ 6=12, weekday($T$,$W$), 1≤$W$, $W$≤5, $W$≠4

b)基于事件演算

Arosha K Bandara 等[56]设计的基于事件演算(Event Calculus)的策略描述与分析，采用一阶逻辑描述基于事件(时序)的策略规则、冲突及冲突解决，策略着眼于事件引起的系统状态变化，以及不同状态对应的主客体权限，具有否定的计算功能。比如：主体 $Subj$ 在时间 $Tm$ 时被允许执行 Action(ParmList)，如果 $Tm<Tn$，那么主体 $Subj$ 在时间 $Tn$ 时产生行为 Action(ParmList)，策略表示如下：

Happens(doAction($Subj$, operation(Targ, Action(ParmList))), $Tn$) ← holdsAt (permit ($Subj$, operation (Targ, Action (ParmList))), $Tm$) ∧ ($Tm<Tn$)

缺省逻辑和博弈计算描述基于状态和知识的权限选择，难以发展用于分层精化计算，因为(1)现有分层系统缺少状态和知识的规范描述以及分层描述的层间映射；(2)如果扩展逻辑分析系统，引入基于状态和知识的精化推导规则，那么规则多难以排序。相比状态和知识，约束内容要少，排序比较容易。基于事件的演算也存在同样的困难。

分层模型、稳定模型和良基模型的权限解释，为决策提供了多种选择，虽然保证叶结点(单个主体或客体)权限没有冲突，但是不能保证叶结点与祖先结点策略一致，如果用于精化计算，不能计算精化完整性，也不具备策略之间其它关联属性分析的能力；而且这类技术冲突分析大多基于主体和客体稳

定的树型关系，而分布式应用系统策略主体之间、客体之间的关系是松散的，并不具有稳定的树型结构关系，所以分布式应用适应性不足。

约束逻辑虽然具有策略（角色）互斥分析能力，但是策略分析基于角色稳定的树型关系，同样存在分布式应用适应性不足的问题。

另外，一个主体访问一个客体，策略精化对于同一访问路径上的相关安全控制单元，主体必须同时具有各个访问控制单元的相应权限才能实现对客体的访问，任何一个权限不成立，访问都不能实现，这可以用“与”关系表示；对于不同访问路径上的策略集合，主体拥有任何一个路径权限集合就可以实现对客体的访问，主体所有路径访问客体均失效，则访问失效，路径策略集合之间的关系可以用“或”表示。如果采用逻辑程序分析这些策略关系属性，策略可能是负文字，文字之间既有“与”关系，又有“或”关系，逻辑程序难以扩展来保证策略分析的可计算性。四值逻辑计算虽然可以计算命题的“与”“或”关系，但是没有谓词的推导计算，难以描述并计算精化推导规则。

再有，由于逻辑程序 Horn 句式的限制，策略之间的关系信息在逻辑计算中丢失，不能呈现策略精化的 SLA 关系。

所以总的来说，逻辑程序冲突分析方法难以扩展来解决精化计算中关联属性的分析与修正问题。

### 3.2 图形计算

一些学者采用图形规则描述主体对客体的权限策略[58-63]。Trent Jaeger[58,59]的访问控制权限空间冲突分析方法比较有代表性。图 10 是一个主体 s1(Subject s1)权限(permission)图形描述示例，permission 的可能值有 prohibited、obligated、permissible、unknown 等，Role r1 与 Role r2 具有互斥关系。图 11 定义一个主体单元对所有客体权限之间应该具有的空间关系，但是实际策略赋值情况可能出现图 12 阴影部分的情况，即一个主体单元对客体的权限多于一种，权限空间交叠，有些形成冲突，比如阴影部分主体对某些客体同时出现禁止(Prohibited)和允许(Specified)权限，出现授权冲突。

根据图 10 计算得到图 13 所示的 s1 权限，发现 s1 同时 permissible(允许)、prohibited(禁止)对 p3 的权限，属于冲突的一种，需要按冲突消解规则消解。

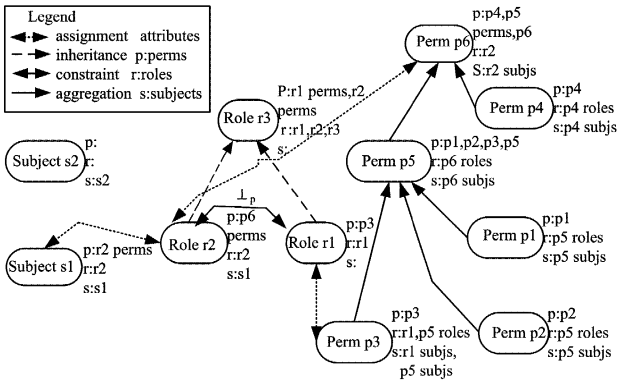



图 10 s1 的权限赋值图形函数

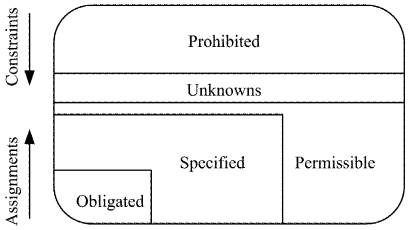



图 11 主体理想的权限空间关系

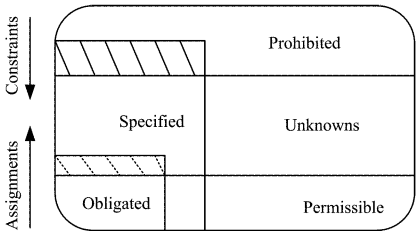



图 12 主体实际的权限分布空间

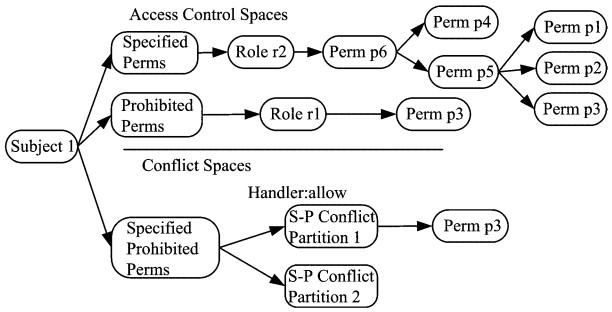



图 13 s1 根据图形赋值计算得到的权限

Giorgio Zanin 等[60]在 SecLinux 平台上以 SecLinux 的类型(type)为权限单元，并以 type 粒度分析主客体的权限及冲突。冲突分析原理与文献[58]相同。

文献[58-60]等的策略冲突分析技术，能直观描述并分析策略之间互斥关系属性，但是逐个主体或客体单元计算其权限、判断是否存在冲突，对于分布式应用而言动态适应能力不足。首先，分布式应用系统各部分策略往往没有统一的授权单元，另外策略往往描述一组主客体集合的权限，即 $p(S,O,a)$形式，其中 $S$ 是主体的集合，$O$ 是客体的集合，$A$ 是权限或约束的集合，$s\in S, o\in O, a\in A$ 是一个权限，$p(S,O,a)=\forall(i\in N)p(s_i,o_i,a)$，其中 $N=\{1,2,\cdots,n\}$，$SO=\{(s_i,o_i)\mid i\in N\}$ 是 $S\times O$ 的子集。逐个计算主体或客体单元的权限显得粒度过细，影响计算性能。

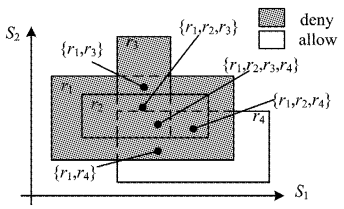
Cataldo Basile 等[61]设计的策略冲突分析技术中，策略 $r=(c,a)$，其中 $c$ 为网络通信源和目的地址(主客体)，$a$ 为执行的操作，比如允许或拒绝通信等。

该技术根据条件 $c$ 相关性构建策略集合的有向无环图，图结点策略集合为条件 $c$ 和条件 $c$ 所有隶属后代的策略，结点的最终策略主客体为 $c$ 减去祖先结点的条件 $c'$、权限为 $c-c'$ 所属的策略集合中优先级最高的。

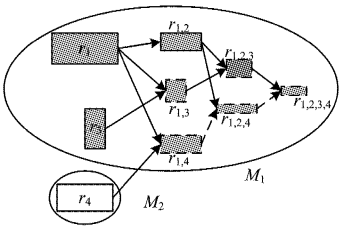
文献[61]认为最先匹配和最后匹配的优先级方式不足以表示实际的决策方式，策略优先级根据应用需要可能为基于策略的创建者、适用范围、创建时间、策略等级等，也可能是否定优先或肯定优先等诸多情况。文献[61]形式描述了多种策略优先排列方式并用于策略分析。比如策略 $p(x)$的最终权限赋值可以表示如下：

$$p(x)=\begin{cases} d, & \text{if } match_R(x)=\Phi \\ a_i, & \text{if } match_R(x)=\{r_i\} \\ \mathcal{R}(\{r_1,r_m,\cdots\}), & \text{if } match_R(x)=\{r_1,r_m,\cdots\} \end{cases}$$

它表示 $p(x)$没有规则匹配时缺省 $p(x)=d$，$p(x)$只有一个匹配规则 $r_i$ 时权限为该规则权限 $a_i$，$p(x)$有一组规则$\{r_1,r_m,\cdots\}$匹配时，根据优先级规则为：$\mathcal{R}(\{r_1,r_m,\cdots\})$，选择优先级最高的规则赋值权限。图 14 显示结点 $r_{1,4}$ 权限因规则优先级排序不同而不同。



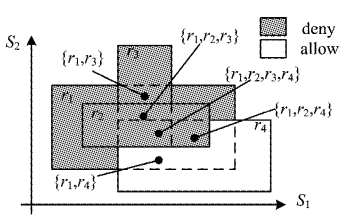

(a) 相关规则集合1



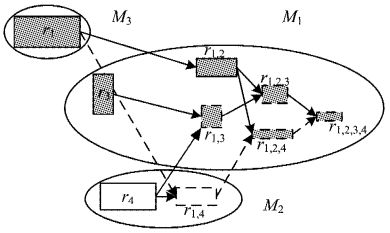

(b) 相关规则集合1的有向无环图

| Rule | Priority |
|---|---|
| $r_1$ | $\pi(r_1)=1$ |
| $r_3$ | $\pi(r_3)=2$ |
| $r_4$ | $\pi(r_4)=3$ |

(c) 相关规则集合1的规则优先级




(d) 相关规则集合2




(e) 相关规则集合2的有向无环图

| Rule | Priority |
|---|---|
| $r_{1,2}$ | $\pi(r_{1,2})=1$ |
| $r_3$ | $\pi(r_3)=2$ |
| $r_4$ | $\pi(r_4)=3$ |
| $r_1$ | $\pi(r_1)=4$ |

(f) 相关规则集合2的规则优先级

图 14

该技术有向无环图策略相关性计算是基于通信地址(条件 c 的形式)的，不适用于通常分布式策略主客体的描述形式，也不具备策略之间关联属性的分析能力。

姚键等[62]设计的策略冲突分析技术中，策略可以简单表示为三元组(*subject*，*target*，*action*)。采用有向无环图表示主体之间、客体之间的结构关系，$G^{\alpha}(V,E)$表示主体域用户、用户组之间的关系构成的有向无环图，其中 $V$ 表示用户组或用户，用户组间弧$(v_A,v_C)$表示 $v_C$ 是 $v_A$ 的成员，如果节点 $v_m$ 有两个父节点 $v_I$，$v_B$，表示 $v_m$ 是 $v_I$，$v_B$ 的共同成员，图中没有回路，表示不存在互为成员的关系，如图 15 所示。

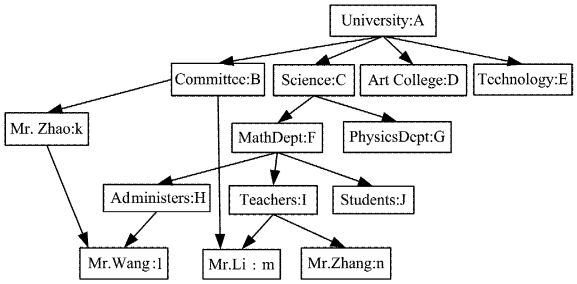



图 15 主体有向无环图示例

$G^T(V,E)$是分布式系统中的客体构成的有向无环图。同一资源可能同时属于多个集合，如图 16 所示。

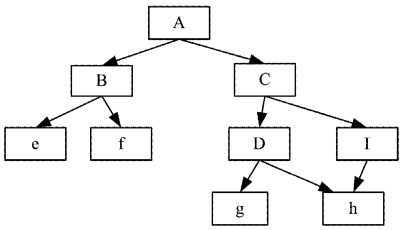



图 16　客体有向无环图示例

假设 $G^{\alpha}(V,E)$中的节点 $v_{A1}$，$S_1$ 和 $v_{A2}$，$S_2$ 分别表示策略 $A_1$ 中的主体 $S_1$ 和策略 $A_2$ 中的主体 $S_2$；$G^T(V,E)$中节点 $v_{A1}$，$T_1$ 和 $v_{A2}$，$T_2$ 分别表示策略 $A_1$ 中的客体 $T_1$ 和策略 $A_2$ 中的客体 $T_2$。在主体有向无环图 $G^{\alpha}(V,E)$上寻找一个能与节点 $v_{A1}$，$S_1$ 和 $v_{A2}$，$S_2$ 都连通的节点 $v\,S$，在客体有向无环图 $G^T(V,E)$上寻找一个能与节点 $v_{A1}$，$T_1$ 和 $v_{A2}$，$T_2$ 都连通的节点 $v\,T$，如果能分别找到这样两个节点，且策略 $A_1$ 与 $A_2$ 的 action 符号相反或存在语义冲突，则策略 $A_1$ 与 $A_2$ 存在冲突。角色冲突分析以角色结构作为主体，计算方法相同。

该技术冲突计算也是基于主体之间和客体之间稳定的规则的树型关系，分布式应用适应性不足，同时也缺少策略之间关联属性的描述和分析能力。

精化完整性和一致性、应用组合与互斥安全约束、纵深防御的路径相关策略一致性等精化需要分析的关联属性，可以通过“与”、“或”关系形式描述，虽然目前策略图形描述限于策略之间的互斥关系，但是“与”、“或”关系采用图形方式是可以描述的，并且基于图形描述的分析通常都是可计算的。所以规约并以图形方式描述策略和策略关联属性，设计适于分布式策略对象特点的高效冲突分析算法，设计基于策略关系分析、修正因冲突消解引起的策略关联失效技术，是提高策略精化计算能力的可能路径和研究方向。

**结束语**　访问控制是分布式应用系统安全的重要内容，策略精化是面向服务分布式应用访问控制策略配置的重要技术。

策略冲突和关联属性分析能力决定策略的实际精化能力。精化技术需要具有分析并修正因冲突消解引起的策略之间关联属性的能力。

另外，面向服务分布式计算的特点是资源弹性变化、环境动态迁移，而访问控制策略会因为资源、环境参数、平台拓扑结构(比如 IPSec 配置、客体的数量与位置等)变化而变化，因此需要与动态特征相适应的高效策略精化计算技术。

本文分析现有精化技术，包括系统和策略分层描述、推导

下层防御协同策略的方法、策略冲突分析与消解方法、策略精化完全性和一致性、纵深防御协同策略一致性、应用策略组合与互斥约束等策略之间的关联属性分析方法，发现现有技术策略关联属性分析能力不足。本文进一步分析现有策略冲突分析技术的冲突与关联属性分析、分布式应用适用性和计算可扩展性。

通过分析发现：规约并以图形方式描述策略及策略之间关联，提高策略冲突分析和消解效率，研究基于图形描述策略关系来分析、修正由冲突消解引起的策略关联属性失效，提供访问控制 SLA 呈现，是提高策略精化技术适应面向服务的分布式应用需要研究的问题。

## 参 考 文 献


[1] http://www.ibm.com/developerworks/webservices/library/ws-soa-design1/

[2] Sloman M. Policy Driven Management For Distributed Systems [J]. Journal of Network and Systems Management, 1994, 2(4)

[3] Zapthink. The SOA Management Landscape [OL]. http://www.zapthink.com/2006/11/30/the-soa-management-landscape/

[4] Wies R. Using a Classification of Management Policies for Policy Specification and Policy Transformation[C]// Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management. May 1995

[5] Maullo M J, Calo S B. Policy management: an architecture and approach[C] // Proceedings of the IEEE First International Workshop on Systems Management. 1993

[6] Pieters W, Dimkov T, Pavlovic D. Security Policy Alignment: A Formal Approach[J]. IEEE System Journal, 2012, 7(2): 275-287

[7] Karat J, Karat C-M, Bertino E, et al. A policy framework for security and privacy management[J]. IBM Journal Research & Development, 2009, 53(2): 4

[8] Phan T, et al. CA Survey of Policy-Based Management Approaches for Service Oriented Systems[C] // 19th Australian Conf. Software Eng. (ASWEC 2008). 2008

[9] Kamienski C, et al. Unleashing the power of policies for service-oriented computing [C] // Network and Service Management (CNSM). 2011

[10] de Albuquerque J P, Krumm H, de Geus P L, et al. Formal validation of automated policy refinement in the management of network security systems[J]. International Journal of Information Security, 2010, 9(2)

[11] The Open Group. SLA Management Handbook [M]. October 2004, 4

[12] Mont C, Baldwin M, Goh A, et al. POWER prototype: towards integrated policy-based management[C] // Network Operations and Management Symposium, IEEE/IFIP. 2000

[13] Menzel M, Meinel C. A Security Meta-Model for Service-oriented Architectures[C] // 2009 IEEE International Conference on Services Computing. 2009

[14] Lang U. OpenPMFSCaaS: Authorization as a Service for Cloud & SOA Applications[C] // 2nd IEEE International Conference on Cloud Computing Technology and Science. 2010

[15] Menzel M, Thomas I, Meinel C. Security Requirements Specification in Service-oriented Business Process Management[C] // 2009 International Conference on Availability, Reliability and Security. 2009

[16] Johnson M, Karat J, Karat C-M, et al. Usable Policy Template Authoring forIterative Policy Re? nement[C] // 2010 IEEE International Symposium on Policies for Distributed Systems and Networks. 2010

[17] Aziz B, Arenas A E, Wilson M. Model-Based Refinement of Security Policies in Collaborative Virtual Organisations[C] // 3rd Int. Symp., ESSoS. Berlin Heidelberg: Springer-Verlag, 2011

[18] Kumari P, Pretschner A. Deriving Implementation-level Policies-for Usage Control Enforcement[C] // CODASPY 2012 ACM. 2012

[19] Zhao Hang, Lobo J, Roy A, et al. Policy Refinement of Network Services for MANETs[C] // 12th IFIP/IEEE International Symposium on Integrated Network Management. 2011

[20] Su Lin-ying, et al. Automated Decomposition of Access Control Policies[C] // POLICY 2005 IEEE. 2005

[21] Basin D, et al. Model Driven Security: From UML Modelsto Access Control Infrastructures[J]. ACM Transactions on Software Engineering and Methodology, 2006, 15(1)

[22] Jayaraman K, Ganesh V, Tripunitara M, et al. Automatic Error Finding in Access-Control Policies[C] // CCS 2011 ACM. 2011

[23] 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

[24] Abadi M, Burrows M, Lampson B, et al. A Calculus for Access Control in Distributed Systems[J]. ACM Transactions on Programming Languages and Systems, 1993, 15(4)

[25] Barrère F, Benzekri A, Grasset F, et al. SPIDERNet: the Security Policy Derivation for Networks tool[C] // 3rd IEEE Latin America Network Operations and Management Symposium (LANOMS)

[26] Davy S, Jennings B. Harnessing Models for Policy Confiict Analysis[C] // Proc. Autonomous Infrastructure, Management and Security, AIMS. 2007

[27] Davy S, Jennings B, Strassner J. Application Domain Independent Policy Confiict Analysis Using Information Models[C] // Proc. IEEE/IFIP Network Operations and Management Symposium. 2008

[28] Barrett K, Davy S, Strassner J, et al. A Model Based Approach for Policy Tool Generation and Policy Analysis[C] // Proc. IEEE Global Information Infrastructure Symposium. 2007

[29] Davy S, Jennings B, Strassner J. Policy Confiict Prevention via Model-driven Policy Refinement[C] // Proc. of the 17th IFIP/IEEE Distributed Systems: Operations and Management, DSOM. 2006

[30] Davy S, Jennings B, Strassner J. On Harnessing Information Models and Ontologies for Policy Confiict Analysis[C] // Integrated Network Management. IM'09. IFIP/IEEE International Symposium. 2009

[31] Fu Zhi, Wu Fe-lix. Automatic Generation of IPSec/VPN Securi-

ty Policies In an Intra-Domain Environment[C]// 12th International Worshop on Distributed Systems. 2001

[32] Fu Z, Wu S F, Huang H, et al. IPSec/VPN Security Policy: Correctness, Conflict Detection and Resolution[C]// Proceedings of IEEE Policy 2001 Workshop. 2001

[33] Lück I, et al. Model-Based Tool-Assistance for Packet-Filter Design[C]// POLICY 2001. 2001

[34] Lück I, Vögel S, Krumm H. Model-Based Configuration of VPNs[C]// Network Operations and Management Symposium. 2002

[35] OMG. MDA Guide Version 1.0.1[S]

[36] McDougall M, Alur R, Gunter C A. AModel-Based Approach to Integrating Security Policies or Embedded Devices[C]// EMSOFT'04. ACM, 2004

[37] Nute D. Defeasible logic[J]. Lecture Notes in Computer Science, 2003, 2543: 151-169

[38] Maher M J. Efficient defeasible reasoning systems [C]// 12th IEEE Int. Conf. on Tools with Artificial Intelligence. Vancouver, 2000: 384-392

[39] Chow R, Golle P, Jakobsson M, et al. Controlling Data in the Cloud: Outsourcing Computation without Outsourcing Control [C]// CCSW'09. ACM, 2009

[40] Boella G, van der Torre L. Security Policies for Sharing Knowledge in Virtual Communities[J]. IEEE Transactions on System, Man, and Cybernetics—Part A: Systems and Humans, 2006, 36(3)

[41] Boella G, van der Torre L. A game theoretic approach to contractsin multiagent systems[J]. IEEE Trans. Syst., Man, Cybern. C, Appl. Rev., 2006, 36(1): 68-79

[42] Broersen J, Dastani M, Hulstijn J, et al. Goal generation in the BOID architecture[J]. Cogn. Sci. Q., 2002, 2(3/4): 428-447

[43] Belnap N D. A useful four-valued logic[J]. Modern Uses of Multiple-Valued Logic, Episteme, 1977, 2: 5-37

[44] Fitting M. Bilattices and the semantics of logic programming [J]. Logic Program, 1991, 11(1/2): 91-116

[45] Ginsberg M. Multi-valued logics: A uniform approach to reasoning[J]. AIComput. Intell., 1988, 4: 256-316

[46] Boella G, van der Torre L. Permission and authorization in policies for virtual communities of agents[C]// Proc. Agents and P2P Computing Workshop AAMAS, Lecture Notes in Computer Science. Berlin, Germany: Springer-Verlag, 2004, 3601: 86-97

[47] Van Emden M H, Kowalski R A. The Semantics of Predicate Logic as a Programming Language[J]. ACM, JACM, 1976, 23(4): 733-742

[48] Baral C, Subrahmanian V. Stable and extension class theory for logic programs and default theories[J]. Automat. Reas., 1992, 8: 345-366

[49] Van Gelder A. The alternating fixpoint of logic programs with negation[C]// Proceedings of the 8th ACM SILACT-SICMOO-SILART Symposium on Principles of Database Systems. 1989

[50] Jajodia S, Samarati P, Sapion M L, et al. Flexible Support for Multiple Access Control Policies[J]. ACM Transactions on Database Systems, 2001, 26(2)

[51] Gelfond M, Lifschitz V. The stable model semantics for logic programming[C]// Proceedings of the 5th International Conference and Symposium on Logic Programming (Seattle, Wash.). 1988

[52] Bertino E, Buccafurri F, Ferrari E, et al. A logical framework for reasoning on data access control policies[C]// Computer Security Foundations Workshop. IEEE, 1999

[53] Przymusinski T. The Well-Founded Semantics Coincides With Three-Valued Stable Semantics[J]. Journal of Fundamenta Informaticae, 1999, 13(4): 445-463

[54] Damianou N, Dulay N, Lupu E C, et al. The Ponder Policy Specification Language[C]// Workshop on Policies for Distributed Systems and Networks. 2001

[55] Barker S, Stuckey P J. Flexible Access Control Policy Specification with Constraint Logic Programming[J]. ACM Transactions on Information and System Security, 2003, 6(4): 501-546

[56] Bandara A K, Lupu E C, Russo A. Using Event Calculus to Formalise Policy Specification and Analysis[C]// Proceedings of the 4th International Workshop on Policies for Distributed Systems and Networks (POLICY'03). IEEE, 2003

[57] 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)

[58] Jaeger T, Zhang Xiao-lan, Edwards A. Policy Management Using Access Control Spaces[J]. ACM Transactions on Information and System Security, 2003, 6(3): 327-364

[59] Tidswell J E, Jaeger T. An Access Control Model for Simplifying Constraint Expression[C]// Proceedings of the 7th ACM Conference on Computer and Communications Security

[60] Zanin G, Mancini L V. Towards a Formal Model for Security Policies Specification and Validation in the SELinux System[C]// SACMAT'04. ACM, 2004

[61] Basile C, Cappadonia A, Lioy A. Network-Level Access Control Policy Analysis and Transformation[J]. IEEE/ACM Transactions on Neworking, 2012, 20(4)

[62] 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(7)

[63] 倪俊, 陈晓苏, 刘辉宇, 等. 网络安全策略求精一致性检测和冲突消解机制的研究[J]. 计算机科学, 2011, 38(2)