一种基于屏蔽码的ABAC静态策略冲突与冗余检测算法

江泽涛 谢 朕 王 琦 张文辉
(桂林电子科技大学计算机与信息安全学院 桂林 541004)

摘 要 针对基于属性的访问控制模型(Attribute-Based Access Control,ABAC)存在的静态策略冲突及冗余问题,提出了一种基于属性集有序化及二进制屏蔽码的静态策略冲突检测算法。该算法能够检测出全部的静态冲突,相对于目前典型的暴力算法与属性分割算法,降低了时间复杂度和空间复杂度;同时支持属性的新增及策略的新增或删除,能够更好地满足现代复杂网络环境的要求。

关键词 ABAC,冲突检测,屏蔽码,静态冲突
中图法分类号 TP309 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2018.02.034

ABAC Static Policy Conflict and Redundancy Detection Algorithm Based on Mask Key

JIANG Ze-tao XIE Zhen WANG Qi ZHANG Wen-hui
(School of Computer Science and Information Security,Guilin University of Electronic Technology,Guilin 541004,China)

Abstract A static policy conflict detection algorithm based on ordered attribute set and binary mask key was proposed. The algorithm can detect all of the static policy conflicts and redundancy in attribute-based access control model. Compared with the typical violence algorithm and the attribute segmentation algorithm,the proposed algorithm can reduce the time complexity and space complexity. Furthermore,it supports adding and removing attributes from set. New algorithm can meet the requirements of modern complex network environments.

Keywords ABAC,Conflict detection,Mask key,Static conflict

计算机安全技术始终是计算机技术发展过程中的重要课题[1];访问控制技术作为解决安全问题的重要手段,其任务是保证资源不被非法使用和非法访问[2];从早期诞生的自主访问控制(Discretionary Access Control,DAC)和强制访问控制(Mandatory Access Control,MAC)到基于角色的访问控制模型(Role-Based Access Control,RBAC),访问控制技术快速而又高效地解决了本地数据的安全问题。随着计算机技术的发展,特别是分布式系统以及云计算技术的发展,数据被分散到了不同的主机甚至不同的控制域[3]。基于属性的访问控制模型(Attribute-Based Access Control,ABAC)以属性为决策访问行为的核心要素,将访问行为抽象为一个属性的集合,通过预先设定的策略对属性集合的判定控制访问行为[4-5],避免了对具体用户或具体文件的繁琐判定。ABAC对复杂信息系统的细粒度访问和大规模用户动态扩展的支持,给网络提供了更为灵活和安全的访问控制技术,该模型非常适合在分布式及云计算环境中使用[6-7]。

但是随着计算机技术的广泛应用,用户数量和数据量不断增加,使得访问控制中的管控策略更加复杂,策略集中策略数量的增加,必然导致策略间出现更多的冲突及冗余。策略冗余并不会影响最后的访问行为决策,但策略冲突会导致对访问行为的决策不一致。解决策略冲突的方式主要有两种:1)在决策过程中消解策略间的冲突;2)在策略建立时检测冲突。

策略消解的主要方式有:自主消解,随机消解,遍历子基法等[8-10],但是无论使用什么方式进行消解,系统都需要额外的开销来处理这些问题。如果能在策略建立时即进行冲突检测,则可以避免后续的额外开销。

Damianou等[11]于2001年提出了基于Ponder语言的冲突检测算法。Campbell[12]提出了基于分离类推理的安全策略冲突检测方法。Davy等[13]提出基于策略连续体(Policy Continuum,PC)对策略定制过程中导致的冲突进行分析。王雅哲等[14]分析了可扩展访问控制标记语言(eXtensible Access Control Markup Language,XACML)中的策略冲突;Huang等提出了基于描述逻辑的方法来检测XACML策略间的冲突[15];Calero等[16]提出了基于对立属性推理的安全策略冲突检测方法。其中,文献[11,14]主要基于描述语言,检测效果及适用范围受限于其策略描述语言,不具备良好的扩展性;文献[13,15]基于形式逻辑,通过逻辑推理进行语义冲突检测,但不易实现,应用中需要开发难度较大的推理验证工具;文献[12,16]属于基于本体推理的冲突检测,本体基于描

到稿日期:2016-12-11 返修日期:2017-02-09 本文受国家自然科学基金(61572147),桂林电子科技大学图像图形智能处理重点实验项目(GIIP201501,GIIP201401),广西可信软件重点实验室项目(kx201502)资助。

江泽涛(1961—),男,博士,教授,主要研究方向为图像处理、计算机视觉、网络信息安全,E-mail:zetaojiang@126.com(通信作者);**谢 朕**(1989—),男,硕士生,主要研究方向为网络信息安全;**王 琦**(1992—),男,硕士生,主要研究方向为计算机视觉;**张文辉**(1970—),女,硕士,副教授,主要研究方向为图像处理、计算机游戏设计。

述逻辑,同时采用本体描述语言(Web Ontology Language,OWL),具备了两者的优点,但是本体的构建比较困难。

近年来,大量的策略冲突研究都集中在理论方面,以上各类方法都是从理论的层面提出相应的策略冲突检测算法,但都由于构建复杂而缺少具体的应用。鉴于这种情况,文献[17]提出了一种适用于 ABAC 的基于属性分解的策略冲突检测算法,其具有较好的应用特性,但是并不能完全检测出所有的静态冲突。本文提出一种新的冲突检测算法,该算法能够在性能与应用性都比较良好的情况下完全检测出所有的静态冲突;同时选择文献[17]中的算法进行性能对比。

1 基于属性的访问控制模型

基于属性的访问控制模型是一种为解决行业分布式应用可信关系的访问控制模型,它利用相关实体属性作为授权的基础来研究如何进行访问控制。

1.1 基本元素

基于属性的访问控制模型以属性作为访问过程控制的核心,主要涉及主体、客体、环境及行为,其核心元素由访问、条件、决策等组成。

主体(Subject):访问行为的发起实体。主体对应的属性称为主体属性(Subject Attribute,SA),例如主体的创建时间等基本信息或主体所具有的安全级别等内部特性。

客体(Object):访问行为的目标对象,可以是数据,也可以是一种服务。客体对应的属性称为客体属性(Object Attribute,OA),例如资源的类型等基本信息或安全级别等内部特性。

环境(Environment):访问行为发生时环境的上下文状态。环境对应的属性称为环境属性(Environment Attribute,EA),可以是系统本身的信息,例如系统 CPU 的使用情况或系统安全级别;也可以是客观存在的信息,例如当前时间或位置信息等。

行为(Action):主要包括对资源的读(Read)、写(Write)、修改(Modify)及运行(Execute)等操作,或对服务的使用(Use)等操作。

访问(Access):由主体发起的对客体的某种行为,例如用户申请播放视频或下载学术论文。访问过程涉及主体、客体、环境及行为,但是并不需要知道具体的主体、客体及环境信息。主体在访问过程中被抽象为一个 SA 的集合,不同的主体可能在某些域内被抽象成相同的集合。例如:GUET 大学的 2014 级学生均可以抽象为:{SA(入学年份)=2014,SA(身份)=学生}。客体在访问过程中被抽象为一个 OA 的集合,不同的客体可能在某些域内被抽象成相同的集合。例如:YK 视频网址的 VIP 视频可以抽象为:{OA(类型)=视频,OA(VIP)=是}。环境本身就是一个 EA 的集合,可以是系统本身的信息,例如系统 CPU 的使用情况或系统安全级别;也可以是客观存在的信息,例如当前时间或位置信息等。

访问过程可以被抽象为一个集合{SA,OA,EA,Action}。例如:

普通用户于 8 点播放普通视频的访问过程可以抽象为:

{SA(类型)=普通客户,OA(类型)=普通视频,EA(时间)=8:00},Action=播放

学生在实验室下载学术论文的访问过程可以抽象为:

{SA(身份)=学生,OA(类型)=学术论文,EA(地点)=实验室},Action=下载

策略(Policy):系统为了对访问行为进行管理,还必须提前定义适用的策略,对访问行为预先设定的判断规则主要包含条件(属性及范围)、行为和决策。

条件(Condition):包括若干的 SA,OA 和 EA 及其对应的适用范围。在访问行为中,若主体、客体及环境的属性均包含在条件给定的属性范围内,则表示本次访问行为是适用于本条策略的。SA,OA 及 EA 并非都必须包含在条件中,但条件中至少要包含一种属性。如:若仅包含 EA(时间)∈[8:00,23:00],则策略退化为一种普适的策略,其对所有在 8:00-23:00 之间的访问行为均具有约束力。

决策(Decision):在条件下访问行为的结果。最常见的决策状态为允许(Allow)和拒绝(Deny)。但是在系统根据策略对访问行为进行判定时,若出现无匹配策略或多条匹配策略,则还有可能返回未知(NA)状态。

例如:允许学生在 8:00-23:00 使用上传和下载的服务的策略定义为:

{SA(身份)∈{学生},OA(服务类型)∈{上传,下载},EA(时间)∈[8:00,23:00]},Action=Use,Decision=Allow

1.2 冲突及冗余

根据访问过程集合在策略集中查找策略,若 $Action_{访问}=Action_{策略}$ 且 $Set_{访问}\in Set_{条件}$,则认为策略适用于本次访问。对同一个访问请求:如果适用于一条以上的策略且产生的判决结果不一致,则认为出现了策略冲突;如果适用于一条以上的策略且产生的判决结果一致,则认为出现了策略冗余。

如果策略在制定完成时无法确定其与已存在的策略发生了冲突,则只能在系统运行的过程中发现冲突,称这种冲突为动态冲突。此时,策略间的条件并未发生交集,只是由于主体、客体或环境的属性同时适配了多条策略。这种情况在实际应用中很难避免,但不属于本文探讨的范围。如果策略在制定完成时就已经可以确定其与已存在的策略发生了冲突,则称这种冲突为静态冲突。静态冲突中,冲突的策略间的属性范围一定发生了相互重叠甚至具有包含关系,这种冲突应该在策略定制时予以避免。若策略间所有的属性均出现了交集,同时策略的行为相同而决策不同,此时访问行为的 SA,OA 及 EA 只要适用于交集内,则必然出现冲突,这种冲突被称为必然冲突。例如:

策略 1={SA(身份)∈{学生},OA(服务类型)∈{上传,下载},EA(时间)∈[8:00,23:00]},Action=Use,Decision=Allow

策略 2={SA(身份)∈{学生},OA(服务类型)∈{下载},EA(时间)∈[22:00,24:00]},Action=Use,Decision=Deny

策略 1 允许学生在 8:00-23:00使用上传和下载的服务。策略 2 不允许学生在 22:00-24:00 使用下载服务。条件的交集部分为{SA(身份)∈{学生},OA(服务类型)∈{下载},EA(时间)∈[22:00,23:00]},即若学生在 22:00-23:00之间使用下载业务,则一定会出现策略冲突。

若策略间所有相同的属性均出现了交集,同时策略的行为相同而决策不同,此时访问行为的 SA,OA 及 EA 尽管适用于交集内,但不一定能适用于所有的策略,因此不一定产生冲突,这种情况被称为或然冲突。例如:

策略 3＝{SA(入学时间)∈[2014，2016]，SA(身份)∈{学生}，OA(服务类型)∈{上传，下载}，EA(时间)∈[8：00，23：00]}，Action＝Use，Decision＝Allow

策略 4＝{SA(宿舍)∈[10，20]，SA(身份)∈{学生}，OA(服务类型)∈{下载}，EA(时间)∈[22：00，24：00]}，Action＝Use，Decision＝Deny

策略 3 允许 2014－2016 级学生在 8：00－23：00 使用上传和下载的服务。策略 4 不允许宿舍号为 10－20 的学生在 22：00－24：00使用下载服务。其交集部分为{SA(身份)∈{学生}，OA(服务类型)∈{下载}，EA(时间)∈[22：00，23：00]}，若 15 号寝室的 2015 级学生在 22：00－23：00 之间使用下载业务则会出现策略冲突，但是 21 号寝室的 2015 级学生或者 15 号寝室的 2013 级学生在 22：00－23：00 之间使用下载业务就不会出现策略冲突。

策略冗余的情况与策略冲突的情况类似，即对同一个访问请求：如果适用于一条以上的策略且产生的判决结果一致，则认为出现了策略冗余。策略冗余只会导致访问重复判定行为，并不会对实际系统的决策过程造成影响，但在对策略进行删除时，其冗余部分可能导致意外的结果。

将策略 4 稍作修改：

策略 4.1＝{SA(宿舍)∈[10，20]，SA(身份)∈{学生}，OA(服务类型)∈{下载}，EA(时间)∈[22：00，24：00]}，Action＝Use，Decision＝Allow

策略 4.1 允许宿舍号为 10－20 的学生在 22：00－24：00 使用下载服务，此时 15 号寝室的 2013 级学生在 22：00－23：00之间使用下载业务的情况可以同时匹配到策略 3 及策略 4.1，两个策略均允许其使用。假设业务规则发生了变动，即禁止宿舍号为 10－20 的学生在22：00－24：00使用下载服务，同时系统默认 $\text{Decision}_{默认}$＝Deny，则管理员选择将策略 4.1 删除，完成规则变动，但是 15 号寝室的 2013 级学生在 22：00－23：00之间依然可以使用下载业务，因为此情况匹配到策略 3。最终导致业务规则修改失败。

由此可见，策略冲突与策略冗余仅仅是 Decision 的不同，其检测过程是类似的。本文的实验将仅针对策略冲突，策略冗余的情况仅需要修改 Decision 的判断，与策略冲突类似。

2　策略冲突检测

假设在控制域内，属性集内包含 M 条属性，策略集内包含 N 条策略，平均每条策略包含 K 条属性。下面将在此域内对各种算法进行时间及空间复杂度的分析。

2.1　暴力检测算法

暴力检测算法是将新加入的策略与已有的策略集的每一条策略依次进行比较。按照策略冲突的定义，若两个策略行为相同、决策相反且条件集合交集不为空，则认为产生了策略冲突。

整个比较过程共进行 N 次比较，以确定是否与策略集的已有策略冲突。每次需要对两个策略的条件集进行对比，两个无序属性集合的比较时间复杂度为 M^2。比较过程中，除了有限变量空间的使用，不需要额外空间，因此时间复杂度为 $O(NM^2)$，空间复杂度为 $O(1)$。

2.2　属性分割算法

刘江等[17]提出了一种将属性分割的冲突检测算法。将属性集中的属性取值按照策略条件进行分割，使得分割后的任意属性值明确地属于或不属于某一条或数条策略，而不会出现属性值部分属于某一条策略的情况。对于新加入的属性，也需要进行属性集的分割，并根据分割后的属性值判断新加入策略与已有策略集的条件相交情况。

由于其具体复杂度与属性分割的精细程度存在很大的关联，这里仅对其进行粗略的评估。对于任意一次新增策略，需要对 M 条属性进行范围拆分，拆分后同属性的范围内需要对策略集合进行合并操作，合并操作后有 M 条属性对应的不同策略集合需要进行交集操作。虽然很难估计每个集合内的数据量，但总体来说其与总策略条数相关，这里估算其平均策略条数为 $O(N)$，而两个无序策略集合并集或交集的复杂度均为 $O(N^2)$。对于需要额外存储的信息，尽管不是每一个策略的所有属性范围都需要进行分割，但分割数的上界为 N 条策略且每条策略分割 M 次。因此，时间复杂度为 $O(N^2M)$，空间复杂度为 $O(NM)$。

但算法存在以下问题：原策略准备完成后，需要保留中间分割状态，否则新增策略时需要重新进行分割；策略集中删除策略后，很难将分割状态进行回退或者回退代价过大，因此必须重建分割状态；检测出策略冲突时，也需要进行分割状态的重运算；只能检测出必然冲突，无法检测出或然冲突，作者在实验中也证实，冲突检测率是低于 100％的。

2.3　基于屏蔽码的冲突检测算法

在传统访问控制模型中，属性集中的每个属性虽然都有着自己的序号，但该序号并没有太多的意义，仅仅是为了便于存储而设置的 ID 号。本文中属性的序号有着重要的意义，针对属性的顺序可以是随机的，但是其有序化的方式必须是确定的且可重复使用的。后续主要的加速过程就是基于属性的有序化的属性编码，具体示例如表 1 所列。

表 1　编号示例

Table 1　Example of number

属性	类型	编号
ATT1	SA/OA/EA	0
ATT2	SA/OA/EA	1
…	…	…
ATTn	SA/OA/EA	$n-1$

屏蔽码(Mask Key，MK)是用来表示策略内涉及属性分配情况的二进制编码。根据属性的 ID 号，为策略生成相应的屏蔽码，用以标记某条策略所涉及到的属性集合，并通过逻辑运算对策略间涉及的属性集合进行比对。

如图 1 所示，MK 是一个长度为 n 位的数据，第 i 位为 1，表示策略条件内包含对编号为 i 的属性的限定。在实际使用过程中，可以适当调整 MK 的长度。例如，访问控制模型属性集中拥有 45 个属性，编码 ID 为 0－44。MK 的长度设计为 64，使用一个无符号 64 位长整形存储。若属性集新增一个属性，则可以顺序将其 ID 设置为 45，这样在已有策略集的策略对应 MK 的 45 位默认为 0，不需要重新计算。

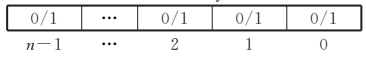

图 1　MK 示意图

Fig. 1　Sketch map of MK

在策略间进行策略冲突检测之前，先对第 i 个和第 j 个

策略的 MK_i 和 MK_j 进行逻辑与运算，得到 MK_{And}。若 $MK_{And}=0$，则表示两条策略条件不涉及相同的属性，不需要进行策略冲突的检测；若 $MK_{And}!=0$，则需要针对两条策略的条件即属性集合进行比对。

属性集合比对的方式有两种：

1)直接将两个集合进行比对。由于集合内属性存在偏序关系，即按照属性 ID 进行排序，其比对速度要快于无序集合。

2)针对两个策略的 MK 进行运算后的结果，其每一个为 1 的位即为需要比对的属性编号。其运算时间只与总属性数(即 MK 的长度)相关，与两个集合的长度无关。

同时，依赖于 MK 进行集合比对时，若 $MK_{And}!=MK_i$ 且 $MK_{And}!=MK_j$，则可以确定一定不会产生必然冲突。在只需要检测必然冲突时有非常良好的加速效果。在选择得当的情况下，集合的比对算法效率较高。

算法流程图如图 2 所示，主要步骤如下：

1)新建策略 S；

2)计算策略 S 对应的屏蔽码 MK；

3)判断策略集中的策略是否已经遍历，若已经遍历则检测流程结束，否则继续；

4)读取策略集的第 i 条策略 S_i 及其屏蔽码 MK_i；

5)判断 S_i 与 S 行为(Action)是否相同，若行为不同，则回到步骤 3)，否则继续；

6)对 MK_i 与 MK 进行逻辑与运算，并判断其运行结果是否为 0，若为 0 则回到步骤 3)，否则继续；

7)判断 S_i 与 S 的条件(Condition)是否有交集，若没有交集则回到步骤 3)，否则继续；

8)判断 S_i 与 S 的决策是否相同，若决策相同则为冗余，若决策不同则为冲突。

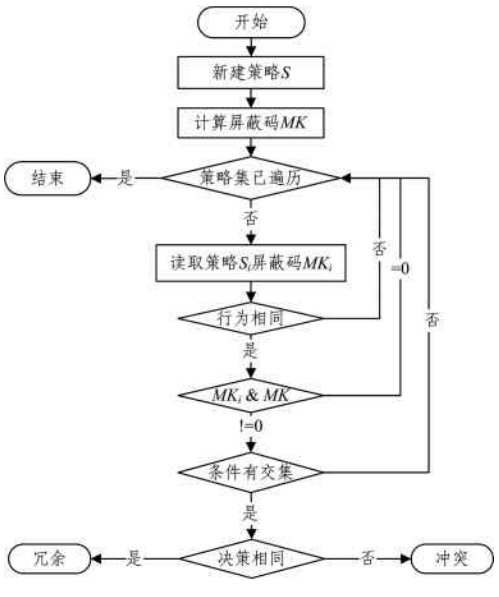

图 2 算法流程图

Fig. 2 Flowchart of the proposed algorithm

步骤 6)中，若将 MK_i & MK 是否为 0 的判定改为 MK_i & $MK==MK$ 或 MK_i & $MK==MK_i$，则可以进行必然冲突的检测。

属性集中的属性有序化只需要额外的 M 个空间记录属性的 ID，时间复杂度为 O(1)，空间复杂度为 O(M)。每一条策略需要一个长度为 M 位的 MK，而其计算过程需要遍历 K 条属性，但是每一条策略只需要在最初加入策略集时进行一次 MK 的计算，时间复杂度为 O(NMK)，空间复杂度为 O(NM)。对任意两个集合的对比，若采用有序集合对比的方式，时间复杂度为 O(NK)，空间复杂度为 O(1)。若使用 MK 辅助进行集合对比，则其复杂度为 O(NM)，空间复杂度为 O(1)。总体时间复杂度为 O(NMK)，空间复杂度为 O(NM)。而且在删除时不需要额外的操作。

3 对比实验结果

本文采用随机生成策略的方式进行策略冲突的对比实验。统一设置属性集内类型范围为[0,100)，枚举类型属性设置为 5 个枚举类，两种属性的比例为 1∶1。将暴力算法、属性分割算法及本文提出的屏蔽码算法分别编号为 A,B,C。

实验 1 选择属性集内 20 个属性，每条策略包含 10～14 个属性，策略条数分别为 1000,2500,5000,7500,10000。实验结果如表 2 所列。

表 2 实验 1 的结果

Table 2 Results of experiment 1

K		20				
N		1000	2500	5000	7500	10000
M		10,14				
A	Conflict	335	1378	3344	5688	7872
	Time	171	880	2647	4423	6997
B	Conflict	0	0	0	0	0
	Time	64	517	4003	13659	30193
C	Conflict	335	1378	3344	5688	7872
	Time	55	234	649	1029	1458

实验 2 选择属性集内 20 个属性，策略数为 5000，每条策略内分别包含 4～8(20%～40%)，10～14(50%～70%)，16～20(80%～100%)个属性。实验结果如表 3 所列。

表 3 实验 2 的结果

Table 3 Results of experiment 2

K		20		
N		5000		
M		4,8	10,14	16,20
A	Conflict	4978	3330	3
	Time	75	2381	11016
B	Conflict	0	0	0
	Time	3256	3492	3235
C	Conflict	4978	3330	3
	Time	72	525	554

实验 3 策略数为 5000，属性集内分别有 10,20,40 个属性，每条策略均包含 50%～70%的属性，属性数为 5～7,10～14,20～28。实验结果如表 4 所列。

表 4 实验 3 的结果

Table 4 Results of experiment 3

K		10	20	40
N		5000		
M		5,7	10,14	20,28
A	Conflict	4895	3369	131
	Time	100	2374	17254
B	Conflict	15	0	0
	Time	20006	3322	1085
C	Conflict	4895	3369	131
	Time	87	523	1440

基于以上 3 个实验结果，从性能及准确性上对本算法进行分析。

本文提出的 C 算法在策略数量上升的情况下，耗时增加不多，对比 A 算法和 B 算法，C 算法具有良好的性能（见图 3）。

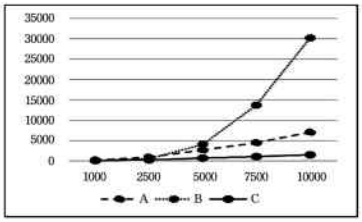

图 3　策略数量与耗时的关系

Fig. 3　Relationship between the number of strategies and time consuming

本文提出的 C 算法计算出的冲突数量无论在何种情况下均与暴力 A 算法一致，具有良好的准确性。而文献[17]中指出 B 算法的检测率仅为 85%左右，因此 B 算法将不参与比较（见图 4）。

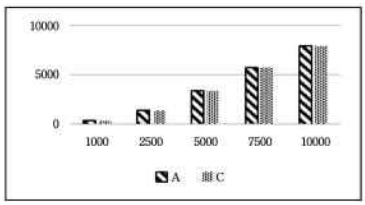

图 4　策略数量与冲突数量的关系

Fig. 4　Relationship between the number of strategies and the number of conflicts

策略检测的核心代码如下。

```
//检测策略冲突或冗余
auto Conflict=[&](const Policy & sA,const Policy & sB)->bool
{
  size_ti=0,j=0;
  //有序集合 sA 与 sB 的遍历对比
  while (i<sA. Attrs. size() && j < sB. Attrs. size())
  {
    //属性相同
    if (sA. Attrs[i]. Name == sB. Attrs[j]. Name)
    {
      //属性值取值范围是否有重合
      if (sA. Attrs[i]. IsRange ? (sA. Attrs[i]. Min < sB. Attrs[j].
      Max && sB. Attrs[j]. Min < sA. Attrs[i]. Max):sA. Attrs
      [i]. Value == sB. Attrs[j]. Value)
      {
        //无重合，继续判断
        ++i;
        ++j;
      }
      else
      {
        //有重合，发生冲突或冗余
        return false;
      }
    }
    //属性不同，根据属性有序性，序列较小的属性做+1 操作
    else if (_Ranks[sA. Attrs[i]. Name] < _Ranks[sB. Attrs[j].
    Name])
    {
      i++;
    }
    else
    {
      j++;
    }
  }
  return true;
};
```

冲突的条件为：

```
_Pol[i]. IsWrite == _Pol[j]. IsWrite && _Pol[i]. IsAllow != _Pol[j]. IsAllow && (_Keys[i] & _Keys[j]) != 0 && Conflict(_Pol[i],_Pol[j])
```

只需要进行简单的修改就可以只进行必然策略检测：

```
_Pol[i]. IsWrite == _Pol[j]. IsWrite && _Pol[i]. IsAllow != _Pol[j]. IsAllow && ((_Keys[i] & _Keys[j]) == _Keys[i] || (_Keys[i] & _Keys[j]) == _Keys[j]) && Conflict(_Pol[i],_Pol[j])
```

或进行条件的修改就可以判定策略冗余：

```
_Pol[i]. IsWrite == _Pol[j]. IsWrite && _Pol[i]. IsAllow == _Pol[j]. IsAllow && (_Keys[i] & _Keys[j]) != 0 && Conflict(_Pol[i],_Pol[j])
```

结束语　本文针对 ABAC 模型下的策略冲突及冗余提出的算法能够在较快速度下检测出全部的静态冲突，但是无法对动态冲突进行快速检测。后续将研究针对动态冲突的快速检测方法。

参 考 文 献

[1] FENG D G，ZHANG M，ZHANG Y，et al. Study on Cloud Computing Security[J]. Journal of Software，2011，22(1)：71-83. (in Chinese)
冯登国，张敏，张妍，等. 云计算安全研究[J]. 软件学报，2011，22(1)：71-83.

[2] WANG Y D，YANG J H，XU C，et al. Survey on Access Control Technologies for Cloud Computing[J]. Journal of Software，2015，26(5)：1129-1150. (in Chinese)
王于丁，杨家海，徐聪，等. 云计算访问控制技术研究综述[J]. 软件学报，2015，26(5)：1129-1150.

[3] LI F H，SU M，SHI G Z，et al. Research Status and Development Trends of Access Control Model[J]. Acta Electronica Sinica，2012，40(4)：805-813. (in Chinese)
李凤华，苏铓，史国振，等. 访问控制模型研究进展及发展趋势[J]. 电子学报，2012，40(4)：805-813.

[4] ZHANG X，LI Y，NALLA D. An attribute-based access matrix model[C]// Proceedings of the 2005 ACM Symposium on Applied Computing. ACM，2005：359-363.

[5] YUAN E，TONG J. Attributed based access control (ABAC) for web services[C]// IEEE International Conference on Web Services (ICWS'05). IEEE，2005.

[6] WANG X M，FU H，ZHANG L G. Research Progress on Attribute-Based Access Control[J]. Acta Electronica Sinica，2010，38(7)：1660-1667. (in Chinese)
王小明，付红，张立臣. 基于属性的访问控制研究进展[J]. 电子学报，2010，38(7)：1660-1667.

[7] ZOU J S,ZHANG Y S,GAO Y. Research of ABAC Model based on Usage Control under Cloud Environment[J]. Application Research of Computers,2014,31(12):3692-3694. (in Chinese)
邹佳顺,张永胜,高艳. 云环境下基于使用控制的 ABAC 模型研究[J]. 计算机应用研究,2014,31(12):3692-3694.

[8] LI R X,LU J F,LI T Y. et al. An Approach for Resolving Inconsistency Conflicts in Access Control Policies[J]. Chinese Journal of Computers,2013,36(6):1210-1223. (in Chinese)
李瑞轩,鲁剑锋,李添翼,等. 一种访问控制策略非一致性冲突消解方法[J]. 计算机学报,2013,36(6):1210-1223.

[9] DUBOIS D,LANG J,PRADE H. Possibilistic logic 1[OL]. http://core. ac. uk/display/20741884.

[10] LANG J. Possibilistic logic: complexity and algorithms[M]// Handbook of defeasible reasoning and uncertainty management systems. Springer Netherlands,2000:179-220.

[11] DAMIANOU N,DUALAY N,LUPU E,et al. The ponder policy specification language[M]// Policies for Distributed Systems and Networks. Springer Berlin Heidelberg,2001:18-38.

[12] CAMPBELL G A. Ontologies for Resolution Policy Definition and Policy Conflict Detection[R]. Department of Computing Science and Mathematics,University of Stirling,2007.

[13] DAVY S,JENNINGS B,STRASSNER J. The policy continuum-Policy authoring and conflict analysis[J]. Computer Communications,2008,31(13):2981-2995.

[14] WANG Y Z,FENG D G. 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.

[15] HUANG F,HUANG Z,LIU L. A DL-based method for access control policy conflict detecting[C]// Proceedings of the First Asia-Pacific Symposium on Internetware. ACM,2009:16.

[16] CALERO J M A,PÉREZ J M M,BERNABÉ J B,et al. Detection of semantic conflicts in ontology and rule-based information systems[J]. Data & Knowledge Engineering,2010,69(11):1117-1137.

[17] LIU J,ZHANG H Q,DAI X D,et al. A Static Policy Conflict Detection Algorithm for Attribute Based Access Control[J]. Computer Engineering,2013,39(6):200-204. (in Chinese)
刘江,张红旗,代向东,等. 一种 ABAC 静态策略冲突检测算法[J]. 计算机工程,2013,39(6):200-204.

(上接第 180 页)

[4] JIANG B,LIU Z C,YAN J G. Research on the Defense on the Theory Code Attacking Compositor Based[J]. Mathematics In Practice and Theory,2013,43(24):75-79. (in Chinese)
姜波,刘志成,严建钢. 基于复杂网络理论的防空节点攻击排序研究[J]. 数学的实践与认识,2013,43(24):75-79.

[5] REN X L,LV L Y. Review of ranking nodes in complex networks[J]. China Science Bulletin,2014,59(13):1175-1197. (in Chinese)
任晓龙,吕琳媛. 网络重要节点排序方法综述[J]. 科学通报,2014,59(13):1175-1197.

[6] REN Z M,SHAO F,LIU J G,et al. Node importance measurement based on the degree and clustering coefficient information [J]. Acta Physica Sinica,2013,62(12):505.

[7] LIU J G,REN Z M,GUO Q,et al. Node importance ranking of complex networks[J]. Acta Physica Sinica,2013,62(17):178901.

[8] ZHANG K,LI P P,ZHU B P,et al. Evaluation method for node important in directed-weighted complex networks based PageRank[J]. Journal of Nanjing University of Aeronautics & Astronautics,2013,45(3):429-434. (in Chinese)
张琨,李配配,朱保平,等. 基于 PageRank 的有向加权复杂网络节点重要性评估方法[J]. 南京航空航天大学学报,2013,45(3):429-434.

[9] LANDERR A,FRIEDL B,HEIDEMANN J. A critical review of centrality measures in social networks[J]. Business & Information Systems Engineering,2010,2(6):371-385.

[10] YU H,LIU Z,LI Y J. Key nodes in complex networks identified by multi-attribute decision-making method[J]. Acta Physica Sinica,2013,62(2):020204.

[11] ZHANG K,MA Y H. Centrality ranking algorithm based on network structure[J]. Application Research of Computers,2016,33(9):2596-2600. (in Chinese)
张凯,马英红. 基于网络结构的节点中心性排序优化算法[J]. 计算机应用研究,2016,33(9):2596-2600.

[12] MEO P D,FERRARAB E,FIUMARA G,et al. A novel measure of edge centrality in social networks [J]. Knowledge-Based Systems,2012,30(6):136-150.

[13] HOLME P,SARAM K J. Temporal networks[J]. Physics Reports,2012,519(3):97-125.

[14] NEWMAN M E J. Networks:an Introduction [M]. Oxford:Oxford University Press,2010:23-45.

[15] MURRAY S,MARK W. Knotty-centrality:finding the connective core of a complex network[J]. Plos One,2012,7(5):e36579.

[16] JIN J,XU K,XIONG N. Multi-index evaluation algorithm based on principal component analysis for node importance in complex networks [J]. IET Networks,2012,1(3):108-122.

[17] ZHANG J,XU X K,LI P. Node importance for dynamical process on networks:a multi-scale characterization [J]. Chaos,2011,21(1):016107.

[18] ZHANG C H,ZHANG X K,DENG H Z. Operation system of systems on effectiveness assessment method based operation loop[J]. Electronic Design Engineering,2012,20(21):62-68. (in Chinese)
张春华,张小可,邓宏钟. 一种基于作战环的作战体系效能评估方法[J]. 电子设计工程,2012,20(21):62-68.

[19] ZHOU X,ZHANG F M,LI K W. Finding Vital node by node importance evaluation matrix in complex networks[J]. Acta Physica Sinica,2012,61(5):485.

[20] ZHANG X P,LI Y S,LIU G,et al. Evaluation method of importance for nodes in complex networks based on importance contribution[J]. Complex Systems and Complexity Science,2014,11(3):26-32. (in Chinese)
张喜平,李永树,刘刚,等. 节点重要度贡献的复杂网络节点重要度评估方法[J]. 复杂系统与复杂性科学,2014,11(3):26-32.

[21] ZHAO Y H,WANG Z L,ZHENG J. Finding most vital node by node importance contribution matrix in communication networks [J]. Journal of Beijing University of Aeronautics and Astronautics,2009,35(9):1076-1079. (in Chinese)
赵毅寰,王祖林,郑晶. 利用重要性贡献矩阵确定通信网中最重要节点[J]. 北京航空航天大学学报,2009,35(9):1076-1079.