基于策略的安全管理系统中一种基于标签的策略协作机制*)

姚 前 陈 舜 谢 立
(南京大学计算机软件新技术国家重点实验室 南京 210093)

摘 要 当前,各研究组织提出的策略信息模型中,策略都是相对独立的,策略之间没有有效的协作机制,难以满足基于策略的安全管理中对策略协作的需求。本文分析了基于策略的安全管理中关于策略协作的需求,并针对策略协作的需求分析了各种研究组织提出的信息模型的不足之处;在此基础上提出了一种基于标签的策略协作机制,提出的策略协作机制是对当前的各种策略信息模型的扩展,可以满足安全管理系统对策略协作的需求。最后给出了实验结果和结论。

关键词 安全管理,基于策略的安全管理,策略协作

A Label-based Policy Cooperation Mechanism in Policy-based Security Management Systems

YAO Qian CHEN Shun XIE Li
(State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093)

Abstract Nowadays, policies in the policy information models proposed by various research groups are isolated and there are no mechanisms for policy cooperation, so the requirements about policy cooperation in policy based security management systems are hard to be satisfied. In this paper, requirements of policy cooperation in policy based secuirty management systems are analyzed, and then the deficiencies of policy information models proposed by different research groups are analyzed according to policy cooperation requirements. Based on these analyses, a label based policy cooperation mechanism is proposed, which is extension to the current various policy information models and can satisfy the requirements of policy cooperation in secuirty management systems. The experiment and conclusions are present at last.

Keywords Security management, Policy-based security management, Policy cooperation

1 引言

基于策略的安全管理可以在不对管理系统进行重新编码并且无需关闭管理系统的情况下动态改变管理系统的行为,从而给管理系统带来了很大的灵活性。近年来基于策略的安全管理逐渐成为研究的热点[1,2]。很多研究组织一直致力于基于策略的安全管理的研究,并提出了不同的策略信息模型,其中 DMTF 提出的 CIM(Common Information Model)模型包含了策略信息模型[3];IETF 基于 CIM 分别提出了策略核心模型 PCIM[4,5](Policy Core Information Model)及针对不同网络或不同应用的策略信息模型如 QoS 策略信息模型等[6];TMF 围绕基于策略的管理方法在 NGOSS 中的应用提出了一种基于 DEN-ng(Directory Enabled Networks-next generation)的策略信息模型[7]。在以上各研究组织提出的策略信息模型中,都提到策略在安全管理系统中有着不同层次的抽象,不同层次的策略之间存在映射关系,而这种映射可能不是一对一的关系。这样多条策略之间就可能存在一种协作关系,因为映射而来的多条策略需要协作共同完成上一层次的策略描述的目标。各研究组织提出的策略信息模型中都包括了策略之间的继承、包含关系,但对策略之间的协作关系的研究还很少。目前为止,各信息模型中还没有对策略协作关系的描述。文[8]提出了一种在被管对象中嵌入标签值来实现策略之间协作的方法,但这种方法仅能实现管理系统中的用来管理被管对象的低级策略之间的协作,如在应用策略对数据分组进行分类时,将标签值嵌入到分组头(如 DiffServ 网络中分组的 DSCP,MPLS 中的 Label 等),另一方面,文[8]提出的方法不能灵活地对有协作关系的策略进行更新,在其实现中所有策略必须同时更新。基于策略的方法除了用来管理安全系统中的被管对象外,还要控制安全管理系统的行为[7],文[8]提出的方法难以实现用来控制安全管理系统行为的策略之间的协作。本文在研究了各组织提出的策略信息模型的基础上,借鉴文[8]的方法,提出了一种基于标签的策略协作的方法。本文提出的方法,简单、灵活,可以实现多条策略之间的协作,并且避免了文[8]提出的方法所存在的问题,可以灵活地更新策略及策略之间的协作关系。

2 基于策略的安全管理中策略协作的分析

在基于策略的安全管理系统的策略抽象层次中,由高级策略映射而来的多条低级策略为了实现高级策略描述的目标,它们之间就可能存在协作关系。另外,基于策略的安全管理系统中,即使不是由同一策略映射而来的策略之间也可能存在协作关系,如系统中的多条策略可能需要协作来完成一定的功能,后实施的策略可能与先实施的策略所带来的结果有关,先实施的策略带来的结果不同,随之而实施的策略可能也不同。当相互协作的策略用来管理系统中的被管对象时,策略之间的协作信息可以由被管对象来传递,如 DiffServ 网

*)国家"十五"科技攻关项目(金融示范工程),课题编号:2001BA102A04。**姚 前、陈 舜** 博士生,主要研究方向为分布式系统和计算机安全;**谢 立** 教授、博士生导师,主要研究方向为分布式计算、并行处理、先进操作系统等。

络[10]中的 DSCP 字段。在 DiffServ 网络中，网元中多个处理单元中所实施的策略共同来实现 IP 流量的分类与整形，这些策略之间的协作可以通过 IP 分组的 DS 域的值来进行，如图 1 所示。

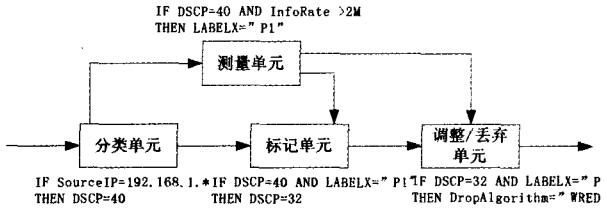

图 1　管理被管对象的策略之间的协作

因为网络中的被管对象都有自身的属性，所以可以利用被管对象的属性来传递管理这些对象的策略之间的协作信息，如图 1 中的 IP 分组中的 DSCP 域及某一字段 LABELX。安全管理系统中策略的实施改变了被管对象的属性，在后续处理中根据被管对象的属性可以了解到已经执行的策略并根据此来确定要执行的策略，这种方法适于应用在网元层管理系统对网元的管理中。应用被管对象进行策略协作信息的传递在文[8]中作了较详细的描述。

在基于策略的安全管理系统中，策略不仅用来管理被管对象，而且还要控制安全管理系统自身的行为[7]，所以文[8]的方法不能完全满足安全管理系统关于策略协作的需求。目前，各策略信息模型中，策略之间是相对独立的，因而通过实施策略完成的功能也常常有限。在目前的信息模型中，一般采用将多策略组成策略组(PolicyGroup)[4,5]的方法，通过策略组的实施来完成复杂的功能。但是，策略组的实施只能保证多条策略在安全管理系统中的共同实施，并不能保证它们之间的协作关系，因为这种方式不能在策略之间传递信息。基于目前的策略信息模型，实现复杂管理功能的另一种方法就是在后续执行的策略中加入已经执行的策略的条件，实现多条策略的共同执行。如图 2 左侧图所示，在安全管理系统中，有 3 个处理过程及在这些处理过程中实施的策略，图 2 中采用自然语言来描述每一过程要执行的策略。当 Policy A 在过程 A 执行后，根据 Policy A 的执行结果来选择下一步的处理，具体的处理需求如图 2 左侧图所示。

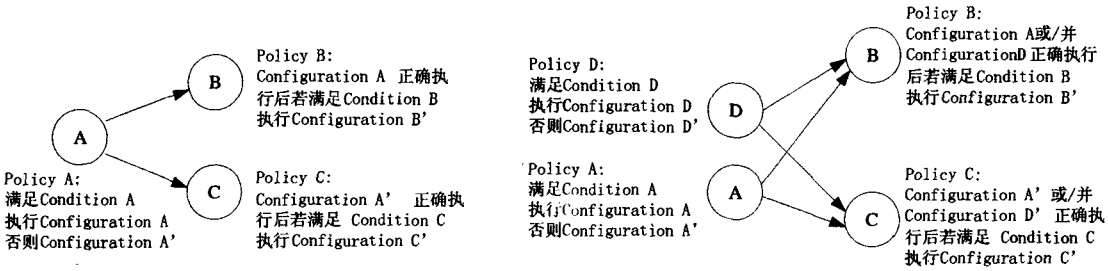

图 2　策略协作需求实例

基于当前的各研究组织提出的信息模型，为实现图 2 左侧图所示的策略协作可以将多条策略组成一个策略组，然后在安全管理系统中实施策略组来保证多个策略的执行(都执行或者都不执行)，如图 3 所示，Policy A、Policy A'、Policy B、Policy C 分别是实现安全管理系统中 3 个处理过程目标的策略。在安全管理系统中，构造两个策略组 PolicyGroup AB 和 PolicyGroup AC，然后在安全管理系统中实施这两个策略组(在同一时间只会有一个策略组被执行)，这样可以保证在系统中共同实施策略组中所包含的策略。但是这种方式存在两方面的缺点，如图 3 所示：(1)当 Condition A 满足，但在 Policy A 实施的过程中出现异常没有正确完成 Configuration A 时，Policy B 仍会被执行。这样就不能满足图 2 左侧图的需求，原因在于策略组中的策略仍然是相对独立的，没有协作信息的传递；(2)策略组中所包含的策略必须在同一个 PEP (Policy Enforcement Point)[11]中实施，而当过程 A 和过程 B 不在同一个 PEP 时，则策略组在每个 PEP 中都可能不会正确执行。

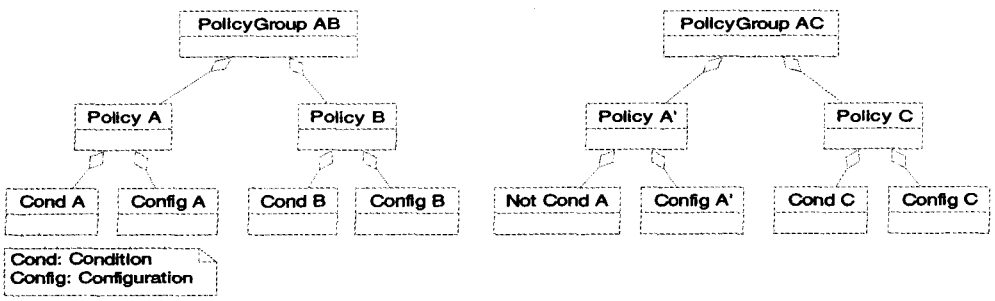

图 3　基于目前的信息模型的策略协作方式 1

基于当前的各策略信息模型，实现图 2 左侧图所示的策略协作的另一种方式就是在后续的策略条件中加入已经执行的策略的条件，后执行的策略如图 4 所示。当同时满足 Condition A 和 Condition B 时才实施 Policy B，这样就保证了安全管理系统能够选择正确的策略来共同实施。当 Condition A 满足说明了已经实施的策略为 Policy A，因此在安全管理系统中 Policy A 和 Policy B 将会共同被实施。这种策略协作的实现方式存在三方面的缺点：(1)如图 4 所示，当 Condition A 满足，但在 Policy A 实施的过程中出现异常没有正确完成 Configuration A 时，Policy B 也会执行，同样不能满足图 2 左

侧图的需求；(2)因为在后续执行的策略中加入了前期执行的策略的条件，使得后续策略条件判断很复杂，并且在安全管理系统中要多次重复判断同一条件；(3)如果当执行 Policy B 时 Condition A 才满足，则系统会认为 Policy A 已经被执行，而实际上 Policy A 可能没被执行。

当基于策略的安全管理中策略协作的需求更加复杂时，如图 2 右侧图所示，则上述的两种实现方式更加难以满足安全管理系统的需求。

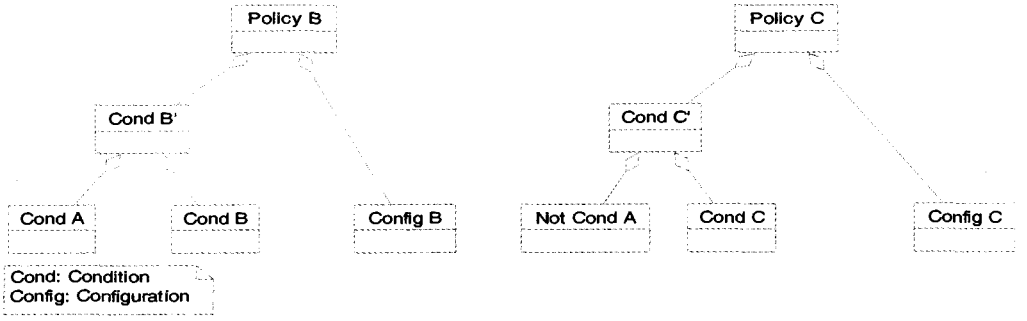

图 4　基于目前的信息模型的策略协作方式 2

3　基于标签的策略协作机制

通过上面的分析可以看出，由于基于策略的安全管理中策略之间难以传递信息，导致策略之间难以实现协作。针对基于策略的安全管理系统中策略协作的需求及目前各策略信息模型的不足，我们提出了一种基于标签的策略协作机制，应用标签传递策略之间的协作信息。

基于标签的策略协作模型主要包括 Label 类、PolicyLabelValue 类及关联类 PolicyVariableInSimplePolicyCondition、PolicyValueInSimplePolicyCondition、PolicyVariableInSimplePolicyAction、PolicyValueInSimplePolicyAction[5]。因为目前的各策略信息模型中没有标签类，我们提出的 Label 类、PolicyLabelValue 类是对目前的策略信息模型的扩展，基于 IETF 提出的策略信息模型的关于标签的扩展如图 5 所示。基于不同的策略信息模型，Label 类、PolicyLabelValue 类的继承体系也会不同，其关联类也有所不同，如基于 TMF 的信息模型中 PolicyVariable 与 PolicyCondition 之间的关系通过关联类 PolicyStatementInPolicyCondition 来描述。

Label 类描述了在策略之间传递信息的实体，Label 类中属性都是继承自其父类，这些父类属于当前提出的策略信息模型，图 5 中的属性在文[5]中有详细描述。PolicyLabelValue 类描述了策略之间传递的具体的信息，即 Label 的取值，我们将其定义为 string 类型，在安全管理系统中关于 Label 的具体取值情况可以通过关联类 ExpectedPolicyValuesForVariable[5] 来约束。关联类 PolicyVariableInSimplePolicyCondition、PolicyValueInSimplePolicyCondition、PolicyVariableInSimplePolicyAction、PolicyValueInSimplePolicyAction 表示标签、标签值与策略条件或策略行为之间的关联关系，文[5]中详细描述了这些关联类。一个 Label 类可以与多个策略的条件或行为关联，因此在对 Label 的生命周期的管理中，只有与 Label 相关联的策略条件或策略行为的数目为 0 时，此标签才可以被删除。

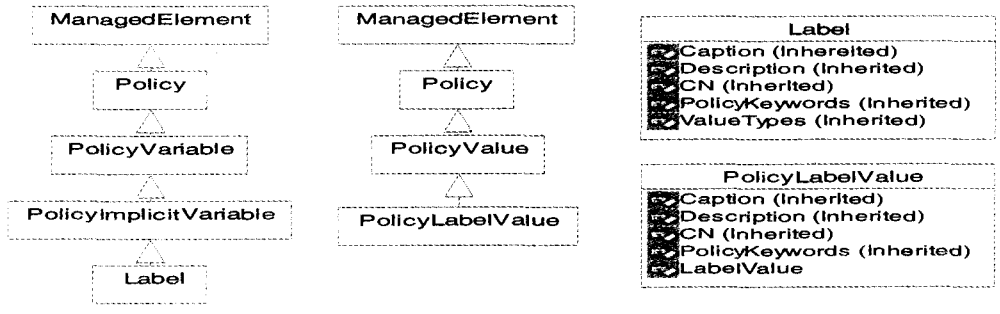

图 5　标签类及其继承体系

在基于策略的安全管理系统中，通过标签传递信息来实现多策略之间的协作。先执行的策略中策略行为要根据策略执行的结果对标签进行赋值，安全管理系统根据系统当前的状态及标签值来决定后续执行的策略。因为先执行的策略的结果作为后续执行的策略的条件，为了保证策略协作的可靠性，安全管理系统可以采用轮询机制来查询标签值。安全管理系统需要周期性查询系统中的标签的值。在基于策略的安全管理系统中，策略作为管理、控制安全管理系统或被管对象的规则被存储在策略库中，通常采用目录服务器作为系统的策略存储库，而系统的管理信息通常被存储在数据库中(如关系数据库)。需要注意的是在策略存储库中只保存策略的信息，其他被管对象的信息保存在系统数据库中。在安全管理系统运行时，根据在系统中执行的策略来改变系统中的管理信息从而控制、管理安全管理系统或被管对象。因此，为实现策略之间的协作，在安全管理系统运行期间，标签的值通常也存储在系统的数据库中，即在实际运行的安全管理系统中标签的具体取值并不是保存在策略库中的，并且在系统的初始化过程中要对这些标签赋初值。

基于标签的策略协作机制可以有效地实现策略之间的协作，能够解决如图 2 所示的安全管理系统中问题。如图 6 所

示，Policy1、Policy2、Policy3 通过标签 Label A 实现协作。在 Process A 中执行 Policy1 时，根据策略的执行结果对标签 Label_A 进行赋值。实施 Policy2、Policy3 时安全管理系统轮询标签 Label_A 的值，根据标签值来确定要执行的策略。在删除与标签关联的策略、策略条件或策略行为时，要查看标签是否与其他的策略相关联，只有当标签不再与其他策略相关联时才可以删除这个标签。需要注意的是先执行的策略的属性 ExecutionStrategy[5,7] 的值必须设置为 3（Do Until Failure），并且对标签进行赋值的策略行为的执行顺序必须设置为最后一项执行。ExecutionStrategy 的值为 Do Until Failure 表示在执行策略的过程中按照指定的顺序来执行策略行为直到出现错误。如果在执行的过程中出现了错误，因为对标签的赋值是最后一项所以标签不会被赋值（对标签赋值这一操作只是对安全管理系统中数据作修改，假设很少或不会出错），所以在先执行的策略出现错误的情况下，后续实施的策略不会被执行，从而实现了策略之间的协作。对如图 2 右侧图所示的复杂的策略协作情况可以用同样的方法来解决。

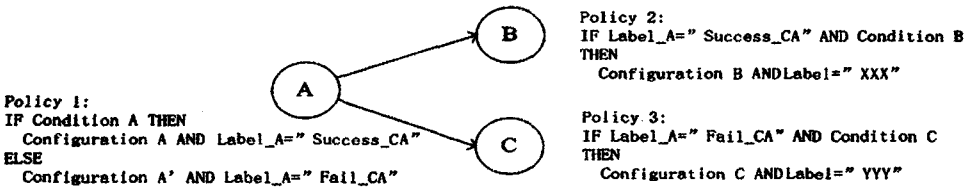

图 6　策略之间的协作实现

采用基于标签的方式来实现策略之间的协作可以应用于安全管理系统中的业务管理层、网络管理层及网元管理层，与仅能应用于网元管理层的采用被管对象传递信息实现策略协作的方法相比，基于标签的方式必须要有标签生命周期的管理如标签的声明、创建、删除。在基于标签的策略协作方式中，因为后续执行的策略采用轮询机制来查询先执行的策略的结果，所以在大规模网络的管理系统中可能存在实时性的问题。

另外，当可以确立多条策略之间的协作关系时，一方面它可以提高策略的灵活性和表达性，另一方面它也可以减少策略描述的复杂性和不明确性[8]，能够从一定程度上减少了策略描述过程中的错误。

4　实验与结果

我们对本文提出的基于标签的策略协作机制进行了验证，在原型系统中我们采用了 IETF 提出的策略信息模型。在原型系统验证的过程中定义了不同数目的策略采用标签的方式分别与另外的对等数目的策略有协作关系，如图 7 所示的是定义 1000 条策略与另 1000 条策略有协作关系时，采用目录服务器浏览器[13]看到存储在策略库中的策略。

在安全管理系统运行时将定义的不同数目的策略读入到系统并保存到策略队列中，然后系统轮询这些策略并根据策略对数据库系统中相应的管理信息进行操作。在我们开发的系统中策略的条件和策略的行为中的变量都是存储在系统数据库中的管理信息。如果策略行为中包括对标签进行赋值，则根据数据库中的信息判断策略条件是否满足，然后对标签进行相应的赋值；如果策略是根据标签值对其他管理信息进行操作，则首先判断标签当前的值然后对数据库中的管理信息进行操作。如在系统中我们定义的两条策略：policycooperation1：IF cond1＝1 THEN Label_1＝“Label_1”，policycooperation2：IF cond2 ＝ 2 AND Label_1 ＝ “Label_1” THEN act1＝1。其中策略 policycooperation1 只是对标签进行赋值，在实际应用中策略 policycooperation1 本身没有意义，但可以验证本文提出的方法。策略中的 cond1、cond2、act1 等都是系统数据库中的分量，它们构成了安全管理系统的管理信息库。

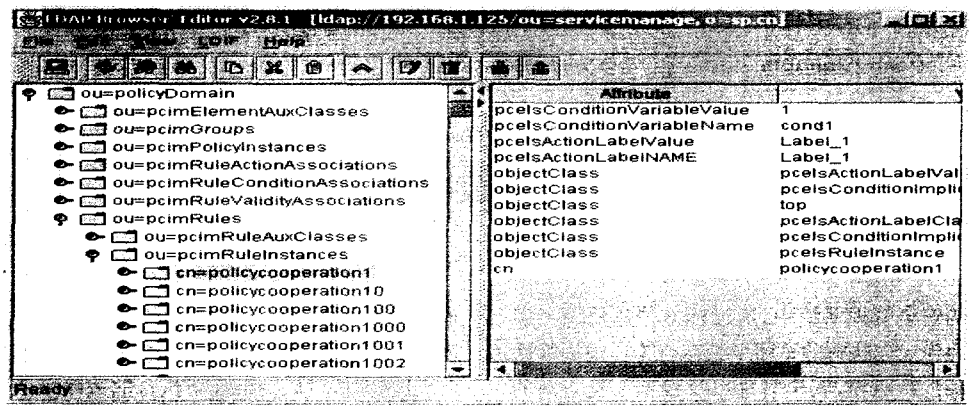

图 7　保存在策略库中的实现协作的策略

在原型系统运行过程中，我们发现系统中不同数目的策略可以实现协作，后续执行的策略都可以根据先执行策略对数据库中的信息进行操作。在系统运行时，有两个过程会影响到系统的性能，一是从策略服务器读取策略然后保存到策略队列中，此过程可以看作是系统的初始化部分；另一过程就是轮询策略队列并根据策略对系统数据库中的管理信息进行操作，因为每一条策略执行时都可能要对系统数据库进行读和写操作。我们对从策略服务器读取策略和轮询队列中的策

略所用的时间在策略数目不同时进行了实验，结果如图 8 所示。

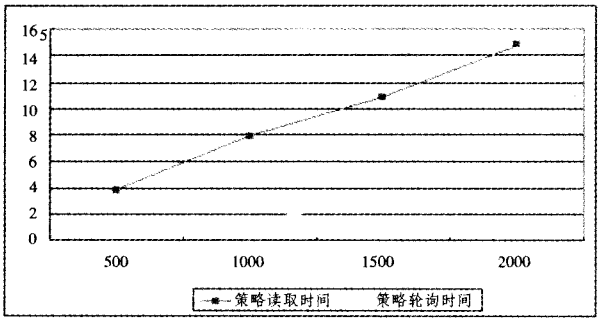

图 8　系统读取策略构造队列及轮询策略队列所用时间(单位:s)

我们的原型系统采用 Java 开发，策略存储库和系统数据库分别采用局域网内的 OpenLDAP[11] 和 Access 承当，系统运行的 PC 机配置为 PIII 800MHZ，128M SDRAM。图 8 中的结果显示系统的响应时间比较长，考虑到我们采用的策略存储库是一种开放源代码的软件，如果采用商业的目录服务器软件，读取策略的时间应该会少一些。图 8 中的轮询策略队列的时间也比较长，我们在实验原型系统中轮询策略队列所用时间时，考虑的是复杂情况下的时间，即策略队列中的每一条策略条件都会满足，每一策略都会被执行并读写系统数据库。在实际系统中，轮询策略队列时一般不会每条策略都被执行，因此轮询策略队列所用时间应该会少一些并且也可以选择其他的系统数据库来提高系统的响应速度。

结论与待研究问题　各研究组织提出的策略信息模型中，策略之间因为缺少传递信息的机制所以难以实现协作。本文分析了基于策略的安全管理系统中策略协作的需求，提出了一种采用标签的方式来传递策略之间的信息从而实现策略协作的方法。本文提出的方法经实验验证能够实现策略之间的协作，但是由于在本文提出的方法中安全管理系统采用轮询的机制来查询策略及其标签的值，所以在大规模安全管理系统中或是安全管理系统需要建立大量标签的情况下，安全管理系统有可能存在实时性的问题。在实际应用中可以采用一些优化算法来提高安全管理系统的响应速度，如可以将策略按照一定的原则保存到多个策略队列中，然后采用多线程的方式来轮询这些策略队列。如何提高安全管理系统的响应速度避免采用本文提出的方法可能带来的实时性问题是我们以后研究的一个方向。

参 考 文 献

1　Verma D C. Simplifying network administration using policy-based management. IEEE Network, 2002, 16(2): 20～26
2　Flegkas P, Trimintzios P, Pavlou G. A policy-based quality of service management system for IP DiffServ networks. IEEE Network, 2002, 16(2): 50～56
3　DMTF. Common Information Model v2. 7, www. dmtf. org/standards/cim
4　IETF RFC3060. Policy Core Information Model -- Version 1 Specification, 2001
5　IETF RFC3460. Policy Core Information Model (PCIM) Extensions, 2003
6　IETF RFC3644. Policy Quality of Service (QoS) Information Model, 2003
7　TMF. Shared Information/Data Model Addendum 1-POL Common Business Entity Definitions - Policy, August 2003
8　Kanada Y, O'Keefe B J. Rule-based Building-Block Architecture for Policy-Based Networking. Journal of Network and Systems Management, 2003, 11(3): 253～275
9　Kanada Y. Policy Division and Fusion: Examples and A Method - or, Multiple Classifiers Considered Harmful. In: 7th IFIP/IEEE International Symposium on Integrated Network Management (IM2001), 2001. 545～560
10　IETF RFC2475. An Architecture for Differentiated Services, 1998
11　IETF RFC3198. Terminology for Policy-Based Management, 2001
12　OpenLDAP homepage. http://www. OpenLDAP. org
13　IETF RFC2252. Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions, 1997
14　LDAP Browser/Editor version 2. 8. 1 http://www. iit. edu/～gawojar/ldap/index. html

(上接第 7 页)

置的策略空间中学习近似最优策略。当然，关于 LOMDPs 和 RMDPs 的文献及其最优策略的算法是较多的，但为了使我们的介绍既简单而又不流于空泛，在我们阅读许多文献后，根据自己的研究，把自己认为最主要的概念和代表不同风格和趋向的算法精选出来，加以较详细的介绍。至于其他作者的工作大都在我们所引的文献中有所涉及，故略而不述，有兴趣的读者可在我们所列的文献中找到有关他们的信息。

最后，对于 LOMDPs 和 RMDPs 今后的发展，我们简单提出几点看法。

(1)所有文献都指出抽象层次上的最优策略是有价值的，但这个论点至今尚未看到清晰的、完整的证明。我们认为，既然 LOMDPs 和 RMDPs 都基于这样一个事实，即逻辑层次上的表述能把基础层次上的状态和行动按某种类似性分类，那么我们就可以运用现代概率论中的条件数学期望概念来论证在抽象层次上的最优策略和实际层次上的具体最优策略之间的关系。我们认为，可以证明抽象层次上最优策略在平均意义上是基础层次上的最优策略。

(2)所有文献都把抽象层次和基础层次交织在一起，这为寻求抽象最优策略带来了很大麻烦。我们认为，也可以把这两个层次"完全"分离，转变为两个层次上的马尔可夫决策过程，这样在抽象层次上，利用它比实际状态数目较少的优点，较简单地运用传统方法求抽象最优策略，再按某种演算具体在实际状态空间实施，可能起到事半功倍之效。这个思路更符合人类思维特点，因为人们总是首先在大体上思考问题解决方案，然后再付诸实践，具体实现之。

参 考 文 献

1　Kersting K, De Raedt L. Logical Markov Decision Programs. In: Working Notes of the IJCAI-2003 Workshop on Learning Statistical Models from Relational Data (SRL-03), Acapulco, Mexico, 2003. 63～70
2　Kersting K, De Raedt L. Logical Markov Decision Programs and the Convergence of Logical TD(λ). In: Proceeding of The 14th International Conference of Inductive Logic Programming, Porto, Portugal, 2004. 180～197
3　van Otterlo M. Reinforcement Learning for Relational MDPs. In: Proceedings of the Annual Machine Learning Conference of Belgium and the Netherlands. Brussels, Belgium, 2004. 138～145
4　Kersting K, van Otterlo M, De Raedt L. Bellman goes to Relational. In: Proceedings of the 21st International Conference on Machine Learning, Banff, Canada, 2004
5　Fern A, Yoon S, Givan R. Approxiamte Policy Iteration With a Policy Language Bias: Solving Relational Markov Decision Processes. Journal of Artificial Intelligence Research, 2006, 25: 75～118
6　Boutilier C, Reiter R, Price B. Symbolic Dynamic Programming for First-Order MDPs. In: Seventeenth International Joint Conference on Artificial Intelligence(IJCAI-01). Seattle, USA, 2001. 690～700