ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.20, No.10, October 2009, pp.2776−2786
doi: 10.3724/SP.J.1001.2009.03491

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

面向分布式证明的信任协商策略语言和方法*

王小峰+, 苏金树, 张 强, 张一鸣

(国防科学技术大学 计算机学院,湖南 长沙 410073)

Distributed Proving Oriented Language and Method for Trust Negotiation

WANG Xiao-Feng+, SU Jin-Shu, ZHANG Qiang, ZHANG Yi-Ming

(College of Computer, National University of Defense Technology, Changsha 410073, China)

+ Corresponding author: E-mail: xfwang.nudt@gmail.com, http://xfwang.nudt.googlepages.com

Wang XF, Su JS, Zhang Q, Zhang YM. Distributed proving oriented language and method for trust negotiation. *Journal of Software*, 2009,20(10):2776−2786. http://www.jos.org.cn/1000-9825/3491.htm

Abstract: Most existing trust negotiation languages can not simultaneously have the following important functions: Distributed trust proving, complicated access control definition and negotiation-related constraints. Based on RT (role-based trust-management) language, this paper proposes a distributed trust proving and negotiation orientated language RTP (role-based trust proving). It can support distributed trust proving, define complicated roles, protect the policy's sensitive information and avoid unrelated credential fetching. Both the syntax and semantics of RTP are introduced. The paper also designs a distributed trust proving and negotiation algorithm based on RTP to demonstrate the efficiency of RTP. Experimental results show that the algorithm supports the functions aimed by RTP, and outperforms the traditional trust negotiation in terms of both time and number of credential transfers.
Key words: trust negotiation; distributed proving; logic signature; credential release; proof hint

摘 要: 现有信任协商语言对复杂的访问控制策略和协商策略以及信任分布式证明方法的支持都不够全面.在RT(role-based trust-management)语言基础上提出一种面向信任分布式证明和协商的策略语言 RTP(role-based trust proving),其特点是能够支持信任分布式证明方法,可以定义复杂角色,保护信任证敏感信息并能避免信任证盲目搜索.给出了RTP语言及其推理规则的语法语义描述,介绍了一种基于RTP语言的信任分布式证明协商示例算法.实验结果表明,该算法支持RTP语言的功能,且比传统信任协商方法有很大的性能提升.
关键词: 信任协商;分布式证明;逻辑签名;信任证释放;启发规则
中图法分类号: TP393 **文献标识码**: A

随着分布式网络应用的日益普及,如何对分布式环境中跨安全域的资源进行访问授权管理,成为当前的研究热点.其中一个主要的研究方向为自动信任协商(automated trust negotiation,简称 ATN).与传统的基于访问控制列表的方法相比,它的特点是通过信任证和访问控制策略的交互披露,资源的请求方和提供方自动地建立信任关系[1].目前,对 ATN 的研究主要有两方面:信任证的分布式放置收集方法以及策略语言的形式化描述分析.

* Supported by the National Natural Science Foundation of China under Grant Nos.90604006, 60303012 (国家自然科学基金); the National Basic Research Program of China under Grant No.2005CB321801 (国家重点基础研究发展计划(973))

Received 2007-11-11; Revised 2008-04-15; Accepted 2008-10-06; Published online 2009-05-07

ATN 策略语言的内容可分为资源访问控制策略和信任协商策略,一般情况下,策略语言将两者分离并分别进行描述[2].

信任证分布式放置收集方法包括传统的信任协商和分布式证明两种方法[3].Bauer[4]分析认为,分布式证明方法比传统的信任协商方法更高效,并能克服后者的一些缺点,如信任求证节点负载过大;策略语言的资源访问控制策略利用角色对权限的授权和委托信息进行定义,角色的描述能力越强,对应用的支持就越大;策略语言的信任协商策略需提供对信任证敏感信息的保护并避免信任证的盲目搜索.综合上面 3 方面的特点,开发出来的一种策略语言具有强大的资源访问控制描述能力,能够保护信任证敏感信息,避免信任证的盲目搜索并能支持信任分布式证明成为本文的目标.

目前一些研究工作都只是对我们目标的某个方面提出解决办法,如文献[4]通过定义 say 谓词提出了一种信任分布式证明算法,但该算法不支持复杂的资源访问控制和信任证信息保护;Li 在文献[5]中提出一种资源访问控制策略 RT(role-based trust-management)语言,支持参数化和连接两种复杂角色,但不能支持敏感信息保护;Li[6]在 RT 语言的基础上通过加密信任证技术支持敏感信息保护,但这两者都不能支持信任的分布式证明方法.以上语言和方法之所以不能对信任分布式证明方法以及复杂的资源访问和信任协商策略提供全面支持,主要是缺少一个功能全面的策略描述语言.

本文提出一种面向信任分布式证明和协商的策略语言 RTP(role-based trust proving),该策略语言对 RT 语言进行改进和功能拓展,其特点是:a) 增加 lsign 语法支持信任分布式证明;b) 访问控制策略延续了 RT 语言描述能力强的优点;c) 信任协商策略通过增加谓词 release 保护信任证信息,增加启发规则避免信任证盲目搜索.另外,本文给出了 RTP 语言及其推理规则的语法语义描述,提出一种基于 RTP 语言的信任分布式证明协商示例算法,分析了该算法对 RTP 语言特点的支持,以及它所带来的性能提升.

文章第 1 节介绍研究相关工作.第 2 节给出 RTP 语言的框架和语法.第 3 节介绍 RTP 语言的推演规则和语义.第 4 节描述、分析一种 RTP 语言信任分布式证明协商算法.最后总结本文工作.

1 相关工作

资源访问控制策略作为策略语言的基本要素,主要描述资源访问权限的授权和委托信息,文献[5–10]从不同方面(如义务[7]及角色管理[8])对该策略进行描述.信任协商策略是协商双方在建立信任关系中所采取的暴露访问控制策略的方式[11],包括协商过程中如何决定向谁索取信任证、索取什么信任证;另外一个重要方面是对信任证敏感信息提供保护[12],如我们不想让别人从身份信任证中知道自己的年龄,涉及的技术主要有隐藏信任证[13]和无记忆属性证书[14]等.

文献[5,15,16]系统地介绍了 RT 系列语言,语言中角色由实体和角色名组成,如 Bob.Fridend 表示实体 Bob 定义的角色名为 Fridend 的角色.RT 语言引入了 4 种基本访问控制策略,定义了参数化角色,如 Bob 的男性朋友(Bob.Fridend(boy))以及连接角色,如 Bob 朋友的老师(Bob.Friend.Teacher)等描述功能强大的角色,能够很好地满足应用需求.该语言的缺点是不能对信任证的敏感信息提供保护.Li[6]在 RT 语言的基础上通过信任证加密技术对访问策略的敏感信息提供保护,但这两者都没有解决协商中盲目索取信任证的问题.

文献[1]定义了传统的分布式协商模型,信任请求方不断地向信任证拥有者请求下载信任证,信任证拥有者判断请求是否满足策略要求,不满足则向请求方发送要求,请求方相应做出反应.如此不断迭代,最终建立信任关系.例如一个信任授权中心 CAS 有 RT 语言描述的规则 *CAS.trust←CAS.honor* 和 *CAS.honor←Alice*,意思是 CAS 将 Alice 认证为自己尊敬的成员,尊敬的成员也是信任的成员.当 Alice 向 CAS 求证自己是否是其信任成员时,两者之间的传统协商过程如下:1) Alice 发送认证请求(?*CAS.trust←Alice*);2) CAS 返回要求(?*CAS.honor←Alice*);3) Alice 发送请求(?*CAS.honor←Alice*);4) CAS 返回信任证(*CAS.honor←Alice*);5) Alice 发送信任证(*CAS.honor ←Alice*);6) CAS 认证成功并发送信任证(*CAS.trust←Alice*).协商的前半部分(步骤 1)~步骤 3))都是交换请求,后半部分(步骤 4)~步骤 6))则交换信任证.基于 RT 语言的方法[5,6,15,16]也采用传统的分布式协商方法,它基于信任证类别收集信任证,收集策略包括前向搜索、后向搜索和混合搜索.

传统的信任协商存在如下几个问题:1) 授权服务器必须时刻处于工作状态,对信任签名请求进行判断并给出签名.这可能导致拒绝服务攻击,不符合信任管理给服务器减负的思想;2) 当信任请求方向其他节点索取信任证时,很可能因为索取太多权限或太少权限而造成失败[2];3) 信任请求方进行信任的全部证明工作,且当有多条访问策略匹配时,只能盲目索取相关的所有信任证,使信任请求方成为影响系统性能的瓶颈.

在上文的例子中,CAS 很显然会得到 *CAS.trust*←*Alice* 的结论,但由于传统信任协商中服务方只提供信任证签名服务并不为请求进行证明,因此,Alice 和 CAS 进行了很多无谓的交互工作.文献[2,4,17]讨论了信任的分布式证明方法,其思想是信任服务方为请求进行力所能及的逻辑证明,从而减少信任签名请求并减轻信任请求方的信任协商负担.PeerAccess[2]给出一个了支持信任分布式证明的语言,但它不能支持 RT 中的参数化角色和连接角色,没有给出分布式信任证明的实现方法.Lujo[4]通过定义 say 语法支持信任分布式证明,并给出一种分布式证明算法,实验从远程请求次数和信任证拥有者被访问的次数两个角度表明了分布式证明带来的性能提升.其缺点是系统策略语言只能实现简单的资源访问控制和委托,不提供信任证敏感信息的保护,算法没有给出证明过程中的证明规则.

2 RTP 语言框架及语法

本文在 RT 语言的基础上进行拓展形成 RTP 语言,其新的特点是能够支持信任的分布式证明、信任证信息保护以及证明的启发式规则.信任证可以认为是策略规则的数字签名[18],因此对信任证和策略规则可采用统一的语言进行描述.本节首先介绍 RTP 语言所需描述知识的框架,然后介绍适应该需求的 RTP 语言语法.

2.1 RTP语言知识框架

在信任协商中,节点(principal)表示一个进程或 Agent.它也可以由一个公钥表示,能够对信任信息进行签名或发出信任请求.系统中所有的节点构成一个集合 N.每个节点 A 都有一个局部知识库 KB(knowledge base),记为 P_A,系统中所有节点的局部知识库的集合组成系统的全局知识库 P.我们的目标信任协商系统需要节点的局部知识库 KB 支持资源访问控制、信任证信息保护、信任协商证明启发式以及信任的分布式证明功能.因此,我们将 KB 设计成一个通用可扩展的结构,如图 1 所示,它包括 5 个部分:信任证释放规则 CRP、证明启发规则 PH、服务访问控制规则 SGP、信任证集 CS 和证明结果集 PS.其中,信任证释放规则和证明启发规则构成 KB 的信任协商策略,服务访问控制规则构成 KB 的访问控制策略.各部分介绍如下:

- 信任证释放规则定义信任证的输出条件,保护信任证敏感信息.
- 证明启发规则指出信任协商时的推理规则,如向谁索取信任证,索取什么信任证.
- 服务访问控制规则是本节点定义的基于角色的访问控制规则.
- 信任证集包含协商过程中从其他节点收到的直接签名信任证.
- 证明结果集包含自己或从其他节点收到的证明中得到的逻辑结论.

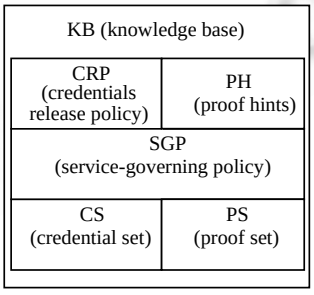

Fig.1 Knowledge base frame for RTP

图 1 RTP 语言局部知识库框架

为了便于理解局部知识库和 RTP 语言,图 2 中的例 1 给出一个书店 Org 的局部知识库,这里我们只介绍各

规则的直观含义,语法由下一节给出介绍.CS 缓存的直接签名规则 *cs*1 表示 Alice 是书店管理员 Reg 的贵宾.PS 的缓存结论 *ps*1 表示 Alice 相信 Bob 是自己的信任者.服务访问规则 *s*1 规定被管理员认为积分大于 1 000 的顾客是书店 Org 的贵宾;*s*2 规定管理员认为的贵宾的信任者是该书店的信任者;*s*3 规定 Carla 是该书店的一个贵宾.信任证释放规则 *c*1 表示,只有书店 Org 的贵宾能把 *s*2 规则发送给其他节点,且只能传给书店的信任者.证明启发式规则 *p*1 表示根据信任证 *cs*1 和规则 *s*2,要证明某人是 Org 的信任者,还需证明该人是 Alice 的信任者.*p*2 表示要证明某人是 Alice 的信任者,必须向 CAS 求证.

Credential Set (CS) Proof Set (PS)
*cs*1. *Reg*(*sign*).*honored*←*Alice* *ps*1. *Alice*(*lsign*).*trust*←*Bob*

Service Governing Policy (SGP)
*s*1. *Org*(*lsign*).*honored*←*Reg*(*lsign*).*customer*(*score*>1 000)
*s*2. *Org*(*lsign*).*trust*←*Reg*(*lsign*).*honored*.*trust*
*s*3. *Org*(*sign*).*honored*←*Carla*

Credential Release Policy (CRP)
*c*1. *Org*(*lsign*).*release*(*s*2,*B*,*A*)←(*Org*(*lsign*).*honored*←*B*∩*Org*(*lsign*).*trust*←*A*)

Proof Hints (PH)
*p*1. *Org*(*lsign*).*prove*(*Org*(*lsign*).*trust*,*cs*1,*s*2),{*Alice*(*lsign*).*trust*})
*p*2. *Org*(*lsign*).*find*(*Alice*(*sign*).*trust*,*CAS*)

Fig.2 Example 1: Knowledge base for Bookstore Org
图 2 例 1:书店 Org 局部知识库 KB

2.2 RTP语言语法

图 3 给出了 RTP 语言的 BNF 描述,下面解释中出现的数字指的是图 3 中该数字行对应的语法.系统中每个节点都有一个局部知识库 Knowledge base(第 4 行),它由 3 种类型的规则组成:服务访问规则、信任证释放规则和证明启发式规则.

1. ⟨set of *X*⟩::=∋|⟨*X*⟩⟨set of *X*⟩
2. ⟨list of *X*⟩::=⟨*X*⟩|⟨*X*⟩“,”⟨list of *X*⟩
3. ⟨conj of *X*⟩::=⟨*X*⟩|⟨*X*⟩“∩”⟨conj of *X*⟩
4. ⟨knowledge-base⟩::=⟨set of policy-expr⟩|⟨set of cred-rele⟩|⟨set of proof-hint⟩
5. ⟨policy-expr⟩::=⟨role⟩“←”⟨policy-body⟩
6. ⟨policy-body⟩::=⟨prin⟩|⟨conj of role⟩
7. ⟨role⟩::=⟨prin⟩(“(sign)”|“(lsign)”)“.”⟨role-terms⟩
8. ⟨role-terms⟩::=⟨role-term⟩|⟨role-term⟩“.”⟨role-terms⟩
9. ⟨role-term⟩::=⟨role-name⟩|⟨role-name⟩“(”⟨list of field⟩“)”
10. ⟨field⟩::=⟨field-name⟩|⟨lelation⟩⟨constant⟩
11. ⟨cred-rele⟩::=⟨rele-stmt⟩“←”“(”⟨conj of policy-expr⟩“)”
12. ⟨rele-stmt⟩::=⟨prin⟩(“(sign)”|“(lsign)”)“.”⟨release-term⟩
13. ⟨rele-term⟩::=“release”“(”(⟨policy-expr⟩|⟨cred-rele⟩)⟨prin⟩⟨prin⟩“)”
14. ⟨proof-hint⟩::=⟨prin⟩(“(sign)”|“(lsign)”)“.”(⟨find-term⟩|⟨prove-term⟩)
15. ⟨find-term⟩::=“find”“(”(⟨role⟩“,”⟨prin⟩“)”
16. ⟨prove-term⟩::=“prove”“(”(⟨role⟩|⟨rele-stmt⟩)“,”⟨grp of policy-expr⟩“,”⟨grp of role⟩“)”
17. ⟨grp of *X*⟩::=“{”⟨list of *X*⟩“}”

Fig.3 BNF description for RTP
图 3 RTP 语法的 BNF 描述

1. 服务访问规则 SGP

服务访问规则(第 5 行)由头部和尾部组成,规则头部为一个角色 *f*,规则尾部(第 6 行)有两种类型:一种只有一个节点标示 prin,表示该节点属于规则头部的角色;另一种形式为 $f_1 \wedge \ldots \wedge f_m$,由一组角色 f_i 的合取组成,表示如

果某节点属于规则尾部所有的角色,则该节点属于规则头部的角色.第 2 种形式的服务访问规则省略了一个变量,等同于($f\leftarrow X$)←($f_1\leftarrow X$)∧…∧($f_m\leftarrow X$).

角色(第 7 行)表示一个节点集合,它有两种类型:直接签名角色和逻辑推导角色.直接签名角色形式为 $A(sign).R$,逻辑推导角色形式为 $A(lsign).R$,其中,A 为节点,R 为一个角色项或角色项的连接(第 8 行).角色项的连接由角色项之间通过点符号连接而成,表示一个连接角色,如例 1 中的 $s2$ 规则.角色项(第 9 行)由角色名或由角色名和一组属性域组成,属性域(第 10 行)包括域名、域值以及变量关系,域值是属性定义域中的常量.带属性域的角色也称参数化角色,如例 1 中 $s1$ 规则:Reg 是节点,customer 是角色名,score 是属性域名,1 000 是域值,>是变量关系.后文中一般用 $R,R_1,\ldots$表示角色项变量,$f,f_1,\ldots$表示角色变量.

如果节点 B 属于直接签名角色 $A(sign).R$,则表示节点 A 签名认定 B 属于自己的角色项 R.如果节点 B 属于逻辑推导角色 $A(lsign).R$,表示某节点通过知识库推导认为 B 属于 A 定义的角色项 R.传统信任协商中的角色都是直接签名角色,角色关系必须由角色定义节点签名才有效,即信任关系须由信任定义节点亲自审核,这给信任定义节点带来负担.逻辑推导角色关系就不必角色定义节点亲自签名,任何节点只要有足够证据就可以认为信任关系成立,这是对直接签名角色的一种放松.例 1 中,由规则 $s2$ 和信任证 $cs1$ 以及结论 $ps1$,Org 可以推导出 $Org(lsign).trust\leftarrow Bob$ 的结论,如果将 $s2$ 规则的尾部角色改成直接签名角色即 $Org(lsign).trust\leftarrow Reg(sign).honored.trust$,则上述结论不能成立,因为 $Reg(sign).honored.trust\leftarrow Bob$ 信任关系需由管理员 Reg 亲自审核签名才能成立,不能由其他节点推导得到.

2. 信任证释放规则 CRP

很多情况下,信任关系的定义节点希望能够控制信任证的传播并保护信任证的敏感信息.为此,系统对 SGP 规则、CRP 规则和 CS 中的信任证的输出进行限制,它们可以由节点 A 发送出去,但必须满足两个条件:i) 信任证所指的规则在节点 A 是正确的;ii) 必须满足节点 A 保存的该信任证的释放规则.

信任证释放规则(图 3 中第 11 行)由释放描述和释放条件两部分组成:释放条件是一组服务访问规则的合取,表示只有当所有服务访问规则都满足时,释放描述中的情况才准许发生;释放描述(图 3 中第 12 行)的定义与角色类似, $A(sign).rele\_term$ 和 $A(lsign).rele\_term$ 分别表示直接签名和逻辑推导的访问规则,$rele\_term$ 是释放规则项.释放规则项(第 13 行)由 $release$ 算子加参数表示,参数包括一条规则 ψ和两个节点 A,B,表示节点 A 可以将规则 ψ传给 B.示例中,$c1$ 规则为信任证访问规则的一个例子,该规则可避免 $s2$ 规则被不相关节点得到.

一个节点只能给自己知识库中的 SGP 规则定义释放规则,当未定义它的访问释放规则时,系统默认为释放条件为真即可以释放.系统不准许节点给知识库中的 CRP 规则以及 CS 中的信任证定义释放规则.CRP 规则的释放规则由系统按照自释放推演规则自动得到,这将在下一节 RTP 语言推演规则中加以讨论.CS 中信任证的释放规则只能从其定义节点得到,当未定义它的访问释放规则时,系统默认为不可以释放,这样信任证签名节点定义的信任证释放规则就能在整个系统中控制信任证的传输.

3. 证明启发式规则 PH

证明启发式规则(见图 3 中第 14 行)也有直接签名和逻辑推导两种形式,包括寻找启发和证明启发两种类型.寻找启发项(见图 3 中第 15 行)由算子 $find$ 和参数组成,参数包括角色 R 和节点 A,表示如果想证明某节点属于角色 R,需向节点 A 寻求帮助;证明启发项(见图 3 中第 16 行)由算子 $prove$ 和参数组成,参数包括角色 f 或只含一个变量的释放规则项 RE、规则集 PS 和角色集 RS,表示如果要证明某节点属于角色 f 或规则项 RE 定义变量的值域,根据规则集 PS,还需证明节点属于角色集 RS 中的所有角色.两种启发式规则示例见图 2 中的规则 $p1$ 和 $p2$.

3 RTP 推演规则语义

3.1 RTP语言推演规则

RTP 语言也是基于 Datalog$^{C[16]}$的逻辑描述语言,因此,一阶逻辑的公理定理 RTP 语言都适用.RTP 语言知识库的推演规则由定义 1 给出.

定义 1(RTP 语言推演规则). 推演规则包括以下 6 条:

1) 实例推演 VI(variable instantiation). 如果规则$\phi \in P_A$,P_A 为节点 A 的知识库,x 为ϕ中一个变量,b 为该变量定义域上的一个常量,则 $P_A \mapsto sub_b^x\phi$, $sub_b^x\phi$ 表示用 b 替代ϕ中所有 x 的出现.若ϕ="$f \leftarrow f_1 \wedge \ldots \wedge f_m$",$f,f_1,\ldots,f_m$ 为角色,则 $sub_b^x\phi$ ="$(f \leftarrow b) \leftarrow (f_1 \leftarrow b) \wedge \ldots \wedge (f_m \leftarrow b)$".

2) 角色连接 RL(role link). 如果规则ϕ="$A(lsign).R_1 \leftarrow B$"且$\phi \in P_A$,则 $P_A \mapsto \phi'$,ϕ'="$A(lsign).R_1.R_2 \leftarrow B(lsign).R_2$".

3) 假言推理 MP(modus ponens). 如果规则ϕ="$S \leftarrow S_1 \wedge \ldots \wedge S_m$"$\in P_A$,$S,S_1,\ldots,S_m$ 为访问控制规则且$\{S_1 \ldots S_m\} \subset P_A$,则 $P_A \mapsto S$.

4) 签名推演 SD(signature derivation). 如果一个直接签名规则$\phi \in P_A$,则 $P_A \mapsto \phi'$,ϕ'为ϕ对应的逻辑签名规则;如果一个被节点 A 逻辑签名的规则$\phi' \in P_A$,则 $P_A \mapsto \phi$,ϕ为ϕ'对应的直接签名规则.

5) 自释放规则 1(SR1(self_release 1)). 如果规则$\phi \in P_A$,ϕ="$B(lsign).release(\psi,C,D) \leftarrow S_1 \wedge \ldots \wedge S_m$",则 $P_A \mapsto \varphi$,φ="$B(lsign).release(\phi,A,C)$".

6) 自释放规则 2(SR2(self_release 2)). 如果$\{\sigma,\psi\} \subset P_A$,$\sigma$="$B(lsign).release(\phi,C,D) \leftarrow S_1 \wedge \ldots \wedge S_m$",$\psi$="$B(lsign).release(\phi,D,E) \leftarrow S_1' \wedge \ldots \wedge S_n'$",则 $P_A \mapsto \varphi$,φ="$B(lsign).release(\psi,A,C)$".

在上面 6 条推演规则中,实例推演和假言推理是一阶逻辑的通用规则,下面,我们对其他 4 条规则的直观意义给出解释.角色连接规则表示,如果节点 A 接受 B 节点作为自己角色 R_1 的成员,则 B 定义的角色 R_2 也将得到 A 的承认.由签名推演规则可以知道,如果一条直接签名规则ϕ在某个节点 A 为真,则ϕ对应的逻辑推导规则ϕ'在 A 也为真;如果一条 A 节点逻辑推导的规则ϕ'在节点 A 为真,则ϕ'对应的直接签名规则ϕ在 A 也为真.也就是说,直接签名规则可以直接推出逻辑推导规则,但逻辑推导规则只有在签名节点才能得到对应的直接签名规则.

CRP 规则的释放规则只能由自释放推演规则得到.自释放规则 SR1 的意思是,如果节点 B 授权节点 C 发布规则ψ,则 B 允许任何节点 A 通知 C 它得到的 B 的授权.自释放规则 SR2 在 SR1 的基础上进一步加长了信任证的授权发布链,意思是,如果节点 B 授权节点 C 发布规则ϕ给节点 D,同时授权 D 发布规则ϕ给节点 E,则 B 允许任何节点 A 将 D 有权给 E 发布规则ϕ的消息通知 C,以便 C 将该消息转发给 D.这样,D 有权给 E 发布规则ϕ的消息就可以沿着 A-C-D 传输,进而让 D 得到 B 的授权.在定义了上面 6 条推导规则之后,全局知识库的推导证明可由定义 2 给出.

定义 2(推导证明). 全局知识库 P 在 A 节点推导出规则ϕ当且仅当存在一个推导序列 $P^0,P^1,\ldots,P^n$,有 $P^0=P$,$\phi \in P_A^n$,其中,$P^{i+1}(0 \le i < n)$由 P^i 实行一个推演规则得到.此时,$P^0,P^1,\ldots,P^n$ 称为ϕ在 A 节点的证明序列,表述为 $P_A \mapsto \phi$. □

3.2 RTP语言语义

定义 3(知识库的 H 域解释 W). 知识库 P 的 Herbrand 域(H 域)解释 W 为知识库原子集(作用在 H 域上的所有角色关系)中各原子命题真假值的设定.

知识库的 H 域包括系统中所有需要判定的对象,如系统中的所有节点.原子命题是不带变量的角色关系,因此,知识库的 H 域解释 W 为知识库中所有角色关系作用在系统中任何节点的真假值设定.利用知识库的 H 域解释可定义知识库的解释和模型如下:

定义 4(知识库解释 I). 全局知识库 P 的解释 I 是一个包含各节点局部解释的集合 $I=\{I_A|A \in N\}$,其中,I_A 是全局知识库 P 在 A 节点的 H 域解释集合.

定义 5(模型|=). 设 I 为知识库 P 的一个解释,I_A 是 P 在节点 A 的 H 域解释集合.

1) 如果规则ϕ="$A(lsign).Role_1.Role_2 \leftarrow B$"是没有变量的规则,$I|=_A\phi$当且仅当对 I_A 中的每个 H 域解释 W,$\exists C(C \in N)$:S_1="$A(lsign).Role_1 \leftarrow C$"$\in W$ 且 S_2="$C(lsign).Role_2 \leftarrow B$"$\in W$.

2) 如果规则ϕ="$S \leftarrow S_1 \wedge \ldots \wedge S_m$"是一个没有变量的逻辑签名规则,$I|=_A\phi$当且仅当对 I_A 中的每个解释 W,要么 $S \in W$,要么$\exists i(1 \le i \le m)$,$S_i \notin W$.

3) 如果ϕ是一个带有变量的逻辑签名规则,$I|=_A\phi$当且仅当对ϕ进行实例化得到的每个没有变量的逻辑签名规则ϕ',$I|=_A\phi'$.

4) 如果ϕ是一个直接签名规则,$I|=_A\phi$当且仅当ϕ是由 A 签名的,且 $I|=_A\phi'$,ϕ'是ϕ对应的逻辑签名规则.

5) 如果Φ是一个规则集,$I|=_A\Phi$当且仅当对任意$\phi\in\Phi$,有 $I|=_A\phi$.此时称解释 I 在 A 节点是Φ的模型.

对全局知识库 P,$I|=P$ 当且仅当对任意节点 $A\in N$,有 $I|=_AP_A$.这种情况下,我们称解释 I 是 P 的模型,记为 $\underline{P}$.模型 $\underline{P}$ 反映的是节点的局部视图,其直观意义就是,系统中各节点根据自己知识库可以知道的所有角色属性关系.模型 $\underline{P}$ 有下面两个定理:

定理 1(模型的可靠性). 对任意全局知识库 P、节点 A 和规则ϕ,如果 $P_A\mapsto\phi$,则 $\underline{P}|=_A\phi$.

证明: $P_A\mapsto\phi$,设有证明序列 $P^0,P^1,\ldots,P^n$, $P^0=P,\phi\in P_A^n$.下面用关于 $i(0\le i\le n)$的第二归纳法证明 $\underline{P}|=_AP^j$ $(0\le i\le n)$,即证明 P^i 比 P^{i-1} 新增的规则在解释 $\underline{P}$ 中为真.假设 P^{i-1} 到 P^i 运行推导规则分别如下:

1) 实例推演.如果一个逻辑签名规则ϕ对解释 $\underline{P}$ 在 A 节点为真,则规则ϕ的所有实例都会出现在 $\underline{P}_A$ 的每个 H 域解释中.因此,实例推演得到的规则对解释 $\underline{P}$ 在 A 节点为真.

2) 角色连接.如果规则ψ="$A(lsign).R_1\leftarrow B$"和规则ϕ="$B(lsign).R_2\leftarrow C$"对解释 $\underline{P}$ 在 A 节点都为真,即ψ和ϕ的任意实例化ψ'和ϕ'出现在 $\underline{P}_A$ 中的每个 H 域解释 W 中,由模型的定义可知,φ="$A(\text{lsign}).R_1.R_2\leftarrow C$"也将出现在 $\underline{P}_A$ 的每个 H 域解释中.因此,规则σ="$A(lsign).R_1.R_2\leftarrow B(lsign).R_2$"为真.

3) 假言推理.假设规则ϕ="$S\leftarrow S_1\wedge\ldots\wedge S_m$"对解释 $\underline{P}$ 在 A 节点为真,S'和 $S_1'...S_m'$ 为规则ϕ任意实例化后 S 和 $S_1\ldots S_m$ 对应的结果,且 $S_1'...S_m'$ 出现在 $\underline{P}_A$ 中的每个 H 域解释 W 中.若 S'不在 W 中,则得到规则ϕ为假,与初始假定矛盾.因此 S'也必在 W 中,即 S 对解释 $\underline{P}$ 在 A 节点为真.

4) 签名推演规则和自释放规则本身即为语义限制所定义,因此它们推导的结果对解释 $\underline{P}$ 为真. □

定理 2(模型的完全性). 对任意知识库 P、节点 A 和规则ϕ,如果 $\underline{P}|=_A\phi$,则 $P_A\mapsto\phi$.

证明:在系统的推演规则中,签名规则、自释放规则和角色连接规则的语义直接与推导证明的语法对应,因此,$\mapsto$ 语法可直接得出.实例推演和假言推理的证明与一阶逻辑证明一样,可以通过一阶逻辑的协调性来证明,因篇幅有限,这里不作具体展开. □

4 基于 RTP 语言的信任分布式证明算法

为体现 RTP 语言的特点,本节介绍一种利用 RTP 语言及其推演规则实现分布式证明协商 DPN(distributed proving and negotiation)的算法,并分析该算法的性能.

4.1 DPN算法描述

DPN 算法如图 4 所示,*recursive_prove* 函数递归地将目标分为子目标进行证明.算法输入为一个证明目标列表,目标的表示形式与 SGP 规则一样,包含头部 h 和尾部 b.输出为证明目标所需的知识库中的规则,如果协商失败,则输出本次证明已经找到的与目标相关的规则和失败 *fail* 消息.函数主要对目标列表的第 1 个目标进行证明(见图 4 中第 3 行),当输入目标为空时,算法证明成功返回(见图 4 中第 2 行).*proof_locate* 函数输入一个规则头部角色,输出为知识库寻找启发式建议的求证节点.算法根据 *proof_locate* 函数的输出采用两种证明方法:本地规则推演(见图 4 中第 10~19 行)和远程证明调用(见图 4 中第 5~9 行).

本地规则推演分两种情况进行证明:一种是当第 1 个目标就在知识库中时(第 10 行),直接证明其他目标(第 11 行),并返回证明中与目标相关的规则(第 12 行);另一种情况是第 1 个目标不能直接得到,需匹配 6 条推理规则进行信任证明(第 13 行),其中,*Rules* 和 P 为待匹配的变量.算法中将推理规则用(*f*|*RE*,*PS*,*RS*)数据结构代表的证明启发规则表示,意为如果要证明某节点属于角色 *f* 或释放规则项 RE 定义的变量值,根据规则集 PS 和某条推理规则,还需证明该节点属于角色集 RS 中的角色.信任证明对所有能够匹配目标头部的推演规则进行试探(第 13 行),将其子目标实例化(第 14 行)并递归证明(第 15 行),如果证明成功,则继续证明其他目标(第 16 行).当子目标和其他目标都证明成功时返回相关规则集(第 17 行),否则任一目标证明失败都记录相关规则集(第 18

行)并尝试匹配下一条规则(第 19 行).如果本地推演的两种情况证明都无法成功,则返回相关规则集和 *fail* 消息(第 19 行).

```
1.  Rule-Set recursive_prove(list goals)
2.    If (goals=[]) then return                    /*No more goals, succeed and return*/

3.    [h,b]←first(goals)                           /*Prove the first goal in the list*/
4.    server←proof_locate(h)                       /*Proof heuristic decides how to prove*/

5.    if (server)                                  /*Need a remote proof */
6.      if (fail∈(α←rpn_client(server,[h,b])))     /*The local API for remote proof*/
7.        return α                                 /*Proof failed, return the rules*/
8.      β←recursive_prove(rest(goals))             /*Prove remainder of goals*/
9.      return α∪β                                 /*Return the rules*/

10.   if [h,b]∈KB                                  /*Prove locally and goal is in the KB*/
11.     β←recursive_prove(rest(goals))             /*Prove remainder of goals*/
12.     return β∪[h,b]                             /*Return the rules in KB*/

13.   for each (h,Rules,P)∈PH                      /*Try to match a proof hint*/
14.     p←instantiate(P,b)                         /*Instance the subgoals*/
15.     if (fail∉(α←recursive_prove(p)))           /*Prove subgoals*/
16.       β←recursive_prove(rest(goals))           /*Prove remainder of goals*/
17.       if (fail∉β then return α∪β∪Rules         /*Proof succeed, return KB rules*/
18.     uncmpt_rs←uncmpt_rs∪α∪β∪Rules             /*No matched proof hints, return rules*/
19.   return uncmpt_rs∪fail                        /*Proof failed, return rules and failure*/
```

Fig.4 Algorithm for distributed proving negotiation

图 4 DPN 分布式证明协商算法

proof_locate 函数的实现如图 5 左所示,如果某个 *find* 启发规则与目标头部的角色相匹配,则返回启发规则建议的节点,否则返回为空.当 *proof_locate* 返回一个节点 *server* 时,*recursive_prove* 函数进行远程信任证明调用,算法首先调用本地代理 *rpn_client*(见图 4 中第 6 行),本地代理向 *server* 发送请求,并等待 *server* 的返回消息.服务节点代理 *rpn_server* 的实现如图 5 右所示,算法调用 *recursive_prove* 函数对目标进行证明,并通过 *negotiation_send* 函数将证明结果包括相关规则集和是否成功的消息保护性地发送给请求节点.*negotiation_send* 函数根据信任证释放规则将满足要求的规则以信任证的形式发送出去,不满足要求的规则不能发送,这样就能提供对信任证信息的保护.

```
20.  address proof_locate(h)
21.    for each ((h′,prin)∈PH)
22.      if (h′=h) return prin
23.  return ⊥
```

```
1.  rpn_server(client,q)
2.  α←recursive_prove(q)
3.  negotiation_send(client,α)
```

Fig.5 Algorithms for remote proving in DPN

图 5 DPN 远程证明的相关算法

4.2 算法实验分析

DPN 算法的特点是规则匹配的逻辑证明,因此我们采用 Amzi Prolog 逻辑语言来实现 DPN 算法,Prolog 语言本身的回溯原理和 Amzi 支持的网络通信使实现变得简单.为了对比分析 DPN 算法的性能,我们实现了 Li[5] 的基于 RT 语言的传统信任协商方法,采用后向搜索 Backward 算法,沿着信任角色定义节点收集信任证.实验模拟上文中例 1 的应用,并增加用户的数量和角色的种类.根据角色的层次关系将所有角色分成 3 个层次,如图 6 所示.一个书店(Org)向它的贵宾、信任客户和自己的员工提供不同的服务,书店的管理员(Reg)将一些服务授权给大学代理(Uni agent).书店(Org)、管理员(Reg)、大学代理(Uni agent)以及每个客户分别运行自己的信任协商代理进程并拥有各自的局部知识库,代理程序包括 *rpn_server* 和 *rpn_client*.

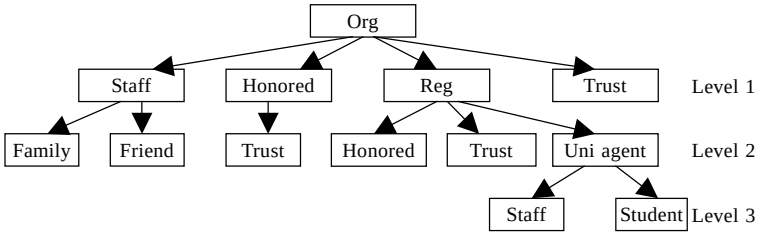

Fig.6 Framework for the roles

图 6 应用角色结构图

根据算法特点,两种算法分别在 4 种情况下进行性能比较:C1) 任何节点都不缓存信任证;C2) 节点缓存第一层次的角色关系;C3) 节点缓存第 1 层和第 2 层的角色关系;C4) 节点缓存 3 个层次的所有角色关系.在缓存的角色关系中,一半为直接签名信任证,另一半为逻辑签名信任证.实验模拟 1 000 个用户分别属于 3 层角色关系中的不同角色,他们向 Org 请求属于各自角色的服务.根据节点所属角色层次的不同,他们的服务请求分别用与其层次对应的 $Q1$,$Q2$ 和 $Q3$ 来表示.

实验在 4 种情况下对各层次的信任关系请求进行信任协商证明,统计两种算法的平均运行时间和节点间交互次数,如图 7 所示.在不缓存信任证的情况下,两种算法对第 1 层次的信任请求问题有相同的交互次数,都为 3 次.信任请求问题每提高一个层次,Backward 算法交互次数即增加 4 次,而 DPN 算法只增加 2 次.即信任请求层次越高,DPN 算法比 Backward 算法越高效.在缓存信任证的情况下,缓存的角色关系每增加一层,DPN 算法交互次数减少 2.对 Backward 算法,当缓存第一层角色关系时,交互次数减少 1;当缓存更高层次的角色关系时,每增加一层缓存信任证,交互次数减少 2.比较缓存信任证导致两种算法交互次数减少的比例可以发现,缓存信任证策略对 DPN 算法更有效.这是因为信任证明能够更有效地利用缓存信任证.

图 7(a)显示,随着 DPN 算法比 Backward 算法交互次数的减少,算法时间也几乎成比例减少,最高减少 50%.这说明两种算法的运行效率主要由算法的交互次数决定.另外,当两种算法交互次数相同时,Backward 算法的运行时间比 DPN 算法要短,这是因为 DPN 算法需要进行更多的规则匹配进行逻辑证明.由此可见,信任证收集算法应尽量减少规则匹配的调用.

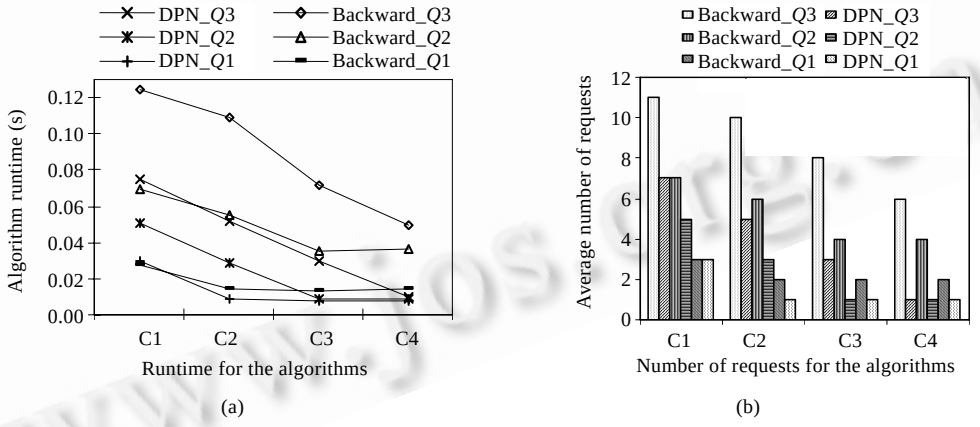

Fig.7 Statistic of the two algorithms under four different conditions

图 7 两种算法在 4 种情况下的运行统计结果

5 结束语

作为跨安全域资源访问管理的主要方法之一,自动信任协商系统需能支持复杂的访问控制和信任协商策

略,并能高效地放置和搜集信任证.本文借鉴已有研究成果,提出面向信任分布式证明和协商的 RTP 语言,它可以支持信任分布式证明方法、定义复杂角色、保护信任证敏感信息并避免信任证盲目搜索.本文介绍了 RTP 语言的语法和语义以及利用它进行信任分布式证明协商的算法,实验结果表明,该算法能够实现 RTP 语言的功能且比传统信任协商方法有很大性能提升.

当前,很多应用需要在限制信任证输出的情况下进行信任协商,本文的信任证释放规则提供了信任证输出限制的功能描述.下一步工作中,我们将在 RTP 语言和 DPN 算法的基础上,增加对信任证加密技术的支持,如隐藏信任证和基于属性的加密技术等.另外,信任的分布式证明方法在信任安全上是对传统信任协商方法的放松,保证信任证明的一致性成为信任协商系统很重要的因素,我们将开展这方面的研究.

致谢 在此,我们对支持本文工作的孙志刚、吴纯清、时向泉、胡晓峰等老师表示感谢.感谢曾迎之和黄清元等同学在课题上对我的启发.同时,特别感谢匿名审稿专家对本文提出的一些建议.

References:

[1] Winsborough WH, Seamons KE, Jones VE. Automated trust negotiation. In: Hilton SC, ed. Proc. of the DARPA Information Survivability Conf. and Exposition. New York: IEEE Press, 2000. 88−102.

[2] Winslett M, Zhang C, Bonatti PA. PeerAccess: A logic for distributed authorization. In: Atluri V, Meadows C, Juels A, eds. Proc. of the ACM Conf. on Computer and Communications Security. New York: ACM Press, 2005. 168−179.

[3] Lee AJ, Winslett M. Safety and consistency in policy-based authorization systems. In: Lin FC, Lee DT, Lin BS, Shieh S, Jajodia S, eds. Proc. of the ACM Conf. on Computer and Communications Security. New York: ACM Press, 2006. 124−133.

[4] Bauer L, Garriss S, Reiter MK. Distributed proving in access-control systems. In: Paxson V, Waidner M, eds. Proc. of the IEEE Symp. on Security and Privacy. Washington: IEEE Computer Society Press, 2005. 81−95.

[5] Li NH, Winsborough WH, Mitchell JC. Distributed credential chain discovery in trust management. In: Herbert AS, ed. Proc. of the 8th ACM Conf. on Computer and Communications Security. New York: ACM Press, 2001. 156−165.

[6] Li J, Li N, Winsborough WH. Automated trust negotiation using cryptographic credentials. In: Atluri V, Meadows C, Juels A, eds. Proc. of the ACM Conf. on Computer and Communications Security. New York: ACM Press, 2005. 46−57.

[7] Irwin K, Yu T, Winsborough WH. On the modeling and analysis of obligations. In: Lin FC, Lee DT, Lin BS, Shieh S, Jajodia S, eds. Proc. of the ACM Conf. on Computer and Communications Security. New York: ACM Press, 2006. 134−143.

[8] Stoller SD, Yang P, Ramakrishnan CR, Gofman MI. Efficient policy analysis for administrative role based access control. In: Ning P, Vimercati S, Syverson PF, eds. Proc. of the ACM Conf. on Computer and Communications Security. New York: ACM Press, 2007. 445−455.

[9] Skogsrud H, Benatallah B, Casati F. Trust-Serv: Model-Driven lifecycle management of trust negotiation policies for Web services. In: Feldman SI, Uretsky M, Najork M, Wills CE, eds. Proc. of the 13th Int'l World Wide Web Conf. (WWW 2004). New York: ACM Press, 2004. 53−62.

[10] Bertino E, Ferrari E, Squicciarini AC. Trust-*X*: A peer-to-peer framework for trust establishment. IEEE Trans. on Knowledge and Data Engineering, 2004,16(7):827−842.

[11] Liao ZS, Jin H, Li CS, Zou DQ. Automated trust negotiation and its development trend. Journal of Software, 2006,17(9): 1933−1948 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/17/1933.htm

[12] Li JX, Huai JP, Li XX. Research on automated trust negotiation. Journal of Software, 2006,17(1):124−133 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/17/124.htm

[13] Bradshaw R, Holt J, Seamons K. Concealing complex policies with hidden credentials. In: Atluri V, Pfitzmann B, McDaniel PD, eds. Proc. of the 11th ACM Conf. on Computer and Communications Security. New York: ACM Press, 2004. 146−157.

[14] Li J, Li N. OACerts: Oblivious attribute certificates. IEEE Trans. on Dependable and Secure Computing, 2006,3(4):340−352.

[15] Li NH, Mitchell JC, Winsborough WH. Design of a role-based trust management framework. In: Heather H, ed. Proc. of the IEEE Symp. on Security and Privacy. Washington: IEEE Computer Society Press, 2002. 114−130.

[16] Li NH, Mitchell JC. Datalog with constraints: A foundation for trust management languages. In: Verónica D, Philip W, eds. Proc. of the Int'l Symp. on Practical Aspects of Declarative Languages. LNCS 2562, Berlin, Heidelberg: Springer-Verlag, 2003. 58−73.

[17] Minami K, Kotz D. Scalability in a secure distributed proof system. In: Proc. of the Int'l Conf. on Pervasive Computing. Berlin: Springer-Verlag, 2006. 220−237. http://www.cs.uiuc.edu/homes/minami/papers/minami_scalability.pdf

[18] Li JX, Huai JP. COTN: A contract_based trust negotiation system. Chinese Journal of Computers, 2006,29(8):1290−1300 (in Chinese with English abstract).

附中文参考文献:

[11] 廖振松,金海,李赤松,邹德清.自动信任协商及其发展趋势.软件学报,2006,17(9):1933−1948. http://www.jos.org.cn/1000-9825/17/1933.htm

[12] 李建欣,怀进鹏,李先贤.自动信任协商研究.软件学报,2006,17(1):124−133. http://www.jos.org.cn/1000-9825/17/124.htm

[18] 李建欣,怀进鹏.COTN:基于契约的信任协商系统.计算机学报,2006,29(8):1290−1300.

王小峰(1982－),男,江苏海安人,博士生,主要研究领域为分布式计算与安全.

张强(1981－),男,博士生,主要研究领域为信息安全.

苏金树(1962－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为计算机网络,信息安全.

张一鸣(1978－),男,博士生,主要研究领域为 P2P 网络,高性能并行处理技术.