ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.20, No.10, October 2009, pp.2822−2833
doi: 10.3724/SP.J.1001.2009.03504

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

利用类型推理验证 Ad Hoc 安全路由协议*

李 沁[1,2], 曾庆凯[1,2+]

[1](南京大学 计算机软件新技术国家重点实验室,江苏 南京 210093)

[2](南京大学 计算机科学与技术系,江苏 南京 210093)

Verifying Mobile Ad-Hoc Security Routing Protocols with Type Inference

LI Qin[1,2], ZENG Qing-Kai[1,2+]

[1](State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China)

[2](Department of Computer Science and Technology, Nanjing University, Nanjing 210093, China)

+ Corresponding author: E-mail: zqk@nju.edu.cn

Li Q, Zeng QK. Verifying mobile ad-hoc security routing protocols with type inference. *Journal of Software*, 2009,20(10):2822−2833. http://www.jos.org.cn/1000-9825/3504.htm

Abstract: A type-inference-based formal method is proposed for verifying an ad-hoc security routing protocol in this paper. A calculus, called NCCC (neighborhood-constraint communication calculus), is defined to specify the protocol. The security property of the protocol is described with typing rules in a type system. Based on the Dolev-Yao model, an attacker model, called the message set of protocol format, is refined. At last, the simplified version of SAODV (secure ad hoc on-demand routing protocol) is verified with this method. With the type-inference-based formal method, not only is the security of protocols verified, but also the attacke examples are predicted. The complexity of inference is reduced significantly for refining the message set of protocol.

Key words: verification of security protocol; ad-hoc protocol; security routing protocol; type inference

摘 要: 提出一种基于类型推理的移动 Ad-Hoc 网络安全路由协议的形式化验证方法.定义了一种邻域限制通信演算 NCCC(neighborhood-constrained communication calculus),包括演算的语法和基于规约的操作语义,在类型系统中描述了移动 Ad-Hoc 网络路由协议的安全属性,定义了近似攻击消息集用以精简 Dolev-Yao 攻击模型.还给出了该方法的一个协议验证实例.基于类型推理,该方法不仅能够验证协议的安全性,也可以得出针对协议的攻击手段.因为攻击集的精简,有效地缩减了推理空间.

关键词: 安全协议验证;ad-hoc 网络协议;安全路由协议;类型推理

中图法分类号: TP393 **文献标识码**: A

移动 Ad-Hoc 网络是一种由具有一定计算和移动能力的节点通过无线连接构成的动态网络.每个节点既是计算节点,又充当路由器的角色.网络中节点间通信通过临时生成的由多个节点构成的多跳路由转发.相对于传

* Supported by the National Natural Science Foundation of China under Grant Nos.60773170, 60721002, 90818022 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No.2006AA01Z432 (国家高技术研究发展计划(863)); the Specialized Research Fund for the Doctoral Program of Higher Education of China under Grant No.200802840002 (高等学校博士学科点专项科研基金)

Received 2007-12-06; Accepted 2008-10-27; Published online 2009-06-09

统网络,移动 Ad-Hoc 网络不需要基础设施的支持,灵活、快速,适用于军事通信、灾后营救等缺乏基础设施的场合,具有广泛的应用前景.但是,其无线通信模式的开放性和节点资源的有限性等特点又使其面临新的安全威胁.其中,针对路由功能的攻击是对网络正常工作基础的严重威胁,攻击者可以采用截取、重放、伪装和篡改等手段骗取其他节点信任并进而在网络中建立错误的路由.因此,陆续提出了一些安全路由协议[1−4],用来发现、维护通信节点间的安全路径,保证通信路径和内容的安全.然而,这些安全协议的安全性证明并不充分.文献[1]虽然利用 Ban logic 进行了安全性的证明,但是由于 Ban logic 自身的缺陷[5],证明并不可靠,故而需要在更严密的形式系统中寻求证明.

已有的安全协议形式化验证主要针对广域或局域网络中实现身份认证、信息流控制等安全属性的安全协议,设计描述被验证协议的形式系统(进程代数、多集重写、串空间等)及其在其中安全属性的表达,研究其验证方法及其自动化(模型检验、定理证明等).然而,对移动 Ad-Hoc 网络的安全协议验证的研究还比较少.由于协议应用场景的不同,Ad-Hoc 网络通信模式和验证的安全属性与传统网络有很大区别:1) 协议所要满足的安全属性与节点存储的路由表状态有关,需要新的描述方式来表达存储状态;2) 节点间以广播方式通信但受到节点间拓扑关系的约束.一般地,只有位于彼此有效通信距离之内才能直接传递消息,除非在合谋的恶意节点间存在其他信道(wormhole),即使是攻击者也会受到通信能力的限制.相比之下,传统安全协议中通信 Agent 的通信方式是点对点的,攻击者可以控制整个网络的通信.因此,安全路由协议的验证对形式化方法提出了新的要求;另外,由于协议的参与者除了通信双方、可信第三方以及攻击者之外,还包括若干中继节点,这不仅导致推理时系统状态大为增加,而且也使系统中可能产生的消息数在数量级上远远超过传统安全协议.庞大的推理空间给模型检验[6]和静态分析[7]等方法的应用带来了性能上的挑战,这个问题在传统安全协议验证中就已经是难点,对于安全路由协议的验证来说则显得更加突出.

类型推理非常适合于安全路由协议的验证:首先,类型系统可以对攻击者的行为进行更加精确的约束.实际上,形式推理面临的庞大状态空间来自于攻击者模型产生消息的任意性,如果用协议规范期待的类型信息来表达消息格式,并利用这个类型信息约束攻击者发出的消息,从而把一些对攻击成功显然无用或对协议无意义的消息排除在推理空间之外(这些消息在所有可能产生的消息中占了绝大多数),便可在一定程度上避免推理的任意性,从而减少推理中回溯的次数;其次,安全路由属性在类型系统中的表达相对自然,把不安全的状态定义为路由存储的不合法,表现为用与存储单元不兼容类型数据对其赋值;最后,类型信息还可以用来精化安全属性的描述,从而达到约简推理空间的目的.根据通信节点在网络中所处物理位置的差异(表达为类型信息)细化安全需求,这样就把推理集中在对实现安全属性最重要的部分(详见第 3.1 节).

本文提出一种基于类型推理的形式化验证方法.提出了领域约束的通信演算(neighborhood-constrained communication calculus,简称 NCCC)及其操作语义,进程在节点中运行,并且它们之间的通信要受到领域关系的约束;在 NCCC 中显式地引入了表示增加存储单元以及引用存储内容的语法构件;定义了类型系统,分析了安全路由属性,并利用子类型、多态、类型依赖在类型系统中对其加以表达,给出了验证方法的正确性证明.另外,攻击者协议消息集被用来描述 Dolev-Yao 攻击者模型[8]在协议约束下的能力,缩减了验证所面临的推理空间.

本文第 1 节介绍相关工作.第 2 节定义 NCCC 的语法、操作语义以及进程同余.第 3 节给出类型系统,证明类型保持(subject reduction)定理.第 4 节定义协议消息集.第 5 节对 SAODV(secure ad hoc on-demand routing protocol)协议进行验证.最后是结论.

1 相关工作

本方法涉及通信协议描述、安全属性定义以及推理技术.

为描述移动计算的分布式系统(进程的运行与进程所在的地点(location)有关),一些形式系统被用来进行相关的形式化验证工作.在进程代数方面移动灰箱演算(mobile ambient calculus)[9]、密封演算(seal calculus)[10]等作为π演算[11]的扩展可以描述诸如同步、Agent 位置、信道上的 Agent 移动等通信模式,并给出了访问控制等安全属性表达.

Klaim[12]提出了一种用于描述 Agent 交互、移动的核语言,网络中 Agent 的存储表达为元组空间(tuple spaces),它可以被节点访问和修改,元组空间及其访问和修改原语是对广播通信的建模.而 $b\pi$[13]则是通过把π演算中的有名信道改为无名广播信道,实现对广播的描述,$b\pi$适合于局域网广播和基于组(group)的通信建模.

文献[7,14,15]描述了移动 Ad-Hoc 网络组播的通信方式,网络拓扑被用来作为通信的限制.文献[7]利用静态分析的方法分析节点存储的数据项是否会包含错误的数据,Dolev-Yao 模型被引入验证过程,但其没有利用协议本身提供的类型信息简化推理;文献[15]则是利用一个与所有 Agent 交互且控制消息传输的交互图灵机来实现中继节点间的通信,但其验证过程完全依赖手工证明,难以实现验证自动化.

类型理论在协议形式化推理和验证中发挥着基础性支持作用.在文献[16]中,类型被用来验证信息流控制;在文献[17]中则被用来定义访问控制的规则.子类型[18]、多态[19]、类型依赖[20,21]在π演算及其各种变体中都有着深入的研究.这些研究虽然不是直接针对 Ad-Hoc 网络路由协议的验证目标,但是它们在其他领域的应用可为建立移动 Ad-Hoc 路由协议的验证方法提供一定借鉴.

2 NCCC

本节给出对移动 Ad-Hoc 网络建模的进程演算 NCCC.在 CBS(calculus of broadcasting systems)[18]的基础上增加了存储单元创建、赋值和引用的语法构件,并且通过节点间的邻域关系限制消息在网络中的传播.

2.1 语 法

NCCC 的语法包括 3 种基本元素:项(term)、进程(process)和网络(network).它们的 BNF 语法见表 1.

Table 1 NCCC syntax

表 1 NCCC 语法

function $f \in \mathcal{F}: \times, sig, dec, enc, aenc, adec$
Term:
$M ::= n \mid x \mid k \mid ?x \mid f(\tilde{x})$, $\tilde{x}$ is a tuple of terms
Process:
$P ::= Nil \mid \langle M \rangle.P \mid (x).P \mid \text{ref } x : X = M.P \mid !P \mid P \mid Q \mid \nu x.P \mid \text{case } M \text{ of } f(\tilde{x})\ P_1 \text{ else } P_2$
Network:
$N ::= n[P]_\mu \mid N_1 \mid N_{2\mu}$

项:$n \in \mathcal{N}$是名字的可数集,$x \in \mathcal{V}$是变量的可数集,$k \in \mathcal{K}$是密钥集,$f \in \mathcal{F}$是有限的构造函数符号集,$arity(f)$指函数 f 的元数.项集$\mathcal{T}$由名字、变量、密钥及形如 $f(t_1,\ldots,t_{arity(f)})$,$t_i \in \mathcal{T}$组成,$?x$ 表示在 x 中存储的内容.构造函数按列表顺序分别是并置、签名、对称解密、对称加密、公钥加密、私钥解密.简便起见,后文中$\times(t_1,t_2)$的乘号$\times$将被省去.

替换 $M[t/x]$是指把项 M 中自由变量 x 的所有发生用项 t 替换,如果牵涉到多个变量的替换,则用 $M[\tilde{t}/\tilde{x}]$ 表示将 M 中的每个变量 x_i 的所有发生各自用 t_i 同时替换.

进程:进程集合$\mathcal{P}$可以归纳定义如下:终止进程由 Nil 表示;$\langle M \rangle.P$ 表示进程输出项 M 后继续进程 P;$(x).P$ 表示进程接受消息 M 而后继续进程 $P[M/x]$;ref x:X=M.P 表示进程创建类型 X 的存储单元 x,并以项 M 对其赋值后继续进程 P.另外,存储单元的声明在节点的作用域内是唯一的,这一点可由μ是一个函数的事实得出;$\nu n.P$ 表示引入新名字 n 后继续进程 P;$P|Q$ 表示进程 P 和 Q 并行执行;$!P$ 表示可数无限个的进程 P 并行执行;case M of $f(\tilde{x})$ P_1 else P_2 表示若 M 形如 $f(\tilde{x})$,则以进程 P_1 继续,否则以进程 P_2 继续.

网络:$n[P]_\mu$表示进程 P 在名字为 n 的节点中运行,函数μ:$\{x@n:x \text{ is a ref variable}, n \in \mathcal{N}\} \to \mathcal{T} \times TYPE$ 负责管理整个网络的存储,x 在节点内被唯一创建;$N_1|N_2$ 表示两个网络并置的结果仍然是网络.

2.2 操作语义

邻域关系是名字对的集合 $G=\{(n,m): n \in \mathcal{N}, m \in \mathcal{N}\}$,它限制了 NCCC 定义的网络规约(reduction)的方式,如表 2 中规则 R-com,只有当节点对属于邻域关系时,发送节点发出的消息才能被另一个节点接收.也就是说,通过邻域关系,我们把节点发送消息的行为限制在其邻域内,而且这种限制也是攻击者所不能违背的.网络间的规约关

系见表 2.

Table 2 Reduction relation

表 2 规约关系

R-rep	$\dfrac{P \to_G Q}{!P \to_G !Q}$
R-new	$\dfrac{P \to_G Q}{\nu x.P \to_G \nu x.Q}$
R-ref	$n[\text{ref } x : X = M.P]_\mu \to_G n[P[?x/M]]_{\mu \sqcup x@n \mapsto (REF(X),M)}$
R-com	$\dfrac{(n,m) \in G, \neg(n,l) \in G}{n[\langle M \rangle.P] \mid m[(x).P'] \mid l[(x).P'']_\mu \to_G n[P] \mid m[P'[M/x]] \mid l[(x).P'']_\mu}$
R-cas 1	$\dfrac{t \text{ consists with } f(\vec{x})}{\text{case } t \text{ of } f(\vec{x})\ P_1 \text{ else } P_2 \to_G P_1}$
R-cas 2	$\dfrac{t \text{ does not consist with } f(\vec{x})}{\text{case } t \text{ of } f(\vec{x})\ P_1 \text{ else } P_2 \to_G P_2}$

规则 R-rep、规则 R-new、规则 R-case1 和规则 R-case2 的语义与π演算[16]类似,只是多了领域关系的限制.规则 R-ref 是指节点中出现存储单元的声明与赋值行为的结果是在函数μ中增加了$(x@n,(REF(X),M))$;规则 R-com 说明只有属于邻域关系的节点对才能通信.

2.3 结构同余

2.3.1 自由变量和存储变量

项、进程和网络的自由变量可以由语法结构归纳定义,见表 3.

Table 3 Free variables

表 3 自由变量

$fn(Nil) = \varnothing$
$fn(x) = \{x\},\ x$ is not a reference variable
$fn(?x) = \{?x\},\ x$ is a refernce variable
$fn((M, M')) = fn(M) \cup fn(M')$
$fn(f(t_1,...,t_k)) = \bigcup_{i=1}^{k} fn(t_i),\ k = arity(f)$
$fn(\langle M \rangle.P) = fn(M) \cup fn(P)$
$fn((x).P) = fn(P) - \{x\}$
$fn(\nu x.P) = fn(P) - \{x\}$
$fn(P \mid Q) = fn(P) \cup fn(Q)$
$fn(!P) = fn(P)$
$fn(\text{ref } x : X = M.P) = fn(P) \cup fn(M) - \{?x\}$
$fn(n[P]) = fn(P)$

函数 *fn* 对给定的参数返回其包含的自由变量的集合,函数 *bn* 对给定的参数返回在其内部被约束的变量.

存储变量是指变量在声明时被保留字 ref 限定为一个存储,每个存储变量在节点内部都是唯一的.进程内的存储变量可以用函数 *ref*(*P*)来表示,其定义与自由变量类似,不再赘述.

2.3.2 结构同余(structure congruence)

网络的结构同余见表 4,进程的结构同余见表 5.第 2 条规则是指进程在α变换下等价.

Table 4 Structural congruence of network

表 4 网络的结构同余

$N \equiv N$
$N \mid n[Nil] \equiv N$
$N_1 \mid N_2 \equiv N_2 \mid N_1$
$N_1 \equiv N_2, N_2 \equiv N_3 \Rightarrow N_1 \equiv N_3$

Table 5 Structural congruence of process

表 5 进程的结构同余

$P \equiv P$
$P \equiv_\alpha P[y/x]\ y \notin name(P), x \in bn(P)$
$P \mid Nil \equiv P$
$P \mid Q \equiv Q \mid P$
$(P \mid Q) \mid R \equiv P \mid (Q \mid R)$
$!P \equiv P \mid !P$
$\nu a.\nu b.P \equiv \nu b.\nu a.P$ if $a \neq b$
$\nu a.P \mid Q \equiv P \mid \nu a.Q$ if $a \notin fn(P)$
$P \equiv Q \Rightarrow P \mid R \equiv Q \mid R$
$P \equiv Q \Rightarrow !P \equiv !Q$
$P \equiv Q \Rightarrow \nu a.P \equiv \nu b.Q$
$P \equiv Q, Q \equiv R \Rightarrow P \equiv R$

3 类型系统

本节给出 NCCC 的类型系统.类型系统利用一系列规则给出如下断言(judgments):

- $\vdash E$ well-formed 是指环境 E 是良形成的;
- $E \vdash \Delta : T$, Δ 代表项 M,进程 P 或在网络 N 在环境 E 中的类型是 T.

利用类型系统对网络进行类型推理,如果在网络中没有错误的路由信息被存储在特定的节点类型上,则推理的结果是 $E \vdash N : \Diamond$,否则为 $E \vdash N : err$.

3.1 Ad-Hoc路由的安全属性

迄今为止,安全路由尚无统一的定义.一般而言,只有当所有节点内部的路由表的存储状态与实际的通信路径一致时,路由过程才被认为是成功的.然而在协议实际执行过程中,这样的要求往往过于严格.从协议执行的结果来看,在存在攻击者的情况下,发起路由发现的源节点可能会:

1) 收到 1 则含有一致路由信息的消息;

2) 收到 1 则含有错误信息的消息;

3) 在预定的时间内收不到消息,意味着不存在它所期望的路径.

3 种情况中,只有第 2 种对攻击者来说是成功的攻击.因为攻击者不能控制整个网络(不同于 Dolev-Yao 模型),它也要受到邻域关系的限制,即使它阻断了某一条路径,也不能防止在源节点和目的节点之间建立起另一条不通过攻击者的路径.再者,如果事实上源节点和目的节点之间不存在路径,如图 1 所示,则节点 A,B 不属于 D 的邻域,不存在源节点 S 和目的节点 D 之间的路由.如果恶意节点 M 转发 B 的消息给 D,使 D 在自己的存储中建立了错误的路由表,但是,由于路由应答消息此时还没有接受检查,则源节点 S 和中继节点 A 并不必然接受错误的路由信息,因此这个存储并不意味着一个成功的攻击.只有当节点 A 也接受 M 的伪造消息并在存储中建立错误的路由表(相应地,S 也会建立一个错误的路由表项)时,才是一次成功的攻击.

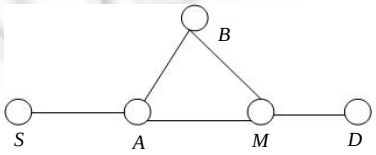

Fig.1 Example of network topology

图 1 网络拓扑示例

由此可知,发生在某些节点的错误的存储行为并不等价于成功的攻击,即存储行为在网络中发生的相对位

置对于攻击成功与否也有影响.就图 1 而言,只有当错误存储发生在 M 的左侧(靠近源节点的一侧)时,攻击才算成功;而对目的节点一侧的存储行为可以不加限制.所以,令节点集为$\mathcal{N}=\mathcal{S}\cup\mathcal{D}\cup\{M\}$,$S$ 是源侧节点集,D 是目的侧节点集,邻域关系 G,函数 *routable* 给出参数节点的路由表集,移动 Ad-Hoc 网路由的安全属性可以被表达为 $\forall x\in S.routable(x)$ respects G,即每一个源侧节点的路由存储与当前网络拓扑一致.同时,对利用 NCCC 规范的网络 N 来说,如果协议满足安全属性,则它的类型为◊,这一点主要体现在后文表 8 中的类型规则 TRrefrt,Trrefrt 1 ~TRrefrt 4.

3.2 环 境

环境是名字或变量的列表,每个名字和变量都有相关联的类型.由于子类型的关系(见第 3.3 节),与名字和变量相关联的类型并不唯一,但是只有当关联与同一变量的类型之间存在子类型的关系时,环境才是良形成的(well-formed),即 $E\vdash n:T_1, E\vdash n:T_2 \Leftrightarrow T_1\le T_2 \vee T_2\le T_1$.有关环境的规则包括:

- $\dfrac{}{\vdash \varnothing \text{ well-formed}}$;
- $\dfrac{\vdash E \text{ well-formed } x\notin dom(E)\vee(x\in dom(E)\wedge E(x)\le T)}{\vdash E,x:T \text{ well-formed}}$

3.3 项的类型

NCCC 的类型见表 6.类型由基本类型、组合类型和通用类型以及存储类型(以 $\star$ 为前缀)组成.基本类型包括节点类型和依赖类型,节点若在源节点侧,则其类型为 IDS,在目的节点侧则为 IDD,二者均为类型 ID 的子类型;组合类型包括积类型和密项类型;通用类型只有 ⊤ 类型,其他所有类型是其子类型,见表 7.依赖类型包括路由表类型和密钥类型,前者是指节点类型的积类型且其是否是路由表类型取决于存储它的节点,后者密钥的类型依赖于密钥的所有者[22].

Table 6 Type of the terms

表 6 项的类型

$TYPE ::= UniType \mid BasicType \mid CompType \mid DependentType \mid refType$
$UniType ::= \top$
$BasicType ::= ID \mid IDS \mid IDD \mid INT \mid DependentType$
$refType ::= \star Type$
$CompType ::= productType \mid EncryType$
$productType ::= BasicType \times productType$
$EncryType ::= \{productType \mid BasicType\}\ with\ KeyType$
$DependentType ::= KeyType \mid RT(n) \quad type(n)\in\{ID, IDS, IDD\}$
$KeyType ::= Sigkey(n) \mid Enckey(n) \mid Deckey(n) \mid Aenckey(n) \mid Adeckey(n)$
$RT(n)\ \square\ IDS\times IDD\times ID\times ID$

Table 7 Subtyping rules

表 7 子类型规则

T-top	$\forall T\in TYPE, T\le\top$
T-id	$IDD\le ID, IDS\le ID$
T-prodepth	$\dfrac{\text{for each } i\ T_i\le S_i}{T_1\times\ldots\times T_n\le S_1\times\ldots\times S_n}$
T-ref	$\dfrac{T\le S}{REF(T)\le REQ(S)}$
T-key	$\dfrac{T\le S}{T \text{ with } K\le S \text{ with } K}$
TRsub	$\dfrac{t:T, T\le S}{t:S}$

3.4 进程和网络的类型规则

进程和网络的类型规则见表 8.

因为接收、发送消息并不确定导致错误的存储行为,只有当一个牵涉到路由存储的类型错误才会使进程的类型为 *err*,所以在进程的输入和输出消息规则中并不关心消息的类型,判定时,利用子类型的关系忽略消息的实际类型(全部当作 ⊤ 类型,但是类型信息仍与项绑定).

值得一提的是规则 TRnew 和 4 个与路由存储有关的规则:TRnew 中的变量类型是 *INT*,我们在 NCCC 中引入语法构件 $vx.P$ 的目的是为了描述安全协议中常见的 nonce.由于 nonce 一般是随机生成整数,所以把引入的变量限定为整型;规则 Trrefrt 1 与 Trrefrt 3 表达的是正常情况下的存储;规则 Trrefrt 2 是指在 IDD 类型的节点存储一个错误路由是可以接受的;规则 Trrefrt 4 表达了唯一会引起进程类型错误的情况,即在 IDS 类型的节点存储一个错误路由使以之为前缀动作的进程的类型为 *err*.另外,规则中存储变量及其内容的类型由μ给出.

Table 8 Typing rules of process and network

表 8 进程和网络的类型规则

TRout	$\dfrac{E,M:\top\vdash_n P:\Diamond}{E\vdash_n \langle M\rangle.P:\Diamond}$
TRin	$\dfrac{E,x:\top\vdash_n P:\Diamond}{E\vdash_n (x).P:\Diamond}$
TRrep	$\dfrac{E\vdash_n P:\Diamond}{E\vdash_n !P:\Diamond}$
TRpar	$\dfrac{E\vdash_n P, E\vdash_n Q}{E\vdash_n P\mid Q}$
TRnew	$\dfrac{E,x:INT\vdash_n P:\Diamond}{E\vdash_n \nu x.P:\Diamond}$
TRcase	$\dfrac{E\vdash t:T, E\vdash f(x):T, E\vdash P_1:\Diamond, E\vdash P_2:\Diamond}{E\vdash \text{case } t \text{ of } f(x)\ P_1 \text{ else } P_2:\Diamond}$
TRref	$\dfrac{E,x:\star X, M:X\vdash_n P:\Diamond, \neg X\in\{RT(n):n \text{ is the name of a node.}\}}{E\vdash_n \text{ref } x:X=M.P:\Diamond}$
TRrefrt 1	$\dfrac{E,x:\star RT(n), M:RT(n)\vdash_n P:\Diamond, E\vdash n:IDD}{E\vdash_n \text{ref } x:RT(n)=M.P:\Diamond}$
TRrefrt 2	$\dfrac{E,x:\star RT(n), M:RT(m)\vdash_n P:\Diamond, E\vdash n:IDD}{E\vdash_n \text{ref } x:RT(n)=M.P:\Diamond}$
TRrefrt 3	$\dfrac{E,x:\star RT(n), M:RT(n)\vdash_n P:\Diamond, E\vdash n:IDS}{E\vdash_n \text{ref } x:RT(n)=M.P:\Diamond}$
TRrefrt 4	$\dfrac{E,x:\star RT(n), M:RT(m)\vdash_n P:\Diamond, E\vdash n:IDS}{E\vdash_n \text{ref } x:RT(n)=M.P:err}$
TRnode 1	$\dfrac{E\vdash_n P:\Diamond}{E\vdash n[P]:\Diamond}$
TRnode 2	$\dfrac{E\vdash_n P:err}{E\vdash n[P]:err}$
TRnet 1	$\dfrac{E\vdash n[P]:\Diamond, E\vdash m[Q]:\Diamond}{E\vdash n[P]\mid m[Q]:\Diamond}$
TRnet 2	$\dfrac{E\vdash n[P]:\Diamond, E\vdash m[Q]:err}{E\vdash n[P]\mid m[Q]:err}$

3.5 一些结论

本节的主要任务是证明类型保持定理,即网络的类型不会随着规约而变化.为此,需要先证明两个引理.

引理 1(细化 narrowing).

1. 假设 $E,n:T\vdash M:R$ 且 $S\le T$,则 $E,n:S\vdash M:R$.
2. 假设 $E,n:T\vdash P:\Diamond$ 且 $S\le T$,则 $E,n:S\vdash P:\Diamond$.

证明:

1. 对 M 的结构利用类型规则作归纳证明.这里仅证明 M 恰为 x 和 $M_1\times M_2$ 的情形.若为前者,则由类型规则 TRsub 可知结论成立.对后者由归纳假设可知 $E,n:S\vdash M_i:R_i, i=1,2$,由类型规则 T-prodepth 可知结论成立.

2. 对 P 的结构利用类型规则归纳证明.这里仅证明 P 形如 ref x:X=m.Q 的情形.由已知条件和类型规则 TRref 有 $E,n:T\vdash m:X$ 以及 $E,n:T\vdash Q:\Diamond$,又由归纳假设可知 $E,n:S\vdash m:X$ 及 $E,n:S\vdash Q:\Diamond$.根据类型规则 T-ref, TRsub 得到 $E,n:S\vdash ref\ x:X=m.Q:\Diamond$. □

引理 2(替换 substitution). 假设 $E\vdash M':T$, $T\le S$, 有如下结论成立:

1. 若 $E,x:S\vdash M:R$,那么 $E\vdash M[M'/x]:R$.
2. 若 $E,x:REF(S)\vdash M:R$,那么 $E\vdash M[M'/?x]:R$.
3. 若 $E,x:S\vdash P:\Diamond$,那么 $E\vdash P[N/x]:\Diamond$.
4. 若 $E,x:REF(S)\vdash P:\Diamond$,那么 $E\vdash P[N/?x]:\Diamond$.

证明:

1. 对项 M 的结构利用类型规则作归纳证明.

1) M 是变量 $y\neq x$ 或名字,由引理 1 可知结论成立.若恰为 x,则 R 为 S 或 T,由类型规则 Trsub 可知结论成立. 2) M 形如(M_1,M_2):$R_1\times R_2$.由归纳假设可知 $E\vdash M_1[M'/x]:R_1$,且 $E\vdash M_2[M'/x]:R_2$.由类型规则 Trprod 可知结论成立.3) M 形如 $f(k,M'')$且其类型为 $R=R'$ with K.由归纳假设可知结论对子项 M'成立,即 $E\vdash M''[M'/x]:R'$,由类型规则 TRktrm 可
知结论成立.

2. 证明除利用类型规则 TRref 与 TRderef 之外与以上证明类似.

3. 对进程 P 的结构作归纳证明.这里仅证明输出进程与引用进程,其余的证明是类似的.

1) P 形如$\langle M\rangle.Q$.由类型规则 Trout 可知 $\exists R.E,x:S\vdash M:R$ 且 $E,x:S\vdash P:\Diamond$.因为本引理第 1 条 $E\vdash M[N/x]:R$,又因为归纳假设及类型规则 TRout, $E\vdash P[N/x]:\Diamond$.

2) P 形如 ref y:Y=$M.Q$,需分 5 种情形讨论.

情形 1: P 由类型规则TRref导出.可知 $E\vdash M:Y$, $E,y:\star Y,M:Y\vdash Q:\Diamond$,由本引理第1条可知 $E\vdash M[N/x]:Y$, 又由归纳假设有 $E,y:\star Y,M[N/x]:Y\vdash Q[N/x]:\Diamond$,最后由类型规则 Trref 可知结论成立.

情形 2: P 由类型规则 TRrefrt 1 导出.可知 $E\vdash M:Y$, $E,n:IDD,y:\star RT(n),M:RT(n)\vdash Q:\Diamond$,因为本引理第一条知 $E\vdash M[N/x]:RT(n)$,又由归纳假设有 $E,n:IDD,y:\star RT(n),M[N/x]:R(n)T\vdash Q[N/x]:\Diamond$,最后由类型规则 TRrefrt 1 可知结论成立.

情形 3: P 由类型规则 TRrefrt 2 导出.与情形 2 类似.

情形 4: P 由类型规则 TRrefrt 3 导出.同上.

情形 5: P 由类型规则 TRrefrt 4 导出.同上.

4. 与结论 3 的证明类似. □

定理 1(subject reduction). 若 $E\vdash N_\mu:\Diamond$ 且 $N_\mu\rightarrow_G N'_{\mu'}$,则 $E\vdash N'_{\mu'}:\Diamond$.

证明:对 N_μ的结构利用类型规则作归纳证明.这里仅就归约规则 R-ref 和 R-com 两种情形给出证明,其余的证明是容易且类似的.

情形 1: N_μ=n[ref x:X=$M.P$]$_\mu$.由条件有 $E\vdash n:ID,E\vdash_n \text{ref}\,x:X=M.P:\Diamond$.因此有 $E\vdash M:X$ 或 $E\vdash M:Y$, Y=$RT(m),m\neq n,X=\star RT(n)$.两种情形下,均可由引理 2 可知 $E\vdash_n P[M/?x]:\Diamond$,继而分别由类型规则 T-ref,T-refrt 1 和 T-refrt 2 知结论成立.

情形 2: $n[\langle M\rangle.P]|m[(x).P']|l[(x).P'']_\mu,(n,m)\in G,(n,l)\notin G$.由条件及类型规则 TRnet 1 可知,式中每个节点的类型均为$\Diamond$,而 $N'_\mu=n[P]\,|\,m[P'[M/x]]\,|\,l[(x).P'']$,显然,节点 n,l 的类型保持不变.又由引理 2 可知 $E\vdash m[P'[M/x]]:\Diamond$.最后由类型规则 TRnet 1 可知结论成立. □

4 攻击者协议消息集

4.1 协议消息集的构造

协议必须在一个存在攻击者的环境中验证.传统的协议验证一般利用经典的 Dolev-Yao 模型作为攻击者模型,这种模型设想攻击者控制协议执行所处的整个网络,能够任意截取、伪造网络中传输的消息,也可以在已知密钥的情况下解密截获的信息.在文献[7]中,攻击者是作为一个由接收、存储、组装、发送消息的进程通过并行和复制组成的,虽然这种描述与 Dolev-Yao 模型相吻合,但由此带来的验证所面对的状态空间却非常复杂.在 NCCC 中,Dolev-Yao 模型除了约束于通信邻域的限制之外(体现于表 2 的规则 R-com),还约束于协议消息格式的类型信息.

对特定协议而言,协议的消息格式本身就是对网络中可传输的消息种类的限制,它设定了在协议交互时每则消息的特定位置上应该出现的特定类型的信息,也规定了消息构造函数迭代作用的层数.例如,SRP(secure routing protocol)协议[3]中目的节点的路由回复消息格式为 $(rrep,S,D,id,sn,(\overrightarrow{X}),mac_D)$,其中,$rrep$ 是回复标

志,S,D 是源节点和目的节点名字,id,sn 是当前路由请求的 *nonce* 和序列号,$(\overrightarrow{X})$ 是节点名字的序列,mac_D 是目的节点用自己的签名密钥作用于消息而生成的散列值.在这则消息中,除最后两项以外,其他项都是原子项,而且前 5 项对于一次路由发现过程来说是不变的(意味着这些位置上出现其他项将使源节点拒绝消息);第 6 项被函数×作用;最后一项则经历了 3 次函数作用(×,×,*hash*).另外,消息格式隐式地声明了在期望的位置出现何种类型的项,在满足了这些要求之后,各项才能被组合成特定的消息.不满足格式的消息无论对于源节点还是对于攻击者来说都是无意义的.由此,我们可以把协议消息格式看作是从项的积空间到消息集的映射集合 $c\in\mathcal{C}$:

$$c:\times_{i\in I}T_i\to\mathcal{M}\cup\{\perp\},T_i\subset\mathcal{T},|I|=arity(c).$$

$\mathcal{C}$是消息格式函数集,I 是指标集,$\mathcal{M}$是消息集,⊥指未定义,当参数不满足消息格式时函数值为⊥.利用消息格式函数,不但限定了攻击者可以发出的消息集的大小,而且可以区分在不同路由阶段攻击者发出的消息种类.显然,对应于 Dolev-Yao 模型来说,协议消息集是 Dolev-Yao 攻击者知识集的子集,其中只包括那些符合协议消息格式的项.

下面讨论协议消息集的计算.首先从原子的初始项集(包括攻击者的初始知识$\mathcal{I}$、子集节点名字$\mathcal{N}$、密钥集$\mathcal{K}$)出发,利用构造函数按一定次序迭代作用于初始项集得到新的项集.根据上面的分析可知,这个迭代过程是有限的,因为由有限构造过程得出的项集是对攻击者所能产生的项集(初始项集的构造函数闭包)的近似,故称为近似知识集$\mathcal{T}$.如果构造函数集为$\mathcal{F}$,消息格式集为$\mathcal{C}$,迭代最大层数为 k,则称为初始项集的$\mathcal{F}k$-closure.

下面仍以 SRP 协议的路由回复消息格式为例(为简单计,忽略回复标志),给出由初始项集得到近似知识集的算法.

Input: function set $\mathcal{F}$, initial term set $\mathcal{I}=\mathcal{N},\{S,D\}\subset\mathcal{N}$,
aid(association id), key set $\mathcal{K}$, topology $\mathcal{G}$, $\mathcal{L}$;
Output: $\mathcal{T}=\mathcal{F}k$-closure of $\mathcal{I}=\mathcal{N}\cup F\cup KT$. $k=3$.
1. $\mathcal{T}=\mathcal{I},KT=\varnothing,F=\varnothing$
2. $\mathcal{N}=\mathcal{N}\cup\{(n_1,...,n_i,...,n_m)\mid n_i\in\mathcal{N},2\le m\le llp(\text{length of logest path in }\mathcal{G}),n_i\ne n_j\}$
3. $F=\{(S,D,aid,t)\mid t\in\mathcal{N}\cup\{(\,)\}\}$
4. for each $key\in\mathcal{K}$
5. for each $t\in\mathcal{T}$
6. $KT=KT\cup\{mac(key,t,aid)\}$
7. $\mathcal{T}=\mathcal{N}\cup F\cup KT$

其次,在已知路由回复消息格式的近似知识集的情况下,令 c 为对应的消息格式函数,有此格式的攻击者消息集:$\mathcal{M}_c=\{c(n,kt)\mid n\in F,kt\in KT\}-\{\perp\}$,从而有攻击者协议消息集 $\mathcal{M}_p=\bigcup_{c\in\mathcal{C}}\mathcal{M}_c$.

4.2 消息集计算量分析

假设$|\mathcal{I}|=n$,即拓扑$\mathcal{G}$中共有 n 个节点,显然 $llp\le n-2$(去掉了源节点、目的节点),则第 1 次迭代后$\mathcal{T}$中所含项的数量为 $|\mathcal{T}|=n+\sum_{l=2}^{llp}l!$,第 2 次迭代后加倍,第 3 次迭代再加倍后,项的总数为 $4(n+\sum_{l=2}^{llp}l!)$,所以,$|\mathcal{T}|=\mathcal{O}((n-2)!)$.虽然知识集的复杂度在计算上不可行,但在算法应用的场景下,n 基本上为个位数,仍然是可以接受的.每个消息格式函数的消息集的大小为 $|\mathcal{M}_c|=|F|\times|KT|=\mathcal{O}((n-2)!^2)$,当 n=5 时,大约会有 10^2 则消息.

5 安全分析

5.1 SAODV安全路由协议

本节用 SAODV 协议[1]的简化版本作为例子探讨类型推理方法的应用.带数字签名的 SAODV 协议用于保证 Ad-Hoc 网络中请求源路由安全的协议,路由的发现与维护基于路由表,每个节点会存储路由的下一跳(the next hop)和上一跳(the previous hop).

路由发现的发起节点生成一则路由请求并在邻域内广播,路由请求包含请求标记 *rreq*,目的节点 D 与请求

节点 S 的身份标识,S 及其自己的公钥 $pk(s)$,并附上用私钥 $sk(s)$的关于$(rreq,S,D,PK(S))$的签名.

每个接收到路由请求的节点首先在公钥基础设施的支持下检查 $pk(s)$与 S 是否一致以及能否用公钥验证签名,如果检查成功,则中继节点建立一个存储$(S,itself,Pre)$,并把请求消息中的前趋节点标识修改为自己的标识后重广播.如果是目的节点收到路由请求,则除建立相应存储以外,构造回复消息,包括 $rrep,D,S$,请求消息的发送邻域节点标识,$D,pk(d)$以及它们的签名.

中继节点收到回复后,首先检查自己是否是指定的接受者:如果不是,则丢弃消息,否则,验证签名并创建存储$(D,itself,Next)$,然后把回复消息中的后继节点域改成自己的标识并转发,直至请求节点 S.

5.2 协议描述和攻击者消息集

首先给出协议运行的拓扑环境.由第 3.1 节和后文的第 5.3 节的分析可知,布置拓扑环境时不需要形成有正常节点构成的路由,可减少重复推理的次数;节点数应尽可能地少,这直接影响到近似知识集的大小.在目的节点侧只需设置目的节点(攻击者的主要目的是欺骗源节点一侧的节点),同时在源节点侧设置一个同时与源节点和攻击节点相邻的节点,提供攻击者伪装的可能.拓扑如图 2 所示.

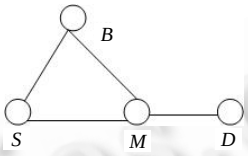

Fig.2 Topology for verification: Malicious M, source S, destination D, normal node B

图 2 协议验证拓扑:恶意节点 M、源节点 S、目的节点 D、普通节点 B

协议的构造函数包括 $aenc,adec$ 和×.第 5.1 节中所描述的 3 个过程在 NCCC 中的规范如下:

源节点发出路由请求:

$Init(D,S)\Box\langle rreq,D,S,S,pk(S),aenc(sk(S),rreq,D,S,pk(S))\rangle$;

路由节点转发路由请求消息或目的节点接收请求后发出回复消息:

$Rreq(n)\Box$

$(req,tar,src,snd,pk,sig).$

case pk of $pk(src)$

case $adec(pk(src),sig)$ of $(tar,src,pk(src))$

ref $prehop:RT(n)=(src,n,snd).$

case tar of n

$\langle rrep,tar,src,snd,n,pk(n),aenc(pk(n),(tar,src,pk(n)))\rangle$

else $\langle rreq,tar,src,n,pk,sig\rangle$

else Nil else Nil

路由节点转发路由回复消息或源节点接收到回复消息:

$Rrep(n)\Box$

$(rrep,tar,src,imm,snd,pk,sig).$

case pk of $pk(tar)$

case $adec(pk(tar),sig)$ of $(tar,src,pk(tar))$

ref $nexthop:RT(n)=(tar,n,snd).$.

case src of n Nil

else case ?$prehop$ of $(src,n,next)$

$\langle rrep,tar,src,next,n,pk,sig\rangle$

else Nil else Nil else Nil

拓扑$\mathcal{G}$的边集是$\{(S,M),(S,B),(M,B),(M,D)\}$.另外,节点的密钥集$\neg\mathcal{K}=\mathcal{K}_p\cup\mathcal{K}_s$,即公钥集与私钥集的并.

协议消息格式包括路由请求格式和路由回复格式,给定一次由 S 发起查找 D 的路由发现,分别是:

- $c_{req}=(rreq,S,D,x,y,z)$,
  $x\in\mathcal{N},y\in\mathcal{K}_p,z\in\{aenc(k_1,S,D,k_2)\mid k_1\in\mathcal{K}_s,k_2\in\mathcal{K}_p,k_1(k_2(x))=x\}$.
- $c_{rep}=(rrep,D,S,x,y,z,u)$,
  $x,y\in\mathcal{N},z\in\mathcal{K}_p,u\in\{aenc(k_1,D,S,k_2)\mid k_1\in\mathcal{K}_s,k_2\in\mathcal{K}_p,k_1(k_2(x))=x\}$.

利用第 4.1 节的方法容易得出攻击者协议消息集 $\mathcal{M}_p=\mathcal{M}_{c_{req}}\cup\mathcal{M}_{c_{rep}}$.

最后,协议在拓扑$\mathcal{G}$上的一次运行可定义为

$$S[Init(D,S)\mid Rrep(S)]\mid D[Rreq(D)]\mid B[Rreq(B)\mid Rrep(B)]\mid M[!_{t_1\in\mathcal{M}_{c_{req}},t_2\in\mathcal{M}_{c_{rep}}}(\langle t_1\rangle.\langle t_2\rangle)]_\mu$$

攻击者节点中的进程没有输入,这是因为攻击者所能接收到的信息已经事先存储在它的消息集中了;而且在实际推理时,目的节点 D 中的进程也不需要考虑,因为它所能发出的消息也已处于攻击者的消息集中.换句话说,在 S 和 B 看来,攻击者 M 代表了除它们自己之外的整个网络(它可以利用定向天线做到这一点).

5.3 类型推理

类型推理是基于经典类型推理算法[23]来进行的:首先把源代码转换为抽象语法树;结合$\mathcal{M}_p$ 中每一则消息和语法树,进而建立对类型变量(与语法项和变量相关联)的等式集;然后对等式集求解 $E\vdash N:\Diamond$ 能否成立.具体算法可参考文献[24].需要注意的是,如果某一次推理的结果是 $E\vdash N:\Diamond$,并不能说明验证是成功的,只能说明协议可以抵御攻击者发出的某种特定的消息序列.换句话说,只有对所有 $!_{t_1\in\mathcal{M}_{c_{req}},t_2\in\mathcal{M}_{c_{rep}}}(\langle t_1\rangle.\langle t_2\rangle)$ 中的消息序列都有 $E\vdash N:\Diamond$ 的结果才能认为通过了验证;反之,只需一次 $E\vdash N:err$ 就可以说明协议是有漏洞的,并且相应的消息序列就代表了相应的攻击手段.根据第 4.1 节、第 4.2 节和第 5.1 节的分析,由源码构建的抽象语法树与消息集 $\mathcal{M}_p$ 均有限,所以整个类型推理过程可终止.

对第 5.2 节中的程序进行手工推理(因为本例中协议消息集很小,相应的自动推理可用文献[25]中的基于 Prolog 的方法来实现,具体可参见文献[25]),发现攻击者可以通过伪装转发的方式使 S 相信 D 是 B 的邻域节点,产生漏洞的直接原因在于协议消息中的签名仅仅针对源节点、目的节点和公钥签名,未包含对路由发现过程中产生的动态或随机信息的签名,使得攻击者无须获得相应签名私钥即可发动冒充及中间人攻击.

6 结 论

本文提出了邻域限制的通信演算 NCCC 以及基于类型推理的形式化方法,用以验证 Ad-Hoc 安全路由协议.给出了类型系统,并利用类型规则定义了协议的安全属性.Dolev-Yao 模型被表示为协议近似消息集,从而显著地减少了推理进行的空间.实例表明,此方法不仅能够验证协议的安全性,也可得出针对协议的攻击手段.

在将来的工作中,将继续研究 NCCC 的扩展,希望能够应用到 Ad-Hoc 网络的其他协议的验证中.另外,研究构造协议消息集方法的普适性,包括对复杂消息结构的简化以及对其他形式化方法的适用性.

References:

[1] Papadimitratos P, Haas ZJ. Secure routing for mobile ad hoc networks. In: Proc. of the SCS Communication Networks and Distributed Systems Modeling and Simulation Conf. San Antonio, 2002. http://www.ece.neu.edu/conf/CNDS/2002/

[2] Pfitzmann B, Waidner M. A model for asynchronous reactive systems and its application to secure message transmission. In: Proc. of the IEEE Symp. on Security and Privacy. IEEE, 2001. 184–200. http://www.ieee-security.org/TC/SP01/program.html

[3] Zapata MG, Asokan N. Securing ad hoc routing protocols. In: Proc. of the ACM Workshop on Wireless Security. 2002. 1–10. http://www.informatik.uni-trier.de/~ley/db/conf/ws/ws2002.html

[4] Hu YC, Perrig A, Johnson D. Ariadne: A secure on-demand routing prortocol for ad hoc networks. Wireless Networks, 2005,11 (1,2):21−38.

[5] Qing SH. Design and logic analysis of security protocols. Journal of Software, 2003,14(7):1300−1309 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/14/1300.htm

[6] Zakiuddin I, Goldsmith M, Whittaker P, Gardiner P. A methodology for mode-checking ad-hoc networks. LNCS 2648, 2003. 624. http://www.springerlink.com/content/ar3q7h8t4423711a/

[7] Nanz S, Hankin C. A framework for security analysis of mobile wireless networks. Theoretical Computer Science, 2006,367(1): 203−227.

[8] Dolev D, Yao AC. On the security of public key protocols. IEEE Trans. on Information Theory, 1983,29(2):198−208.

[9] Cardelli L, Gordon AD. Mobile ambients. Theoretical Computer Science, 2000,240(1):177−213.

[10] Castagna G, Vitek J, Nardelli FZ. The seal calculus. Information and Computation, 2005,201(1):1−54.

[11] Milner R, Parrow J, Walker D. A calculus of mobile processes, Part 1 and 2. Information and Computation, 1992,100(1):1−77.

[12] Nicola RD, Ferrari G, Pugliese R. Klaim: A kernal language for agents interaction and mobilty. IEEE Trans. on Software Engineering, 1998,24(5):315−330.

[13] Ene C, Muntean T. A broadcast-based calculus for communicating system. In: Proc. of the 15th Int'l Parallel and Distributed Processing Symp. IEEE, 2001. 1516−1525. http://www.ipdps.org/ipdps2001/

[14] Nanz S, Nielson F, Nielson HR. Topology-Dependent abstractions of broadcast networks. Technical Report, Lyngby: Informatics and Mathematical Modelling, Technical University of Danmank, 2007.

[15] Buttyán L, Vajda I. Towards provable security for ad hoc routing protocols. In: Proc. of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks. Washington: ACM, 2004. 94−105. http://www.cs.gmu.edu/sasn/

[16] Abadi M. Secrecy by typing in security protocols. Journal of the ACM, 1999,46(5):749−786.

[17] Cardelli L, Gordon AD. Types for mobile ambients. In: Proc. of the 26th ACM SIGPLAN-SIGACT Symp. on Principles of Programming Languages (POPL'99). ACM Press, 1999. 79−92. http://www.cs.princeton.edu/~appel/popl99/

[18] Pierce BC. Behavioral equivalence in the polymorphic Pi-calculus. Journal of the ACM, 2000,47(3):532−584.

[19] Deng Y, Sangiorgi D. Towards an algebraic theory of typed mobile processes. Theoretical Computer Science, 2006,350(2,3): 188−212.

[20] Hennssy M, Merro M, Rathke J. Towards a behavioral theory of access and mobility control in distributed systems. Theoretical Computer Science, 2003,322(3):615−669.

[21] Hennessy M, Rathke J. Bisimulations for a calculus of broadcast systems. Theoretical Computer Science, 1998,200(1,2):225−260.

[22] Cervesato I. Typed multiset rewriting specifications of security protocols. Electronic Notes in Theoretical Computer Science, 2001, 40:8−51.

[23] Milner R. A theory of type polymorphism in programming. Journal of Computer and System Sciences, 1978,17:348−375.

[24] Mitchell JC. Concepts in Programming Languages. Cambridge: Cambridge Press, 2002.

[25] Giovannetti E. Types inference for mobile ambients in prolog. Electronic Notes in Theoretical Computer Science, 2004,91:96−115.

附中文参考文献:

[5] 卿斯汉.安全协议的设计和逻辑分析.软件学报,2003,14(7):1300−1309. http://www.jos.org.cn/1000-9825/14/1300.htm

李沁(1976－),男,安徽芜湖人,博士生,主要研究领域为安全协议形式化分析.

曾庆凯(1963－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为信息安全,分布计算.