安全协议逻辑程序不停机性快速预测的动态方法

周 倜[1),2)] 李梦君[1)] 李舟军[3)]

[1)](国防科学技术大学计算机学院 长沙 410073)
[2)](北京航天飞行控制中心软件室 北京 100094)
[3)](北京航空航天大学计算机学院 北京 100083)

摘 要 基于一般逻辑程序停机性刻画的动态方法,研究了解形式不动点不停机的一种动态刻画方法,给出了安全协议 Horn 逻辑扩展模型解形式不动点不停机性的一个充分条件.基于这个充分条件给出了一种不动点计算不终止的预测方法,该方法能够根据新产生的解形式逻辑规则,预测不动点计算的不终止性,同时定位模型中导致解形式不动点无穷计算的解形式逻辑规则.解形式不动点不停机性的预测结果将作为选择精确验证方法或者抽象验证方法验证安全协议的基本依据.相关实验结果表明文中给出的预测算法是高效的.

关键词 安全协议;验证;不动点计算;不停机性;预测
中图法分类号 TP309 **DOI 号**: 10.3724/SP.J.1016.2011.01275

A Dynamic Approach on Quickly Predicting Non-Termination of Logic Program of Security Protocol

ZHOU Ti[1),2)] LI Meng-Jun[1)] LI Zhou-Jun[3)]

[1)](*School of Computer Science, National University of Defense Technology, Changsha* 410073)
[2)](*Department of Software, Beijing Aerospace Control Center, Beijing* 100094)
[3)](*School of Computer Science & Engineering, Beihang University, Beijing* 100083)

Abstract Based on the dynamic approach to characterizing termination of generic logic program, a dynamic approach to characterizing termination of solved-form fixpoint is researched in this paper, and a sufficient condition of non-termination of solved-form fixpoint of the extended Horn logic model of security protocol is presented. Based on this condition, an approach is represented to predict non-termination of computation of solved-form fixpoint. This approach can predict the non-termination of computing fixpoint by checking new generated rules, and localize the solved-form rules in the extended Horn logic model which cause infinite computations of solved-form fixpoint. The non-termination prediction results will provide the criteria for choosing different verification methods of security protocols, such as accurate methods and abstract methods. The experiment results show the effectiveness of the prediction algorithms.

Keywords security protocol; verification; fixpoint computation; non-termination; prediction

1 引 言

基于逻辑程序的安全协议验证方法[1-5]能够高效地验证协议,有效地处理协议无穷会话的并发交叠运行以及对称密钥、非对称密钥、Hash 函数、消息认证码等协议原语.但是,安全协议正确性验证问题是不可判定的[6],基于逻辑程序的安全协议验证

收稿日期:2010-07-05;最终修改稿收到日期:2011-03-27.本课题得到国家自然科学基金(60973105,90604007,90104026,90718017,60703075)资助.**周 倜**,男,1981 年生,博士,工程师,主要研究方向为形式化方法与技术、安全协议验证、仿真建模与验证.E-mail:tzhou99@gmail.com.**李梦君**,男,1975 年生,博士,副教授,主要研究方向为形式化方法与技术、信息安全技术.**李舟军**,男,1963 年生,博士,教授,主要研究领域为形式化方法与技术、信息安全、数据挖掘.

方法不保证验证过程的停机性.例如,对于 Needham-Schroeder 共享密钥协议、Woo-Lam 共享密钥单向认证协议的版本 Π_3 等,基于逻辑程序的安全协议验证方法的验证过程都不停机[5].

抽象解释理论[7-10]是 Cousot 等人于 1977 年提出的构造和逼近程序不动点语义的理论,它为计算机科学中的不可判定问题和复杂问题的逼近求解提供了系统性的构造方法和有效算法.抽象解释本质上是在计算效率和计算精度之间取得均衡,以损失计算精度求得计算可行性,再通过迭代计算增强计算精度的一种逼近方法.基于抽象解释理论,文献[11]提出了一个停机的安全协议抽象验证过程,该抽象验证过程基于变种 $depth(k)$ 抽象域[12]对安全协议 Horn 逻辑扩展模型解形式不动点进行可靠的抽象逼近(抽象不动点是解形式不动点的上界逼近).

基于逻辑程序的安全协议验证方法和抽象验证方法各有利弊,一方面,基于逻辑程序的安全协议验证方法不能保证验证过程停机,但是该方法对于多数安全协议是停机的,验证过程不需要迭代精化计算;另一方面,抽象验证方法保证验证过程停机,但是它基于抽象和精化的迭代验证过程,在多次迭代计算后才能够验证安全协议正确或者构造出反例,对于解形式不动点停机的安全协议,它们的抽象验证过程也是基于抽象和精化的迭代验证过程,而且为了构造出反例,必须用未被抽象的规则再验证一次安全性质.为了充分发挥两种验证方法的优势并克服它们各自的不足,在实际验证过程中可以综合两种验证方法以提高验证效率,即对于解形式不动点停机的安全协议使用基于逻辑程序的协议验证方法,对于解形式不动点计算过程不停机的安全协议使用抽象验证方法,解形式不动点计算过程的停机性是选择验证方法的基本依据.因而,安全协议 Horn 逻辑扩展模型解形式不动点的停机性预测是一项重要的研究工作.

本文借鉴了一般逻辑程序停机性刻画的动态方法[13],给出了安全协议 Horn 逻辑扩展模型[14-15]解形式不动点计算不终止的一个充分条件,根据这个条件给出了不停机性判定的一个简单有效的预测算法.一般逻辑程序停机性刻画的静态方法借助于逻辑规则原子公式中项的大小以及其它的一些静态信息刻画停机性,而动态方法借助于逻辑推导树结构中的相关动态信息刻画停机性[13].文献[13]中目标关于一般逻辑程序的推导问题采用了 SLDNF-推理方法,基于广义的 SLDNF-树给出了一个目标关于一般逻辑程序的推导过程停机性刻画的动态方法.本文基于解形式不动点计算过程的特点,给出了解形式不动点停机性的预测算法.与文献[13]中的停机性刻画方法相比较,本文的方法能够定位模型中导致解形式不动点无穷计算的逻辑规则,并且使得本文的停机性预测算法更加精确,也具有更高的计算效率.文献[13]中的停机性刻画的动态方法用于判定一个目标关于一般逻辑程序推导过程的停机性,本文停机性预测方法用于判定解形式不动点的停机性,解形式不动点停机性的预测结果将作为选择安全协议验证方法的基本依据.为了解决文献[16]中的一些可导致不动点计算不停机的协议的验证问题,文献[17]给出了通过静态语法检查给存在形如"$\cdots P(f(x))\cdots\rightarrow P(f(g(x)))$"的规则前后件原子公式加标记的方法防止不停机情况的出现.这种方法依赖于人的主观认识,并且如果模型中没有这样的结构,则文献[17]不会做保证计算终止的处理.同时,由于在安全协议逻辑模型中存在以 $P(f(x))$ 的实例为逻辑后件的规则,该规则表示协议运行时有某个主体发出了 $P(f(x))\theta$ 的实例所描述的动作,由该规则与形如"$\cdots P(f(x))\cdots\rightarrow P(f(g(x)))$"的规则进行消解,得到以 $P(f(g(x))$ 的实例为逻辑后件的规则,新规则又可与形如"$\cdots P(f(x))\cdots\rightarrow P(f(g(x)))$"的规则进行消解,……这样的行为循环往复,从而造成不动点计算不终止的情况.而正则逻辑程序中产生的规则不一定会存在此种形式,所以该方法只针对安全协议的逻辑程序,对一般逻辑程序则难于用此方法消除不停机性.

本文通过研究安全协议不停机的动态特性,给出了一种动态刻画方法.根据逻辑程序本身存在的不停机特征,该方法在协议验证过程中给出协议验证的不停机预测.动态预测算法独立于人的主观意识,在验证过程中动态的检测新生成的规则,对不动点计算是否停机做出一定的预测.事实上,文献[17]用静态语法检查发现的不停机性只是一种特殊情况,而本文给出的预测方法完全包含了这种特例.同时,由于本文给出的动态预测算法是独立于安全协议模型、根据不动点计算出的规则特征而进行不终止预测的,因此该预测方法不仅适用安全协议逻辑程序的不停机预测,也可用于正则逻辑程序的不停机预测,为是否选择抽象验证以及如何进行局部抽象验证提供理论依据.首先可采用文献[14-15]提出的协议验证方法进行不动点计算,每次迭代计算结束时采用本文给出不停机性预测算法进行不停机性

预测.如果预测不动点计算不终止,则启动文献[11]中的抽象验证算法,否则继续进行迭代计算与预测过程,直到得到验证结果或发现计算不终止的证据为止.由于本文给出的预测算法还能精确定位导致不终止计算的规则,因此还可以进行局部抽象验证[18],即只对逻辑程序中导致不动点计算不终止的规则进行抽象,而其它规则仍然按精确验证的方法进行消解.局部抽象验证可以有效地减少抽象-精化迭代的次数,节省总的验证时间.

本文第 2 节描述安全协议 Horn 逻辑扩展模型;第 3 节描述安全性质的验证方法,它们是基于逻辑程序的安全协议验证方法中协议模型以及验证方法的变型;第 4 节研究协议验证过程的动态特性,并给出解形式不动点不停机性的预测算法;第 5 节采用不停机性预测算法,对典型安全协议 Horn 逻辑扩展模型解形式不动点的停机性进行预测,实验结果表明预测算法的有效性;第 6 节对本文内容进行小结,阐述进一步的研究工作.

2　安全协议基于 Horn 逻辑的扩展模型

安全协议基于 Horn 逻辑扩展模型的语法描述如表 1 所示.在 Horn 逻辑扩展模型语法中,如果原子公式 attacker($role(\langle M,N,tag\rangle,M')$)、begin($M,M'$)、end($M,M'$)中的项$M'$中没有变量,则称它们为闭原子公式.原子公式 begin(M,M')和 end(M,M')是对应性断言序偶,用于描述认证性,其中 M 是对应性断言标识符.严格按照协议描述进行消息交换和消息处理的角色称为诚实角色,除了安全协议攻击者之外的所有角色都是诚实角色;安全协议攻击者在安全协议形式化分析中扮演两种角色:其一是诚实角色,其二是攻击者角色,扮演攻击者角色时进行消息处理和消息交换的行为由 Dolev-Yao 模型刻画.

表 1　Horn 逻辑扩展模型的语法

表达式	名称
tag::=	标志
0,1,2,…	非负整数
M,N,U,V,S,T::=	项
x,y,z	变量
$a[M_1,\cdots,M_n]$	名
$f(M_1,\cdots,M_n)$	函数
F,C,A::=	原子公式
attacker($role(\langle M,N,tag\rangle,M')$)	attacker 谓词
begin(M,M')	begin 谓词
End(M,M')	end 谓词
R,R'::=	规则
$F_1\wedge\cdots\wedge F_n\rightarrow F$	逻辑规则

2.1　安全协议诚实角色模型

对应于安全协议描述中每一个诚实角色 A 的每一个消息发送动作,假设发送的消息为 M',则生成一条逻辑规则:attacker($role(\langle M_1,A,k\rangle,M_1')$)$\wedge\cdots\wedge$attacker($role(\langle M_n,A,k\rangle,M_n')$)$\rightarrow$attacker($role(\langle A,N,k\rangle,M')$);其中,根据安全协议描述的上下文,$A$ 认为消息 M'的接收者为 N(理想接收者),$M_1',\cdots,M_n'$是 A 在发送消息 M'之前应该依次接收到的消息序列,并且根据安全协议描述的上下文,A 认为消息 $M_i'(i=1,2,\cdots,n)$的发送者(理想发送者)为 M_i.安全协议描述中 A 生成的每一个新鲜值(nonce)和新鲜加密密钥(fresh encryption key)都被 Skolem 化为一个同名的 Skolem 函数,Skolem 函数的参数包括:一个会话标识符变量,A 生成新鲜值或新鲜加密密钥之前应该依次接收到的消息序列,k 表示该规则是加入到诚实角色模型中的第 k 条规则.

在诚实角色模型中,逻辑规则前件中形如 attacker($role(\langle M,N,tag\rangle,M')$)的原子公式中 tag 的取值均为大于 0 的整数,以方便定位基于 Horn 逻辑扩展模型中导致解形式不动点计算过程出现无穷计算的逻辑规则.

2.2　安全协议攻击者模型

攻击者进行消息处理和消息交换的行为由 Dolev-Yao 模型刻画,描述为以下的一组逻辑规则:

(1)设 S 表示安全协议的公共知识集合,对每一个 $M\in S$,生成如下的一条逻辑规则:

$\rightarrow$attacker($role(\langle host(k_I[\,]),host(v),n\rangle,M)$);

它表示的含义是:攻击者能够向任何参与者发送属于公共知识集合的消息 M.

(2)攻击者具有生成新鲜值的能力,用下面的一条逻辑规则刻画:

$\rightarrow$attacker($role(\langle host(k_I[\,]),host(v),n\rangle,N_I[i])$);

攻击者生成的新鲜值被 Skolem 化为 Skolem 函数 $N_I[i]$,其中 i 是自然数集合上的变量,上述逻辑规则的含义是:攻击者能够生成无穷多个新鲜值,并能向任何参与者发送生成的新鲜值.

(3)对于安全协议中的每一个构造算子 f,攻击者具有使用 f 构造新消息的能力,用下面的逻辑规则刻画:

attacker($role(\langle host(v_1),host(k_I[\,]),n\rangle,x_1)$)$\wedge\cdots\wedge$attacker($role(\langle host(v_n),host(k_I[\,]),n\rangle,x_n)$)$\rightarrow$attacker($role(\langle host(k_I[\,]),host(v),n\rangle,f(x_1,\cdots,$

$x_n)))$；

构造算子包括消息组合算子和加密算子等.上述逻辑规则的含义是：攻击者能够利用接收到的消息构造新的消息,并能够向任何参与者发送构造的新消息.

(4)对于安全协议中的每一个析构算子 g 和 g 的每一个消减式 $g(M_1,\cdots,M_n)=M$,攻击者具有使用 g 析构消息的能力,用下面的一条逻辑规则刻画：

$\text{attacker}(role(\langle host(v_1),host(k_I[]),n\rangle,M_1))\wedge\cdots\wedge\text{attacker}(role(\langle host(v_n),host(k_I[]),n\rangle,M_n))\rightarrow\text{attacker}(role(\langle host(k_I[]),host(v),n\rangle,M))$；

析构算子包括消息分解算子和解密算子等.上述逻辑规则的含义是：攻击者能够析构接收到的消息,得到新消息,并能够向任何参与者发送析构得到的新消息.

在攻击者模型中,tag 取值为 n 表示逻辑规则是 Horn 逻辑扩展模型中的第 n 条逻辑规则.描述保密性时,扩展模型中不出现原子公式 begin(M,M')和 end(M,M')；描述认证性时,begin(M,M')只出现在逻辑规则的逻辑前件中,end(M,M')只出现在逻辑规则的逻辑后件中.

2.3 安全性质描述

定义 1(逻辑蕴涵). 设 $R_1=H_1^1\wedge\cdots\wedge H_m^1\rightarrow C_1$ 和 $R_2=H_1^2\wedge\cdots\wedge H_n^2\rightarrow C_2$ 是两条逻辑规则,$C_1=\text{attacker}(role(\langle M_1,N_1,tag_1\rangle,M^1))$,$C_2=\text{attacker}(role(\langle M_2,N_2,tag_2\rangle,M^2))$,或者 $C_1=\text{end}(M,M^1)$,$C_2=\text{end}(M,M^2)$,则 R_1 逻辑蕴涵 R_2,记作 $R_1\Rightarrow R_2$,当且仅当：存在置换 θ 使得 $M^1\theta=M^2$,并且对每一个 $H_i^1=\text{attacker}(role(\langle M_i^1,N_i^1,tag_i^1\rangle,M_i'))\in\{H_1^1,\cdots,H_m^1\}$,存在 $H_j^2=\text{attacker}(role(\langle M_j^2,N_j^2,tag_j^2\rangle,M_j''))\in\{H_1^2,\cdots,H_n^2\}$,对每一个 $H_i^1=\text{begin}(M,M_i')\in\{M_1^1,\cdots,H_m^1\}$,存在 $H_1^2=\text{begin}(M,M_j'')\in\{H_1^2,\cdots,H_n^2\}$,使得 $M_i'\theta=M_j''$.

定义 2(逻辑可推导). 设 F 是一个闭原子公式,B 是一组逻辑规则,F 关于 B 逻辑可推导当且仅当存在满足以下条件的一棵有限树：

(1)除了根结点外,每一个结点均用 B 中的一条逻辑规则标记,并且每一条边均用一个闭原子公式标记；

(2)若树中的一个结点用 R 标记,结点的入边用闭原子公式 F_0 标记,n 条出边分别用闭原子公式 $F_1,\cdots,F_n$ 标记,则 $R\Rightarrow F_1\wedge\cdots\wedge F_n\rightarrow F_0$；

(3)根结点有唯一一条出边并且用闭原子公式 F 标记；

满足上述条件的有限树称为 F 关于 B 的逻辑可推导树.

定义 3(保密性). 设 P 是安全协议 Horn 逻辑扩展模型,$F=\text{attacker}(role(\langle M,N,tag\rangle,M'))$ 是一个闭原子公式,如果 F 关于 P 逻辑不可推导,则称安全协议关于 M' 满足保密性.

认证性用对应性断言 begin(M,M')和 end(M,M')刻画,令 $B_b=\{\rightarrow\text{begin}(M_1,M_1'),\cdots,\rightarrow\text{begin}(M_n,M_n')\}$.

定义 4(认证性). 设 P 是安全协议 Horn 逻辑扩展模型,begin(M,M')和 end(M,M')是刻画认证性的对应性断言序偶,end(M,M')是一个闭原子公式,如果 end(M,M')关于 $P\cup B_b$ 逻辑可推导,那么 begin(M,M')$\in B_b$,称安全协议关于 begin(M,M')和 end(M,M')满足认证性.

3 安全协议验证方法

定义 5(消解). 设 $R=H\rightarrow F$ 和 $R'=H'\rightarrow F'$ 是两条逻辑规则,如果 $F=\text{attacker}(role(\langle M_1,N_1,tag_1\rangle,M_1'))$(或 $F=\text{end}(M_1,M_1')$),并且 H' 中存在原子公式 $F_0=\text{attacker}(role(\langle M_2,N_2,tag_2\rangle,M_2'))$(或 $F_0=\text{end}(M_1,M_2')$),使得 M_1' 能够与 M_2' 进行合一化,则 R 与 R' 的消解记作 $R'\cdot R$,定义为 $(H\wedge(H'-F_0))\theta\rightarrow F'\theta$,其中 $\theta=\text{mgu}(M_1',M_2')$ 是 M_1' 和 M_2' 的最一般合一化算子.

定义 6(目标). 逻辑规则前件中形如 $\text{attacker}(role(\langle M,N,tag\rangle,x))$($x$ 是任意一个变量)和 begin(M,M')的原子公式称为假目标,形如 $\text{attacker}(role(\langle M,N,tag\rangle,M'))$($M'$ 不是变量)和 end(M,M')的原子公式称为目标.

定义 7(解形式逻辑规则). 设 $H\rightarrow C$ 是一个逻辑规则,如果 H 中的原子公式都是假目标,则称 $H\rightarrow C$ 是解形式逻辑规则.用 $SolvedForm$ 表示解形式逻辑规则集合,用 $UnSolvedForm$ 表示非解形式逻辑规则集合.

定义 8(X-消解). 设 $R=H\rightarrow F$ 和 $R'=H'\rightarrow F'$ 是两条逻辑规则,$R\in SolvedForm$,$R'\in UnSolvedForm$,$F=\text{attacker}(role(\langle M_1,N_1,tag_1\rangle,M_1'))$,$F_0=\text{attacker}(role(\langle M_2,N_2,tag_2\rangle,M_2'))$ 是 H' 中满足 M_1' 能够与 M_2' 进行合一化的目标,则 R 与 R' 的 X-消解记作 $R'\circ R$,定义为 $(H\wedge(H'-F_0))\theta\rightarrow F'\theta$,其中 $\theta=\text{mgu}(M_1',M_2')$.

设 R 表示一个逻辑规则，B 表示一个逻辑规则集合，定义 $addRule(R,B)$ 如下：

If $\exists R'\in B, R'\Rightarrow R$, then $addRule(R,B)=B$；

else $addRule(R,B)=\{R\}\cup\{R'\mid R'\in B, R\not\Rightarrow R'\}\cup\{marked(R'')\mid R''\in B, R\Rightarrow R''\}$

并且定义 $addRule(\{R_1,\cdots,R_n\},B)=addRule(\{R_2,\cdots,R_n\},addRule(R_1,B))$.

上述定义中，$marked(R'')$ 表示逻辑规则 R'' 不再参与 X-消解计算，本文用 Marked 表示不再参与 X-消解计算的逻辑规则的集合，UnMarked 表示不属于 Marked 的逻辑规则的集合.

设 $R=F_1\wedge\cdots\wedge F_n\rightarrow C$，$F_i=\text{attacker}(role(\langle M_i,N_i,tag_i\rangle,M_i'))(i=1,\cdots,n)$，定义 $elimdup(R)$ 为满足以下条件的逻辑规则 R'：(1) 如果对任意的 $j<i$，$M_j'\neq M_i'$，则 F_i 是 R' 逻辑前件中的一个原子公式；(2) C 是 R' 的逻辑后件.

设 P 是安全协议 Horn 逻辑扩展模型，归纳定义：

$Rule^0(P)=\{elimdup(R)\mid R\in P\}$；

$T^0(P)=Rule^0(P)\cap SolvedForm$ $C^0(P)=Rule^0(P)\cap UnSolvedForm$；

$X\_Resolution^1(P)=\{elimdup(R)\mid R=R''\circ R', R'\in T^0(P), R''\in C^0(P)\}$；

$Rule^{n+1}(P)=addRule(X\_Resolution^{n+1}(P),Rule^n(P))$；

$T^{n+1}(P)=Rule^{n+1}(P)\cap SolvedForm$；

$C^{n+1}(P)=Rule^{n+1}(P)\cap UnSolvedForm$；

$X\_Resolution^{n+1}(P)=\{elimdup(R)\mid R=R''\circ R', R'\in T^n(P), R''\in C^n(P)\}$.

定义 9(解形式不动点). 设 P 是安全协议 Horn 逻辑扩展模型，定义 $fixpoint(P)=\bigcup\{T^n(P)\mid n\geq 0\}\cap UnMarked$，$fixpoint(P)$ 称为 P 的解形式不动点.

设 P 是安全协议 Horn 逻辑扩展模型，R 是一个逻辑规则，B 是一个逻辑规则集合，对于保密性，定义 $derivablerec(R,B,P)$ 如下：

if $\exists R'\in B, R'\Rightarrow R$, then $derivablerec(R,B,P)=\varnothing$

else if $R=\rightarrow C$, then $derivablerec(R,B,P)=\{\rightarrow C\}$

else $derivablerec(R,B,P)=\{derivablerec(elimdup(R\cdot R'),\{R\}\cup B,P)\mid R'\in fixpoint(P)\}$.

对于认证性，定义 $derivablerec(R,B,P)$ 如下：

if $\exists R'\in B, R'\Rightarrow R$, then $derivablerec(R,B,P)=\varnothing$

else if $R=\text{begin}(M_1,M_1')\wedge\cdots\wedge\text{begin}(M_n,M_n')\rightarrow\text{end}(M,M')$, then $derivablerec(R,B,P)=\{R\}$

else $derivablerec(R,B,P)=\bigcup\{derivablerec(elimdup(R\cdot R'),\{R\}\cup B,P)\mid R'\in fixpoint(P)\}$.

对于形如 attacker($role(\langle M,N,tag\rangle,M')$)和 end($M,M'$)的闭原子公式 F，定义 $derivable(F,P)=derivablerec(F\rightarrow F,\varnothing,P)$.

定理 1[15]. 设 P 是安全协议 Horn 逻辑扩展模型，F 是一个闭原子公式，则 $derivable(F,P)$ 的计算过程停机.

定理 2[15]. 设 P 是安全协议 Horn 逻辑扩展模型，则

① 保密性：若 F 是形如 attacker($role(\langle M,N,tag\rangle,M')$)的闭原子公式，则 F 关于 $fixpoint(P)$ 逻辑可推导当且仅当 $\rightarrow F\in derivable(F,P)$.

② 认证性：设 F 是形如 end(M,M')的闭原子公式，F 关于 $fixpoint(P)\cup B_b$ 逻辑可推导，begin(M,M')$\in B_b$ 当且仅当：存在 $H_1\wedge\cdots\wedge H_n\rightarrow F\in derivable(F,P)$，其中 $H_i\in B_b$ 均为形如 begin(M_i,M_i')的原子公式，并且 begin(M,M')$\in\{H_1,\cdots,H_n\}$.

4　不停机性的动态预测

设 P 是安全协议 Horn 逻辑扩展模型，由定理 1 可知，只要 P 的解形式不动点的计算过程停机，则安全协议的验证过程一定停机，下面将给出安全协议逻辑模型中解形式不动点计算过程的不停机性预测方法.

安全协议 Horn 逻辑扩展模型中的逻辑规则数目是有穷的，解形式不动点计算不停机的原因在于安全协议逻辑程序模型中的一些逻辑规则被无穷次地应用于计算 X-消解. 设 A 和 B 是两个原子公式，如果删除 A 中增加的函数符号、名和变量，原子公式 A 就是原子公式 B 或 B 的变量换名，则称 A 是 B 的递归增加. 一条逻辑规则被无穷次地应用于计算 X-消解，由最一般合一化算法可知，将表现为逻辑规则中一些原子公式的变量换名在解形式不动点中无穷次地出现，或者一些原子公式在解形式不动点中无穷次地递归增加[13]. 解形式不动点中的规则都是解形式逻辑规则，解形式逻辑规则逻辑前件中的原子公式都是形如 attacker($role(\langle M,N,tag\rangle,x)$)($x$ 是任意一个变量)和 begin(M,M')的简单原子公式，因而解形式不动点不停机将表现为：安全协议 Horn 逻辑扩展模型中的一些逻辑规则的逻辑后件的变量换名在解形式不动点中无穷次地出现，或者这些逻辑规则的逻辑后件在解形式不动点中无穷次地递归增加.

定义 10(符号字符串). 设 T 是一个项(或是形如 attacker($role$(⟨M, N, tag⟩, M'))的原子公式),则 T 的字符串 S 是 T(或 M')中的函数符号、名、变量按照从左到右的顺序连接在一起构成的字符串,T 的符号字符串 S_T 是将 S 中的所有变量全部用新符号 χ 替换后得到的字符串.

符号字符串按照从左到右的顺序依次抽取出原子公式中的函数符号、名和变量,并将变量用新符号 χ 替换.例如,对于原子公式 attacker($role$(⟨$host(k_{SS}[\,])$, $host(k_{BS}[\,])$, n⟩, $encrypt$($2tuple$($host$($k_{AS}[\,]$), $2tuple$($host$ ($k_{AS}[\,]$), $2tuple$($host$($k_{AS}[\,]$), v_6))), $k_{BS}[\,]$))),它的符号字符串为 $encrypt$ · $2tuple$ · $host$ · k_{AS} · $2tuple$ · $host$ · k_{AS} · $2tuple$ · $host$ · k_{AS} · χ · k_{BS},其中"·"表示字符串的连接.

定义 11(投影). 设 S_{T1} 和 S_{T2} 是两个符号字符串,如果 S_{T1} 是将字符串 S_{T2} 中的 0 个或多个符号删除得到的字符串,则称 S_{T1} 是 S_{T2} 的投影,记作 $S_{T1}\subseteq_{\text{proj}}S_{T2}$.

投影关系用于刻画一个原子公式 A 是原子公式 B 的变量换名,或者原子公式 A 是原子公式 B 的递归增加.例如,原子公式 attacker($role$(⟨$host(k_{SS}[\,])$, $host(k_{BS}[\,])$, 13⟩, $encrypt$($2tuple$($host$($k_{AS}[\,]$), $2tuple$($host$($k_{AS}[\,]$), v_6)), $k_{BS}[\,]$))) 的符号字符串是 attacker($role$(⟨$host(k_{SS}[\,])$, $host(k_{BS}[\,])$, 13⟩, $encrypt$($2tuple$($host$($k_{AS}[\,]$), $2tuple$($host$($k_{AS}[\,]$), $2tuple$($host$($k_{AS}[\,]$), v_6))), $k_{BS}[\,]$))))的符号字符串的投影.

定义 12(循环到). 设 A_1 = attacker($role$(⟨M_1, N_1, tag_1⟩, M_1')) 和 A_2 = attacker($role$(⟨M_2, N_2, tag_2⟩, M_2'))是两个原子公式,如果 $S_{A1}\subseteq_{\text{proj}}S_{A2}$ 并且 $tag_1=tag_2$,则称 A_1 循环到 A_2,记作 $A_1\rightsquigarrow_{\text{loop}}A_2$.

与文献[13]中的定义不同,本文中循环到的定义增加了约束条件 $tag_1=tag_2$,$tag_1=tag_2$ 约束原子公式 A_1 和 A_2 必须是安全协议 Horn 逻辑扩展模型中同一个逻辑规则的逻辑后件的不同实例.

定义 13(规则的长幼关系). 若规则 $R_i=H_i\rightarrow F_i(i=1,2,3)$ 满足 $R_3=R_1\circ R_2$,则 R_1,R_2 与 R_3 之间具有长幼关系 $R_1\prec R_3$,$R_2\prec R_3$,在不引起歧义时可写作 $F_1\prec F_3$,$F_2\prec F_3$.用 $\prec^c$ 表示 $\prec$ 的传递闭包.

定义 14(循环目标). 设 $H_j\rightarrow A_j$,$H_i\rightarrow A_i$ 均为解形式规则,如果 $A_i\prec^c A_j$ 且 $A_i\rightsquigarrow_{\text{loop}}A_j$,则称 A_j 是 A_i 的循环目标.

引理 1[13]. 设 $\{A_i\}_{i=1}^{\infty}$ 是有穷字母表 $\sum$ 上字符串的一个无穷序列,则存在一个无穷的递增整数序列 $\{n_i\}_{i=1}^{\infty}$,使得对于任意的 i,均有 $A_{n_i}\subseteq_{\text{proj}}A_{n_{i+1}}$.

定理 3. $\forall n\in N$, $Rule^n(P)$ 为有限集.

证明. 用归纳法证明.

(1) 当 $n=0$ 时,因为安全协议的逻辑程序模型 P 是有限集,所以 $Rule^0(P)=\{elimdup(R)\mid R\in P\}$ 是有限集,所以当 $n=0$ 时,结论成立.

(2) 假设当 $n=k$ 时结论成立.当 $n=k+1$ 时,由归纳假设,$Rule^k(P)$ 是有限集,所以其子集 $T^k(P)$,$C^k(P)$ 均为有限集,则由两个有限集进行 X-消解得到的新规则集 $X\_Resolution^{k+1}(P)=\{elimdup(R)\mid R=R''\circ R',R'\in T^k(P),R''\in C^k(P)\}$ 为有限集,所以 $Rule^{k+1}(P)=addRule(X\_Resolution^{k+1}(P), Rule^k(P))$ 为有限集.

综上所述,$\forall n\in N$, $Rule^n(P)$ 为有限集. 证毕.

定理 4. 设 P 是一个安全协议的逻辑程序模型,如果存在无穷解形式逻辑规则序列 $H_1\rightarrow A_1,\cdots, H_{g_1}\rightarrow A_{g_1},\cdots, H_{g_2}\rightarrow A_{g_2},\cdots, H_{g_i}\rightarrow A_{g_i},\cdots, H_{g_{i+1}}\rightarrow A_{g_{i+1}},\cdots$,其中 $A_{g_{i+1}}$ 是 $A_{g_i}(i=1,2,\cdots)$ 的循环目标,则 P 的解形式不动点计算不终止.

证明. 假设 P 的解形式不动点计算终止,则 $\exists m\in N$,使得 $T^m(P)=T^{m+1}(P)$, $C^m(P)=C^{m+1}(P)$. 则 $\forall n\in N$ 且 $n>m$, $Rule^n(P)=Rule^m(P)$. 又由定理 3 知,$Rule^n$ 为有限集. 所以 $\bigcup\{Rule^n(P)\mid n\geq 0\}=\bigcup_{n=0}^{m}Rule^n(P)$ 为有限集,这与题设中存在无穷解形式逻辑规则序列矛盾. 所以假设不成立,即必有 P 的解形式不动点计算不终止.

证毕.

定理 4 给出了不动点计算不终止的一个充分条件,该条件表明安全协议 Horn 逻辑扩展模型解形式不动点计算过程的不停机性可以刻画为:存在无穷选择序列 $A_1,\cdots,A_{g_1},\cdots,A_{g_2},\cdots,A_{g_i},\cdots,A_{g_{i+1}},\cdots$,使得对于任意的 $i\geq 1$,$A_{g_{i+1}}$ 是 A_{g_i} 的循环目标.检测满足上述条件的无穷选择序列是不现实的,为了可在计算机中实现,可以采用近似的思想:选定一个正整数 k 作为阈值,检测安全协议逻辑程序解形式不动点的计算过程中是否存在选择序列 $A_1,\cdots,A_{g_1},\cdots,A_{g_2},\cdots,A_{g_k}$,使得对于任意的 $i(1\leq i\leq k)$,$A_{g_{i+1}}$ 是 A_{g_i} 的循环目标.

将满足上述条件的选择序列称为不停机条件.如果满足不停机条件,则预测安全协议 Horn 逻辑扩展模型解形式不动点的计算过程不停机,否则预测解形式不动点的计算过程停机.显然,k 值越大,预测的准确度越高,需要的时空代价越高;反之,预测速度越快,预测的准确度越低.如果满足上述条件

的选择序列中原子公式的 tag 项的取值均为 n_0，则预测安全协议 Horn 逻辑扩展模型中第 n_0 条逻辑规则是导致解形式不动点无穷计算的逻辑规则. 此时也可针对标号为 n_0 的规则进行局部抽象验证，这样可以更加逼近精确模型，减少抽象-精化迭代的次数，从而降低验证一个协议所需花费的总的时间开销.

5　实　验

采用本文提出的不停机性预测算法，选取阈值 $k=3$，基于安全协议验证工具 SPVT，在 Intel(R) Pentium(R) D 2.66 GHz CPU 和 1 GB 内存的 PC 机上，对典型安全协议的 Horn 逻辑扩展模型解形式不动点的停机性进行了预测. 实验过程在计算安全协议逻辑程序解形式不动点的同时检测不停机条件是否满足，如果不停机条件得到满足，则预测解形式不动点不停机，如果不停机条件不满足，则继续迭代计算解形式不动点，直到完全计算出解形式不动点，或者预测解形式不动点不停机. 表 2 列出了采用预测算法帮助协议验证的实验结果.

表 2　协议验证的预测结果

安全协议	停机性	预测算法判定结果($k=3$)	预测时间/s	总时间/s
Needham-Schroeder 共享密钥协议	不停机	不停机	0.0	2.859
Yahalom 协议	停机	停机	0.016	2.61
Otway-Rees 协议	停机	停机	0.015	9.203
Woo-Lam 认证协议版本 II	停机	停机	0.0	0.641
Woo-Lam 认证协议版本 II_1	停机	停机	0.015	6.438
Woo-Lam 认证协议版本 II_2	停机	停机	0.0	0.578
Woo-Lam 认证协议版本 II_3	不停机	不停机	0.0	2.859
Woo-Lam 认证协议版本 II_f	停机	停机	0.031	6.0
公钥 Kerberos 协议认证服务阶段	停机	停机	0.015	—
Denning-Sacco 协议	停机	停机	0.016	0.813
Neuman-Stubblebine 协议	停机	停机	0.0	2.672
Andrew secure RPC 协议	不停机	不停机	0.0	30.812

表 2 中的总时间是指解形式不动点计算、安全性质检测、反例构造等阶段的全部时间开销. 需要说明的是，公钥 Kerberos 协议认证服务阶段的验证过程是终止的，但是在构造反例时，由于 *derivable* 函数的可选择性多，造成了组合爆炸，完成反例构造需要的时间过长，因此表中不能给出总时间. Needham-Schroeder 共享密钥协议、Woo-Lam 认证协议版本 II_3 以及 Andrew secure RPC 协议的解形式不动点被预测为不停机，实验中预测计算花费的时间非常少，以至于计算机近似认为时间为 0s. 其它安全协议的解形式不动点被预测为停机，由于停机的协议在达到不动点时才停止预测，因此预测所检测的规则数比不停机的协议要多得多，所以 SPVT 能计算出预测时间. 从表中可以看出，验证一个协议的总时间开销比预测时间多得多，所有协议的停机性预测花费的时间基本上可以忽略不计. 上述实验结果表明：安全协议的逻辑模型的解形式不动点计算过程的停机性在选取阈值 $k=3$ 时，使用本文的不停机性预测算法得到的判定结果都是正确的，从而表明停机性预测算法的实用性.

下面以 Needham-Schroeder 共享密钥协议(简称为 NS 协议)为例，说明不停机性预测算法的具体应用. NS 协议的形式化描述如下：

(1) $A\rightarrow S$：A,B,Na；

(2) $S\rightarrow A$：$\{Na,B,Kab,\{Kab,A\}_{Kbs}\}_{Kas}$；

(3) $A\rightarrow B$：$\{Kab,A\}_{Kbs}$；

(4) $B\rightarrow A$：$\{Nb\}_{Kab}$；

(5) $A\rightarrow B$：$\{Nb-1\}_{Kab}$.

由于消息(4)、(5)会导致消解过程中不断出现形如$\{Nb-1-\cdots-1\}_{Kab}$的消息，从而造成不动点计算不终止. 在 NS 协议的逻辑程序的不动点计算过程中，SPVT 产生了如下逻辑程序规则：

规则 4. 〈Ready〉: Begin($Aparam$[($host$(key(kas())), $s1$)]) ∧ Attacker($LKerb$($host$(key(kss())), $host$(key(kas())), "4", $encrypt$((Na($s1$), $host$(key(kbs())), $x4$, YM), key(kas())))) ∧ Attacker($LKerb$($host$(key(kbs())), $host$(key(kas())), "4", $encrypt$($x3$, $x4$))) → Attacker($LKerb$($host$(key(kas())), $host$(key(kbs())), "4", $encrypt$($minus$($x3$), $x4$))).

规则 37. 〈Ready〉: Begin($Aparam$[($host$(key(kas())), $z16$)]) ∧ Attacker($LKerb$($host$(key(kbs())), $host$(key(kas())), "4", $encrypt$($z7$, $Akey$(($host$(key(kas())), $host$(key(kbs())), Na($z16$)), $z2$)))) → Attacker($LKerb$($host$(key(kas())), $host$(key(kbs())), "4", $encrypt$($minus$($z7$), $Akey$(($host$

(*key*(*kas*( )), *host*(*key*(*kbs*( )), *Na*(*z*16)), *z*2)))).

规则 43. 〈Solved〉: Begin(*Aparam*[(*host*(*key*(*kas*( )), *z*16)]) → Attacker(*LKerb*(*host*(*key*(*kas*( ))), *host*(*key*(*kbs*( ))), "4", *encrypt*(*minus*(*Nb*(*encrypt*((*Akey*((*host*(*key*(*kas*( ))), *host*(*key*(*kbs*( ))), *Na*(*z*16)), *z*2), *host*(*key*(*kas*( )))), *key*(*kbs*( ))), *z*6)), *Akey*((*host*(*key*(*kas*( ))), *host*(*key*(*kbs*( ))), *Na*(*z*16)), *z*2)))).

规则 45. 〈Solved〉: Begin(*Aparam*[(*host*(*key*(*kas*( )), *z*16)]) → Attacker(*LKerb*(*host*(*key*(*kas*( ))), *host*(*key*(*kbs*( ))), "4", *encrypt*(*minus*(*minus*(*Nb*(*encrypt*((*Akey*((*host*(*key*(*kas*( ))), *host*(*key*(*kbs*( ))), *Na*(*z*16)), *z*2), *host*(*key*(*kas*( )))), *key*(*kbs*( ))), *z*6))), *Akey*((*host*(*key*(*kas*( ))), *host*(*key*(*kbs*( ))), *Na*(*z*16)), *z*2)))).

规则 46. 〈Solved〉: Begin(*Aparam*, (*host*(*key*(*kas*( )), *z*16)) → Attacker(*LKerb*(*host*(*key*(*kas*( ))), *host*(*key*(*kbs*( ))), "4", *encrypt*(*minus*(*minus*(*minus*(*Nb*(*encrypt*((*Akey*((*host*(*key*(*kas*( ))), *host*(*key*(*kbs*( ))), *Na*(*z*16)), *z*2), *host*(*key*(*kas*( ))))*key*(*kbs*( ))), *z*6)))), *Akey*((*host*(*key*(*kas*( ))), *host*(*key*(*kbs*( ))), *Na*(*z*16)), *z*2)))).

规则 4 是安全协议模型中的初始规则,描述的是协议中的第(5)条消息,即 Alice 将收到的 *Nb*−1 再发给 Bob. 随着不动点计算的进行,规则 4 逐步消除前件中的目标而变成规则 37. 非解形式规则 37 与解形式规则 43 进行 *X*-消解得到解形式逻辑规则 45,规则 45 继续与规则 37 进行 *X*-消解得到解形式规则 46. 而规则 46 的逻辑后件是规则 45 的逻辑后件的循环目标,规则 45 的逻辑后件是规则 43 的逻辑后件的循环目标. 事实上,规则 37 还能继续和规则 46 进行 *X*-消解生成新的解形式逻辑规则,且新规则的逻辑后件是规则 46 的逻辑后件的循环目标. 这个过程会一直进行下去而不断地生成新的解形式规则. 事实上,这些规则的逻辑后件都是规则 4 的逻辑后件的具体实例. 根据本文的预测算法,NS 协议的解形式不动点的计算过程不停机,并且标记为 4 的第 4 条逻辑规则是模型中导致解形式不动点计算过程不停机的逻辑规则.

6 结束语

借鉴文献[13]中一般逻辑程序停机性刻画的动态方法,本文给出了安全协议 Horn 逻辑扩展模型解形式不动点计算不停机性的预测算法,该算法为验证工具自动选择精确验证或抽象验证提供了依据. 停机性算法同时能定位模型中导致解形式不动点无穷计算的逻辑规则,在这种精确定位的帮助下,可对协议进行局部抽象验证,即只对造成不停机原因的规则进行抽象,而其他规则不进行抽象. 局部抽象验证模型更接近精确模型,因而可以有效地减少抽象-精化迭代的次数. 相关实验结果表明了本文提出的预测算法是实用、高效的.

进一步的研究工作包括:继续深入研究安全协议 Horn 逻辑扩展模型解形式不动点停机性的动态刻画方法,最终给出不动点计算不终止的充要条件;研究反例构造的启发式算法,尽量减少组合爆炸;对实用安全协议(如 Kerberos 协议等)进行验证.

参 考 文 献

[1] Blanchet B. An efficient cryptographic protocol verifier based on prolog rules//Proceedings of the 14th IEEE Computer Security Foundations Workshop (CSFW-14). Cape Breton, Nova Scotia, Canada, 2001: 82-96

[2] Blanchet B. From secrecy to authenticity in security protocols//Proceedings of the 9th International Static Analysis Symposium (SAS'02). Madrid, 2002: 242-259

[3] Abadi M, Blanchet B. Analyzing security protocols with secrecy types and logic programs//Proceedings of the 29th ACM Symposium on Principles of Programming Languages (POPL'02). Portland, 2002: 33-44

[4] Allamigeon X, Blanchet B. Reconstruction of attacks against cryptography protocols//Proceedings of the 18th IEEE Computer Security Foundations Workshop (CSFW18). Aix-en-Provence, France, 2005: 140-154

[5] Blanchet B, Podelski A. Verification of cryptographic protocols: Tagging enforces termination. Theoretical Computer Science, 2005, 333(1-2): 67-90

[6] Durgin N, Lincoln P, Mitchell J, Scedrov A. Undecidability of bounded security protocols//Proceedings of the Formal Methods and Security Protocols, FLOC Workshop. Trento, 1999

[7] Cousot P, Cousot R. Abstract Interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints//Proceedings of the 4th POPL. Los Angeles, CA, 1977: 238-252

[8] Cousot P, Cousot R. Systematic design of program analysis frameworks//Proceedings of the 6th POPL. San Antonio, TX, 1979: 269-282

[9] Cousot P, Cousot R. Abstract interpretation frameworks. Journal of Logic and Computation, 1992, 2(4): 511-547

[10] Cousot P, Cousot R. Basic concepts of abstract interpretation//Proceedings of the Building the Information Society. Toulouse, France, 2004: 359-366

[11] Li M, Zhou T, Li Z, Chen H. An abstract and refinement framework for verifying security protocols based on logic programming//Cervesato I ed. Proceedings of the ASIAN2007. LNCS: 4846. Springer, 2007: 166-180

[12] Gori R, Lastres E, Moreno R, Spoto F. Approximation of the Well-Founded Semantics for Normal Logic Programs using Abstract Interpretation//Proceedings of the APPIA-GULP-PRODE'98 Conference. Spain, 1998: 433-441

[13] Shen Y, You J, Yuan L, Shen S, Yang Q. A dynamic approach to characterizing termination of general logic programs. ACM Transactions on Computational Logic, 2003, 4(4): 417-430

[14] Li Meng-Jun, Li Zhou-Jun, Chen Huo-Wang. SPVT: An efficient verification tool for security protocol. Journal of Software, 2006, 17(4): 898-906(in Chinese)
(李梦君，李舟军，陈火旺. SPVT:一个有效的安全协议验证工具. 软件学报，2006，17(4)：898-906)

[15] Li Meng-Jun, Li Zhou-Jun, Chen Huo-Wang. Security protocol's extended Horn logic model and its verification method. Chinese Journal of Computers, 2006, 29(9): 1666-1678 (in Chinese)
(李梦君，李舟军，陈火旺. 安全协议的扩展 Horn 逻辑模型及其验证方法. 计算机学报，2006，29(9)：1666-1678)

[16] Clark J, Jacob J. A survey of authentication protocol literature. Technical Report 1.0, 1997. http://citeseer.ist.psu.edu/clark97 survey.html

[17] Blanchet B, Podelski A. Verification of cryptographic protocols: Tagging enforces termination. Theoretical Computer Science, 2005, 333(1-2): 67-90

[18] Zhou Ti, Li Meng-Jun, Li Zhou-Jun. Local abstract verification and refinement of security protocols//Vitaly Shmatikov ed. Proceedings of the FMSE. ACM, 2008: 21-30

ZHOU Ti, born in 1981, Ph. D., engineer. His main research interests include formal methods and technology, verification of security protocols, and model and verification of simulation.

LI Meng-Jun, born in 1975, Ph. D., associate professor. His main research interests include formal methods and technologies, and information security technologies.

LI Zhou-Jun, born in 1963, Ph. D., professor. His main research interests include formal methods and technology, information security, data mining.

Background

With the development of network, security problems in the internet are becoming more and more urgent and important. Security properties encounter more and more challenges. Security protocol is one of the effective methods to solve the security issues. The open internet with security protocols can realize various security functions. Security protocol is a type of communication protocol based on cryptography, which runs over the computer network or distributed systems, and arranges the execution steps and the execution rules with the cryptographic algorithms among the participants who implement tasks. The correctness of security protocols is critical for the transactions over the open networks. However, security protocols designed and used in the real world do not always achieve their objectives. Many protocols have been found having flaws after a long application period.

Therefore, it's a hot field in formal method to design and develop an efficient automatic tool for the verification of security protocols, and it's a key step in verification of complex security protocols.

This paper presents a sufficient condition of non-termination of the solved-form fixpoint of the extended Horn logic model of security protocols. The termination of generic logic program is characterized by Shen, but there is no effective methods in predicting non-termination of computation of solved-form fixpoint of the extended Horn logic model except ours. Based on this sufficient condition, an approach is represented to predict non-termination of computation of solved-form fixpoint in this paper.

In recent years, we research several formal methods in modeling and verifying security protocols, and present some methods to model and verify protocols on the fly, and show algorithms for constructing conter-examples, design and implement the security protocol verifier SPVT, analyze and verify some protocols, present abstraction and iterative refinement framework, and give the framework to model and verify time sensitive security protocols. This paper focuses on the termination of verification. The non-termination prediction results will provide the criteria for choosing different verification methods.

This paper is supported by the National Natural Science Foundation of China under grant Nos. 60973105, 90604007, 90104026, 90718017, 60703075.