文章编号:1001 - 9081(2007)02 - 0303 - 05

一种基于问题求解理论的密码协议形式模型

赵 宇,王亚弟,韩继红,范钰丹,赵 琦
(信息工程大学 电子技术学院,河南 郑州 450004)
(zhaoyn83@ hotmail. com)

摘 要:提出了一种基于问题求解理论的密码协议模型,给出了模型的基本语法以及基于 ρ 演算的形式语义,明确了模型推理过程中涉及到的一些关键性的概念和命题。该模型具有以下特点:能够对密码协议进行精确的形式化描述;具有合理可靠的可证明语义;对密码协议安全性的定义精确合理;便于实现自动化推理。所有这些均确保了基于该模型的密码协议安全性分析的合理性和有效性,为正确的分析密码协议的安全性提供了可靠依据。

关键词:密码协议;形式模型;问题求解理论;ρ 演算;运算语义
中图分类号: TP309 **文献标识码**:A

Formal model for cryptographic protocols based on problem-solving theory

ZHAO Yu, WANG Ya-di, HAN Ji-hong, FAN Yu-dan, ZHAO Qi
(*Institute of Electronic Technology, Information Engineering University, Zhengzhou Henan* 450004, *China*)

Abstract: This paper proposed a problem-solving theory based formal model, introduced the basic syntax and ρ-calculus based semantics of the model, and presented some pivotal concepts and propositions in the deduction of the model. The model has some properties as follows: can give accurate formal specifications for cryptographic protocols; has provable semantics which is reasonable and sound; can define the security properties precisely and reasonably; is easy to realize automatic deductions. All of those aforementioned make sure that the security analysis of cryptographic protocols based on this model is reasonable and efficient, and provide a dependable basis for analyzing the security of cryptographic protocols correctly.

Key words: cryptographic protocols; formal model; problem-solving theory; ρ-calculus; operational semantics

0 引言

密码协议形式模型是对密码协议形式化分析的基础和依据,协议形式模型的好坏直接影响密码协议形式化分析过程的正确性和可靠性。如果依赖于一个不完善的形式模型对密码协议进行安全性分析,即使协议本身确实存在着缺陷,而分析的结果却可能与之相反,如早期的 BAN 逻辑模型[1]等。在对一些典型的密码协议形式模型[3~10]进行了研究的基础上,通过归纳分析,可知一个合理有效的协议模型应当具备下述特点:1)能够对密码协议进行精确的形式化描述;2)具有合理可靠的可证明语义;3)对密码协议安全特性的定义应当精确合理;4)便于实现自动化推理。

在已有研究工作的基础上,本文提出了一种基于人工智能领域问题求解理论[2]的密码协议模型。该模型很好地吻合了前面所提出四种模型需求,为正确的分析密码协议的安全性提供了可靠保障。

1 密码协议安全性问题

对密码协议安全性进行分析的过程,事实上就是验证密码协议在运行过程中是否可能出现一些不安全的情形或状态。如果协议运行中没有不安全的情形出现,则说明相应的密码协议能够满足其预期的安全性需求或目标;否则,说明密码协议中不满足预期的安全性需求,亦即密码协议中存在着安全缺陷或攻击。因此,对密码协议的安全性进行分析的过程可以建模对这样一种问题进行求解的过程:问题的初始状态为密码协议初始运行时的状态;问题的动作为协议主体(包括攻击者)的动作;问题的目标为违背协议预期安全特性的状态。

1.1 模型概述

密码协议安全性问题可以形式化地描述为一个三元组 $\Pi = < InitStates, ActionsSet, Goals >$。其中:

1) $InitStates = \{s_1 \cup s_2 \mid s_1 \in InitStates_P, s_2 \in InitStates_I\}$ 为初始状态集合,用于描述协议初始运行时的合法主体和攻击者所处的状态及所拥有的知识。

2) $ActionsSet = ActionsSet_P \cup ActionsSet_I$ 为动作集合,用于描述协议运行过程中合法主体和攻击者的可能行为。其中,$InitStates_P$、$ActionsSet_P$ 结合在一起共同用于描述密码协议运行过程中合法主体的状态、知识和行为;$InitStates_I$、$ActionsSet_I$ 结合在一起共同用于描述协议运行过程中攻击者的状态、知识和行为。

3) *Goals* 为目标集合,用于描述协议运行过程中可能出现的不安全状态。

如果上述问题有解,即存在一个从问题初始状态到目标状态的动作序列,则说明相应的密码协议是不安全的,问题解

收稿日期:2006 - 08 - 08;**修订日期**:2006 - 10 - 27 **基金项目**:国家 973 计划项目资助(TG19990358.01)

作者简介:赵宇(1983 -),男,山东成武人,硕士研究生,主要研究方向:计算机网络安全、密码协议自动化验证技术; 王亚弟(1953 -),男,甘肃兰州人,教授,博士生导师,主要研究方向:计算机网络安全、信息系统安全; 韩继红(1966 -),女,山西定襄人,副教授,主要研究方向:计算机网络安全、信息系统安全; 范钰丹(1982 -),女,河南南阳人,硕士研究生,主要研究方向:计算机网络安全、密码协议自动化验证技术; 赵琦(1981 -),男,山东沂水人,助教,硕士,主要研究方向:计算机网络安全.

中的动作序列则对应着一次针对协议的攻击。

密码协议安全性问题的定义中没有考虑耗散函数,主要是由于对密码协议安全性问题进行求解时,关心的是解(即针对密码协议的攻击)的存在性,因此不需要关心问题解的质量(攻击路径的优劣)。

模型的描述采用了一阶逻辑的方法。下面首先给出两个基本概念。

定义 1 事实空间:模型 $\Pi = < InitState, ActionsSet, Goals >$ 中所有可能的事实构成模型的事实空间,记为 F。

定义 2 状态空间:模型 $\Pi = < InitState, ActionsSet, Goals >$ 中所有可能的状态构成模型的状态空间,记为 S。

Needham-Schroeder 公钥协议的自然描述如下所示:

Step1 $A \rightarrow B: \{A, N_A\}_{pubKB}$

Step2 $B \rightarrow A: \{N_A, N_B\}_{pubKA}$

Step3 $A \rightarrow B: \{N_B\}_{pubKB}$

1.2 模型基本语法及语义

模型描述中的语法结构主要包括项、逻辑事实、状态、ρ 演算规则等,生成规则如下所示:

$Term ::= AtomTerm \mid CompoundTerm$

$AtomTerm ::= Var \mid Num \mid PName \mid Key \mid Nonce$

$Var ::= a \mid b \mid k \mid i \mid r \mid m \mid n \mid x \mid y \mid z \mid \cdots$

$Num ::= 0 \mid 1 \mid 2 \mid \cdots$

$PName ::= A \mid B \mid C \mid S \mid T \mid I \mid Var \mid \cdots$

$Key ::= symK \mid pubK \mid privK \mid \cdots$

$Nonce ::= N \mid Var \mid \cdots$

$CompoundTerm ::= \{Term\}_{Key} \mid [Term(,Term)^*] \mid Hash(Term)$

$Fact ::= Await(PName, PName, Knowledge, Stepid, Sessionid) \mid Send(PName, PName, Term, Stepid, Sessionid) \mid Receive(PName, PName, Term, Stepid, Sessionid) \mid Shared(Term, PSet) \mid Inverse(Key, Key) \mid Intrude(Term) \mid Begin(PName, PName, Authid(,Term)^+) \mid End(PName, PName, Authid(,Term)^+) \mid Not\ Fact$

$Knowledge ::= \{Term(,Term)^*\}$

$Stepid ::= Num$

$Sessionid ::= Num$

$PSet ::= \{Name(,Name)^*\}$

$Authid ::= auth_1 \mid auth_2 \mid \cdots$

$State ::= \{\} \mid \{Fact(,Fact)^*\}$

$\rho - Rule ::= PreCond\ (\ PostCond$

$PreCond ::= \{\} \mid \{Fact(,Fact)^*\}$

$PostCond ::= \{\} \mid \{Fact(,Fact)^*\}$

其中,Num 表示常数,主要用于标识不同的协议步 $Stepid$ 和会话 $Sessionid$;Var 表示变量;$Term$ 表示协议中的消息项,分为原子消息 $AtomTerm$ 和复合消息 $CompoundTerm$ 两类;原子消息主要包括变量 Var、常数 Num、主体名 $PName$、密钥 Key、随机数 $Nonce$ 等;复合消息主要通过加密 $\{Term\}_{Key}$、连接 $[Term(,Term)^*]$ 和散列 $Hash(Term)$ 产生;密钥 Key 包括对称密钥 $symK$、公开密钥 $pubK$ 和私有密钥 $privK$;事实 $Await(A, B, Knowledge, Stepid, Sessionid)$ 表示在协议会话标识为 $Sessionid$、协议步为 $Stepid$ 时主体 B 拥有知识 $Knowledge$ 且等待着主体 A 发送消息;事实 $Send(A, B, Term, Stepid)$ 表示在协议步为 $Stepid$ 时主体 A 向主体 B 发送协议消息 $Term$;事实 $Receive(A, B, Term, Stepid)$ 表示在协议步为 $Stepid$ 时主体 A 接收来自主体 B 的消息 $Term$;事实 $Shared(Term, PSet)$ 表示消息 $Term$ 是一个为 $PSet$ 集合中的主体所共享的秘密;事实 $Inverse(pubK, privK)$ 表示 $pubK$ 和 $privK$ 为一对公私钥,互为加解密密钥;事实 $Intrude(Term)$ 表示攻击者获取了消息 $Term$,亦即 $Term$ 属于攻击者的知识;事实 $Begin(A, B, Authid(, Term)^+)$ 和事实 $End(A, B, Authid(, Term)^+)$ 主要用于对密码协议认证性的建模,A 和 B 分别表示认证的主体,$Authid$ 用于标识一次特定的认证过程,$(, Term)^*$ 表示进行一致性检查的数据;$Not\ f$ 表示对事实 f 取反;$State$ 是由逻辑事实组成一个多重集,用于描述协议运行过程中的状态,当协议运行处于某一状态 s 时,对于一个特定事实 f,要么 $f \in s$,要么 $Not\ f \in s$;一阶 ρ 演算规则 $\rho - Rule$ 用于建模协议步和攻击者的计算能力,其中的 $PreCond$ 称为 $\rho - Rule$ 的规则前件,$PostCond$ 称为 $\rho - Rule$ 的规则后件。

1.3 协议自身描述

在模型中,对密码协议自身的形式描述包括初始状态 $InitStates_P$ 和动作集合 $ActionsSet_P$ 两个部分。其中,$InitStates_P$ 主要用于建模协议初始运行时协议合法主体的知识、状态等;$ActionsSet_P$ 主要用于建模协议步。

1) 构建初始状态集合 $InitStates_P$

考虑两个 Needham - Schroeder 公钥协议会话并行运行时的情况,可以令:

$InitStates_P = \{s_0 \mid s_0 = \{Await(A, A, \{A, B, pubK_A, privK_A, pubK_B\}, 0, 1), Await(A, B, \{B, A, pubK_B, privK_B, pubK_A\}, 1, 1), Await(A, A, \{A, C, pubK_A, privK_A, pubK_C\}, 0, 2), Await(A, C, \{C, A, pubK_C, privK_C, pubK_A\}, 1, 2), Inverse(pubK_A, privK_A), Inverse(pubK_B, privK_B), Inverse(pubK_C, privK_C)\}$。

其中,事实 $Await(A, A, \{A, B, pubK_A, privK_A, pubK_B\}, 0, 1)$ 与 $Await(A, A, \{A, C, pubK_A, privK_A, pubK_C\}, 0, 2)$ 分别表示主体 A 在协议会话 1 和协议会话 2 中扮演协议发起者的角色,其初始知识中包括其自身的身份标识、对方的身份标识、其自身的公钥和私钥、以及对方的公钥等;事实 $Await(A, B, \{B, A, pubK_B, privK_B, pubK_A\}, 1, 1)$ 与 $Await(A, C, \{C, A, pubK_C, privK_C, pubK_A\}, 1, 2)$ 则分别表示主体 B 和主体 C 在两个协议会话中均扮演协议响应者的角色;谓词 $Inverse$ 则用于声明协议中使用的公私钥对。

2) 构建动作集合 $ActionsSet_P$

协议模型中的动作采用一阶 ρ 演算规则进行描述。在执行的过程中,为了增强 ρ 演算规则的描述能力,特在其中引入了 Spi 演算[14] 中所的语法结构 (vn),用于描述随机数的生成,并保留了原始的形式语义。协议规则具体如下所述。

(1) 对于协议中的第一步:

$Rule_1 = \{Await(i, i, \{i, r, pubK_i, privK_i, pubK_r\}, 0, sid)\} \rightarrow (vn_i)\{Await(r, i, \{i, r, n_i, pubK_i, privK_i, pubK_r\}, 2, sid), Send(i, r, \{[i, n_i]\}_{pubKr}, 1), Begin(i, r, auth\_n_i, n_i)\}$

表示协议发起者 i 向协议响应者 r 发送消息 $\{[i, n_i]\}_{pubKr}$。其中,n_i 为主体 i 产生的新鲜数,$Begin(i, r, auth\_n_i, n_i)$ 表示主体 i 发起了一次与主体 r 的协议会话。

$Rule_2 = \{Await(i, r, \{r, i, pubK_r, privK_r, pubK_i\}, 1, sid), Send(i, r, \{[i, n_i]\}_{pubKr}, 1)\} \rightarrow \{Await(i, r, \{r, i, pubK_r, privK_r, pubK_i\}, 1, sid), Receive(r, i,$

$\{[i,n_i]\}_{pubKr},1)\}$

表示协议响应者 r 接收来自发起者 i 的消息 $\{[i,n_i]\}_{pubKr}$。

(2) 对于协议中的第二步:

$Rule_3 = \{Await(i,r,\{r,i,pubK_r,privK_r,pubK_i\},1,sid),Receive(r,i,\{[i,n_i]\}_{pubKr},1),Inverse(pubK_r,privK_r)\} \rightarrow (vn_r)\{Await(i,r,\{r,i,n_r,n_i,pubK_r,privK_r,pubK_i\},3,sid),Send(r,i,\{[n_i,n_r]\}_{pubKi},2),Begin(r,i,auth\_n_r,n_r),Inverse(pubK_r,privK_r)\}$

表示主体 r 在接收到来自主体 i 的消息后,解密消息并获取其中的随机数 n_i,然后生成一个随机数 n_r,并将 n_r 和 n_i 一起以消息 $\{[n_i,n_r]\}_{pubKi}$ 的形式发送给主体 i。

$Rule_4 = \{Await(r,i,\{i,r,n_i,pubK_i,privK_i,pubK_r\},2,sid),Send(r,i,\{[n_i,n_r]\}_{pubKi},2)\} \rightarrow \{Await(r,i,\{i,r,n_i,pubK_i,privK_i,pubK_r\},2,sid),Receive(i,r,\{[n_i,n_r]\}_{pubKi},2)\}$

表示主体 i 接收来自主体 r 的消息 $\{[n_i,n_r]\}_{pubKi}$。

(3) 对于协议中的第三步,即 *Step*3:

$Rule_5 = \{Await(r,i,\{i,r,n_i,pubK_i,privK_i,pubK_r\},2,sid),Receive(i,r,\{[n_i,n_r]\}_{pubKi},2),Inverse(pubK_i,privK_i)\} \rightarrow \{Await(r,i,\{i,r,n_r,n_i,pubK_i,privK_i,pubK_r\},4,sid),Send(i,r,\{n_r\}_{pubKr},3),End(r,i,auth\_n_r,n_r),Inverse(pubK_i,privK_i)\}$

表示主体 i 在接收到来自 r 的消息 $\{[n_i,n_r]\}_{pubKi}$ 后,解密消息,完成对内容的检验,并获得随机数 n_r,然后将 n_r 以加密消息 $\{n_r\}_{pubKr}$ 的形式重新返送给主体 r,结束此次协议会话。

$Rule_6 = \{Await(i,r,\{r,i,n_r,n_i,pubK_r,privK_r,pubK_i\},3,sid),Send(i,r,\{n_r\}_{pubKr},3)\} \rightarrow \{Await(i,r,\{r,i,n_r,n_i,pubK_r,privK_r,pubK_i\},3,sid),Receive(r,i,\{n_r\}_{pubKr},3)\}$

表示主体 r 接收来自主体 i 的消息 $\{n_r\}_{pubKr}$。

$Rule_7 = \{Await(i,r,\{r,i,n_r,n_i,pubK_r,privK_r,pubK_i\},3,sid),Receive(r,i,\{n_r\}_{pubKr},3),Inverse(pubK_r,privK_r)\} \rightarrow \{Await(i,r,\{r,i,n_r,n_i,pubK_r,privK_r,pubK_i\},5,sid),End(i,r,auth\_n_i,n_i),Inverse(pubK_r,privK_r)\}$

表示主体 r 在接收到来自主体 i 的消息 $\{n_r\}_{pubKr}$ 后,解密该消息,并完成对消息内容的检查,结束协议会话。

综上所述,协议模型中的动作集合 $ActionsSet = \{Rule_1,Rule_2,Rule_3,Rule_4,Rule_5,Rule_6,Rule_7\}$。

1.4　攻击者描述

对协议攻击者的描述,仍然沿用 Needham – Schroeder 公钥协议的例子。

1) 构建攻击者初始状态集合 $InitStates_I$

攻击者在任一时刻的状态可以通过攻击者在该时刻所拥有的知识进行描述。对公钥协议进行分析时,通常假设在协议初始运行时,攻击者所拥有的知识中包括攻击者自身的身份标识和公私钥对、合法用户的身份及公钥等。

因此,可以令攻击者的初始状态集合:

$InitStates_I = \{s_0' \mid s_0' = \{Intrude(I),Intrude(pubK_I),Intrude(privK_I),Intrude(A),Intrude(B),Intrude(C),Intrude(pubK_A),Intrude(pubK_B),Intrude(pubK_C)\}\}$

其中,I 为攻击者的身份标识;$pubK_I$ 为攻击者的公开密钥;$privK_I$ 为攻击者的私有密钥。

2) 构建攻击者动作集合 $ActionsSet_I$

网络中针对密码协议的攻击者通常具备一定的计算能力,包括截获消息、伪造消息、转发消息、重放消息等。具体描述如下。

$Rule_1' = \{Send(a,b,m,stepid)\} \rightarrow \{Send(a,b,m,stepid),Intrude(m)\}$

表示攻击者可以转发网络中的消息。

$Rule_2' = \{Send(a,b,m,stepid)\} \rightarrow \{Intrude(m)\}$

表示攻击者可以截获网络中的消息。

$Rule_3' = \{Intrude(m),Intrude(k)\} \rightarrow \{Intrude(m),Intrude(k),Intrude(\{m\}_k)\}$

表示攻击者在已知消息 m 和密钥 k 的情况下,可以生成加密消息 $\{m\}_k$。

$Rule_4' = \{Intrude(m_1),Intrude(m_2),\cdots,Intrude(m_n)\} \rightarrow \{Intrude([m_1,m_2,\cdots,m_n]),Intrude(m_1),Intrude(m_2),\cdots,Intrude(m_n)\}$

表示攻击者在已知消息 m_1、m_2、…、m_n 的情况下,可以连接生成消息 $[m_1,m_2,\cdots,m_n]$。

$Rule_5' = \{Intrude(\{m\}_k),Intrude(k)\} \rightarrow \{Intrude(m),Intrude(\{m\}_k),Intrude(k)\}$

表示攻击者在已知消息 $\{m\}_k$ 和密钥 k 的情况下,可以解密获得消息 m,此处的加密方式为对称加密。

$Rule_6' = \{Intrude(\{m\}_k),Intrude(k'),Inverse(k,k')\} \rightarrow \{Intrude(m),Intrude(\{m\}_k),Intrude(k'),Inverse(k,k')\}$

表示密钥 k 与密钥 k' 互为加解密密钥,则攻击者在已知消息 $\{m\}_k$ 和密钥 k' 的情况下,可以解密获得消息 m,此处的加密方式为非对称加密。

$Rule_7' = \{Intrude([m_1,m_2,\cdots,m_n])\} \rightarrow \{Intrude([m_1,m_2,\cdots,m_n]),Intrude(m_1),Intrude(m_2),\cdots,Intrude(m_n)\}$

表示攻击者在已知消息 $[m_1,m_2,\cdots,m_n]$ 的情况下,可以对其拆分,获得消息 m_1、m_2、…、m_n。

$Rule_8' = \{Intrude(m),Intrude(a),Intrude(b)\} \rightarrow \{Send(a,b,m,stepid),Intrude(m),Intrude(a),Intrude(b)\}$

表示攻击者在已知主体 a、主体 b 以及消息 m 的情况下,可以假冒 a 向 b 发送消息 m。

综上所述,模型中关于攻击者的动作集合

$ActionsSet_I = \{Rule_1',Rule_2',Rule_3',Rule_4',Rule_5',Rule_6',Rule_7',Rule_8'\}$

1.5　模型目标描述

模型的目标集合 *Goals* 主要用于描述协议运行过程中可能出现的违背协议预期安全性的状态。在本文提出的模型中,主要定义了两类违反协议安全性的目标状态。

1) 违反协议秘密性的目标状态

$SecrecyViolation = \{\cdots,Intrude(Term),Not\ Shared(Term,I),\cdots\}$

该状态表示攻击者获得了其不应该获得的秘密消息项 *Term*,因此破坏了协议的秘密性,为一个不安全的状态。

2) 违反协议认证性的目标状态

模型中关于认证性的概念借鉴了 Gavin Lowe 的思想[11]。如果密码协议在运行时到达某一状态,该包含有事实 $End(a,b,authid(\_,Term)^*)$,但却不包含与之相对应的事实 $Begin(a,b,authid(\_,Term)^*)$,则该状态违反了协议的认证性定义。

针对 Needham-Schroeder 公钥协议,可以定义两种不安全的目标状态:

$$AuthenticityViolation_1 = \{\cdots, End(A,B,auth\_N_A,N_A), Not\ \ Begin(A,B,auth\_N_A,N_A), \cdots\}$$

表示主体 A 对主体 B 的认证;

$$AuthenticityViolation_2 = \{\cdots, End(B,A,auth\_N_B,N_B), Not\ \ Begin(B,A,auth\_N_B,N_B), \cdots\}$$

表示主体 A 对主体 B 的认证。

2 模型运算语义

上述模型在求解过程中主要涉及以下两类运算,即规则应用和规则合一化。

2.1 规则应用

定义 3 规则应用:对于一阶 ρ 演算规则 $Rule = PreCond \rightarrow PostCond$ 和状态 $State$,如果存在置换算子 Θ,使得 $\Theta PreCond \subseteq State$,即对 $\forall fact \in PreCond$,总有 $\Theta fact \in State$,则称 ρ 演算规则 $Rule$ 可以应用于状态 $State$。

若 ρ 演算规则 $Rule$ 可以应用于状态 $State$,则记这一运算过程为 $[Rule]_\Theta(State)$。当不强调 Θ 时,也可以简记为 $[Rule](State)$。其中,Θ 为定义 1 中提到的置换算子。将 $Rule$ 应用于 $State$ 后所生成的新的状态记为 $State'$,则有:

$$\begin{aligned} State' &= [Rule]_\Theta(State) \\ &= (State \backslash \Theta PreCond) \cup \Theta PostCond \\ &= \{f \mid f \in State \wedge f \notin \Theta PreCond \vee f \in \Theta PostCond\} \end{aligned}$$

例如,对于上文所提到的 Needham – Schroeder 公钥协议的初始状态 s_0 以及协议动作集中的 ρ 演算规则 $Rule_1$,存在置换算子 $\Theta = \{A/i,\ B/r,\ pubK_A/pubK_i,\ privK_A/privK_i,\ pubK_B/privK_r,\ 1/sid,\ N_A/n_i,\ auth\_N_A/auth\_n_i\}$,使得 $pre(\Theta Rule_1) \subseteq s_0$,$pre(\Theta Rule_1)$ 表示获取 $\Theta Rule_1$ 的规则前件,故 ρ 演算规则 $Rule_1$ 可以应用于初始状态 s_0。如果记生成的新状态为 s_1,则有:

$$\begin{aligned} s_1 &= [Rule_1]_\Theta(s_0) \\ &= \{ Await(B,A,\{A,B,N_A,pubK_A,privK_A,pubK_B\},2,1), Send(A,B,\{[A,N_A]\}_{pubKB},1), Begin(A,B,auth\_N_A,N_A), Await(A,B,\{B,A,pubK_B,privK_B,pubK_A\},1,1), Await(A,A,\{A,C,pubK_A,privK_A,pubK_C\},0,2), Await(A,C,\{C,A,pubK_C,privK_C,pubK_A\},1,2), Inverse(pubK_A,privK_A), Inverse(pubK_B,privK_B), Inverse(pubK_C,privK_C)\} \end{aligned}$$

另外,值得说明的是此处使用的置换算子并不唯一,$\Theta' = \{A/i, C/r, pubK_A/pubK_i,\ privK_A/privK_i, pubK_C/privK_r, 2/sid, N_A/n_i, auth\_N_A/auth\_n_i\}$ 也可以满足条件。

定义 4 可达性:对于模型 $\Pi = < InitStates, ActionsSet, Goals >$ 状态空间中的某一状态 $s \in S$,如果存在动作序列 $R_1, R_2, \cdots, R_n \in ActionsSet$ 以及置换算子 $\Theta_1, \Theta_2, \cdots, \Theta_n, n > 0$,使得 $s = [R_n]_{\Theta n}(\cdots([R_2]_{\Theta 2}([R_1]_{\Theta 1}(s_0)))\cdots), s_0 \in InitStates$,则称状态 s 在模型 Π 中是可达的,规则序列 $R_1, R_2, \cdots, R_n$ 则称为由 s_0 至 s 的路径。

定义 4′ n 步可达:若定义 3 中的整数 n 是符合条件的最小整数,则称状态 s 在模型 Π 中是 n 步可达的。

定义 5 有解性:对于模型 $\Pi = < InitStates, ActionsSet, Goals >$,若存在状态 $g \in Goals$,且 g 在模型 Π 中是可达的,则称模型 Π 有解,相应的路径称为模型的解。

定义 6 解空间:由模型 Π 的所有的解构成的集合称为模型 Π 的解空间,记为 $Solutions$。

若模型 Π 有解,则与模型相对应的密码协议存在着攻击,是有缺陷或不安全的,即协议不满足其预期的安全性需求。

2.2 规则合一

规则合一是模型推理过程中所使用的另外一个重要的运算形式,可以大大减化模型推理过程的复杂性。在引出规则合一之前,首先对规则增量的概念做以简要介绍。

定义 7 规则增量:对于模型 Π 的动作集合 $ActionsSet$ 中的 ρ 演算规则 $Rule = PreCond \rightarrow PostCond$ 以及事实集合 $F' \subseteq F$,如果对于任意的置换算子 Θ,均有 $\Theta PreCond \cup F' \neq \Omega$,且 $\Theta PostCond \cup F' \neq \Omega$,则规则 $Rule' = PreCond \cup F' \rightarrow PostCond \cup F'$ 是有意义的,称规则 $Rule'$ 为规则 $Rule$ 的增量规则,而 F' 则称为规则 $Rule'$ 相对于规则 $Rule$ 的增量。

如果规则 $Rule'$ 为规则 $Rule$ 的增量规则,而 F' 为规则 $Rule'$ 相对于规则 $Rule$ 的增量,亦可记 $Rule' = Rule + F'$。

命题 1 自反性:每条规则均是其自身的增量规则。

当规则增量 F' 为空集 $\varnothing$ 时,易证命题 1 成立。

命题 2 传递性:若规则 $Rule_1$ 是规则 $Rule_2$ 的增量规则,规则 $Rule_2$ 是规则 $Rule_3$ 的增量规则,则规则 $Rule_1$ 是规则 $Rule_3$ 的增量规则。

命题 3 若规则 $Rule_1$ 是规则 $Rule_2$ 的增量规则,且规则 $Rule_2$ 可以应用于某一状态 $s \in S$,则规则 $Rule_1$ 也可以应用于状态 s,并且 $[Rule_1](s) = [Rule_2](s)$。

依据定义 7,易证命题 2、3 成立。

定义 8 规则合一:对于规则 $Rule_1 = PreCond_1 \rightarrow PostCond_1$ 和规则 $Rule_2 = PreCond_2 \rightarrow PostCond_2$,如果 $Rule_1$ 存在增量规则 $Rule_1 + F'$,使得 $PreCond_1 \cup F'$ 与 $PreCond_2$ 可以合一化,且 $Unify(PostCond_1 \cup F', PreCond_2) = \Theta$,则称 $Rule_1$ 与 $Rule_2$ 可以进行规则合一运算,运算后生成新规则 $\Theta PreCond_1 \cup \Theta F' \rightarrow \Theta PostCond_2$,记为 $Rule_1 \circ Rule_2$。

若原始的两条规则是有意义的,则规则合一后生成的新规则仍然是有意义的。

例如,在对 Needham – Schroeder 公钥协议进行描述的 ρ 演算规则中,$Rule_2$ 具有增量规则 $Rule_{2'}$。其中:

$$\begin{aligned} Rule_2' = \{&Await(i,r,\{r,i,pubK_r,privK_r,pubK_i\},1,sid), Send(i,r,\{[i,n_i]\}_{pubKr},1), Inverse(pubK_r,privK_r)\} \rightarrow \{Await(i,r,\{r,i,pubK_r,privK_r,pubK_i\},1,sid), Receive(r,i,\{[i,n_i]\}_{pubKr},1), Inverse(pubK_r,privK_r)\} \end{aligned}$$

可以看出,$Rule_{2'}$ 的规则后件与 $Rule_3$ 的规则前件相同,即 $Unify(post(Rule_2'), pre(Rule_3)) = 1$。其中,$post(Rule_2')$ 表示 $Rule_2'$ 的规则后件;$pre(Rule_3)$ 表示 $Rule_3$ 的规则前件;1 表示恒等置换。故规则 $Rule_2$ 与规则 $Rule_3$ 可以进行合一化运算,结果为:

$$Rule_{2,3} = Rule_2 \circ Rule_3 = \{Await(i,r,\{r,i,pubK_r,privK_r,$$

$pubK_i\}$,1,sid),$Send(i,r,\{[i,n_i]\}_{pubKr}$,1),
$Inverse(pubK_r,privK_r)\} \rightarrow (vn_r)\{Await(i,r,$
$\{r,i,n_r,n_i,pubK_r,privK_r,pubK_i\}$,3,$sid$),
$Send(r,i,\{[n_i,n_r]\}_{pubKi}$,2),$Begin(r,i,$
$auth\_n_r,n_r)$,$Inverse(pubK_r,privK_r)\}$

另外,规则 $Rule_4$ 与规则 $Rule_5$ 可以进行合一化,规则 $Rule_6$ 与规则 $Rule_7$ 也可以进行合一化。

定义一个新的动作集合,然后按照以下步骤对其进行扩充:

1)从集合 $ActionsSet$ 中任取规则 $Rule$,若 $R \notin ActionsSet^E$,则将规则 $Rule$ 添加到集合 $ActionsSet^E$ 中,然后执行 2);否则,重复执行 1)。

2)对于集合 $ActionsSet^E$ 中任意规则 $Rule'$,若规则 $Rule'$ 与规则 $Rule$ 可以进行合一化,则将合一化后生成的新规则添加到 $ActionsSet^E$ 中,然后返回 1)继续执行。

3)当系统到达一个不动点时,终止上述过程。

构建新的问题模型 $\Pi^E = < InitStates, ActionsSet^E, Goals >$,其中的 $InitStates$ 和 $Goals$ 与原有模型 $\Pi = < InitStates, ActionsSet, Goals >$ 中的定义相同;而 $ActionsSet^E$ 为上述扩充过程完成后得到的动作集合。称 Π^E 为 Π 的扩展模型。

定义 9　解等价:令 Φ 和 Φ' 均为基于问题求解理论的密码协议模型,如果模型 Φ 有解,当且仅当模型 Φ' 有解,且两者的解之间存在着某种偏序对应关系,则称模型 Φ 和模型 Φ' 是解等价的。

引理 1　令规则 $Rule$ 与 $Rule'$ 可以进行合一化,且合一化后生成的规则为 $Rule \circ Rule'$,则若规则 $Rule$ 可以应用于状态 $s \in S$,当且仅当 $Rule \circ Rule'$ 可以应用于状态 s,并且有 $[Rule']([Rule](s)) = [Rule \circ Rule'](s)$。

证明:令 $Rule = PreCond_1 \rightarrow PostCond_1$,$Rule' = PreCond_2 \rightarrow PostCond_2$,$Rule \circ Rule' = \Theta PreCond_1 \cup \Theta F' \rightarrow \Theta PostCond_2$,其中,$F'$ 为规则 $Rule$ 的规则增量,Θ 为规则 $Rule$ 和规则 $Rule'$ 进行合一化时使用的合一化算子。

充分性:若规则 $Rule \circ Rule'$ 可以应用于状态 s,则依据定义 3,可知存在一个置换算子 θ,使得 $\theta(\Theta PreCond_1 \cup \Theta F') \in s$,从而得出 $\theta\Theta PreCond_1 \in s$,故规则 $Rule$ 可以应用于状态 s。

必要性:若规则 $Rule$ 可以应用于状态 s,则依据命题 3,可知规则 $Rule + F'$ 亦可以应用于状态 s,则存在置换算子 θ',使得 $\theta'(PreCond_1 \cup F') \in s$。令 $\theta'' = \theta' \cdot \Theta^{-1}$,则 $\theta''(\Theta PreCond_1 \cup \Theta F') = \theta'\Theta^{-1}\Theta(PreCond_1 \cup F') = \theta'(PreCond_1 \cup F') \in s$,故规则 $Rule \circ Rule'$ 可以应用于状态 s。

同时,由于:

$$
\begin{aligned}
&[Rule']([Rule](s)) = [Rule']([Rule + F'](s)) \\
&\quad = [Rule']((State \backslash \theta'(PreCond_1 \cup F')) \cup \\
&\qquad \theta'(PostCond_1 \cup F')) \\
&\quad = (((State \backslash \theta'(PreCond_1 \cup F')) \cup \theta'(PreCond_1 \cup \\
&\qquad F')) \backslash \theta' PreCond_2) \cup \theta' PreCond_2 \\
&\quad = (((State \backslash \theta'\Theta^{-1}\Theta(PreCond_1 \cup F')) \cup \\
&\qquad \theta'\Theta^{-1}\Theta(PostCond_1 \cup F')) \backslash \theta'\Theta^{-1}\Theta PreCond_2) \cup \\
&\qquad \theta'\Theta^{-1}\Theta PostCond_2) \\
&\quad = (((State \backslash \theta'\Theta^{-1}\Theta(PreCond_1 \cup F')) \cup \\
&\qquad \theta'\Theta^{-1}\Theta PostCond_2) \\
&\quad = [Rule \circ Rule'](s)
\end{aligned}
$$

故引理 1 得证。

命题 4　协议模型 Π 与其扩展模型 Π^E 是解等价的。

依据定义 5、定义 9 及引理 1,易证命题 3 成立。此处就不再对证明过程加以赘述。

3　结语

本文基于问题求解理论提出了一种密码协议模型,并结合 ρ 演算给出了模型的形式语义。在下一步的研究工作中,将重点研究模型的自动化推理算法,并对其中可能发生的不可终止现象进行处理,开发基于该模型的自动化验证软件系统。

参考文献:

[1] BURROWS M, ABADI M, NEEDHAM RM. A Logic of Cryptographic[J]. ACM Transactions on Computer Systems, 1990, 8(1): 18 - 36.

[2] RUSSELL S, NORVIG P. Artificial Intelligence (A Modern Approach (Second Edition)[M]. Pearson Education Prentice Hall, 2003. 49 - 146.

[3] AIELLOL C, MASSACCI F. Verifying Security Protocols as Planning in Logic Programming[J]. ACM Transactions on Computational Logic, 2001, 2(4): 542 - 580.

[4] BLANCHET B. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules[A]. In 14th IEEE Computer Security Foundations Workshop(CSFW-14)[C]. IEEE Computer Society, Los Alamitos, CA, 2001. 82 - 96.

[5] SONG D. Athena: a New Efficient Automatic Checker for Security Protocol Analysis[A]. In Proceedings of the 12th IEEE Computer Security Foundations Workshop (CSFW'99)[C]. June 1999. 192 - 202.

[6] CLARKE EM, JHA S, MARRERO W. Verifying Security Protocols with Brutus[J]. ACM Transactions on Software Engineering and Methodology (TOSEM), 2000, 9(4): 443 - 487.

[7] MEADOWS C. The NRL Protocol Analyzer: an Overview[J]. Journal of Logic Programming, 1996, 26(2): 113 - 131.

[8] LOWE G. Casper: A Compiler for the Analysis of Security Protocols [A]. In Proceedings of the 1997 IEEE Computer Society Symposium on Research in Security and Privacy[C]. 1997. 18 - 30.

[9] STOLLER SD. A Reduction for Automated Verification of Authentication Protocols[R]. Technical Report 520, Computer Science Department, Indiana University, 1998.

[10] AlessandroArmando and Luca Compagna . SATMC : a SAT - based Model Checker for security protocols[A]. In Proceedings of the 9th European Conference on Logics in Artificial Intelligence (JELIA 2004)[C]. LNAI 3229, Lisbon, Portugal, Springer-Verlag. September 2004.

[11] LOWE G. A Hierarchy of Authentication Specifications[A]. In Proceedings of the 10th Computer Security Foundations Workshop (CSFW'97)[C]. Rockport, Massachusetts, June 1997.

[12] 季庆光,冯登国. 对几类重要网络安全协议形式模型的分析[J]. 计算机学报, 2005, 28(7): 1071 - 1083.

[13] DOLEV D, YAO AC. On the Security of Public-Key Protocols[J]. IEEE Transactions on Information Theory, 1983, 2(29): 198 - 208.

[14] ABADI M, GORDON A. A Calculus for Cryptographic Protocols: The Spi Calculus [R]. Technical Report 149, SRC, Palo Alto, California, 1998.

[15] CIRSTEA H, KIRCHNER C. The Rewriting Calculus[J]. L. J. of the IGPL, Oxford University Press, 2001, 9(3): 339 - 375.