基于经典逻辑的安全协议模型检测方法*)

徐 畅 刘吉锋 孙吉贵
(吉林大学计算机科学与技术学院 长春 130012)
(吉林大学符号计算与知识工程教育部重点实验室 长春 130012)

摘 要 本文分别以安全协议模型检测器 SATMC 和 ProVerif 为例,介绍了基于经典逻辑的安全协议模型检测两种方法:SAT 方法和归结方法,并简要地给出了我们设计实现的基于 SAT 方法的安全协议模型检测器 JLU-PV。

关键词 安全协议,模型检测,SAT 求解,归结

Security Protocol Model Checking Based on Classcial Logic

XU Chang LIU Ji-feng SUN Ji-gui
(College of Computer Science and Technology, Jilin University, Changchun 130012, China)
(Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Changchun 130012, China)

Abstract The security protocol model-checking based on classcial logic is introduced in this paper, using the model-checker SATMC and the model-checker ProVerif as examples, and a security protocol model-checker JLU-PV exploited by us based on SAT is also introduced here.

Keywords Security protocol, Model-checking, Propositional satisfiability, Resolution

1 引言

安全协议是建立在密码体制基础上、运行在网络或分布式系统中、借助于密码算法为有安全需求的各方提供一系列步骤以达到密钥分配、身份认证、信息保密等目的的协议。安全协议是信息技术在日常生活中广泛应用的基础和保障之一,尤其在国防、金融和政府机关等安全敏感部门中,对安全协议的要求更加严格。但是安全协议的设计却是困难的、易出错的,原因在于对安全协议设计进行人工检查和测试无法保证验证的完备性,例如著名的 Needham-Schroeder 认证协议在发布 17 年之后才被发现存在一个针对它的严重攻击[1]。因此,研发出自动化的方法和工具来验证安全协议是否可以维护某种安全属性已经成为当前计算机科学与技术研究所关注的热点问题之一。

模型检测是当前最重要的验证技术之一,在 20 世纪 90 年代分别由 Edmund M. Clark 和 E. Allen Emerson, Jeen-Pierre Queille 和 Joseph Sifakis 提出[2,3]。模型检测将被验证系统建模为 $\Xi=(F,A,I,O)$,其中 F 是系统状态的有限集合;I 是系统初始状态的集合;$O\subseteq F\times F$ 是状态转换关系的集合;$A:F\to 2^{AP}$ 是一个标记函数,AP 是表示某些性质的原子命题的集合,A 标明每个状态上哪些原子命题成立。基于安全协议模型检测的过程是,从 I 出发根据 O 生成 Ξ 所有的状态转换路径,如果这些状态转换路径满足特定要求,就表明被验证系统满足被验证的属性,验证过程输出证明并终止,给出特定的攻击;否则,表明被验证系统不满足被验证的属性,表明协议不存在攻击。

当前国际上突出的基于经典逻辑的安全协议模型检测方法主要有两种,分别为 SAT 方法和归结方法。Alessandro Armando, Luca Compagna 和 Roberto Carbon 开发的安全协议模型检测器 SATMC[4-9] 是基于 SAT 方法的安全协议模型检测的代表;Bruno Blanchet 开发的安全协议模型检测器 ProVerif[10] 是基于归结方法的安全协议模型检测的代表。

本文分别以 SATMC 和 ProVerif 为例来介绍基于 SAT 方法和归结方法的安全协议模型检测。针对 SATMC 的效率,我们简要地给出了一个改进的新算法和设计实现的安全协议模型检测器 JLU-PV。

2 基于 SAT 方法的安全协议的模型检测器 SATMC

SATMC 使用 Dolev-Yao 模型[11] 对入侵者建模,对于给定的界限 $K>0$ 将协议安全问题建模为可到达性问题 $\Xi=\langle\langle F,A,I,O\rangle,B\rangle$,使用便于处理的语言 IF 来表示,并且把可到达问题转化成为规划问题,不断利用规划图构建命题公式,每构建一个命题层 k 之后就构建对应的命题公式 $[\![\Xi]\!]_k$ $(0\leqslant k\leqslant K)$。构建命题公式的过程是:规划图的命题层向前扩展一层,编码根据规划图向前扩展一层[12,13],然后调用 SAT 求解器判断命题公式是否可满足。如果命题公式可满足,那么根据它的模型可以获得攻击对应的状态转换路径,此时 SATMC 再现攻击并终止;否则如果这个命题公式不可满足,则 SATMC 构建包含更多步状态转换的命题公式继续验证,直到找到攻击或达到人为设定的界限为止。

SATMC 用于求解有限非循环会话的协议安全问题(protocol insecurity problems with a bounded number of acyclic sessions,简称为 BNAS 问题),所以 SATMC 的过程一定可以终止。

2.1 协议安全问题转化为规划问题

用符号 $\widetilde{\Xi}$ 表示协议安全问题 Ξ 对应的规划问题,$\widetilde{M}$ 表示

*)国家自然科学基金(60473003)、教育部"新世纪优秀人才支持计划"、博士点基金(20050183065)资助课题。徐 畅 硕士研究生,主要研究方向是模型检测;刘吉锋 博士研究生,主要研究方向是模型检测;孙吉贵 博士生导师,主要研究方向是人工智能、约束程序、决策支持系统。

规划系统，$\tilde{F},\tilde{A},\tilde{I},\tilde{R}$ 分别表示规划系统中的状态集，动作集，初始状态集和规则集，$\tilde{R}$ 具体形式为

$$Pre \overset{Act}{\Rightarrow} Add;Del$$

其中 $Act\in\tilde{A},Pre\in\tilde{F},Add\in\tilde{F},Del\in\tilde{F}$，它们分别是操作 Act 的前件、添加效果和删除效果。$\tilde{B}$ 表示规划问题的目标状态，则：

$$\tilde{\Xi}=\langle\tilde{M},\tilde{B}\rangle=\langle\langle\tilde{F},\tilde{A},\tilde{I},\tilde{R}\rangle,\tilde{B}\rangle$$

规划问题和协议安全问题的转化对应关系为：

(1)$\tilde{F}=F$；

(2)$\tilde{A}=A$；

(3)$\tilde{R}=\{(L\sigma \overset{b}{\Rightarrow} R\sigma;L\sigma\backslash R\sigma):(L \xrightarrow{l} R)\in O,\sigma$ 为基替换$\}$；

(4)$\tilde{I}=I$；

(5)$\tilde{B}=B$。

即协议安全问题转化为规划问题时协议安全问题的状态集、动作集、初始状态集和目标状态集分别转化为规划问题的状态集、动作集初始状态集和目标状态集。协议安全问题的规则集转化为规划问题的规则集时要对协议安全问题操作集中的变量加以实例化，然后得到规划问题的规则集，且 SATMC 当前只允许一个初始状态。

2.2 规划问题转化为 SAT 方法求解

2.2.1 规划图

一个规划问题 $\tilde{\Xi}=\langle\tilde{M},\tilde{B}\rangle$ 的 k-规划图[14]是一个由 $k+1$ 个命题层和 k 个动作层组成、命题层和动作曾交叉出现的有向无环图。N_f^i 是命题层 i 中命题的集合，N_a^i 是动作层 i 中动作的集合，规划图的第一个层是命题层 0，N_f^0 中包括了表示初始状态集合的命题的集合。N_a^i 中的所有动作的前提条件都包含在 N_f^i 中，且一个动作的所有前提条件之间不存在任何互斥关系，N_a^i 中动作的效果构成了命题层 $i+1$。

互斥关系的定义如下：

1. 动作层 i 中的动作 a_1 与 a_2 互斥，如果满足下列条件之一：

(C1)a_1 删除 a_2 的添加效果或 a_1 删除 a_2 的前提条件(静态互斥)；

(C2)分别存在 a_1 和 a_2 的前提条件 f_1 和 f_2，f_1 和 f_2 在命题层 i 中互斥；

2. 命题层 i 中的命题 f_1 与 f_2 互斥，如果满足下列条件之一：

(C3)$f_1,f_2\in N_f^0$ 并且 f_1 与 f_2 分别属于不同的初始状态；

(C4)动作层 $i-1$ 中任意两个分别添加 f_1 和 f_2 的动作 a_1 与 a_2 都互斥。

2.2.2 利用规划图构建命题公式

问题求解需要构建的命题公式表示为：

$$[\![\Xi]\!]_k=I(f^0)\wedge\bigwedge_{i=0}^{k-1}T_i(f^i,a^i,f^{i+1})\wedge B(f^k)$$

f 和 a 是命题变量，分别用于表示规划问题的状态和状态转换关系；$T_i(f^i,a^i,f^{i+1})$ 是表示在第 i 步状态转换中所有可应用的状态转换关系的命题公式，其中表示状态转换关系的命题变量带有时间步标识 i，表示状态转换前提的命题变量带有时间步标识 i，表示状态转换结果的命题变量带有时间步标识 $i+1$；B_k 是代表攻击的目标状态，符号 $I(f^0)$ 是表示初始状态集合 I 的命题公式，其中的命题变量 f 带有时间步标识 0。

根据规划图构建 $T_i(f^i,a^i,f^{i+1})$ 的过程为：

1. 对于命题 a^i 有：

(1) a^i 为真蕴涵，则 f^i 为真；

(2) a^i 为真蕴涵，则添加效果 f^{i+1} 赋值为真；

(3) a^i 为真蕴涵，则删除效果 f^{i+1} 赋值为假；

(4) a^i 的添加效果如果不包含在当前命题层中，那么把它赋值为真后加入命题层 $i+1$ 中。

2. 对于命题 $p_1,p_2\in N_a^i\cup N_f^i$，如果 p_1 与 p_2 互斥，则 p_1^i 与 p_2^i 的合取范式 $p_1^i\wedge p_2^i$ 赋值为假。

由于 SATMC 当前只允许一个初始状态，根据互斥关系的定义，事实间的互斥关系最终都由互斥条件(C1)导致。而且，SATMC 中命题公式的编码方式可以直接实现由互斥条件(C1)导致的互斥，所以 SATMC 只考虑由互斥条件(C1)导致的互斥即静态互斥。默认情况下 SATMC 在构建规划图时不考虑事实间的互斥关系，判断一条重写规则是否可应用时只考虑它的前提条件是否可以全部与命题层中的事实合一，$[\![\Xi]\!]_k$ 中也只编码规则实例间的静态互斥关系。

即使 SATMC 只考虑静态互斥关系，由于处理静态互斥关系仍需要占用大量处理时间，所以 SATMC 允许使用一种抽象/细化策略[15-17]。这种策略的具体方法是构建 $[\![\Xi]\!]_k$ 时先不考虑静态互斥关系。如果 $[\![\Xi]\!]_k$ 是不可满足的，则即使在 $[\![\Xi]\!]_k$ 中编码了静态互斥关系，$[\![\Xi]\!]_k$ 也是不可满足的，所以验证过程继续构建规划图，构建 $[\![\Xi]\!]_{k+1}$ 并判定其可满足性；若 $[\![\Xi]\!]_k$ 是可满足的，则判定 $[\![\Xi]\!]_k$ 的可满足解所代表的规划图的子图中是否存在某些静态互斥关系。如果在这个子图中存在某些静态互斥关系，那么这些静态互斥关系都被违反，$[\![\Xi]\!]_k$ 的赋值不合理，需要将这些静态互斥关系补充到 $[\![\Xi]\!]_k$ 中，重新判定 $[\![\Xi]\!]_k$ 的可满足性；如果在这个子图中不存在任何静态互斥关系，则表明 $[\![\Xi]\!]_k$ 的赋值是合理的，根据 $[\![\Xi]\!]_k$ 的模型可以获得一个真实可行的攻击过程。使用抽象/细化策略可以大量节约 SATMC 的处理时间。

在命题公式 $[\![\Xi]\!]_k=I(f^0)\wedge\bigwedge_{i=0}^{k-1}T_i(f^i,a^i,f^{i+1})\wedge B(f^k)$ 被构建后，调用 SAT 求解器判定其可满足性。如果 $[\![\Xi]\!]_k$ 是可满足的，则表明某个目标状态可到达，$[\![\Xi]\!]_k$ 的可满足解就表现了达到目标状态的规划解，根据 $[\![\Xi]\!]_k$ 的模型可以建立 M 满足 B_k 的证明，即再现攻击；否则如果 $[\![\Xi]\!]_k$ 是不可满足的，则表明根据当前展开的状态转换无法找到攻击，此时验证过程将 k 值加一，构建新的 $[\![\Xi]\!]_k$ 并判定其可满足性。

2.3 SATMC 的可判定性和终止性

安全协议的会话数、会话的参与者数、会话运行中产生的消息数是无限的，并且在协议的会话中允许循环的出现，因此求解协议安全问题是不可判定的[18]。

如果增加一些限定条件，求解协议安全问题就可以转化为半可判定问题[19]。因为 SATMC 求解 BNAS 问题，求解此类问题是可判定的，所以 SATMC 的过程一定可以终止[20,21]。

SATMC 的终止状态分为如下三种情况：

(1) 如果找到攻击，那么验证过程再现攻击并终止；

(2) 如果已经遍历 S 中的所有状态也无法达到代表攻击可以实现的目标状态，则被验证协议是安全的，验证过程终止；

(3) 如果 k 值达到人为限定的最大值，而且此时无法达

到目标状态，则在限定范围内无法找到针对被验证协议的攻击，验证过程终止。

3 基于 SAT 方法的安全协议模型检测器 JLU-PV

我们吉林大学研究组提出一种新的基于 SAT 方法的安全协议模型检测算法，在此基础上，设计开发了安全协议模型检测器 JLU-PV[22]。实验结果表明 JLU-PV 可以验证所有 SATMC 可以验证的测试用例，并在验证速度上明显优于 SATMC，在绝大多数测试用例上比 SATMC 快两个数量级以上，而且在大部分测试用例上也领先于另外两个安全协议模型检测器 OFMC[23] 和 CL-AtSe[24]。

新算法改进了 SATMC 的方法，目的在于提高基于 SAT 方法的安全协议模型检测的速度。其思想是在构建规划图时重用尽可能多的信息，包括已经构建的命题层、动作层和找到的成功合一，且仅在必要时构建命题公式$[\![\Xi]\!]_k$，在保证验证能力的前提下尽可能地提高验证速度。

出现在命题层 i 中的事实 f_1 因为空动作的作用，一定会出现在命题层 $i+1$ 中，所以在构建命题层 $i+1$ 的时候可以重用命题层 i 的事实。

由于在 SATMC 中考虑一条重写规则是否可能应用的时候只考虑它的前提条件是否可以全部与命题层中的事实合一，而一旦一个事实出现在命题层中就会出现在所有后续命题层中，所以一个可能应用的规则实例如果出现在动作层 i 中，它也会出现在动作层 $i+1$ 中，因此可以在构建动作层 $i+1$ 的时候安全地重用动作层 i 的规则实例。

设符号 $Add^i(f_1)$ 是 N_a^{i-1} 中在命题层 i 中添加事实 f_1 的规则实例的集合，$Del^i(f_1)$是 N_a^{i-1} 中在命题层 i 中删除事实 f_1 的规则实例的集合，$Mutex_a^i(a_1)$ 是在动作层 i 中与规则实例 a_1 静态互斥的规则实例的集合。这些都是构建规划图和命题公式$[\![\Xi]\!]_k$所必须的信息。因为一个事实一旦出现在某一命题层中就会出现在后面所有的命题层中；一个规则实例一旦出现在某一动作层中就会出现在后面所有的动作层中，所以下列断言成立：

(1) $Add^{i+1}(f_1) \supseteq Add^i(f_1)$

(2) $Del^{i+1}(f_1) \supseteq Del^i(f_1)$

(3) $Mutex_a^{i+1}(a_1) \supseteq Mutex_a^i(a_1)$

即 $Add^i(f_1)$，$Del^i(f_1)$和 $Mutex_a^i(a_1)$可以分别在 $Add^{i+1}(f_1)$，$Del^{i+1}(f_1)$和 $Mutex_a^{i+1}(a_1)$中被重用。

对于一条重写规则 A，在 SATMC 构建动作层 i 的时候，需要判断 A 的前提条件可以与 N_f^i 中的哪些事实合一，如果 A 的所有前提条件都可以与 N_f^i 中的事实合一，则表明 A 可能应用于命题层 i，A 的规则实例可以被添加到动作层 i 中。如果独立地用 A 的每个前提条件与 N_f^i 中的每个事实尝试合一，那么一定可以用这样找到的合一组合出构建 A 的规则实例所需的替换。在构建动作层 $i+1$ 时，由于 N_f^{i+1} 中包含了 N_f^i 中的事实，而且此时 A 的前提条件已经与 N_f^i 中所有事实尝试了合一，所以 A 的每个前提条件只需再与 $N_f^{i+1} \backslash N_f^i$ 中的事实尝试合一，就可以找到 A 的前提条件与 N_f^{i+1} 中所有事实的合一，用这些合一就可以组合出构建在 N_a^{i+1} 中新增的规则实例所需的替换。

除此之外，我们的新算法只在必要的时候构建命题公式$[\![\Xi]\!]_k$。SATMC 每构建一个新的命题层 k 都要构建对应的$[\![\Xi]\!]_k$。事实上并不需要被如此频繁地构建$[\![\Xi]\!]_k$，因为在许多情况下对于命题层 k 来说所有的目标状态都是不可能到达的。这首先表现为在表示任意一个目标状态的事实都无法全部与命题层 k 中的事实合一时，即使构建了$[\![\Xi]\!]_k$，$[\![\Xi]\!]_k$ 也是不可满足的；如果表示一个目标状态的所有事实都可以与命题层 k 中的事实合一，那么这个目标状态在命题层 k 中才是可能到达的。新算法由于延续了在 SATMC 默认的构建规划图的方法中只考虑静态互斥关系这种方法，所以只能判断目标状态在命题层中是否可能到达。在新算法中，只有当目标状态在新构建的命题层 k 中可能到达时才构建对应的$[\![\Xi]\!]_k$，然后通过判定$[\![\Xi]\!]_k$ 的可满足性来判定目标状态是否真正可到达。这种方法极大地减少了处理命题公式的工作量，而且对于许多 BNAS 问题，由于目标状态在所有命题层中都不可能到达，所以在验证的过程中不需要消耗任何处理时间来构建命题公式$[\![\Xi]\!]_k$。

对于给定的 BNAS 问题 Ξ 和界限 $K>0$，新算法首先初始化 k 值并根据初始状态构建规划图的命题层 0，然后进入不断增加 k 值并以 k 值为界限的循环。

对于新的命题层 k，如果存在可能到达的目标状态，则验证过程构建命题公式$[\![\Xi]\!]_k$，并调用 SAT 求解器判定其可满足性。

如果$[\![\Xi]\!]_k$ 是可满足的，则验证过程根据$[\![\Xi]\!]_k$ 的可满足解来再现攻击并退出。

如果$[\![\Xi]\!]_k$ 不可满足或者在命题层 k 中不存在可能到达的目标状态，则开始判断规划图是否已经稳定，如果规划图已经稳定[25]，则验证过程显示找不到攻击并退出。如果规划图没有稳定，则验证过程应该继续构建规划图。k 值指示了需要构建的动作层的序号，当 k 不等于 0 时，在构建动作层 k 时可以重用 N_a^{k-1} 中的规则实例及相关的规则实例间的静态互斥信息，只需再补充进那些在动作层 k 中首次出现的规则实例及相关的规则实例间的静态互斥信息就可以获得完整的动作层 k。而且，N_f^k 中的事实和相关的添加、删除信息也被重用于构建不完整的命题层 $k+1$，只需再补充进在命题层 $k+1$ 中首次出现的事实和相关的添加、删除信息就可以获得完整的命题层 $k+1$。

本质上求解 BNAS 问题是 NP 完全的，虽然 JLU-PV 的算法使得在验证终止时重写规则前提条件尝试合一的次数大大降低，但由于算法中需要组合出构建可能应用的规则实例所需的替换，而且还要构建命题公式并调用 SAT 求解器判定其可满足性，所以 JLU-PV 的算法仍旧是 NP 完全的。

4 基于归结方法的安全协议模型检测

基于归结方法的模型检测以法国的 B. Blanchet 开发的模型检测器 ProVerif 为代表，ProVerif 也使用 Dolev-Yao 模型对入侵者建模，但使用 Horn 子句集描述协议的安全问题。然后从代表攻击的子句出发逆向进行归结，构建逻辑推导树。如果逻辑推导树的所有分支都可以终止于表示初始状态的子句，则表明从初始状态出发可以实现对被验证协议的攻击；否则就意味着从初始状态出发无法实现攻击，被验证协议是安全的。ProVerif 在验证协议安全问题时不限制并行会话的数量、允许协议的循环，但限制消息的数量，这使得 ProVerif 的验证过程是半可判定的，而且有时会返回虚假的攻击，需要使用者对攻击进行人工分析[26-28]。

4.1 协议的描述

模型检测器 ProVerif 描述协议时使用谓词 $attacker(M)$ 表示“入侵者获得消息 M”，描述协议使用的密码原语用两种

函数表示，构造函数和析构函数。构造函数用来构造项，析构对项进行分解。

下面是这两种函数的介绍：

(1) f 是一个 n 维的构造函数，那么 Horn 子句 $attacker(x_1) \wedge, \cdots, \wedge attacker(x_n) \rightarrow attacker(f(x_1, \cdots, x_n))$，表示入侵者在获得消息 $x_1, \cdots, x_n$ 的情况下，能够获得消息 $f(x_1, \cdots, x_n)$。$f(x_1, \cdots, x_n)$ 是入侵者自身通过构造函数 f 构造的消息。例如：构造函数 $pencrypt(M, Pk)$ 表示对消息 M 用公钥 Pk 进行加密。如果入侵者知道公钥，那么入侵者具有用公钥对消息进行加密的能力。如果入侵者知道消息 x 和密钥 y，它便可以对 x 用 y 进行加密。ProVerif 把它这种能力描述为

$$attacker(x) \wedge attacker(y) \rightarrow attacker(pencrypt(x, y))$$

(2) g 是一个析构函数，那么 Horn 子句 $attacker(M_1) \wedge, \cdots, \wedge attacker(M_n) \rightarrow attacker(M)$ 表示入侵者在获得消息 $M_1, \cdots, M_n$ 的情况下，能够通过析构函数对 $M_1, \cdots, M_n$ 进行析构，进而产生的消息 M。例如：析构函数 $pdecrypt(pencrypt(M, Pk), N)$ 表示以私钥 N 对用公钥 Pk 加密的明文 M 进行解密。当入侵者获得密钥和用密钥加密的密文，它便可以对密文进行解密，从而获得消息的明文。ProVerif 把这种能力描述为：

$$attacker(pencrypt(x, y)) \wedge attacker(y) \rightarrow attacker(x)$$

协议中消息的传输过程也是用 Horn 子句表示，例如：

(1) $B \rightarrow A : \{N_B\}_K$

(2) $A \rightarrow B : \{N_B - 1\}_K$

表示用户 B 对用户 A 发送了以密钥 K 对新鲜性因子 N_B 加密的密文 $\{N_B\}_K$，A 在收到密文 $\{N_B\}_K$ 后，对它进行解密，然后对 B 发送密文 $\{N_B - 1\}_K$。定义一个函数 $minusone(x) = x - 1$ 后，用符号 H 表示用户 A 以前收到的其他的消息，那么(2)式用 Horn 子句表示为：

$$H \wedge attacker(pencrypt(n, k)) \rightarrow attacker(pencrypt(minusone(n), k))$$

和 IF 表示法相比使用 Horn 子句对协议进行描述更加简单，这种描述有利于节省存储空间和简化数据结构，能够大幅度提高算法速度，但这种表示法的代价是表示能力稍弱，不及 IF 语言表示能力强。

4.2 归结方法的执行过程

在规则库中，如果一条规则被另一条规则蕴含，那么这条规则是冗余的规则，首先需要去掉规则集中冗余的规则，下面是其中使用的规则蕴含：

规则蕴含：

$(H_1 \rightarrow C_1) \Rightarrow (H_2 \rightarrow C_2)$，当且仅当 $\exists \sigma, \sigma C_1 = C_2, \sigma H_1 \subseteq H_2$。其中 σ 是一个替换。去掉规则库中的冗余规则可以减少归结时的工作量，加快运行速度。

ProVerif 利用下面的逻辑推导树来判断一个事实是否能够在初始条件下被推导出来，如果一个事实可以由初始条件推导出，那么这个事实会被入侵者获取，由此判定协议存在攻击。

逻辑推导树：

设 F 是一个基原子公式，B 是一组规则(一个 Horn 子句即为一条规则)，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；

(2)每一条边均用一个基原子公式标记；

(3)若树中的一个结点用 R 标记，指向结点的边用基原子公式 F_0 标记，n 条结点发出的边又分别用基原子公式 $F_1 F_2, \cdots, F_n$ 标记，则 $R \Rightarrow F_1 \wedge F_2 \wedge, \cdots, \wedge F_n \rightarrow F_0$；

(4)根结点有唯一一条出边并且用基原子公式 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。

ProVerif 通过初始状态和规则库 B 来判断攻击状态是否可以到达。Proverif 求解算法以描述协议的 Horn 子句作为初始规则库 B_0，对 B_0 中的规则进行归结，产生新的规则。例如对于规则 $R = H \rightarrow C$ 和 $R' = H' \rightarrow C'$，可以进行归结产生新的规则 $\sigma(H \cup (H' - F_0)) \rightarrow \sigma C'$，条件是：

(1) C 和 $F_0 \in H'$ 是可归一的，且 σ 是它们的最一般归一化因子；

(2)确保 F_0 中不含有可以和初始状态合一的状态，也不允许 F_0 形式为 $attacker(x)$，其中 x 是一个变量，前者会导致把可以初始状态合一的状态归结掉，后者会导致项 $attacker(x)$ 中变量 x 会和任何项进行替换而进入死循环。

通过对规则库 B 里的规则进行归结，把两条相关规则归结成为一条规则，直到规则图稳定，无法产生新的规则，减少了构建逻辑可推导树时的工作量。

在判断协议的保密性是否被破坏的时候，对需要保密的消息 m 只需检测 $attacker(m)$ 是否是逻辑可推导的，即 m 是否可以被入侵者得知。具体作法是 ProVerif 从代表攻击可以实现的子句出发逆向进行归结，如果所有分支都可以终止于表示初始状态的子句，则表明从初始状态出发可以实现对协议的攻击，否则就意味着从初始状态出发无法实现攻击。在这个过程中使用的数据结构是逻辑推导树，逻辑推导树的每一个父亲节点和它的所有子节点分别对应的是一条规则的后件和它的所有前件。这个过程首先在规则库中找到后件可以和代表攻击实现的子句的文字合一的规则，接着扩展逻辑推导树，以这条规则的前件作为根节点的子节点，然后考察根节点的子节点是不是都可以被推导出。如果根节点的子节点都可以被推导则说明根节点可以被推导，目标状态可以达到，存在针对被验证协议的攻击；如果目标状态有一个子节点不能推导，那么根节点不可被推导，则攻击无法实现；如果无法判断目标状态的子节点是否可推导，则在规则库中找到后件和这个子节点可以合一的规则，然后扩展逻辑推导树把这条规则的前件作为这个节点的子节点。对于任意一个节点，如果它的子节点都可以推导出，那么这个子节点可以推导出，如果它有一个子节点不可以推导出，那么这个节点就是不可以推导的，这样不断继续，直到逻辑推导树的所有叶子节点都可以判断出来是否可以推导。在逻辑推导树中可以看到，如果逻辑推导树的某一个叶子节点不可以推导则它的父节点不可推导，向上逐层判断会判断出根节点就是不可以推导的。

4.3 ProVerif 的优势和缺陷

基于归结方法的验证方法具有以下优点：和有限状态的技术相比，它无需对协议的会话数量或者消息的长度进行限制，这使得我们能够给出协议安全性的证明；在实际的协议验证中，它具有很高的时间和空间效率。

ProVerif 使用 Horn 子句来描述协议，Horn 子句之间没有顺序可言，而协议的步骤之间却有严格的时序。因此 ProVerif 返回的攻击可能是不合理的，所以如果 ProVerif 的结果表明被验证协议可以维护某种安全性质，那么这个协议确实可以维护该安全性质。但如果 ProVerif 得到针对被验证

协议的一个攻击，那么这个攻击可能是伪攻击，需要对得到的攻击进行人工分析，以确认它的正确性。

4.4 ProVerif 的终止性

大多数协议在被 ProVerif 验证过程中终止，只有小部分协议不终止。

具体情况分为：

1. 算法自然终止

即逻辑推导树的每个叶节点都可以推导那么说明存在攻击，算法终止；否则如果存在不可以被推导叶节点，得出不存在攻击的结论，算法终止。

2. 算法不终止

如果存在一条规则，它的前件可以和它本身后件合一，在进行规则归结时这条规则的前件可能不断与这条规则的后件合一，导致这条规则被不断应用，进入死循环，使算法不终止。

下面是几种强迫终止的方法：

(1) 可以通过把这种规则的前件加入到限定归结的事实集中，避免它不断进行自己的前件与后件的归结，从而保证终止；

(2) 限制项的深度可以强迫 ProVerif 的过程终止，如果一个项的深度超过限制，那么就用一个新的变量代替；

(3) 在算法运行时限制规则中出现的项的深度，不限制入侵者构造的项的深度；

(4) 通过在协议的消息中加标记[29]来使算法终止。

结束语 在诸多验证方法中，模型检测因为高度自动化、对使用者要求不高、覆盖系统全部状态和可以生成反例等原因而受到越来越多的关注。本文以 SATMC 和 ProVerif 为例分别介绍了基于经典逻辑的安全协议模型检测方法，并简要介绍了我们设计实现的一种新的高效的基于 SAT 方法的安全协议模型检测器 JLU-PV。SATMC 和我们的 JLU-PV 都是采用有界模型检测的方法验证协议安全问题，并使用规划图分析状态转换，简化命题公式的编码。归结方法通过从目标出发的逆向搜索来判定是否存在攻击。目前基于经典逻辑的安全协议模型检测还处在起步阶段，距离真正成熟还有许多工作要做。

参 考 文 献

[1] Lowe G. Breaking and Fixing the Needham- Schroeder Public-Key Protocol using FDR. In Tools and Algorithms for the Construction and Analysis of Systems, LNCS 1055, Springer, 1996:147-166

[2] Clarke E M, Emerson E A. Design and synt- hesis of synchronization skeletons using branch- ing time temporal logic, Logic of Programs. LNCS 131, 1981

[3] Queille J P, Sifakis J. Specification and verification of concurrent systems in CESAR// Proceedings of the 5th International Symposium on Programming. LNCS 137, 1982:337-351

[4] Compagna L. SAT-based Model-Checking of Se curity Protocols. Ph. D. thesis. Universitâ degli Studi di Genova and the University of Edinbur gh (joint programme), 2005

[5] Armando A, Compagna L. SAT-based Model- Checking for Security Protocols Analysis, to be published on International Journal of Information Security

[6] Armando A, Compagna L. Automatic SAT- Compilation of Protocol insecurity Problems via Reduction to Planning// Proceedings of FORTE 2002. LNCS 2529, 2002:210-225

[7] Armando A, Compagna L. Abstraction-driven SAT-based Analysis of Security Protocols . Theory and Applications of Satisfiability Testing, LNCS 2919, 2004:257-271

[8] Armando A, Compagna L. SATMC: a SAT- based Model Checker for security protocols //Proceedings of JELIA'04. 2004, LNAI 3229: 730-733

[9] Armando A, Compagna L, Ganty P. SAT-based Model-Checking of Security Protocols using Planning Graph Analysis// Proceedings of FME'03. LNCS 2805, 2003: 875-893

[10] Blanchet B. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules// Proceedings of CSFW'01. 2001:82-96

[11] Dolev D , Yao A C . On the security of public key protocols . IEEE Transactions on Information Theory, 1983, IT-29 (12): 198-208

[12] Ernst M D, Millstein T D, Weld D S. Automatic SAT-compilation of Planning Problems// Proceedings of the 15th International Joint Conference on Artificial Intelligence(IJCAI-97). 1997: 1169-1177

[13] Kautz H, McAllester D, Selman B. Encoding Plans in Propositional Logic// Proc. of KR. 1996:374-384

[14] Blum A, Fures M. Fast planning through planning graph analysis// Proceedings of IJCAI'95. 1995:1636-1642

[15] Ball T, Rajamani S K. Automatically validating temporal safety properties of interfaces// The 8th International SPIN Workshop on Model Checking of Software. Toronto, Canada, 2001

[16] Clarke E, Grumberg O, Jha S, et al. Counterexample-guided abstraction refine- ment// The 13th Conference on Computer Aided Verification. Chicago, USA, 2000

[17] LIU Ji-Feng , SUN Ji-Gui . Software Model Checking Based on Abstract-verify-refine Paradigm. Computer Science, 2006, 33 (12):12-17(in Chinese)

[18] Even S, Goldreich O. On the security of multi- party ping pong protocols// Proceedings of FOC S'83 . 1983

[19] Comon H, Shmatikov V. Is it possible to decide whether a cryptographic protocol is secure or not? Journal of Telecommunications and In- formation Technoloey, 2002, 4:5-15

[20] Biere A, Clarke E. Bounded Model Checking, Advances in Computers, 2003, 58:118-149

[21] Rusinowitch M, Turuani M. Protocol Insecurity with Finite Number of Sessions is NP- complete// Proceedings of the CSFW' 01. 2001:174-190

[22] Liu Jifeng, Sun Jigui, Gu Binbing, et al . A New SAT-based Security Protocol Model Checking Algorithm (发表中)

[23] Basin D, Mödersheim S , Viganò L. OFMC: A Symbolic Model-Checker for Security Protocols. International Journal of Information Security , 2005, 4(3):181-208

[24] Chevalier Y, Vigneron L. Automated Unbounded Verification of Security Protocols// Proceedings of CAV'02. 2002: 324-337

[25] Blum A, Fures M. Fast planning through plan- ning graph analysis// Proc. 14th Int. Joint Conf. AI. 1995:1636-1642

[26] Allamigeon A, Blanchet B. Reconstruc- tion of Attacks against Cryptographic Protocols// Proceedings of the 18th IEEE Computer Security Foundations Workshop (CSFW18). 2005: 140-154

[27] Li Meng-Jun, LiZhou-Jun, Chen Huo-Wang. SPVT: An efficient verification tool for security protocol. Journal of Software, 2006, 17(4): 898-906(in Chinese)

[28] Li Meng-Jun, LiZhou-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)

[29] Blanchet B , Podelski A . Verification of Cryptographic Protocols: Tagging Enforces Termination// A. Gordon, ed. Foundations of Software Science and Computation Structures (FoSSaCS'03). LNCS 2620, 2003:136-15