时间敏感的安全协议建模与验证:研究综述

周 俐[1] 李舟军[2] 王志勇[3] 王巾盈[4]
(国防科技大学计算机学院 长沙 410073)[1] (北京航空航天大学计算机学院 北京 100083)[2]
(空军雷达学院基础部)[3] (北京跟踪与通信技术研究所 北京 100094)[4]

摘 要 安全协议用于实现开放互连网的通讯安全,时间戳可以保证协议传输消息时的新鲜性。但目前对含有时间特性的协议的研究还很不成熟,还没有有效的方法来验证带时间戳的安全协议。这使得一些大规模复杂协议的安全性质无法通过形式化方法进行全面的验证。详细说明了时间戳的起因和研究时间戳的原因;详细介绍了国际上时间戳特性的几种主流研究方法——MSR方法、归纳法、CSP方法和BAN逻辑在时间敏感安全协议验证方面的工作,对它们的优缺点进行了评述,并指出了进一步的研究方向。

关键词 安全协议,形式化验证,时间模型,时间戳
中图法分类号 TP309 **文献标识码** A

Survey on Modelling and Verification of Time Sensitive Security Protocol

ZHOU Ti[1] LI Zhou-jun[2] WANG Zhi-yong[3] WANG Jin-ying[4]
(School of Computer Science, National University of Defence Technology, Changsha 410073, China)[1]
(School of Computer Science and Engineering, Beijing University of Aeronautics and Astronautics, Beijing 100083, China)[2]
(Basic Course Department, Radar Academy of Air Force)[3]
(Beijing Institute of Tracking and Communications Technology, Beijing 100094, China)[4]

Abstract Security protocols are used to provide secure communication over open network. Time-stamp can make sure the freshness of the message in the protocol, but it is not enough to research the time sensitive protocols, and there are no effective methods that can verify these protocols. So it's very difficult to verify all aspects of those huge and complex protocols in formal ways. This paper described why using time-stamp and researching it, and discussed several popular methods in this field which are MSR method, induction method, CSP method and BAN logic method in verification of time sensitive security protocols. This paper gave out their estimations. Finally we also stated the possible new directions of time sensitive security protocols verification.

Keywords Security protocols, Formal verification, Timed model, Time-stamp

1 引言

在过去的几十年里,网络和分布式系统的发展为各类组织与用户提供了更多的交流与服务,也使资源的利用率进一步得到了提高。为了从系统中获益,参与者必须按一定方式证明自己的身份,这就产生了协议。协议是指两个或者两个以上的参与者为完成某特定任务而相互约定的步骤和规则。安全协议是指在网络环境中完成主体身份认证和密钥分发等任务的具有安全功能的协议。由于网络上参与者们只能靠发送、接受消息来判断对方的身份,这就使得网络上的身份认证、数据保密等要求不能用现实世界中的办法来保证。安全协议使用加密技术在开放的互联网上实现密文传送、身份认证、授权等功能,它是实现安全电子商务的基本保证。但是在技术发展的同时,安全协议的安全性质也受到越来越多的挑战,不少协议相继发现了攻击,这使得人们对协议的形式化验证产生了极大的兴趣,随之产生了许多形式化分析方法[1,2]。

同时,由于有些协议修改了一些内容(例如引入了异或算子等),这使得它的安全性质需要重新评估。

著名的Needham-Schroeder对称密钥认证协议是基于可信任第三方的对称加密协议,由于会话密钥仅在一次协议运行中使用,因此它通常是随机产生的,并不复杂,从而降低了破解它的难度。从密码学的角度来说,破译该会话密钥的可能性很大,因而可以认为通过一段较长的时间后,攻击者能够掌握该密钥。Denning和Sacco在文献[13]中建议使用时间戳来解决该问题。

目前有很多协议都带有时间戳,但时间戳并不能解决人们所遇到的一切问题,因而关注带时间戳的安全协议的安全性质验证问题成为大家所关心的热点问题。在最早的协议

到稿日期:2008-10-08 返修日期:2009-03-13 本课题受国家自然科学基金(60473057,90604007)资助。
周 俐(1981—),男,博士,CCF会员,研究方向为安全协议形式化验证;E-mail:tzhou@nudt.edu.cn;李舟军(1963—),男,博士,教授,博士生导师,研究方向为进程代数理论、数据挖掘、信息安全;王志勇(1977—),男,硕士,讲师,研究方向为信息安全、运筹学;王巾盈(1980—),女,硕士,工程师,研究方向为网络安全。

中，没有消息的新旧特性，这使得协议容易受到重放、延迟等攻击。为了解决这些问题，人们在协议中逐步引入了新鲜值、计数器和时间戳。

新鲜值　一个为了同样目的仅用一次的随机数，主要用于防止重放攻击。在协议中，主体 A 产生一个新鲜值 N_A，发送给另一个主体 B，B 返回一个消息 $m=f(N_A,\cdots)$。这时，通过检测消息 m 中的 N_A，主体 A 可以断定 m 是最近产生的，并且产生时间晚于 N_A，从而保证了 m 的新鲜性。它的缺点是必须通过交互的方式才可能达到新鲜性，增加了消息的传送量。

计数器　协议中发送方与接收方共同保持一个计数器，随着每次消息的发送，计数器的值增加一次。其缺点是潜在的通讯方均须保持状态信息。一旦信道出现问题，计数器的管理也成问题。

时间戳　消息的发送者给消息加上发送时的时间，接收者检测收到的消息上的时间信息，并与当地时间比较，如果收到的时间戳位于一定的时间范围内，则可以断定消息是新鲜的。但是这里主要的困难在于，时间戳的应用要求通讯主体间保持时钟的同步。

在上述 3 种保持消息新鲜性的方法中，以带时间戳的验证最为复杂，原因是新鲜值只使用一次，且随机产生；计数器产生连续整数，容易验证处理；而时间戳是与时刻对应的，它的取值是所有实数(或正实数)。这就存在以下两个问题：

1)如何对时间戳建模？在建模时，应当体现时刻的特性，即时间戳之间的关系是全序关系。

2)验证过程如何处理时间戳？这是带时间戳的安全协议验证的核心问题。它主要体现在时间戳的本质是时刻，而时刻是无穷多的、稠密的，任何两个不同时刻之间可以插入任意多的时刻。基于时刻的这种特性，人们根据目前已有的验证方法提出了各种近似解决方案。

时间戳应用广泛的原因在于与新鲜值相比，时间戳更自然，更反映时间的本质属性；与计数器相比，时间戳更具有可操作性。由于时间戳关于“时间先后”关系构成全序，因此任何主体都可以在任何时间点(无论协议参与者是否存在)检查消息的新旧程度。主体只需知道时间戳的生命期，而生命期一般都是已知的。如果协议使用新鲜值，则期望检查消息新旧程度的角色必须在等待该消息之前就已经存在并发出了新鲜值，因而其他主体就可能在消息中插入同样的新鲜值。

很多实际应用的协议都使用时间戳防止攻击，例如 Kerberos 协议。该协议是作为 MIT 的 Athena 计划的认证服务而开发的，是目前在 Internet 上被广泛采用的一种安全验证机制。Windows@2000 就使用了公钥验证的扩展来实现 Kerberos 5。国际上安全协议研究的攻击者模型都是基于广泛应用的 Dolev-Yao 模型，该模型刻画了在攻击者完全控制网络媒介的情况下攻击者的行为特性。目前对时间的处理方法各异，没有一个行之有效的、公认的建模和验证方法，因此在这个蓬勃发展但尚未成熟的领域里，仍然有不少的问题需要加以认真研究。

2　多集重写方法研究时间戳

多集重写(Multi-Set Rewriting，简记为 MSR)源于简单面向逻辑的语言，其目的是在各种假设下对协议分析的判定性问题进行研究[9]，现已演化为一个精确、强大、灵活，且对复杂安全协议的规约依然相对简单的框架。该框架使用一阶逻辑公式上的强类型多集重写规则来表达协议行为，并且它依赖一种存在量化形式，该形式对新鲜值和其他新鲜数据进行符号化建模。多集有时也称为多重集。文献[8]用 MSR 对 Kerberos5 的某些性质做了形式化分析。这充分说明了该方法具有较好的分析大规模复杂协议的能力。MSR(C)的规约 S 是一个五元组$\langle\mathcal{P},\Sigma,\mathcal{V},\mathcal{I},\mathcal{R}\rangle$，其中 $\mathcal{P}$ 是一个谓词符号集，Σ 是一阶符号表，$\mathcal{V}=\mathcal{V}_t\cup\mathcal{V}_i$ 是变量集，$\mathcal{V}_t$ 是项变量集，$\mathcal{V}_i$ 是整数变量集，并且 $\mathcal{V}_t\cap\mathcal{V}_i=\emptyset$。$\mathcal{I}$ 是一个“初始格局”集合，$\mathcal{R}$ 是带标规则的有穷集。使用 $\cdot\mid\cdot$ 和 ε 作为多重集的构造子，用$\oplus$表示多重集的并。用 $Fv(t)$ 表示项(或公式)t 中自由变元组成的集合。如果项(或公式)t 中没有变量，则称 t 是基的。一个格局是一个由基原子公式构成的多重集，例如 $p(ts(3))\mid q(ts(4),h(5))$。

规则具有形式：$\alpha:\mathcal{M}\rightarrow\mathcal{M}':\varphi$。其中 α 是标号；$\mathcal{M}$ 和 $\mathcal{M}'$ 是两个由原子公式构成的多重集；φ 是满足 $Fv(\varphi)\subseteq Fv(\mathcal{M}\oplus\mathcal{M}')$ 的一个线性算术约束。

$\mathcal{S}$ 的操作语义定义成格局上的重写二元关系$\Rightarrow_{\mathcal{R}}$。给定两个格局 $\mathcal{M}_1$ 和 $\mathcal{M}_2$，$\mathcal{M}_1\Rightarrow_{\mathcal{R}}\mathcal{M}_2$ 当且仅当存在格局 $\mathcal{Q}$ 使得 $\mathcal{M}_1=\mathcal{M}\oplus\mathcal{Q}$，$\mathcal{M}_2=\mathcal{M}_2\oplus\mathcal{Q}$ 并且 $\mathcal{M}\rightarrow\mathcal{M}_2$ 是 $\mathcal{R}$ 中的某条规则的一个基实例。规则 $\mathcal{M}\rightarrow\mathcal{M}':\varphi$ 的一个基实例是通过使用对 $Fv(\mathcal{M},\mathcal{M}')$ 的一个基替换在满足 φ 的解集合上的扩展替换而获得的，其中的基替换是指替换的值域是由基项构成的。设 $\mathcal{M}$ 是一个格局，如果存在 $\mathcal{M}_0\Rightarrow\mathcal{I}$ 使得 $\mathcal{M}_0\Rightarrow_{\mathcal{R}}{}^*\mathcal{M}$，其中$\Rightarrow_{\mathcal{R}}{}^*$是$\Rightarrow_{\mathcal{R}}$的传递闭包，则称格局 $\mathcal{M}$ 是可达的。

MSR(C)对时间敏感协议的建模如下：

• 诚实的主体有一个大于 0 的整数作为它身份的标识；攻击者具有一个特殊 0 的身份标识 0。

• 诚实的主体的当前状态用形如 $r_i(id(n),\langle t_1,\cdots,t_n\rangle)$ 的原子公式表达，它的意思是：诚实主体 $n>0$ 作为角色 r 已经执行完第 i 步并且准备执行下一步；$t_1,\cdots,t_n$ 是表示该角色的当前已知的识。知识以一阶项的形式表示，例如：$id(\cdot)$(主体标识)，$sk(\cdot)$(密钥)，$sk(\cdot,\cdot)$(共享密钥)，$ts(\cdot)$(时间戳)，$enc(\cdot,\cdot)$(加密消息)，$\langle\cdot,\cdots,\cdot\rangle$(元组构造子)等等。

• 攻击者的知识用形如 $men(t)$ 的原子公式的集合来表示，攻击者的行为可以用 Dolev-Yao 模型描述。在开放网络上传送消息 message 可以表示为公式 $net(sender,receiver,message)$；加密数据用项 $enc(key,t)$ 表示，其中 key 是形如 $sk(a,b)$ 的项，表示 key 是 a，b 间的共享密钥，t 是代表明文的项。

• 采用一个使用共享变量同步参与者的全局时钟 global clock 来建模时间戳。这个全局时钟的当前值用一个原子公式 $clock(\cdot)$ 表示，时间信息采用如下规则非确定地更新：

time：$clock(ts(now))\rightarrow clock(ts(next)):next>now$

这条规则使得发送者和接收者之间可以有任意的时间延迟。诚实主体的行为用如下形式的规则定义：$\alpha:clock(ts)\mid fresh(sk(y))\mid r(id(a),k)\mid net(id(b),id(a),m)\rightarrow clock(ts')\mid fresh(sk(y'))\mid r'(id(a),k')\mid net(id(a),id(b'),m'):\varphi$ 规则的意义是：在时刻 ts 主体 a 接受到一条消息 m 并认为该消息是从 b 发出的，同时将自己的状态从 r 更新为 r'，将知识 k 更

新为 k'；在时刻 ts' 他发送新消息 m'。如果假设单一主体执行一步只需非常少的时间，则可认为 $ts=ts'=ts(now)$，其中整型变量 now 表示当前时刻。

服务器(S)和响应者(B)都要求检查相应时间戳的有效性，即检查 $now-\delta_i \leqslant T_i \leqslant now (i=1,2)$，其中 δ_1，δ_2 分别代表消息的网络延时。

为了定义协议的多重集的符号化表示，先引入一个特殊的项构造子 $sup(t)$，它表示以 t 作为子项的所有项。

定义 1（扩展项） 变量集 $\mathcal{V}$ 和符号表 Σ 上的扩展项定义如下：

- Σ 中的常量和 $\mathcal{V}$ 中的变量都是扩展项；
- 如果 t 是扩展项，则 $sup(t)$ 也是扩展项；
- 如果 t 是扩展项构成的表，f 是 Σ 中的符号，则 $f(t)$ 也是项。

定义 2（扩展项的指称语义） 基扩展项 t 的指称语义定义如下：

$[\![c]\!]=\{c\}$，其中 c 是常量；

$[\![sup(t)]\!]=\{s \mid s$ 是包含 t 的基项$\}$；

$[\![f(t_1,\cdots,t_n)]\!]=\{f(s_1,\cdots,s_n) \mid s_1 \in [\![t_1]\!],\cdots,s_n \in [\![t_n]\!]\}$。

定义 3（符号化格局） 如果 $p_1,\cdots,p_n$ 是谓词符号，t_i 是由扩展项构成的元组$(i=1,\cdots,n)$，可满足约束 φ 满足 $Fv(\varphi) \subseteq Fv(t_1,\cdots,t_n)$，则称公式 $p_1(t_1) \mid \cdots \mid p_n(t_n):\varphi$ 为符号化格局。

使用 M,N 来表示符号化格局。符号化格局 $M=A_1 \mid \cdots \mid A_n:\varphi$ 的基指称语义定义如下：

$[\![M]\!]=\{[\![A_1\sigma]\!] \mid \cdots \mid [\![A_n\sigma]\!] \mid \sigma$ 是基替换，且 $\sigma_{\mid Fv(\varphi)} \in Sol(\varphi)\}$。

为了考察包含任意数目的并发协议会话的格局，定义符号化格局的集合的向上闭合指称语义(upward closed denotation)如下：

$$\langle\!\langle S\rangle\!\rangle=\{\mathcal{N} \mid \exists \mathcal{M} \in [\![S]\!] s.t.\ \mathcal{M} \subseteq \mathcal{N}\}$$

这样的扩展语义在局部上十分有效地描述了协议安全性质遭到破坏时的情况(例如，表达泄露的秘密在攻击者和有限主体之间共享)，并且与会话和参与者数目无关。扩展项的使用促使参数化程度进一步提高，例如，能够局部地表达某个数据片断在一条消息的内部出现。扩展项可以使用文献[7]给出的合一化算法进行合一化计算。

给定一个 MSR 的格局 S，定义前导操作如下：

$$Pre_{\mathcal{R}}(S)=\{\mathcal{M} \mid \mathcal{M} \in S \text{ s. t. } \mathcal{M} \Rightarrow_{\mathcal{R}} \mathcal{M}\}$$

定义 4（最大约束合一化子） 设 $M:\varphi$ 和 $N:\psi$ 是两个符号化格局，如果 σ 是 M,N 的最大的一般合一化子，且 σ' 是 σ 在整型变量上的最大子约束时，$\gamma \equiv \varphi \wedge \psi \wedge \sigma'$，则称序偶 $\langle\sigma,\gamma\rangle$ 为 $M:\varphi$ 和 $N:\psi$ 的最大约束合一化子(简写为 m. c. u.)。

定义符号化前导操作为：$Pre_{\mathcal{R}}(S)=\{\mathcal{A}\sigma \oplus \mathcal{D}\sigma:\xi \mid \exists \mathcal{A} \rightarrowtail \mathcal{B}:\psi \in \mathcal{R}, \exists \mathcal{M}:\varphi \in S, \mathcal{M}=\mathcal{M}' \oplus \mathcal{D}, \exists \mathcal{D}.\ \mathcal{B}=\mathcal{B}' \oplus \mathcal{D}, \langle\sigma,\gamma\rangle$ 为 $\mathcal{M}':\varphi$ 和 $\mathcal{B}':\psi$ 的 m. c. u.，$\xi \equiv \exists x.\ \gamma$，其中 $x=Fv(\gamma) \setminus Fv(\mathcal{A}\sigma \oplus \mathcal{D}\sigma)\}$

定义 5（符号化回朔可达图） 设 U 是一个符号化格局的集合。符号化操作可用来构造一个符号化回朔可达图 $G=\langle N,E\rangle$ 如下：结点集 N 初始化为 U，对任意的 $M \in N, Q \in Pre_{\mathcal{R}}(\{M\})$，如果存在已访问过的符号化格局 $O \in N$，使得 $\langle\!\langle Q\rangle\!\rangle \subseteq \langle\!\langle O\rangle\!\rangle$，则删除 Q 并将边 OM 加入边集 E 中，否则将 Q 加入结点集 N 中，并将边 QM 加入图中。

定理 1（保密性） 设 S 是一个 $MSR(C)$ 规约，U 是由违反安全性质的符号化格局构成的集合。如果基于 S 和 U 的符号化回朔可达图计算终止，终态时的图的结点集 N 不包含初始格局，则协议没有受到保密性攻击。此时，对任意数目的新鲜值、角色和并发会话，保密性都成立。

3 归纳方法

3.1 归纳证明方法简介

在密码协议运行环境中要考察协议安全属性 P 的正确性，即证明在任何给定的观察事件集合的任何扩展下 P 都是安全的。Paulson 定义了 3 种运算：parts，analz 和 synth。$parts(M)$ 是消息集合 M 的所有消息的构成部分的集合。$analz(M)$ 是利用攻击者得到的密钥对 M 中的消息进行解密所能够得到的消息的集合。而 $synth(M)$ 描述的是攻击者利用集合 M 中已有的消息伪造新的消息的行为。后两种运算实际上是攻击者能力的刻画。

在归纳证明方法中有 3 个重要概念：事件、轨迹和主体。

事件是协议中的通讯动作，它有两种形式：一种是 Say A B X，表示主体 A 发送消息 X 给主体 B；另一个是 Note A X，表示主体 A 内部存储 X。

轨迹是事件序列。[]表示空事件；如果 evs 是一条轨迹，ev 是一个事件，则在符合协议要求的情况下，将事件 ev 连接在轨迹前面(记作 $ev\#evs$)仍然是一条轨迹。

主体分 3 类：友好主体 Friend i(i 是自然数，表示第 i 个出现的友好主体)、服务器 S 以及攻击者 Spy。它们的初始知识集合定义如下：

initState S = {所有共享密钥}

initState(Friend i) = $\{shrK(\text{Friend } i)\}$

initState Spy = $\{shrK(A) \mid A(bad\}$

其中 $shrK(A)$ 是 A 的共享密钥。bad 是在协议运行过程中被攻击者控制的主体的集合。

3.2 归纳方法研究时间戳

文献[5]详细阐述了如何用归纳方法建模时间戳。它认为新鲜值和时间戳在操作上的不同在于后者可被攻击者猜测，因而将时间戳建模为合法消息成分中的猜测数。由此引入的代价仅仅是在引入新的消息构造子、证明一些技术性的引理以及对某些已有定理做少量的更新。该方法在很大程度上具有一般性。

建模猜测数的过程包括 3 个阶段：

1)在消息的构造中扩展一个新的构造子 Number，其参数是自然数。时间戳 T 以消息的成分 Number T 的形式表现。

2)在 $synth$ H 中加入下列规则以允许攻击者任意伪造随机数：$Number\ N \in synth\ H$

3)由于包含了新的成分，其它操作子需要做如下修改：

$parts(\{Number\ N\} \cup H) = \{Number\ N\} \cup (parts\ H)$

$analz(\{Number\ N\} \cup H) = \{Number\ N\} \cup (analz\ H)$

文献[5]认为协议以两种方式使用时间：用当前时间生成时间戳以及根据当前时间检测时间戳是否有效。由于只需考虑特殊时刻，因此只需要考察连续时间的离散样本。假设有一个所有进程都使用的安全的全局时钟，用“轨迹”(trace)建模网络历史的当前时间，从而将时间嵌入轨迹中。

轨迹是线性增长的，由于轨迹模型使用的是交叠并发模型，因此如果在一条轨迹中事件 ev 出现在事件 ev' 之后，则在现实世界中 ev' 出现在 ev 之后。轨迹所对应的一个样本是指，样本的第 i 个值表示的时间是轨迹的第 i 个事件发生的时间($i=1,2,3,\cdots$)。

因而可以用自然数将样本标准化：存在一个从时间的所有离散样本所构成的集合到自然数幂集之间的函数。空样本对应集合{0}，任何单元素集都对应集合{1}，任何两元素集都对应集合{1,2}，等等。由于一个轨迹的当前时间是对应时间样本的最大值，因此在标准化后，长度为 n 的轨迹的当前时间就是 n。定义函数 ct: $event\ list \rightarrow nat$ 为 $ct\ evs \equiv length\ evs$。长度为 n 的轨迹代表发生了 n 个事件的网络的历史，所以从直观上也可以看出轨迹的当前时间为 n。为了形式化描述会话密钥和认证器的生命期，定义两个自然数 seskLife 和 authLife。协议参与者将时间戳与当前时间对照，以确认消息是否在有效期内。这种检测可形式化描述为以下两个二元谓词 expiredS，expiredA：[nat，event list]：

$$expiredS\ Tk\ evs \equiv (ct\ evs) - Tk > seskLife$$

$$expiredA\ Ta\ evs \equiv (ct\ evs) - Ta > authLife$$

例如，$expiredS\ Tk\ evs$ 成立，意味着从 Tk 时刻起，evs 记录的历史时间要比 seskLife 长。Tk 和它所关联的会话密钥在第二条协议消息中产生，所以它不必显式形式化。

协议模型的形式化模型为轨迹的集合，它由以下几部分归纳定义得到：Base 确定归纳基础；Fake 建模攻击者的非法动作；BK1－n 建模协议发起者执行协议第 1－n 步；Oops 规则允许在任意时刻偶然泄漏会话密钥。

文献[5]指出，安全协议形式化推理在信念和知识间存在混乱。知道(know)某事，则有充分的证据证实该事件成立；相信(believes)则无需证据。BAN 逻辑只抓住了信念的特性，而经常与知识(knowledge)相混淆。归纳法通过定理的形式提供主体的信念和知识的元形式化方法。从主体角度阐述的这些定理表明：如果主体能够验证定理的所有假设，则定理所给出的结论就是主体的知识；如果主体不能够验证定理的所有假设，则定理所给出的结论就是主体的信念。该文使用两种方法，即轨迹检查方法和消息接收方法，分别讨论了带时间戳的协议的性质。

创建消息的主体显然知道消息的所有组成成分，包括最终加密消息的密钥。但是由于 Says 事件可能在向前传递消息时出现，因此它不适合表达创建消息。然而，如果消息 X 在事件 Says $A\ B\ X$ 发生前从未出现过(也不是其他消息的一部分)，则 A 是 X 的真实创建者。这可通过检查事件之前的历史状态而确定。由于历史是由轨迹记录的，因此只需检查轨迹就可以做出正确判断。

假设消息 X 是消息 Y 的组成部分，且事件 Says $A\ B\ Y$ 在轨迹 evs 上出现。A 是 X 的真实创建者，当且仅当可证明“A Issues B with X on evs”成立。并且假设在 A 实际创建 X 之前 spy 不能伪造它，即：如果 X 是合成消息，则假定 spy 不能从网络中分析(或伪造)出它的组成成分；如果 X 是密文，则须保证加密密钥没有泄露，且 A 不是 spy。证明知识的策略如下：

- 简化由 Issues 定义的主要子目标。
- 通过回溯，将 Issues 定义的前两个合取支分离。
- 利用假设证明第一个合取支 Says $A\ B\ Y$。利用操作子 parts 的符号演化证明第二个合取支 $X \in \text{parts}\{Y\}$。
- 运用协议模型的结构归纳法验证协议定义中所有步骤保持性质：事件 Says $A\ B\ Y$ 在轨迹中的出现蕴含 Y 在此事件从未出现过。
- 简化所有子目标。
- 证明关于事件 Says $A\ B\ Y$ 发生的协议步骤的子目标。

4 CSP 研究时间戳

1996 年 Lowe 首先采用 CSP 和模型检验技术对安全协议进行形式化分析，他应用 CSP 模型和检验工具 FDR 分析 NSPK 协议，发现了一个未知的攻击。从此以后，CSP 作为一个形式化分析安全协议的新途径而被广泛使用。

CSP 中进程的轨迹模型是描述进程行为的重要手段。某个进程的一个有限轨迹实际上是这个进程通讯至某个时刻为止发生的可见事件的序列。所以，CSP 的进程 P 的轨迹模型是它的轨迹集合。

文献[14]将 CSP 嵌入到定理证明器 PVS(Prototype Verification System)，向 CSP 中引入基于事件的时间，从而实现了大嘴青蛙协议的 CSP 模型的半自动分析。Evans 向进程字母表中新增一个特殊的事件 tock，由此得到的语言即为 tock-CSP。tock 事件用于表达经过一个单位时间，并且 tock-CSP 要求系统中的所有进程同步以反映所有进程以相同的速率流逝时间。延迟就通过事件 tock 的发生来表达。例如，下面就是一个期望在一个单位时间内收到输入否则重传消息 v 的进程：$P(v)$=in? $x \rightarrow Q(x)$ □ tock→out! $v \rightarrow P(v)$。为了更准确地描述协议的交互性，定义紧急事件 urgent 为不可延误事件，即一旦遇到 urgent 事件，则该事件必然在任何 tock 事件之前。下面以 CSP 中拷贝进程 OPC 为例，具体说明紧急事件与普通事件的区别。OPC 定义为：OPC=in? x→out! x→Stop。由定义知，该进程不允许任何延迟，因为在它的定义中没有 tock 事件发生。因为 in 不应是紧急事件，所以它应该允许任意多的 tock 事件在其之前发生。如果允许输出事件受阻，则可描述如下：

TOPC=in? x→OPC(x) □ tock →TOPC

TOPC(x)=out! $x \rightarrow \text{RUN}_{\text{tock}}$ □ tock→TOPC(x)

如果输出是紧急事件，则可描述如下：

TOPC′=in? x →out! $x \rightarrow \text{RUN}_{\text{tock}}$ □ tock→TOPC′

在进程 TOPC′中，一旦 in 发生，则 out 必然立即发生。文献[20]介绍了一种从传统 CSP 的进程到 tock-CSP 的进程的转换机器 Ψ，转换后的进程都不具有紧急事件，例如 ψ(OPC)=TOPC。该机器可以处理所有不具有紧急事件行为的 CSP 进程，并将其转换为 tock-CSP。

在建模时间网络模型的时候，依然采用 Dolev-Yao 模型，但是协议的参与者对时间可能非常敏感，原因如下：

- 他们产生的值(如时间戳)依赖当前时间。因此必须保持当前时钟的踪迹，并且在每个 tock 事件发生时增加当前时间。
- 对依赖时间的特殊消息进行响应。一般说来，要将当前时间与这种消息所带的时间进行比较以检测消息是不是最新的。
- 协议的实现可能包含时间依赖行为，例如时间戳或延

迟等。

基于上述考虑，时间行为被描述为 tock-CSP 进程，时间的流逝是以 tock 为单位的，较好地模拟了现实世界的情况。Evans 以大嘴青蛙协议为例具体说明了该过程，而且时间属性也被引入到认证性检查中来。如果时间戳出现在消息中，则这个性质可以在现有框架中表述，即用消息集合 T 认证另一个消息集合 R。例如，如果 m 是消息，l 是时间戳，则 $\{rec.\ B.\ A.\ m.\ l\}$ 可以认证 $\{trans.\ A.\ B.\ m.\ l' \mid l-d \leqslant l' \leqslant l\}$，后一集合是时间戳在 $l-d$ 到 l 之间发送的消息的集合。

5 BAN 逻辑方法

文献[21]在原有的 BAN 逻辑中增加模态词■（之前必然）和◆（之前可能）。■的意思是在一条状态路径上，当前点之前的所有结点都满足它的辖域中的公式；◆的意思是在一条状态路径上，当前点之前存在某个结点满足它的辖域中的公式。设 φ 和 ψ 是公式，则有下列公理：

K	$\blacksquare(\varphi \supset \psi) \supset (\blacksquare\varphi \supset \blacksquare\psi)$
4	$\blacksquare\varphi \supset \blacksquare\blacksquare\varphi$
D	$\blacksquare\varphi \supset \blacklozenge\varphi$
L	$\blacksquare\varphi \wedge \blacksquare(\varphi \supset \psi) \vee \blacksquare(\psi \wedge \blacksquare\psi \supset \varphi)$
Z	$\blacksquare(\blacksquare\varphi \supset \varphi) \supset (\blacksquare\blacklozenge\varphi \supset \blacksquare\varphi)$

以及推理规则：Nec 如果 $\vdash \varphi$，则 $\vdash \blacksquare\varphi$

K 和 Nec 确保所使用的是模态逻辑系统。其他公理分别描述了时间的特性。4 保证了传递律；D 保证时间的连续性（即不会用完时间）。给定路径 r 和时刻 k：

$(r,k) \vdash \blacklozenge\varphi$ iff $(r,k') \vdash \varphi\ \exists k' < k$

下面以 Two-party 密钥分发协议为例说明该方法是如何应用的。该协议形式化描述如下。

A→B:	A, N_a
B→S:	$B, N_b, \{N_b, N_a, A\}_{Kbs}$
S→A:	$S, B, N_s, \{N_a, N_s, B\}_{Kas}$
A→S:	$A, \{Kab, N_s, B\}_{Kas}$
S→B:	$S, \{N_b, Kab, A\}_{Kbs}$
B→A:	$\{N_a+1\}_{Kab}$

目标约束（Causal Requirements，简记为 CR）如下：

1. $(B\ sees\ (S, \{N_b, Kab, A\})) \supset \blacklozenge(S\ said\ (S, \{N_b, Kab, A\}_{Kbs}))$

2. $(S\ sees\ (A, \{Kab, N_s, B\}_{Kas})) \supset \blacklozenge(A\ said\ (A, \{Kab, N_s, B\}_{Kas}))$

3. $(A\ sees\ (S, B, N_s, \{N_a, N_s, B\}_{Kas})) \supset \blacklozenge(S\ said\ (S, B, N_s, \{N_a, N_s, B\}_{Kas}))$

4. $(S\ sees\ (B, N_b, \{N_b, N_a, A\}_{Kbs})) \supset \blacklozenge(B\ said\ (B, N_b, \{N_b, N_a, A\}_{Kbs}))$

5. $(B\ sees\ (A, N_a)) \supset \blacklozenge(A\ said\ (A, N_a))$

公平性约束（Faithfulness Assumptions，简记为 FA）为：

1. $(B\ said\ (B, N_b, \{N_b, N_a, A\}_{Kbs})) \supset \blacklozenge(B\ sees\ (A, N_a))$

2. $(S\ said\ (S, B, N_s, \{N_a, N_s, B\}_{Kas})) \supset \blacklozenge(S\ sees\ (B, N_b, \{N_b, N_a, A\}_{Kbs}))$

3. $(A\ said\ (A, \{K_{ab}\}, N_s, B\}_{Kas})) \supset \blacklozenge(A\ sees\ (S, B, N_s, \{N_a, N_s, B\}_{Kas}))$

4. $(A\ said\ (A, \{Kab, N_s, B\}_{Kas})) \supset \blacklozenge(A\ said\ (A, N_a))$

5. $(S\ said\ (S, \{N_b, Kab, A\}_{Kbs})) \supset \blacklozenge(A\ sees\ (A, \{Kab, Ns, B\}_{Kas}))$

6. $(S\ said\ (S, \{N_b, Kab, A\}_{Kbs})) \supset \blacklozenge(S\ said\ (S, B, N_s, \{N_a, N_s, B\}_{Kas}))$

因此，对协议验证而言，只需保证公式 FA⊃CR 是有效的。文献[21]从语义上说明了该公式不是有效的，但是没有从语法上给予证明，也没能给出反例。

结束语 用何种方法对时间敏感的安全协议进行建模和验证，以及如何设计相应的自动验证工具是本领域亟待解决的问题。本文介绍了国际上流行的几种研究安全协议的时间戳特性的主要工作。多集重写方法是典型的模型检验方法，它的优势在于使用简单的逻辑推导作为验证手段，实现了对带时间戳的协议的自动验证，其不足在于该方法要求计算符号化回朔可达图，而这种图的计算复杂度很高，不利于验证大规模复杂协议。文献[11]通过在多集重写（MSR）中采用一个全局时钟和时间更新规则来确定绝对时间，该方法具有启发性，但是不直观，并且很繁琐，有很大的改进空间，类似的工作还有文献[27]。归纳方法是典型的定理证明方法，其优势在于它的正确性源于数学归纳法，可靠性高。文献[5]根据“轨迹”的长度来确定各自的时间，但是它用历史踪迹刻画时间的做法可能导致遗漏攻击，所以它虽然能对一些大规模复杂协议进行验证，但是由于模型对时间建模的简化，使得这种验证方法的适用范围受到了约束，如何在这种方法中降低时间的抽象程度是该方法能否进一步推广的关键。文献[14]用改进的 CSP 分析了大嘴青蛙协议基于时间的认证性。CSP 方法对时间的刻画非常类似现实世界的时间，但是它依赖 PVS 的不动点计算，因而该过程是半可判定过程，无法在其计算不终止时给出判断。绝大多数协议在计算时都可能不终止，从而使该方法不能广泛使用。文献[17]分析了带时间信息的协议，但是其重点并没有放在时间戳本身的特性上，而是主要在离散时间下考虑超时与重传问题，文献[18]给出了 tock-CSP 的指称语义模型来研究多安全级别的系统中的消息流。BAN 逻辑建模存在模糊性，不同的人对同一性质（或协议）给出的描述可能存在很大的差异，模态逻辑的自动化推理也非常困难，因而使用 BAN 逻辑做时间敏感协议的验证难度大，不易实现自动化验证。文献[22]将模态词加入到 BAN 逻辑中来，从而使 BAN 逻辑具有验证带时间特性的协议的正确性的能力。但是由于这种方法涉及模态词，因此它的自动化实现的难度很大。目前上述各种方法均没有自动构造反例的算法。对带有时间的协议建模也随着验证工作的开展而不断发展，文献[15]介绍了一种在移动环境下对系统的实时特性建模的代数，给出了时间迁移系统语义（TLTS）。文献[23]阐明了如何对以时间自动机为模型的时间系统进行验证，它提出的时间抽象互模拟使时间自动机由无限状态变成了有穷的。文献[10]利用时间自动机实现了对安全协议的建模与验证。研究超时与重传这两种时间特性的文献比较多，文献[10]考虑连续时间的建模与验证问题，主要关注的是协议在超时与重传时可能出现的攻击，用 Uppaal 实现了对简化的 Needham-Schroeder 协议和 Yahalom 协议的自动验证。文献[15]将超时算子加入到 π 演算中以达到建模移动环境中系统的实时特性的目的。文献[19]给出了实时进程代数来对协议中依赖时间的特性进行分析，研究了关于超时的建模。

在时间敏感安全协议验证领域的进一步研究工作包括：

（下转第 48 页）

Spectrum Access Networks. Piscataway, NJ, USA, 2005: 170-179

[6] Cabric D, Mishra S M, Brodersen R W. Implementation issues in spectrum sensing for cognitive radios[C]// Thirty-Eighth Asilomar Conference on Signals, Systems and Computers. Piscataway, NJ, USA, 2004: 772-776

(上接第 7 页)

(1)设计一个简洁的符号表示方案表示时间,从而达到有效建模的目的;(2)在时间建模的基础上,研究一套更有效的符号化验证方法对时间敏感协议进行验证,例如上节提出的在验证中加入约束条件等;(3)对存在攻击的协议构造相应的反例;(4)根据反例生成的约束条件,给出避免反例产生时的约束,从而能通过控制有效期等因素有效地避免攻击的产生;(5) 对时间同步协议的认证性进行验证;(6)设计自动验证工具自动完成验证和反例构造工作;(7) 自动验证带时间特性的大规模复杂安全协议。

参 考 文 献

[1] 薛锐,冯登国. 安全协议的形式化分析技术与方法[J]. 计算机学报,2006:1-20

[2] 冯登国. 国内外安全协议研究现状及发展趋势[C]// 信息安全国家重点实验室安全协议研讨会文集. 2004,10

[3] Sangiorgi D, Walker D. The π-calculus: a Theory of Mobile Processe[M]. Cambridge University Press, 2001

[4] Abadi M, Gordon A D. A calculus for cryptographic protocols: The spi calculus[J]. Information and Computatiuon, Academic Press, 1999: 148(1): 1-70

[5] Bella G. Inductive Verification of Cryptographic Protocols[D]. University of Cambridge, 2000

[6] Berezin S. Model Checking and Theorem Proving : a Unified Framework[D]. Carnegie Mellon University, 2002

[7] Bozga L, Lakhnech Y, Périn M. Pattern-based Abstraction for Verifying Secrecy in Protocols[C]// TACAS 2003. 2003: 299-314

[8] Butler F, Cervesato I, Jaggard A, et al. A Formal Analysis of Some Properties of Kerberos 5 Using MSR[C]//Proceedings of the 15th IEEE Computer Security Foundations Workshop(CSFW'02). 2002

[9] Cervesato I, Durgin N A, Lincoln P D, et al. A Meta-notation for Protocol Analysis[C]// Proc. of the Twelfth IEEE Computer Security Foundations Workshop. 1999: 55-69

[10] Corin R, Etalle S, Hartel P H, et al. Timed Model Checking of Security Protocols[C]// Proceedings of the 2004 ACM workshop on formal methods in security engineering. ACM Press, 2004: 23-32

[11] Delzanno G, Ganty P. Automatic Verification of Time Sensitive Cryptographic Protocols[C]// TACAS 2004, Held as Part of the Joint European Conferences on Theory and Practice of Software. ETAPS 2004. Barcelona, Spain, March 2004: 342-356

[12] Dolev D, Yao A. On the security of public-key protocols[J]. IEEE Transaction on Information Theory, 1983, 2(29): 198-208

[13] Denning, Sacco. Timestamps in Key Distribution Protocols[J]. Communications of the ACM, 1981, 24(8): 533-536

[14] Evans N, Schneider S. Analysing Time Dependent Security Properties in CSP using PVS[C]// Frédéric Cuppens, Yves Deswarte, Dieter Gollmann, Michael Waidner, eds. ESORICS, volume 1895 of LNCS. Springer, 2000: 222-237

[15] Lee J, Žic J. On Modeling Real-time Mobile Processes[C]//25th Australasian Computer Science Conference(ACSC2002). 2002

[16] Lowe G. A Hierarchy of Authentication Specifications [C]// Proc. 10th IEEE Computer Security Foundations Workshop. 1997: 31-43

[17] Lowe G. Casper: A Compiler for the Analysis of Security Protocols[C]//Proceedings of 10th IEEE Computer Security Foundations Workshop. 1997

[18] Lowe G, Ouäknine J. On timed models and full abstraction[C]// Proceedings of the Twenty-first Conference on the Mathematical Foundations of Programming Semantics(MFPS '05). ENTCS, 2005

[19] Gorrieri R, Locatelli E, Martinelli F. A Simple Language for Real-Time Cryptographic Protocol Analysis[C]// Pierpaolo Degano, ed. 12th European Symposium on Programming. ESOP 2003, volume 2618 of LNCS. Heidelberg, Springer-Verlag, 2003: 114-128

[20] Schneider S A. Concurrent and Real-time Systems[M]. Wiley, 1999

[21] Syverson P F. Adding Time to a Logic of Authentication[C]// CCS '93. 1993: 97-101

[22] Syverson P, Meadows C, Cervesato I. Dolev-Yao is no better than Machiavelli[C]// WITS'00. Workshop on Issues in the Theory of Security. 2000

[23] Tripakis S, Yovine S. Analysis of Timed Systems Using Time-Abstracting Bisimulations[C]// Formal Methods in System Design. Springer Science+Business Media B. V., Formerly Kluwer Academic Publishers B. V, 2001

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

[25] Blanchet B. An efficient cryptographic protocol verifier based on prolog rules[C]//Proc. of the 14th Computer Security Foundation Workshop(CSFW14). IEEE Computer Society Press, 2001: 82-96

[26] Blanchet B. From Secrecy to Authenticity in Security Protocols [C]// 9th International Static Analysis Symposium(SAS'02). Vol 2477 of LNCS. Springer-Verlag, September 2002: 242-259

[27] Bozga L, Ene C, Lakhnech Y. A Symbolic Decision Procedure for Cryptographic Protocols with Time Stamps (Extended Abstract)[C]//CONCUR 2004. LNCS 3170. 2004: 177-192