http://www.phei.com.cn

计算机密码学
——通用方案构造及安全性证明

◎田 园 著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

计算机密码学

——通用方案构造及安全性证明

田 园 著

電子工業出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书阐述计算机密码方案与密码协议的安全性证明理论，包括基于计算复杂度概念的计算密码学方法和基于符号演算的形式分析与验证方法。安全性证明是一个技术复杂而思想活跃的领域，本书选择少数典型、普适、有发展前途，同时又不特别复杂艰涩的方法进行论述。在选择这些方法时，作者也充分考虑到这些方法所解决的问题本身都有着相当的理论与应用价值，使读者通过仔细学习这些安全证明，能更深入地理解这些密码方案。

图书在版编目（CIP）数据

计算机密码学：通用方案构造及安全性证明 / 田园著. —北京：电子工业出版社，2008.6

ISBN 978-7-121-06658-0

Ⅰ. 计… Ⅱ. 田… Ⅲ. 电子计算机－密码术 Ⅳ.TP309.7

中国版本图书馆 CIP 数据核字（2008）第 065831 号

策划编辑：董亚峰
责任编辑：裴　杰
印　　刷：
装　　订：北京市李史山胶印厂
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编：100036
开　　本：787×1092　1/16　印张：20.25　字数：615 千字
印　　次：2008 年 6 月第 1 次印刷
印　　数：3000 册　　定价：36.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

序　言

这是一部计算机密码学专著，阐述密码方案及密码协议的分析与证明理论。对可证实的安全性质（Provable Security）的精确描述与理论分析是信息安全领域近十年来所取得的重要进展之一，在这方面所取得成功的概念、结论和方法应该成为该领域每一位研究者的知识库中所涵盖的重要内容。然而，这方面的绝大多数成果仍然分散于研究论文之中，其中绝大多数是外文文献，而系统反映这些最新进展的专著几乎没有，作者希望通过本书对这方面的重要成果进行一个较系统的归纳。

本书内容覆盖很广，但可以划分为两大部分，第一部分为第 2～6 章，主要论述对象是（非交互式）密码方案，涵盖了最典型的密码方案，包括消息认证，数字签名、对称加密和公钥加密（包括 2001 年以来才开始使用的基于身份的公钥加密方案），建立了关于这些密码方案的最典型的安全性质，各种安全性质之间的精确关系，如何通过典型的变换技术从弱密码方案构造强密码方案，以及对这些通用构造技术的严格的安全性证明等，如著名的分支引理（Forking Lemma）、Fiat-Shamir 变换、Fujisaki-Okamoto 变换和最近提出的 Canetti-Halevi-Katz 变换的安全性证明。这一部分是当代计算机密码学理论中最重要的组成部分，其论述完全是计算密码学式的（Computational Setting），即基于计算复杂度描述安全概念，对所证明的每一个结论都给出了安全强度的定量估计（Concrete Security）。

第二部分为第 7～13 章，论述对象是（交互式）密码协议，主要涵盖身份鉴别和密钥交换两类协议，以及如何组合两类协议以实现安全信道。在论述方式上，第 7 章和第 8 章仍然是计算密码学式的论述，而第 9～11 章则是形式模型式（Symbolic Setting）的论述。第 7～9 章汇集了这两个领域里一些最重要、具有普遍意义的成果，特别是第 8 章，充分阐述了 Canetti 所建立的 UC-理论，证明了这一理论最重要的结论——UC-复合定理，并作为具体应用导出了著名的 Bellare-Canetti-Krawczyk 理论。UC-理论是一种非常有前途的理论模型，值得密码学理论工作者深入研究。在第 12 章和第 13 章中，将计算密码学理论和形式模型理论两种方法融合起来解决复杂协议的分析与证明问题，阐述并证明了经典的 Abadi-Rogaway 定理，并将其推广到非自由消息代数和主动攻击模型，最后以作者建立的 Dolev-Yao 刚性复合-稳定性定理和语义相似-对偶定理结束。如何有效地综合计算密码学理论和形式分析技术来证明复杂密码协议，是目前计算机密码学领域最富有挑战性的研究方向之一，作者希望这里的论述能激起读者进一步的研究热情。

本书不是面向初学者的，读者至少需要具备如 Stinson 的《密码学理论与实践》和 Menezes、Ooschot 与 Vanstone 所著《应用密码学手册》[①]中的大部分知识，并在一定程度上熟悉计算机密码学的归结式论证。此外，为正确理解密码协议的形式分析技术，读者最好具备一些基本的数理逻辑知识，如谓词演算、形式证明和模型概念。本书主要阐述通用

① 这两部名著的中译本均已由电子工业出版社于 2004 年出版，译者分别为冯登国和胡磊。

密码方案/协议的构造技术及其安全性证明，而非具体方案的分析与证明，但作为对普遍性理论方法和结论的具体应用，我们在书中给出了丰富的具体密码方案/协议的实例，几乎每个重要的普遍性结论都有实例加以解释。对这些具体的密码方案及其安全性条件，我们都给出了详细描述，但除很少几个之外都没有给出证明，否则将偏离我们的主题，而且书的篇幅将增长一倍以上。相反，除少数高度相似的情形之外，书中对每一个普遍性的结论都给出了详细证明，其中有些证明比原始证明更简化，而另一些证明因为对理解结论的含义很重要，作者为此精心增加了许多较直观的解释，包括给出一些图示，以帮助读者准确理解其论证的结构并看清推广的要领。

虽然本书不适合于初学者，但作者的经验表明在导师适当指导下它完全适合于用做密码学领域研究生的讨论班材料。本书参考了大量原始文献，同时也包含作者自己在密码学领域的部分工作，如第 6 章除 6.2 节以外的大部分内容，以及第 13 章的绝大部分内容都属于作者自己的研究成果。

在本书即将完稿之际，幸逢英国 UCL 大学 Yvo Desmedt 教授对我院为期一周的访问交流，与这位对密码学领域贡献丰厚的大师当面交流使作者收益良多，感谢大师对本书材料提出的许多建设性意见。最后，作者要衷心感谢大连理工大学软件学院领导集体，特别是沈宏书院长对作者工作的热情支持，感谢他们为大连理工大学软件学院创造的优良的工作环境，使作者无论是研究、写作还是教学都始终身心愉快。谨以此书献给热情鼓励和支持作者的人们！

作者期望通过本书与读者分享计算机密码学这一高雅而深刻的智力创造，同时欢迎来自读者的批评意见并对此表示感谢。

田　园

于大连理工大学软件学院

目 录

第1章 导 论

信息安全技术所要解决的问题可以概括为如何在不可信任的环境中实现完全可信任的过程或行为，如在不可信任的开放信道上实现完全可信任的通信(保密而且防止任何篡改)，在互不信任的双方之间实现完全可信任的计算，等等。计算机密码学为此提供精确的、可验证的解决方案，其关键在于可验证性，即在特定条件下精确的、数学意义上的理论证明，这是正确实施一切信息安全技术的前提。

在当代网络环境中，实现信息安全方案的具体机制就是各种类型的安全协议（或称密码协议)。遗憾的是，这一学科的历史表明，如何准确验证各种各样的密码协议是否真正具有设计者所期望的安全性质不仅非常困难，而且常常容易出错。这与早期对各种安全性质的精确概念缺乏正确的理论认识有直接关系，而计算机密码学的安全证明理论的建立很大程度上正是源于解决这一难题。本书论述了密码方案和协议安全性质的精确的数学证明理论。然而在开始阐述严格的理论方法之前，我们先不借助于任何专门技术，完全从直观出发讨论一些经典的例子。这些例子都是近 20 年内被发明的各种密码协议，在建立之初都被“证明”过是“安全”的，而多年之后却被指出有严重的安全漏洞。本章的目的是在开始阐述严格的理论分析与证明技术之前给读者一些实际例子，使读者看到直观的论证无法走远，它在建立密码方案或协议这类特殊的计算机算法方面是十分不可靠的，从而认识到严格、精确的数学理论是不可或缺的。另一方面，这些例子所展示出的许多攻击类型为理论分析模型提供了基本素材，是任何安全分析都必须考虑的重要因素。我们从下一章开始阐述严格的方案构造和安全证明理论。读者将看到，为达到精确分析和证明中等复杂程度密码协议的目的，我们将要花去很长的篇幅才能比较完善地建立起必要的理论基础。

这些例子本身也是有趣的，对其中某些修改过的协议在后面的章节将给予严格的分析和证明。本章另一个目的是通过许多协议实例引进一些今后用到的重要概念，如 strand-图、消息表达式等，使我们在论述严格的分析方法之前对这些重要概念有一个直观上的理解。

1.1 Needham-Schroeder 协议

第一个有趣而又非常简单的例子是 Needham-Schroeder 协议，这是一个基于公钥体制的身份验证协议[①]，在其发表以后的很长一段时间内都被认为是安全的，直到该协议发表

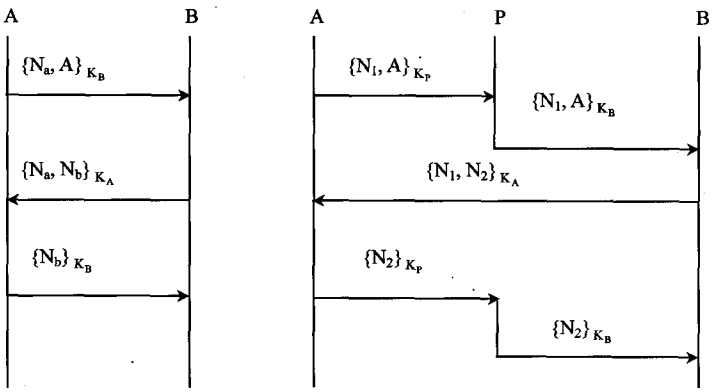
① 历史上有两个 Needham-Schroeder 协议，都发表于 1978 年，我们这里讨论的是基于公钥体制的 Needham- Schroeder 协议。另一个 Needham-Schroeder 协议基于对称加密体制，早年就被发现有被重放攻击的漏洞，改正后发表于 1987 年，与同时发表的 Otway-Rees 协议很相似。对称 Needham-Schroeder 协议的安全漏洞在 1989 年被 BAN 逻辑重新发现。改正后的对称 Needham-Schroeder 协议就是 Kerberos 协议早期版本的基础。耐人寻味的是，在发表 BAN 逻辑的同一篇论文中，对公钥 Needham-Schroeder 协议分析的结果并没有发现其安全漏洞，但这只能说明早期分析方法和技术不够成熟，“没能发现安全漏洞”和严格证明“不存在安全漏洞”是截然不同的事情！

17 年后的 1995 年才被 Gavin Lowe 发现了一个严重的安全漏洞。改正后的协议称为 Needham-Schroeder-Lowe 协议。

原始的 Needham-Schroeder 协议非常简单，表示如下。A、B 是两个用户或进程的名字（ID），N_a、N_b 是随机数，K_A、K_B 分别表示 A、B 的公钥，$\{m\}_K$ 表示以 K 对消息 m 加密后的密文，X、Y 表示字 X 和字 Y 的联结，这些都是典型的消息表达式。Needham-Schroeder 协议的消息交换过程为：

$$A \rightarrow B: \{N_a, A\}_{K_B}$$
$$B \rightarrow A: \{N_a, N_b\}_{K_A}$$
$$A \rightarrow B: \{N_b\}_{K_B}$$

该协议也可以用分布式计算系统中进程的 Lamport 图表达，见图 1-1（a），这种图形表示法就是后面将要介绍的 strand-图理论的基本模型，是一种很直观并且可以精确化的表示方法，今后两种表示方法将交替使用，而且不断地转换。



(a) Needham-Schroeder 协议　　(b) 对 Needham-Schroeder 协议的一种攻击

图 1-1　Needham-Schroeder 协议及其安全漏洞

直观上看，A、B 通过 Needham-Schroeder 协议向对方证实自己的存在。A 将自己生成的随机数 N_a 用 B 的公钥加密后发送到 B；B 生成一个自己的随机数 N_b 后连同 A 的随机数 N_a 一起传递回 A，整个消息以 A 的公钥加密；A 以自己的私钥解密该消息，验证其中包含的第一个数确实是自己在第一个消息中向 B 传递的那个随机数（否则 A 立刻终止该协议），然后将该消息中的第二个数以 B 的公钥加密后传递回 B；B 以自己的私钥解密该消息，验证其中所含的恰是其在第二条消息中生成的一个随机数 N_b。

因为所有消息都以接收方的公钥加密，因此只有接收方才能识别消息的内容。因为两个随机数同时出现在第二条消息中，因此 A、B 都能肯定对方与自己同属于当前的协议会话（这是随机数在协议中的重要用途之一，所谓随机数，其基本属性就是每次随机、独立地生成，因此可以合理假定协议会话中出现的所有随机数是互不相同的，本书今后在讨论形式方法时将始终遵循这一假定）。

如果上面所叙述的一切验证都成立，则协议将正常结束，A、B 在协议结束时都应得出结论：对方存在于刚刚完成的协议会话中，并且该会话中的两个随机数 N_a、N_b 只有自己和对方知道。这就是 Needham-Schroeder 协议所追求的安全性质（但直到 BAN 逻辑发表

之前，一个协议的安全性质却从未被精确地表达出来过）。

这里要强调的是，在以上协议（实际上在任何协议）中，每个参与者只能是根据其自身所发送-接收到的所有消息进行推断，密码协议的目的就是使在协议进行到特定时刻时，参与者能根据这些消息推断出特定的结论。更具体地讲，Needham-Schroeder 协议的目标就是使 A 在接收到第 2 条消息时，能根据前两条消息$+\{N_a, A\}_{K_A}$ $-\{N_a, N_b\}_{K_A}$（+号表示发送，-号表示接收，本书今后将一直沿用这一表示，并且在介绍 strand-图时精确化）推断结论“B 存在于当前的会话中，且 N_b 由 B 生成”。同样，在接收到第三条消息时，B 能根据$-\{N_a, A\}_{K_B}$ $+\{N_a, N_b\}_{K_A}$ $-\{N_b\}_{K_B}$ 推断“A 存在于当前的会话中，且 N_a 由 A 生成”。

果真如此吗？毕竟，以上这些“论证”不是严格和精确的分析。首先，该协议的安全目标的精确含义究竟如何表达？其次，该协议的行为本身应该如何精确表示？最后，如何根据这两者精确地验证或证明该模型确实有所要求的性质？这些问题都是形式分析方法和计算密码学方法要回答的问题，我们将在下一章开始学习。不过现在暂时回到我们的直观论证上来，看 Needham-Schroeder 协议是如何被破解的。

值得特别注意的是第二条消息$\{N_a, N_b\}_{K_A}$，它要表达的意思是“A、B 当前存在于一个时效由随机数 N_a、N_b 界定的会话中”，但注意该消息表达式中实际上没有任何信息可以使接收方据此推断发送方是 B 而不是其他参与者。细心的读者或许会争辩说该消息中的项 N_a 来自 A 发送的第一条消息$\{N_a, A\}_{K_B}$，既然该消息以 B 的公钥加密，当然只有 B 能解密从而提取出 N_a，因此当 A 看到 N_a 出现于$\{N_a, N_b\}_{K_A}$中时，能据此推断出是 B 而不是其他进程发送了该消息。

问题是，这一切需要在一个严格的形式体系和概念框架中精确地证明。实际上，以上“论证”并不成立，图 1-1（b）中的攻击恰是对 Needham-Schroeder 协议这一漏洞的巧妙利用。

A、B 以及其他符号如前述，P 表示攻击者，假定 P 以前破译出过一个合法参与者的私钥，其对应的公钥为 K_P。当 A 期望向 P（从现在起对 P 及其破译出私钥的参与者不加区分，用同一符号表示）证实身份时，向 P 发送消息$\{N_1, A\}_{K_P}$，P 提取出 A 的随机数 N_1 后以 A 的身份向 B 发送消息$\{N_1, A\}_{K_B}$，B 根据协议的规定向 A 发送消息$\{N_1, N_2\}_{K_A}$，注意如前所述，A 并不能根据该消息本身推断出该消息的实际发送者是 B 而不是 A 所期望的 P，于是 A 接收该消息后能正确地通过验证并向 P 发送消息$\{N_2\}_{K_P}$，P 提取出 N_2 后向 B 发送$\{N_2\}_{K_B}$。至此，根据 Needham-Schroeder 协议，A 断定“P 存在于 N_1、N_2 所界定的会话中，且 N_1、N_2 仅有 A、P 知道”，B 断定“A 存在于 N_1、N_2 所界定的会话中，且 N_1、N_2 仅有 A、B 知道”，而事实上，两者的结论都是错误的。

在实际应用中，该错误可能导致严重后果，因为随机数 N_1、N_2 可能在身份验证之后用于合成会话密钥，当 B 以这样合成的密钥向 A 发送敏感消息时，P 能轻易地破译这些消息，这当然不是协议设计者所期望的结局。

对 Needham-Schroeder 协议的改正非常简单，这就是 Needham-Schroeder-Lowe 协议：

$$A \rightarrow B : \{N_a, A\}_{K_B}$$
$$B \rightarrow A : \{N_a, N_b, B\}_{K_A}$$
$$A \rightarrow B : \{N_b\}_{K_B}$$

读者将在学习 strand-图方法时看到，Needham-Schroeder-Lowe 协议的安全性质能被严格表达和证明。

1.2 更多的例子

1.1 节较详细地解释了对 Needham-Schroeder 协议的一种攻击，在发现该攻击之前该协议曾长期被认为是安全的。事实上，这类例子是很多的，本节将给出更多的例子，同时我们也试图通过这些例子总结出一些经验性的原则，从直观的角度初步分析为什么会有安全漏洞。

1.2.1 其他身份鉴别协议的例子

（1）Denning-Sacco 协议

这是一个基于公钥体制的会话密钥交换协议，参与的进程是准备进行会话的双方 A、B 和一个双方共同信任的服务器 S，A、B 的公钥分别是 K_a、K_b，对应的私钥分别为 d_a、d_b，且 A、B 分别有证书 CA、CB。在下面的分析中 CA、CB 的具体形式无关紧要，消息交换过程如下：

$A \rightarrow S:\ A,B$

$S \rightarrow A:\ CA,CB$

$A \rightarrow B:\ CA,CB,\{\{K,\ T_a\}d_a\}K_b$

在最后一条消息中，A 生成会话钥 K 和一个表示时效的随机数 T_a，两者相关联后共同以 A 的私钥签字，再以 B 的公钥加密，目的是想提供对 K 的发送方的身份鉴别和对 K 的保密——只有 B 能从该消息提取出正确的 K，并且能判定该消息来自于 A。

然而，消息 CA,CB,$\{\{K, T_a\}_{d_a}\}_{K_b}$ 对该含义的表达并不充分，具体地说，消息中的子项 $\{\{K, T_a\}_{d_a}\}_{K_b}$ 可以被一个意图假冒发送者身份的进程所重用。设想 C 是这样一个进程，在与 A 以 Denning-Sacco 协议获取了会话密钥 K_{ac} 后，C 再与 B 进行一次 Denning-Sacco 协议，C 向 B 发送的最后一条消息：

$C \rightarrow B:\ CB,CC,\{\{Kac,\ T_a\}d_a\}K_b$

于是，根据 Denning-Sacco 协议 B 将判定 K_{ac} 来自 A，而实际上 C 也知道该密钥，当 B 以 K_{ac} 加密其敏感数据与 A 会话时，将立刻被 C 破译。

读者不难画出 Denning-Sacco 协议及以上攻击的完整的 Lamport 图。

对 Denning-Sacco 协议可以做如下改正：

$A \rightarrow S:\ A,B$

$S \rightarrow A:\ CA,CB$

$A \rightarrow B:\ CA,CB,\{\{A,B,K,T_a\}d_a\}K_b$

（2）Woo-Lam 协议

这是一个基于对称加密体制的身份鉴别协议，S 是被共同信任的服务器进程，进程 B 期望验证进程 A 当前的确存在，A、B 双方的消息交换过程为（注意被验证的 A 是协议过程的发起方）：

A → B: A

B → A: N_b

A → B: $\{N_b\}K_{as}$

B → S: $\{A,\{N_b\}K_{as}\}K_{bs}$

S → B: $\{N_b\}K_{bs}$

N_b是随机数，K_{as}、K_{bs}分别是A、S之间和B、S之间的共享密钥。B解密最后一条消息$\{N_b\}_{K_{bs}}$，将提取出的明文与它在第二条消息中发送的随机数比较，如果相同则B判定A当前确实存在。

果真如此吗？如果最后一条消息$\{N_b\}_{K_{bs}}$中的N_b确实来自第三条消息$\{N_b\}_{K_{as}}$中的N_b，则A确实存在，但注意B在接收到消息$\{N_b\}_{K_{as}}$时并不能对其明文做任何推断（请思考为什么），需要借助于S从密文$\{N_b\}_{K_{as}}$提取出N_b后再以形式$\{N_b\}_{K_{bs}}$传递给B，而观察整个协议，S并没有任何依据能判定什么样的N_b的密文才是合法的（以正确的密钥加密）。因此，直观的分析表明B并没有充分的依据判定最后的消息$\{N_b\}_{K_{bs}}$中的N_b确实是A发送的。

以上分析可以归纳为：从B的观点看，消息4到消息5是多对一的关系，即形如$\{A',\{N_b\}_{Ka's}\}_{K_{bs}}$和$\{A'',\{N_b\}_{Ka''s}\}_{K_{bs}}$的消息4都对应相同形式的消息5$\{N_b\}_{K_{bs}}$，利用这一弱点，可以构造出对该协议的一种攻击如下。

A、B、C是三个进程，A当前并未运行，C向B同时发起两个Woo-Lam协议过程，其中一个假冒A的身份，另一个以自己的真实身份运行，并截获所有发向A的消息：

C → B: A　　C假冒A的身份

C → B: C

B → A: N_b　　C截获该消息

B → C: N'_b

C → B: $\{N_b\}K_{cs}$　　C假冒A的身份

C → B: $\{N_b\}K_{cs}$

B → S: $\{A,\{N_b\}K_{cs}\}K_{bs}$

B → S: $\{C,\{N_b\}K_{cs}\}K_{bs}$

S → B: $\{N''\}K_{bs}$

S → B: $\{N_b\}K_{bs}$

N″是S以K_{as}解密$\{N_b\}_{Kcs}$的结果，一般地有$N''\neq N_b$、N'_b，但注意在以上协议过程中，B不能根据最后一条消息本身准确判定$\{N''\}_{Kbs}$、$\{N_b\}_{Kbs}$中的哪一个和$\{N_b\}_{Kcs}$对应，即$\{N''\}_{Kbs}$、$\{N_b\}_{Kbs}$分别属于两个并发协议过程中的哪一个。根据其中有一条对应，另一条不对应的情况，B会得出结论认为当前A在运行而C没有（请读者特别注意C故意将B发送给A的随机数N_b重复用于两条消息$\{N_b\}_{Kcs}$中），结果与事实恰好相反。

对Woo-Lam协议的改正如下：

A → B: A

B → A: N_b

A → B: $\{N_b\}K_{as}$

B → S: $\{A,\{N_b\}K_{as}\}K_{bs}$

$S \rightarrow B: \{A, N_b\}K_{bs}$

即将其最后一条消息改为$\{A,N_b\}_{Kbs}$，使之明确包含B期望验证其存在性的那个进程的ID。

（3）SSL协议

早期的SSL(Secure Socket Layer）协议按以下过程交换会话密钥K：

$A \rightarrow B: \{K\}K_b$

$B \rightarrow A: \{N_b\}K$

$A \rightarrow B: \{CA, \{N_b\}d_a\}K$

K是A生成的会话密钥，d_a是与公钥K_a对应的私钥，N_b是随机数，CA是A的证书，具体形式与下面的分析无关。随机数N_b的目的是用以表达K的时效，但注意以上第三条消息并不能完全表达“K仅用于A、B之间的会话，且K的时效与N_b相同”这一命题。考虑以下两个交错的SSL协议过程：

$A \rightarrow B: \{K\}K_b$

$B \rightarrow C: \{K\}K_c$　　B假冒A

$C \rightarrow A: \{N\}K$

$A \rightarrow B: \{CA, \{N\}d_a\}K$

$B \rightarrow C: \{CA, \{N\}d_a\}K$

在协议终止时，A认为A与B当前有会话钥K，C则认为A、C之间当前有会话钥K，但C并未意识到B也知道K。

改正的SSL协议：

$A \rightarrow B: \{K\}K_b$

$B \rightarrow A: \{N_b\}K$

$A \rightarrow B: \{CA, \{A,B,K,N_b\}d_a\}K$

1.2.2 加密的运用

在设计密码协议时，无论是对称密码体制还是公钥体制都被以各种方式广泛应用，这里以Kerberos协议为例来解释加密在密码协议中的主要应用方式。

Kerberos协议（早期版本）过程如下，其中T_s、T_a是时戳，L是会话钥K的寿命，K_{as}、K_{bs}是会话双方A、B与其共同信任的验证服务器S之间的共享密钥：

$A \rightarrow S: A,B$

$S \rightarrow A: \{T_s, L, K, B, \{T_s, L, K, A\}K_{bs}\}K_{as}$

$A \rightarrow B: \{T_s, L, K, A\}K_{bs}, \{Ta, A\}K$

$B \rightarrow A: \{T_a+1, A\}K$

在继续讨论之前，先借此协议介绍一种协议进程的表示法。在分析协议时，重要的是明确每个参与协议的进程从自身观点都“看到”了什么、“没有看到”什么，即进程按特定时间顺序接收/发送的一组消息（以后称为进程上的消息序列或strand），以及这些消息的内部结构。

按上小节讲过的表示法，进程A上的消息序列（或称strand-A）是：