计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.yg20220101

一种新的密码本原:棘轮密钥交换的定义、模型及构造

冯登国
中国科学院软件研究所可信计算与信息保障实验室　北京 100190

摘　要　在传统密码学应用中,人们总假定终端是安全的,并且敌手只存在于通信信道上。然而,主流的恶意软件和系统漏洞给终端安全带来了严重和直接的威胁和挑战,例如容易遭受存储内容被病毒破坏、随机数发生器被腐化等各种攻击。更糟糕的是,协议会话通常有较长的生存期,因此需要在较长的时间内存储与会话相关的秘密信息。在这种情况下,有必要设计高强度的安全协议,以对抗可以暴露存储内容和中间计算结果(包括随机数)的敌手。棘轮密钥交换是解决这一问题的一个基本工具。文中综述了密码本原——棘轮密钥交换,包括单向、半双向和双向等棘轮密钥交换的定义、模型及构造,并展望了棘轮密钥交换的未来发展趋势。

关键词:密码本原;棘轮密钥交换;单向棘轮密钥交换;半双向棘轮密钥交换;双向棘轮密钥交换;安全消息传递协议
中图法分类号　TP309

New Cryptographic Primitive:Definition,Model and Construction of Ratched Key Exchange

FENG Deng-guo
Lab of TCA,Institute of Software of CAS,Beijing 100190,China

Abstract　In the application of traditional cryptography,people always assume that the endpoints are secure and the adversary is on the communication channel. However,the prevalence of malware and system vulnerabilities makes endpoint compromise a serious and immediate threat. For example,it is vulnerable to various attacks such as memory content being destroyed by viruses,randomness generator being corrupted,etc. What's worse,protocol sessions usually have a long lifetime,so they need to store session-related secret information for a long time. In this situation,it becomes essential to design high-strength security protocols even in the setting where the memory contents and intermediate values of computation (including the randomness) can be exposed. Ratchet key exchange is a basic tool to solve this problem. In this paper,we overview the definition,model and construction of ratchet key exchange,including unidirectional ratcheted key exchange,sesquidirectional ratcheted key exchange and bidirectional ratcheted key exchange,and prospect the future development of ratchet key exchange.

Keywords　Cryptographic primitive,Ratcheted key exchange,Unidirectional ratcheted key exchange,Sesquidirectional ratcheted key exchange,Bidirectional ratcheted key exchange,Secure-messaging protocol

1　研究背景

在实际应用中谈安全,离不开应用环境,也离不开计算环境。例如,在传统密码学应用中,人们总假定终端是安全的,并且敌手只存在于通信信道上。然而,主流的恶意软件和系统漏洞给终端安全带来了严重和直接的威胁和挑战。在这种情况下,如何解决安全问题是一个现实且重要的问题。安全消息传递协议是目前解决这一安全问题的重要手段之一。安全消息传递协议的目标是允许通信双方在存在任意交叉的异步通信信道上安全地交换消息,使得敌手不能读取、改变或插入新消息。

Borisov 等[1]尝试在阅读即焚(Off-the-record,OTP)通信系统中通过规律性地更新加密密钥来实现消息的安全传递,Langley[2]将这种机制称为棘轮(ratcheted)机制。棘轮机制在开放耳语系统的 Signal 协议[3]中得以实现,并在 WhatsApp 和其他安全消息传递系统中得到应用。在棘轮机制中,加密密钥是周期性更新的,每加密一条消息就更新一次密钥。在 OTP 中[1]是这样定义的:

$$B\rightarrow A:g^{b_1};A\rightarrow B:g^{a_1},E(k_1,M_1);B\rightarrow A:g^{b_2},E(k_2,M_2);\cdots$$

其中,a_i 和 b_i 是分别由 A 和 B 选择的随机指数;$k_1=H(k,g^{b_1a_1})$,$k_2=H(k_1,g^{a_1b_2})$,…;H 是一个杂凑函数;E 是一个加密函数,输入为密钥和消息,输出为密文;g 是所使用的群的生成元;k 是一个共享对称密钥。每个参与方只要不再使用这些指数和密钥进行加解密,就会立刻删除它们。

近年来,棘轮机制的研究引起了学术界的关注,具有代表性的工作主要包括:Bellare 等[4]对棘轮机制的目标进行了形式化建模,给出了棘轮加密和棘轮密钥交换的定义,以及满足形式化定义的可证明安全协议,然而这种模型是非常受限的,如它只考虑了单向通信和单方暴露的场景;Cohn-Gordon 等[5]

到稿日期:2021-08-12　返修日期:2021-08-24
通信作者:冯登国(fengdg@263.net)

评估了 Signal 协议实际达到的安全性，认为它有可能弱于其应该达到的安全性；Poettering 等[6]拓展了 Bellare 等[4]的工作，给出了棘轮密钥交换的另一种安全性定义，提出了单向、半双向和双向 3 种棘轮密钥交换，形式化地给出了安全消息传递协议可以达到的最优安全性；Jaeger 等[7]也形式化地给出了与文献[6]类似的工作，主要区别在于其考虑了计算中间值(包括随机数)的暴露；Jost 等[8]给出了次优安全属性的特征定义和一个高效的安全消息传递协议，该协议在双方的存储和中间计算结果(包括随机数)可以暴露的条件下仍然可以达到几乎最优的安全属性；Durak 等[9]也给出了一个非常高效的异步通信协议，且其具备次优安全性，该协议显式排除了计算中间值的暴露，特别是随机数的暴露。

本文主要介绍了棘轮机制中的核心技术——棘轮密钥交换(Ratcheted Key Exchange，RKE)协议，简称为棘轮密钥交换，这是一种新的密码本原，是实现消息安全传递的关键。本文第 2 节比较详细地介绍了单向棘轮密钥交换的定义、模型及构造；第 3 节简要介绍了半双向棘轮密钥交换的定义、模型及构造；第 4 节简要介绍了双向棘轮密钥交换的定义、模型及构造；最后展望了棘轮密钥交换的未来发展趋势。

2 单向棘轮密钥交换

文献[6]将文献[4]中定义的棘轮密钥交换称为单向棘轮密钥交换(Unidirectional RKE，URKE)，并使用不同的命名规则给出了单向棘轮密钥交换的定义，该定义与文献[4]中的定义基本一致，只是更加明确地区分了(会话)密钥、(会话)状态、密文，分别对应于文献[4]中的输出密钥、会话密钥及发送者/接收者密钥、更新信息。

2.1 URKE 的定义

定义 1(URKE)[6] 设 $\mathcal{K}$ 是密钥空间，$\mathcal{AD}$ 是相关数据空间，$\mathcal{S}_A$ 是发送者状态空间，$\mathcal{S}_B$ 是接收者状态空间，$\mathcal{C}$ 是密文空间。则 URKE 是一个满足如下要求的三元算法组 $R=(init, snd, rcv)$。

(1) $init$：一种随机初始化算法，其输出是发送者状态 $S_A\in\mathcal{S}_A$ 和接收者状态 $S_B\in\mathcal{S}_B$。

(2) snd：一种随机发送算法，其输入是状态 $S_A\in\mathcal{S}_A$ 和相关数据串 $ad\in\mathcal{AD}$，输出是更新状态 $S_A'\in\mathcal{S}_A$、密钥 $k\in\mathcal{K}$ 和密文 $c\in\mathcal{C}$。

(3) rcv：一种确定性接收算法，其输入是状态 $S_B\in\mathcal{S}_B$、相关数据串 $ad\in\mathcal{AD}$ 和密文 $c\in\mathcal{C}$，输出是更新状态 $S_B'\in\mathcal{S}_B$ 和密钥 $k\in\mathcal{K}$，或者是 $\perp$(特殊符号 $\perp$ 表示拒绝)。

设 (k_i) 和 (c_i) 是 A 使用相关数据串序列 (ad_i) 连续派生的密钥序列和密文序列，(k_i') 是 B 使用同样的相关数据串序列 (ad_i) 和密文序列 (c_i) 生成的密钥序列。如果对于任意相关数据串序列 (ad_i)，A 和 B 的密钥均相匹配，即对于所有的 i，都有 $k_i=k_i'$ 成立，则称 URKE 满足正确性。

现在我们采用“游戏”(也称为“实验”)的方式来形式化定义 URKE 的正确性。

游戏 1：$FUNC_{\mathrm{URKE}}(\mathcal{A})$

(1)置 $s_A\leftarrow 0$，$r_B\leftarrow 0$(变量 s_A 和 r_B 分别记录发送和接收操作的数目)；$\boldsymbol{adc}_A[\cdot]\leftarrow\perp$(联合向量 $\boldsymbol{adc}_A$ 记录相关数据串和 A 产生的密文)；$is_B\leftarrow$True(标志位 is_B 用于标记 B 是否仍然处于同步状态)；$\boldsymbol{key}_A[\cdot]\leftarrow\perp$(联合向量 $\boldsymbol{key}_A$ 记录 A 建立的密钥)。

(2)初始化 A 和 B 的状态，即 $(S_A, S_B)\leftarrow_{\$} init$。

(3)运行敌手算法 $\mathcal{A}^{SndA(\cdot),RcvB(\cdots)}$，敌手 $\mathcal{A}$ 可以调用谕示器 $SndA(\cdot)$ 和 $RcvB(\cdot,\cdot)$(谕示器(oracle)也译为“预言机”或“预言或谕示”)。

(4)如果敌手成功，则游戏终止并输出 1，记为“Stop with 1”；如果敌手失败，则游戏终止并输出 0，记为“Stop with 0”。

游戏 1 中假定发送者 A 和接收者 B 使用同一个 $init$ 算法进行联合初始化。在该游戏中，敌手可以选择 A 和 B 处理的相关数据串和密文，其目标是使通信双方计算出不匹配的密钥。

下文将描述谕示器 $SndA(\cdot)$ 和 $RcvB(\cdot,\cdot)$。

谕示器 $SndA(ad)$：

(1)运行 A 的发送算法，生成密文和密钥，更新 A 的状态，即 $(S_A, k, c)\leftarrow_{\$} snd(S_A, ad)$。

(2)在联合向量 $\boldsymbol{adc}_A$ 中记录本次操作的相关数据串和密文，即 $\boldsymbol{adc}_A[s_A]\leftarrow(ad, c)$。

(3)在联合向量 $\boldsymbol{key}_A$ 中记录本次操作建立的密钥，即 $\boldsymbol{key}_A[s_A]\leftarrow k$。

(4)将 A 的发送操作数目 s_A 累加 1，即 $s_A\leftarrow s_A+1$。

(5)返回密文 c。

谕示器 $RcvB(ad, c)$：

(1)要求 B 此前没有拒绝接收密文，即要求 $S_B\neq\perp$。

(2)如果当前 is_B 标识 B 仍处于同步状态且收到不匹配的相关数据串或密文，则 is_B 标记 B 进入失步状态，即如果 $is_B\wedge\boldsymbol{adc}_A[r_B]\neq(ad, c)$，则 $is_B\leftarrow$False。

(3)运行 B 的接收算法，更新状态，建立密钥，即 $(S_B, k)\leftarrow rcv(S_B, ad, c)$。

(4)如果 B 的接收算法产生拒绝状态，则返回 $\perp$，即如果 $S_B=\perp$，则返回 $\perp$。

(5)敌手成功仅 A 和 B 在同步状态下($is_B=$True)建立了不同的密钥($k\neq\boldsymbol{key}_A[r_B]$)，即如果 $is_B\wedge k\neq\boldsymbol{key}_A[r_B]$，则 Stop with 1。

(6)将 B 的接收操作数目 r_B 累加 1，即 $r_B\leftarrow r_B+1$。

定义 2(URKE 的正确性)[6] 如果对于任意敌手 $\mathcal{A}$，都有 $\Pr[FUNC_{\mathrm{URKE}}(\mathcal{A})\Rightarrow 1]=0$，则称 URKE 是正确的。其中，$\Pr[G\Rightarrow 1]$ 表示游戏 G 终止并输出 1 的概率。

2.2 URKE 的安全性

URKE 的安全性可通过 URKE 的密钥不可区分性来定义。也就是说，在敌手看来，发送者 A 建立的密钥和接收者 B 恢复的密钥应当均匀随机地分布于整个密钥空间。文献[6]给出的 URKE 的安全性强于文献[4]中的安全性。

下文通过游戏的方式刻画 URKE 的安全性。游戏 2 给出了 URKE 的密钥不可区分性游戏 $KIND^b_{\mathrm{URKE}}$，其中 $b\in\{0, 1\}$ 是挑战比特，这里要求特殊符号即占位符 $\diamondsuit\notin\mathcal{K}$，假定敌手除了可以通过谕示器 $SndA$ 和 $RcvB$ 进行常规的 URKE 操作外，还可以进行如下 4 种谕示器问询：$ExpA$，$ExpB$，Rvl 和 $Chll$，这 4 种问询分别用于暴露用户以获得他们的当前状态、

获得建立的密钥以及对建立的密钥进行 real-or-random 挑战。我们将敌手 $\mathscr{A}$ 的密钥区分优势定义为：$Adv^{kind}_{\mathrm{URKE}}(\mathscr{A}):=|\Pr[KIND^1_{\mathrm{URKE}}(\mathscr{A})\Rightarrow 1]-\Pr[KIND^0_{\mathrm{URKE}}(\mathscr{A})\Rightarrow 1]|$。

游戏 2：$KIND^b_{\mathrm{URKE}}(\mathscr{A})$

(1)置$s_A\leftarrow 0$，$r_B\leftarrow 0$(变量s_A和r_B分别记录发送和接收操作的数目)；$\boldsymbol{adc}_A[\cdot]\leftarrow\perp$(联合向量$\boldsymbol{adc}_A$记录相关数据串和$A$产生的密文)；$is_B\leftarrow$True(标志位$is_B$用于标记$B$是否仍然处于同步状态)；$\boldsymbol{key}_A[\cdot]\leftarrow\perp$；$\boldsymbol{key}_B[\cdot]\leftarrow\perp$(联合向量$\boldsymbol{key}_A$和$\boldsymbol{key}_B$分别记录$A$和$B$建立的密钥)；$XP_A\leftarrow\emptyset$(集合$XP_A$记录敌手暴露$A$状态的时刻，即已发生发送操作的数目)；$TR_A\leftarrow\emptyset$，$TR_B\leftarrow\emptyset$(集合$TR_A$和$TR_B$分别记录敌手暴露状态可以恢复的$A$和$B$的密钥的序号)；$CH_A\leftarrow\emptyset$，$CH_B\leftarrow\emptyset$(集合$CH_A$和$CH_B$分别记录挑战问询过的$A$和$B$的密钥的序号)。

(2)初始化A和B的状态，即$(S_A,S_B)\leftarrow_{\$} init$。

(3)运行敌手算法$b'\leftarrow\mathscr{A}^{SndA(\cdot),RcvB(\cdot,\cdot),ExpA,ExpB,Rvl(\cdot,\cdot),Chll(\cdot,\cdot)}$，敌手$\mathscr{A}$可以调用谕示器$SndA(\cdot)$，$RcvB(\cdot,\cdot)$，$ExpA$，$ExpB$，$Rvl(\cdot,\cdot)$和$Chll(\cdot,\cdot)$。

(4)要求$TR_A\cap CH_A=\emptyset$(避免平凡攻击3，4，5，即敌手对可恢复的A的密钥进行挑战)。

(5)要求$TR_B\cap CH_B=\emptyset$(避免平凡攻击3，4，5，即敌手对可恢复的B的密钥进行挑战)。

(6)游戏终止并输出敌手对挑战比特b的猜测b'，记为“Stop with b'”。

游戏2中使用变量集合定义等方式避免了敌手的如下5种平凡攻击策略：

(1)敌手要求对泄露过的密钥进行挑战或对相同的密钥进行两次挑战。

(2)敌手要求泄露一个A的密钥，并对同步的B的密钥进行挑战。

(3)敌手要求暴露B的状态，并使用获得的状态恢复未来的A的密钥，再对可恢复的密钥进行挑战。

(4)敌手要求暴露B的状态，并使用获得的状态恢复未来同步的B的密钥，再对可恢复的密钥进行挑战。

(5)敌手要求暴露A的状态，并使用获得的状态向B伪装成A发送密文，可恢复未来B通过接收操作建立的所有密钥，再对可恢复的密钥进行挑战。

下文描述谕示器$SndA(\cdot)$，$RcvB(\cdot,\cdot)$，$ExpA$，$ExpB$，$Rvl(\cdot,\cdot)$和$Chll(\cdot,\cdot)$。

谕示器$SndA(ad)$：

(1)运行发送算法，生成密文和密钥，更新A的状态，即$(S_A,k,c)\leftarrow_{\$} snd(S_A,ad)$。

(2)在联合向量$\boldsymbol{adc}_A$中记录本次操作的相关数据串和密文，即$\boldsymbol{adc}_A[s_A]\leftarrow(ad,c)$。

(3)在联合向量$\boldsymbol{key}_A$中记录本次操作建立的密钥，即$\boldsymbol{key}_A[s_A]\leftarrow k$。

(4)将A的发送操作数目s_A累加1，即$s_A\leftarrow s_A+1$。

(5)返回密文c。

谕示器$RcvB(ad,c)$：

(1)要求B此前没有拒绝接收密文，即要求$S_B\neq\perp$。

(2)如果当前is_B标识B仍处于同步状态且收到不匹配的相关数据串或密文，则is_B标记B进入失步状态，即如果$is_B\wedge\boldsymbol{adc}_A[r_B]\neq(ad,c)$，则$is_B\leftarrow$False。

(3)如果敌手暴露了对应的A的状态，则可以伪装成A，此后B建立的所有密钥都可以被敌手恢复出来，即如果$r_B\in XP_A$，则$TR_B\xleftarrow{\cup}[r_B,\cdots]$($TR_B\xleftarrow{\cup}[r_B,\cdots]$表示$TR_B\leftarrow TR_B\cup[r_B,\cdots]$)。

(4)运行B的接收算法，更新状态，建立密钥，即$(S_B,k)\leftarrow rcv(S_B,ad,c)$。

(5)如果B的接收算法产生拒绝状态，则返回$\perp$，即如果$S_B=\perp$，则返回$\perp$。

(6)对同步B的密钥使用特殊符号重新赋值，使得敌手不能泄露或挑战同步B的密钥，避免平凡攻击2，即如果is_B，则$k\leftarrow\diamondsuit$。

(7)在联合向量$\boldsymbol{key}_B$中记录本次操作建立的密钥，即$\boldsymbol{key}_B[r_B]\leftarrow k$。

(8)将B的接收操作数目r_B累加1，即$r_B\leftarrow r_B+1$。

谕示器$ExpA$：

(1)在集合XP_A中记录敌手暴露A的状态的时刻，即$XP_A\xleftarrow{\cup}\{s_A\}$。

(2)返回当前的A的状态S_A。

谕示器$ExpB$：

(1)当B的状态发生暴露时，未来所有B的密钥都可恢复，即$TR_B\xleftarrow{\cup}\{r_B,\cdots\}$。

(2)如果B处于同步状态，则暴露B的状态会导致未来所有A的密钥都可恢复，即如果is_B，则$TR_A\xleftarrow{\cup}\{r_B,\cdots\}$。

(3)返回当前的B的状态S_B。

谕示器$Rvl(u,i)$，$u\in\{A,B\}$：

(1)要求泄露或挑战过的密钥不能二次挑战或泄露，避免平凡攻击1，即要求$\boldsymbol{key}_u[i]\in\mathscr{K}$。

(2)取出要泄露的密钥值，即$k\leftarrow\boldsymbol{key}_u[i]$。

(3)对泄露过的密钥使用特殊符号进行重新赋值，即$\boldsymbol{key}_u[i]\leftarrow\diamondsuit$。

(4)返回要泄露的密钥值k。

谕示器$Chll(u,i)$，$u\in\{A,B\}$：

(1)要求泄露或挑战过的密钥不能二次挑战或泄露，避免平凡攻击1，即要求$\boldsymbol{key}_u[i]\in\mathscr{K}$。

(2)根据挑战比特b选择真实密钥或随机生成密钥作为挑战密钥输出值，即若$b=1$，则$k\leftarrow\boldsymbol{key}_u[i]$；若$b=0$，则$k\leftarrow\$(\mathscr{K})$，记为“$k\leftarrow b?\ \boldsymbol{key}_u[i]:\$(\mathscr{K})$”。

(3)对挑战过的密钥使用特殊符号进行重新赋值，即$\boldsymbol{key}_u[i]\leftarrow\diamondsuit$。

(4)集合CH_u记录挑战过的密钥序号，即$CH_u\xleftarrow{\cup}\{i\}$。

(5)返回挑战密钥输出值k。

定义3(URKE的安全性)[6] 如果所有实际敌手的密钥区分优势都是可忽略的，则称URKE具有密钥不可区分性，即URKE是安全的。

2.3 URKE 的构造

本节主要介绍了两种构造 URKE 的方法，这两种方法分别来自文献[4]和文献[6]。

方案 1[4]：构造模块包括一个具有选择消息攻击下强不可伪造性的伪随机函数族 F 和一个随机谕示器 H，$\mathbb{G}$表示一个阶为 $p\in\mathbb{N}$的循环群，$\mathbb{G}^*$ 表示生成元的集合，$F.kl$ 表示 F 的密钥长度。三元算法组 $Alg_1=(init,snd,rcv)$的描述如下：

$(1)init$：随机选择 $K\leftarrow_{\$}\{0,1\}^{F.kl}$，$g\leftarrow_{\$}\mathbb{G}^*$，$y\leftarrow_{\$}\mathbb{Z}_p$；置$S_A\leftarrow(0,K,g^y)$，$S_B\leftarrow(0,K,y)$；返回$(S_A,S_B)$。

$(2)snd(i_A,K_A,Y)$：随机选择 $x\leftarrow_{\$}\mathbb{Z}_p$并计算 $X\leftarrow g^x$，$\sigma\leftarrow F(K_A,X)$，$(k,K_A')\leftarrow H(i_A,\sigma,X,Y^x)$；置 $S_A\leftarrow(i_A+1,K_A',Y)$，$C\leftarrow(X,\sigma)$；返回$(S_A,k,C)$。

$(3)rcv((i_B,K_B,y),(X,\sigma))$：如果 $\sigma\neq F(K_B,X)$，则返回$\perp$；否则，计算$(k,K_B')\leftarrow H(i_B,\sigma,X,X^y)$，置$S_B\leftarrow(i_B+1,K_B',y)$，返回$(S_B,k)$。

在文献[4]提出的 URKE 模型中，尽管敌手可以在游戏中问询 A 的状态，但不能泄露 B 的任何秘密信息。如果模型允许这样的问询，则方案 1 实际上是不安全的。事实上，利用 A 仅使用 B 的静态公钥 Y 来进行加密的条件，敌手可以给出如下完全被动的攻击：敌手首先泄露 A 的会话状态，包含 B 的长期不变的公钥 Y；然后，让 A 进行若干次消息发送过程，并将消息密文不经修改就转发给 B；最后，敌手选择暴露 B 的秘密信息，就可以使用暴露的指数 y 得到 B 过去的会话密钥。对于游戏 2 定义的一个安全的 URKE 协议，这些会话密钥对于敌手来说应当是秘密的：A 应当可以从状态暴露中恢复，并且前向安全性应当在即使 B 的状态发生暴露的情况下也成立。

方案 2[6]：构造模块包括一个密钥封装机制 KEM，即$K=(gen_K,enc,dec)$、一个消息认证码 MAC，即 $M=(tag,vfy_M)$，以及一个随机谕示器 H。三元算法组 $Alg_2=(init,snd,rcv)$的描述如下：

$(1)init$：随机选择$(sk,pk)\leftarrow_{\$}gen_K$，$K\leftarrow_{\$}\mathcal{K}$，$k.m\leftarrow_{\$}\mathcal{K}$；置 $t\leftarrow\epsilon$，$S_A\leftarrow(pk,K,k.m,t)$，$S_B\leftarrow(sk,K,k.m,t)$；返回$(S_A,S_B)$。

$(2)snd(S_A,ad)$：置$(pk,K,k.m,t)\leftarrow S_A$；随机选择$(k,c)\leftarrow_{\$}enc(pk)$，$\tau\leftarrow_{\$}tag(k.m,ad\|c)$；置 $C\leftarrow c\|\tau$，$t\overset{\|}{\leftarrow}ad\|C$（即 $t\leftarrow t\|ad\|C$）；计算$(k.o\|K\|k.m\|sk)\leftarrow H(K,k,t)$，$pk\leftarrow gen_K(sk)$；置$S_A\leftarrow(pk,K,k.m,t)$；返回$(S_A,k.o,C)$。

$(3)rcv(S_B,ad,C)$：置$(sk,K,k.m,t)\leftarrow S_B$，$c\|\tau\leftarrow C$；要求 $vfy_M(k.m,ad\|c,\tau)$（即如果$vfy_M(k.m,ad\|c,\tau)$不成立，则终止并输出 0），$k\leftarrow dec(sk,c)$；要求 $k\neq\perp$，$t\overset{\|}{\leftarrow}ad\|C$；计算$(k.o\|K\|k.m\|sk)\leftarrow H(K,k,t)$；置$S_B\leftarrow(sk,K,k.m,t)$；返回$(S_B,k.o)$。

方案 2 通过组合杂凑链和 KEM 来同时达到前向安全性和状态暴露的可恢复性。与方案 1 的关键区别是：方案 2 中 B 的公钥每次使用后都会更换。上述给出的对方案 1 的攻击不适用于方案 2，敌手在暴露 B 状态后获得的解密私钥是没有用的。

已证明[6]：由方案 2 给出的 URKE，如果函数 H 建模为随机谕示器，KEM 提供单向(OW)安全性，MAC 提供强不可伪造(SUF)安全性，并且 KEM 的会话密钥空间足够大，则它可以达到密钥不可区分性，即方案 2 具有游戏 2 定义的可证明安全性。

3 半双向棘轮密钥交换

文献[6]引入了半双向棘轮密钥交换(Sesquidirectional RKE，SRKE)，可视作 URKE 的推广，这两个本原的基本功能相同：会话包含两个参与方 A 和 B，其中 A 建立密钥，并通过向 B 发送密文，与 B 安全地共享密钥。与 URKE 场景不同，SRKE 中的参与方 B 同样可以产生并向 A 发送密文。然而 B 的发送操作不是为了建立密钥，而是为了增强 A 建立密钥的安全性。事实上，在 SRKE 中 B 可以从状态暴露攻击中恢复。

3.1 SRKE 的定义

定义 4(SRKE)[6] 设 $\mathcal{K}$ 是密钥空间，$\mathcal{AD}$ 是相关数据空间，$\mathcal{S}_A$和$\mathcal{S}_B$分别是发送者和接收者的状态空间，$\mathcal{C}$ 是密文空间。则 SRKE 是一个满足如下要求的五元算法组 $R=(init,snd_A,rcv_B,snd_B,rcv_A)$：

$(1)init$：一种随机初始化算法，其输出是发送者状态 $S_A\in\mathcal{S}_A$和接收者状态$S_B\in\mathcal{S}_B$。

$(2)snd_A$：一种随机发送算法，其输入是发送者状态$S_A\in\mathcal{S}_A$和相关数据串 $ad\in\mathcal{AD}$，输出是更新状态 $S_A'\in\mathcal{S}_A$、密钥 $k\in\mathcal{K}$ 和密文 $c\in\mathcal{C}$。

$(3)rcv_B$：一种确定性接收算法，其输入是接收者状态 $S_B\in\mathcal{S}_B$、相关数据串 $ad\in\mathcal{AD}$ 和密文 $c\in\mathcal{C}$，输出是更新状态 $S_B'\in\mathcal{S}_B$和密钥 $k\in\mathcal{K}$，或者$\perp$。

$(4)snd_B$：一种随机发送算法，其输入是$S_B\in\mathcal{S}_B$和相关数据串$ad\in\mathcal{AD}$，输出是更新状态$S_B'\in\mathcal{S}_B$和密文$c\in\mathcal{C}$。

$(5)rcv_A$：一种确定性接收算法，其输入是状态$S_A\in\mathcal{S}_A$、相关数据串 $ad\in\mathcal{AD}$ 和密文 $c\in\mathcal{C}$，输出是更新状态$S_A'\in\mathcal{S}_A$，或者$\perp$。

SRKE 的正确性定义也可通过一个游戏来刻画。SRKE 中 B 到 A 方向的通信直观上允许 B 不定时地刷新它的状态并通知 A。其目标是使得 B 可以从状态暴露中恢复。在 SRKE 的正确性定义中引入了“时段(epoch)”的概念，B 每次向 A 发送消息时会开启一个新时段，时段按顺序编号，$init$ 算法隐式开启第一个时段(起始编号为 0)。A 每次接收 B 的消息都会更新 A 端记录的时段，并且随后向 B 发送消息时，会在生成的密文中隐式包含最新记录的时段。最后，当 B 接收 A 的密文消息时，会在 B 端结束所有早于该密文隐含的时段。

3.2 SRKE 的安全性

SRKE 安全模型将 URKE 安全模型提升为半双向(更确切地说是一又二分之一向)通信环境。敌手目标仍然是区分建立的会话密钥与随机密钥。

SRKE 的安全性也可通过一个游戏来刻画，这个游戏是游戏 2 和定义 SRKE 的正确性游戏的自然混合体。考查本文 2.2 节中给出的 URKE 的 5 种平凡攻击，对于 SRKE，平凡攻击(1)和(2)不需要修改也成立，平凡攻击(3)和(4)需要一些

小修改,以反映时段更新操作对状态暴露威胁的修复过程。平凡攻击(5)需要增加 A 的同步性要求,因为平凡攻击(5)仅当 A 处于同步状态才成立(在 URKE 场景下恒成立,但是在 SRKE 场景下并非如此)。此外,SRKE 中还存在一种新的平凡攻击(6),即如果 B 的状态暴露了,敌手也可以通过伪装成 B 向 A 实施攻击。因此,定义 SRKE 的安全性游戏应避免如下 6 种平凡攻击策略。

(1)敌手要求对泄露过的密钥进行挑战或对相同的密钥进行两次挑战。

(2)敌手要求泄露一个 A 的密钥,并对同步的 B 的密钥进行挑战。

(3)敌手要求暴露 B 的状态,并使用获得的状态恢复当前 B 支持时段下的 A 的密钥,再对可恢复的密钥进行挑战。

(4)敌手要求暴露 B 的状态,并使用获得的状态恢复当前 B 支持时段下同步的 B 的密钥,再对可恢复的密钥进行挑战。

(5)敌手要求暴露同步的 A 的状态,并使用获得的状态向 B 伪装成 A 发送密文,可恢复未来 B 通过接收操作建立的所有密钥,再对可恢复的密钥进行挑战。

(6)敌手要求暴露 B 的状态,并使用获得的状态向 A 伪装成 B 发送消息,可恢复未来 A 通过发送操作建立的所有密钥,再对可恢复的密钥进行挑战。

3.3 SRKE 的构造

在 SRKE 中,B 可以向 A 发送更新密文,这可以帮助其从状态暴露中恢复。文献[6]中的协议可以处理通信双方完全并发操作(特别地,密文在通信中会产生“交叉”)。然而,这会极大地增加复杂度,因为协议中的算法需要同时处理多个“时段”。有趣的是,为了处理这种更加复杂的通信场景,构造需要用到更强的密码本原:设计的 SRKE 基于一种特殊类型的 KEM,即支持所谓的密钥更新(基于 HIBE 可以给出这种本原的构造)。

简言之,在 SRKE 的构造中,B 通过如下方式从状态暴露中恢复:不定期产生一个新的可更新的 KEM 密钥对,并将公钥发送给 A。A 使用密钥更新功能向当前状态“快速推送”这些密钥,从而使得它们可以识别如下密文:在密钥由 B 发送之后但由 A 接收之前交换的密文。在 A 之后的发送操作中,使用混合新旧公钥来进行封装。

文献[6]给出了一种具体的构造 SRKE 的方法,这种方法的核心思想是:与方案 2 相同,A 到 B 的密文使用 KEM 密文,对应的公私钥分别由 A 和 B 持有,并且公私钥在每次使用完成后都会演化为新密钥。此外,为了让 B 可以从状态暴露威胁中恢复,该方法允许 B 主动更新它的状态。主动更新方式是由 B 产生一个新的 KEM 密钥对,并将对应的公钥(使用 SRKE 特有的 B 到 A 的连接)传递给 A。为了简便,一般使用相同的符号 $\mathcal{K}$ 标记 MAC 的密钥空间和链式密钥的空间。如果期望向量中的某个条目包含一个密文,但是不关心密文的具体取值,则用特殊符号即占位符◇代替。

4 双向棘轮密钥交换

在 URKE 和 SRKE 这两个密码本原中,两个参加者是不对称的,因此文献[6]对 URKE 和 SRKE 这两个概念进行了自然扩展,引入了双向棘轮密钥交换(Bidirectional RKE,BRKE)。在 BRKE 中,双方都可以发送密文,而且双方是对等的,即所有密文都可以用来建立密钥以及从状态暴露中恢复。

BRKE 可视作是两个 SRKE 的混合,每个方向各一个。游戏中的部分变量可以合二为一,从而使得游戏定义更加紧凑。

4.1 BRKE 的定义

定义 5(BRKE)[6] 设 $\mathcal{K}$ 是密钥空间,$\mathcal{AD}$ 是相关数据空间,$\mathcal{S}$ 是状态空间,$\mathcal{C}$ 是密文空间。则 BRKE 是一个满足如下要求的三元算法组 $R=(init,snd,rcv)$:

(1)$init$:一种随机初始化算法,其输出是一对状态 $(S_A,S_B)\in\mathcal{S}\times\mathcal{S}$。

(2)snd:一种随机发送算法,其输入是状态 $S\in\mathcal{S}$ 和相关数据串 $ad\in\mathcal{AD}$,输出是更新状态 $S'\in\mathcal{S}$、密钥 $k\in\mathcal{K}$ 和密文 $c\in\mathcal{C}$。

(3)rcv:一种确定性接收算法,其输入是状态 $S\in\mathcal{S}$、相关数据串 $ad\in\mathcal{AD}$ 和密文 $c\in\mathcal{C}$,输出更新状态 $S'\in\mathcal{S}$ 和密钥 $k\in\mathcal{K}$,或者 $\perp$。

BRKE 的正确性定义也可通过一个游戏来刻画。在 BRKE 中,参与者的作用是对称的。

4.2 BRKE 的安全性

BRKE 的安全性也可通过一个游戏来刻画,这个游戏将 SRKE 安全模型提升为双向的通信环境,并将谕示器 $SndA$ 和 $SndB$ 以及谕示器 $RcvA$ 和 $RcvB$ 分别合并为单个谕示器 Snd 和 Rcv。唯一值得注意的是,要求在 BRKE 中的两个参与者都可以建立密钥,游戏管理每个用户密钥数组的两个副本:用 $\boldsymbol{key}_u[\mathcal{S},\cdots]$ 表示用户 u 作为发送者角色建立的密钥,用 $\boldsymbol{key}_u[\mathcal{R},\cdots]$ 表示用户 u 作为接收者角色建立的密钥。

4.3 BRKE 的构造

文献[10]给出了两种构造 BRKE 的方法,一种是通过在相反方向融合两个 SRKE 实例的方式来构造,为了实现安全性,两个实例需要周密融合,可通过一次签名实现这一点;另一种是通过交叉组合 SRKE 中构造模块的方式得到,这种方法不太通用但更加高效。

5 未来展望

本文主要介绍了一种新的密码本原——棘轮密钥交换,这些研究结果主要来源于文献[4]和文献[6]。棘轮密钥交换是实现安全消息传递的关键工具,它的提出不仅丰富了密码本原,而且激发了一些密码本原的发展与提升。棘轮密钥交换是一个较新的概念,其研究还很不成熟,仍有很多问题需要研究,归纳起来主要有以下几个方面:

(1)新工具。目前棘轮密钥交换的正确性和安全性主要是基于游戏来刻画的,有很多攻击策略还需要通过排除法来处理,如何将其刻画得更加完备和简洁,是一个值得进一步研究的问题,包括语义定义、研究工具开拓、安全模型刻画等。

(2)新设计。目前的设计需要很多其他密码本原,而且有的密码本原的效率很低,如何设计高效、安全、简洁的棘轮密

钥交换是一个极具挑战性的问题。

(3)新功能。棘轮密钥交换的功能相对单一,需要从单一功能向多功能拓展,如同时具有可认证性、匿名性等。这方面的研究主要关注两点:1)可证明安全的设计方法;2)方案的性能分析与评估。

(4)新应用。探索棘轮密钥交换的新应用,主要包括两个方面:1)在理论上还可用于设计哪些方案或协议;2)在哪些环境中有更显著的应用价值或前景,可以更好地解决哪些安全问题。

(5)新方向。在实际应用中谈安全,离不开应用环境,也离不开计算环境。对于一类协议,如果应用环境有 x 种情形,计算环境有 y 种情形,则这类协议就有 xy 种情形。特别地,对于棘轮密钥交换,如果应用环境分为终端是安全的和终端是不安全的两种情形,计算环境分为具有传统计算能力和量子计算能力两种情形,则棘轮密钥交换就有 4 种情形:1)终端是安全的且具有传统计算能力,此时棘轮密钥交换就是传统的密钥交换,这种协议已有很多,如 DH 密钥交换、MQV 密钥交换等;2)终端是安全的且具有量子计算能力,此时棘轮密钥交换就是抗量子密钥交换,这种协议正在发展和完善之中,如基于格的密钥交换、基于格的口令认证密钥交换等;3)终端是不安全的且具有传统计算能力,此时棘轮密钥交换就是本文介绍的密钥交换,这种协议主要来源于文献[4]和文献[6],但其还很不成熟;4)终端是不安全的且具有量子计算能力,此时棘轮密钥交换就是抗量子棘轮密钥交换,这种协议还未见报道,需要结合抗量子计算数学难题进行系统研究,是一个值得深入研究的方向。

参 考 文 献

[1] BORISOV N, GOLDBERG I, BREWER E. Off-the-record communication, or, why not to use pgp[C]// Proceedings of the ACM Workshop on Privacy in the Electronic Society (WPES). 2004:77-84.

[2] LANGLEY A. Pond GitHub repository, README. md [OL]. https://github.com/agl/pond/commit/7bb06244b9aa121d367a6d556867992d1481f0c8.

[3] Open Whisper Systems: Signal protocol library for java/android GitHub repository [OL]. https://github.com/WhisperSystems/libsignal-protocol-java.

[4] BELLARE M, SINGH A C, JAEGER J, et al. Ratcheted Encryption and Key Exchange: The Security of Messaging[C]// CRYPTO 2017. Springer, 2017:619-650.

[5] COHN-GORDON K, CREMERS C, DOWLING B, et al. A Formal Security Analysis of the Signal Messaging Protocol[C]// 2017 IEEE Euro S&P 2017. Paris, France, 2017:451-466.

[6] POETTERING B, RÖSLER P. Towards Bidirectional Ratcheted Key Exchange[C]// CRYPTO 2018. Santa Barbara, USA, Springer, 2018:3-32.

[7] JAEGER J, STEPANOVS I. Optimal Channel Security Against Fine-Grained State Compromise: The Safety of Messaging[C]// CRYPTO 2018. Santa Barbara, USA, Springer, 2018:33-62.

[8] JOST D, MAURER U, MULARCZYK M. Efficient Ratcheting: Almost-Optimal Guarantees for Secure Messaging[C]// EUROCRYPT 2019. 2019:159-188.

[9] DURAK F B, VAUDENAY S. Bidirectional asynchronous ratcheted key agreement without key-update primitives[OL]. https://eprint.iacr.org/2018/889.

[10] POETTERING B, ROSLER P. Asynchronous ratcheted key exchange[OL]. https://eprint.iacr.org/2018/296.

冯登国,1965 年生,中国科学院软件研究所研究员、博士生导师。长期从事网络与信息安全研究工作,在 Theor. Comput. Sci、J. Cryptology、IEEE IT 等国内外重要期刊和会议上发表论文多篇。

(责任编辑:李亚辉)