基于零知识集的群组密钥分配方案

孙海波,林东岱
(中科院软件所信息安全国家重点实验室,北京 100080)

摘 要: 零知识集是一种新的具有零知识性质的集合.这种集合的构造使得证明者对于任意元素都可以给出一个证明,证明该元素属于这个集合或者不属于这个集合,同时不泄漏额外的信息.本文基于 Pedersen 承诺设计一种新的群组密钥分配方案,利用零知识集的性质实现密钥分配.协议不仅保证了组成员可以安全动态的获得组密钥,而且保证了组成员除了获取组密钥,不会得到群组中其它成员的相关信息.与先前工作相比,本文提出的方案提供了更高的安全特性,适合应用于一些较特殊的网络应用,如网络秘密会议.

关键词: 零知识集; Pedersen 承诺; 群组密钥分配协议
中图分类号: TP393 **文献标识码**: A **文章编号**: 0372-2112 (2005) 02-0345-05

A New Group Key Exchange Protocol Based on Zero-Knowledge Set

SUN Hai-bo, LIN Dong-dai
(State Key Laboratory of Information Security, Institute of Software, CAS, Beijing 100080, China)

Abstract: Zero-knowledge set is a set that has zero-knowledge quality. The structure of the set makes that a prover can commit to any arbitrary finite set of strings and for any string, reveal with a proof whether a given element is in the set without revealing any knowledge beyond the verity of these membership assertions. In this paper, we propose a new Group Key Distribution protocol based on zero-knowledge set and Peterson commitment so that the identity and number of the group members can be concealed and realize key distribution at the same time. The protocol not only provides a dynamic distribution on a group key, but also guarantees nobody can get additional information about other members except the group key. Compared with previous work, our protocol can provide more security, and is suitable for some special network application, such as secret network meeting.

Key words: zero-knowledge set; Pederson commitment; group key distribution protocol

1 引言

一般的群组密钥交换协议主要包括群组密钥分配协议和群组密钥协商协议.目前的群组协议研究主要集中于对于密钥分发的方式,以及对于其一般的安全性的讨论,而没有涉及到群组本身的零知识性.当前所提出的群组协议都是基于认证的,在各种方案中群组成员都需要彼此进行认证,也就是对于将要会话的对象都是确切知道的,而一些特殊的应用领域中,例如需要匿名的进行群组通信需要对成员的身份进行隐藏.为此本文提出的群组协议中首次引入了零知识性,以增加群组密钥交换协议可提供的安全性,扩大群组协议的应用范围.

零知识性的概念早在二十世纪八十年代就被提出并被应用于密码学领域,但是过去的研究都没有对集合的零知识性进行讨论,直至 2003 年 Silvio Micali, Michael Rabin 和 Joe Kilian[1]首次提出了零知识集合的概念.对于零知识集合的任意一个有限子集,证明者都可以构造一个承诺使得对于任意元素,证明者都可以给出一个证明来证明该元素是否属于这个集合,同时不泄漏任何关于该集合的信息,如集合的大小.文献[1]中给出了一个具体的基于 Pederson 承诺构造零知识集合的方案,该方案对于承诺的计算采用了二叉树的结构,验证过程采取非交互式的证明.本文将承诺树的生成过程与群组协议会话密钥的生成过程进行结合,基于同样的承诺方案构造一种新的群组密钥分配协议,使得群组密钥分配协议具有这种良好的零知识性质.

2 零知识集合及 Pederson 承诺

2.1 基本概念

在传统的零知识证明中,对于给定的有 n 个元素的集合 $S=\{x_1, x_2, \cdots, x_n\}$ 和元素 x 可以证明 $x \neq x_1, \cdots, x \neq x_n$,但是在证明的同时泄露了关于集合元素个数的一些信息,即对于 $x_i \in S$ 的证明是不完备的.文献[1]中提出的零知识集合则有效的在提供证明的同时不泄漏额外的信息.

定义 1 如果集合 S 满足下面三个条件:

收稿日期:2004-04-28;修回日期:2004-08-20
基金项目:国家自然科学基金(No. 90204016, No. 60373048);国家 863 项目(No. 2003AA144030)

(1) 该集合的元素个数是有限的.

(2) 对于 S 可以计算一个承诺值 C_s.

(3) 对于任意给定的元素 x_i 都可以依据承诺 C_s 给出一个非交互的证明来证明 $x_i \notin S$ 或者 $x_i \notin S$,同时不泄漏关于 S 的其它信息.

则称集合 S 具有零知识性.

对于集合 S,我们定义映像 $D:\{0,1\}^* \rightarrow \{0,1\}^*$,其中若 $x \in S$,则 $D(x) = X$,否则 $D(x) = \perp$.

定义 2 如果对于任意一个正多项式 $P(x)$,有 $\lim \varepsilon(x) P(x) = 0$,则称 $\varepsilon(x)$ 为可忽略函数.

定义 3 如果对于任意的多项式时间算法 A,和多项式可计算函数 H,都有 $Pr\{\forall x, y, x \neq y, H(x) = H(y)\} = \varepsilon(k)$,并且 H 函数的逆运算是计算上不可行的,则称 H 为碰撞自由(collision-free)的哈希函数[4].

我们约定:

(1) $x \overset{R}{\leftarrow} S$ 表示在 S 中随机选取一个元素 x,这里 S 可以是有限集合也可以是一个概率空间.

(2) $P_1[x_1 \overset{R}{\leftarrow} S_1; x_2 \overset{R}{\leftarrow} S_2; \cdots: p(x_1, x_2, \cdots)]$ 表示 $x_1, x_2, \cdots$ 分别在 $S_1, S_2, \cdots$ 中取值后断言 $p(x_1, x_2, \cdots)$ 成立的概率.

零知识证明系统一般包括证明者提供证明和验证者进行验证两个过程. 在我们的方案中类似的包括证明者对集合 S 提供承诺 C_s 和验证者验证承诺值的过程. 为叙述方便我们定义如下两个函数:$\{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^*$:COMMIT$(\sigma, x) = (c, r)$,其中 σ 是一个公开的随机串作为系统参数,x 是承诺的对象,(c, r) 表示承诺值.

$\{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^*$:VERIFY$(\sigma, c, r) = D(x)$,表示对 x 进行验证.

基于以上给出的定义和记号,我们将后文中讨论的三种安全性:完备性,合理性和零知识性,形式化的描述如下:

完备性 $\forall \sigma, x \in \{0,1\}^*$. $Pr\{(c, r) \overset{R}{\leftarrow} \text{COMMIT}(\sigma, x): \text{VERIFY}(\sigma, c, r) = D(x)\} = 1$,完备性表示证明者可以对集合进行承诺并且对于任意的元素 x 都可以有效的验证承诺的正确性.

合理性 对于任意给定的承诺函数 COMMIT, $Pr\{(c, r), (c', r') \overset{R}{\leftarrow} \text{COMMIT}(\sigma, x): \text{VERIFY}(\sigma, c, r) \neq \text{VERIFY}(\sigma, c', y') \neq \perp\} = \varepsilon(k)$,合理性表示对于同一个 x,证明者不能产生两个不同的承诺使得验证者得到的结果不唯一,也就是说证明者不能提供假的承诺值欺骗验证者.

零知识性 $\forall m_1, m_2 \in \{0,1\}^*$. $C(\sigma, m_1) = C(\sigma, m_2)$,这里 $C(\sigma, m) = \{(c, r) \overset{R}{\leftarrow} \text{COMMIT}(\sigma, m): c\}$,零知识性表示验证者通过证明者做出的承诺和提供的证明只能得到 $D(x)$ 的值,除此之外得不到任何其它的信息.

以上我们给出了零知识集合的概念和性质. 在我们的群组密钥分配方案当中采用的是树状结构,集合的元素被分配在叶子结点上并且每一个结点中都需要存储一定的值,采用文献[1]中的 Merkle 树结构,因此我们首先介绍一下 Merkle 树[1]的概念.

2.2 Merkle 树

我们设计的方案中组密钥的分配以及承诺的生成和验证过程依据的都是二叉树的结构. 我们用 T_k 表示有 2^k 个叶结点的完全二叉树,将一个结点的层数定义为该结点到根结点的距离(该结点到根结点的边的个数). 对于一个内部结点 v,我们用 v_0 和 v_1 分别表示该结点的左右子结点,用 parent(v) 表示该结点的父结点. 对于任意一个叶结点 v,定义由 v 到根结点路径上的所有结点的集合为 S_v,显然对于第 k 层上的叶结点 v, S_v 的元素的个数为 k. 定义集合 $FS_v = \{u \mid u \notin S_v$,且 parent$(u) \in S_v\}$,即这个集合中的元素是 S_v 上每一个结点的兄弟结点.

Merkle 树是一种特殊的二叉树,它在二叉树的每个结点中存储一个值:对于每一个叶结点 v 存储一个随机值 V_v,对于每一个内部结点 v,该结点存储的值 $V_v = H(V_{v_0}, V_{v_1})$,其中 V_{v_0}, V_{v_1} 分别表示 v 的左右子节点中存储的值. 这里所使用的哈希函数 H 就是我们前面介绍的碰撞自由哈希函数,也就是说每一个内部结点存储的值等于它的左右子结点存储值的哈希函数值. 本文的方案对于集合的承诺的计算也是基于 Merkle 树的这种思想,对于结点 v 验证的过程依赖于 Merkle 树中集合 S_v 和 FS_v 中的结点的存储值,但不同的是,我们在每个结点当中存储的数据不止一个,并且对于两个子结点的父结点的值的计算也略有不同. 为了方便起见,我们仍然称本文提出的方案中使用的二叉树为 Merkle 树.

有了 Merkle 树的结构,我们就可以首先将零知识集合中的元素存储到叶结点当中. 对于每一个中间结点直到整棵树的根结点的承诺的计算依赖于 Pederson 承诺[2]实现,而最后根结点的承诺作为公开的对该集合的承诺. 下面我们介绍 Pederson 承诺.

2.3 Pederson 承诺

1991 年,Pederson 提出了一种基于离散对数问题的承诺方案[2],这个承诺方案对于任意一个消息可以进行有效的承诺和验证,而伪造承诺是计算上不可行的. 具体的对于消息 m 的承诺描述如下:

假设存在公开的四元组 (p, q, g, h),这里 p, q 是素数,q 整除 $p-1$,g 和 h 是群的阶为 q 的循环子群的生成元. 相应的承诺算法和验证算法如下:

PED_COMMIT$((p, q, g, h), x) = (c, r)$,这里 r 是随机选取的 Z_q 中的元素,$c = g^x h^r$ (x 是要承诺的对象).

PED_VERIFY$((p, q, g, h), c, x, r)$:如果 $c = g^x h^r$,算法接受,否则拒绝.

Pederson 承诺中用来进行承诺的函数为 $H(ab) = g^a h^b \bmod p$,在离散对数问题假设前提下,显然该函数也是一个碰撞自由哈希函数,因此 Pederson 承诺函数也叫做 Pederson 哈希函数.

我们提出的新的群组密钥分配协议方案中使用的系统参数和 Pederson 承诺中给出的一样,不同的是生成元 g 和 h 虽

然是公开的,但是 g 和 h 之间的指数关系是秘密的不被任何人知道的,即 $\log_g h$ 和 $\log_h g$ 是秘密的. 且方案中使用的哈希函数是 $H(ab) = \alpha^{ab} \bmod p$,其中 α 和 g, h 的定义相同,也是 Z_p^* 的一个阶为 q 的循环子群的生成元,同时它和 g, h 之间的指数关系也是保密的. 显然对于任意的多项式时间算法,找到 a_1, b_1, a_2, b_2,使得 $a_1 b_1 \neq a_2 b_2$,且 $H(a_1 b_1) = H(a_2 b_2)$ 是计算上不可行的,并且因为该函数是一个碰撞自由哈希函数,因此其计算是不可逆的.

3 基于 Pederson 承诺构造的群组密钥分配协议方案

由于本文将重点放在零知识性相关的安全性讨论上,因此我们假设在协议的所有通信方之间存在一个可认证的安全通信信道,这个安全通道提供了用户之间通信的秘密性,认证性和数据完整性,这个安全信道可以使用签名等技术实现,并已经有了成熟的解决方案,这里不再赘述.

假设总共有 n 个用户参与协议并且有一个可信的承诺服务器,记 n 个用户为 M_1, M_2, …, M_n,记可信服务器为 Prover. 首先对于每个用户 M_i,生成随机数 r_i,计算 α^{r_i},并由安全通道发送给服务器 Prover. 服务器解密收到的信息,选择群组大小 m 并选取参与密钥分配的 $m(m < n)$ 个用户,不失一般性设为 M_{j+1}, …, M_{j+m}. 对于这 m 个用户 Prover 产生随机数 R 并计算 $\alpha^{k^R}(j+1 \leq k \leq j+m)$. 对于其它的 $n - m$ 个用户,Prover 产生随机数 R 并计算 $\alpha^{k^{R'}}(1 \leq k < j+1$ 或 $j+m < k < n)$. 然后服务器构造一个 k 层的 Merkle 树满足 $(2^{k-1} \leq n < 2^k)$,在 2^k 个叶结点中随机选取 n 个结点 $(v_1, \cdots, v_n)$ 代表 n 个用户. Merkle 树的每一个叶结点 v 中存储一个值 m,这里我们约定如果该结点表示上面 n 个用户中的一个(不失一般性假设为 M_i),则 $m = \alpha^i$,否则 $m = 0$(存储 0 的结点表示除代表用户的 n 个结点之外的空结点). 我们把表示被选中的群组成员(m 个)的叶结点称为实结点,并且对于任何一个内部结点如果它的两个子结点中至少有一个结点为实结点,则该结点也称为实结点. 实结点以外的所有结点称为虚结点.

对于树中代表用户的每一个叶结点 v, Prover 选择 $e_v = m_v$,对于空结点,Prover 随机选择 e_v,然后如下计算值 h_v:如果该结点为实结点,则 $h_v = h^{e_v}$,否则 $h_v = g^{e_v}$. 对于一个叶结点,有了 m 和 h_v, Prover 再选取 Z_p^* 中的一个元素 r_v,计算该结点的承诺值 $c_v = g^{m_v} h_v^{r_v} \bmod p$. 至此在一个叶结点 v 中一共存储三个值 (m_v, c_v, h_v),这样对于所有叶结点的承诺就完成了. 注意对于每一个实叶结点,承诺 c_v 只能是对于 m_v 的真实的承诺. 因为 Prover 不知道 $\log_g h$,所以对于任意不同的 m 值, Prover 无法计算相同的承诺值. 但是对于一个虚结点, Prover 可以根据需要对任意的结点选择适当的 r_v 作出相同的承诺,因为此时他知道 $\log_g h_v$.

下面我们给出如何计算中间结点的承诺值. 对于任意一个中间结点 v 也需要存储三个值 (m_v, c_v, h_v). 对中间结点的计算顺序由叶结点向上逐层计算直到根结点. 这三个值的计算如下:h_v 的计算方法和叶结点相似. 对于任意的中间结点(无论实结点还是虚结点), Prover 随机产生 e_v,如果该中间结点是实结点,则 $h_v = h^{e_v}$,否则 $h_v = g^{e_v}$. 对于 m_v 的计算我们采用文献[6]中树状群组密钥协商的计算方法(即碰撞自由函数 $H(ab) = \alpha^{ab} \bmod p$),计算 $m_v = \alpha^{(c_{v_0}, c_{v_1})}$ (c_{v_0} 和 c_{v_1} 分别表示该结点左右子结点中存储的承诺值). 同样的对于该结点承诺值 c_v 的计算与叶结点相同,$c_v = g^{m_v} h_v^{r_v} \bmod p$. 如果中间结点是虚结点,则意味着它的两个子结点都是虚结点. 不失一般性我们考虑第 $k-1$ 层的虚结点,它的两个子结点都是叶结点并且是虚结点. 这时依照上面的计算规则 $m_v = \alpha^{(c_{v_0}, c_{v_1})}$,我们可以看到其中 α 的指数都是关于 g 的幂次,$m_v = \alpha^{e_{v_0}(r_{v_0}+1) + e_{v_1}(r_{v_1}+1)}$,其中 e_{v_0}, r_{v_0}, e_{v_1}, r_{v_1} (这些值是在计算 c_{v_0} 和 c_{v_1} 时服务器选择的参数)都是由 prover 选取的. 这样 prover 可以选取适当的 e_{v_0}, r_{v_0}, e_{v_1}, r_{v_1} 使得 $m_v = \alpha$. 如果该结点的父亲结点仍然是虚结点,那么对于该结点的参数 e_v, r_v 的选择仍然遵循相同的原则使得 $m_{\text{parent}}(v) = \alpha$. 如果该结点的父亲结点是实结点,则 e_v, r_v 可以随机选取,有了 e_v,和 r_v, c_v 的计算是相同的. 这样由叶结点逐层计算上去直到根结点 e. 最后公开承诺值 c_e.

方案对承诺的计算中,每个叶结点只需三次模幂操作,每个内部结点只需四次模幂操作. 显然对于有 n 个用户参与的该方案的群组密钥分配的承诺是多项式时间可计算的. 同时由于 Prover 不知道 $\log_g h$,他不能给出一个错误的承诺值,除非他能够求解离散对数问题或者发现哈希函数 $H(ab) = \alpha^{ab} \bmod p$ 的一个碰撞,由我们的前提假设这是计算上不可行的.

下面我们给出用户的验证方法:

对于用户验证的路径与承诺计算的顺序相同,都是由叶结点逐层向上验证直到根结点. 我们用 $P(v)$ 表示由结点 v 到根结点的路径.

对于被选中的群组成员 M_i, Prover 在发送 α^{i^R} 的同时,发送代表该成员的叶结点 v 中存储的 h_v 和相应的 r_v,以及在路径 $P(v)$ 上除叶结点和根结点之外每一个结点 w ($w \in S_v$) 存储的 m_w, c_w, h_w, e_w, r_w 以及每一个结点 w 的兄弟结点 l ($l \in FS_v$) 中存储的 c_l. 当收到这些消息之后,用户 M_i 首先验证叶结点中存储的信息 $h_v = h^{e_v}$ 还是 $h_v = g^{e_v}$. 我们知道在服务器和用户之间消息的传递都是通过安全通道完成的,因此对于 M_i 之外的其它用户无法验证 h_v 的值. 如果 $h_v = h^{e_v}$,那么该用户可以初步认为自己被选中为群组成员,从而知道验证路径 $P(v)$ 上的每一个结点都是实结点. 此时用户得到 m_v, h_v, e_v, r_v,可以计算相应的 c_v. 接下来对于 $P(v)$ 上的每一个结点 u(除了叶结点和根结点), M_i 根据 Prover 提供的值验证:

(1) $m_u = \alpha^{(c_{u_0}, c_{u_1})}$,其中 c_{u_0}, c_{u_0} 分别表示结点 u 的左右子结点中存储的值;

(2) $h_u = h^{e_u}$;

(3) $c_u = g^{m_u} h_u^{r_u} \bmod p$.

如上递归的验证 $P(v)$ 上每一个中间结点的承诺值直到根结点 e,计算 c_e,验证这个值与公开的承诺值是否相等,如果相等则验证结束. 该用户可以确定服务器给出了正确的承诺并且自己已经被选中为群组成员,这时该用户可以用他得到的

α_i^{R}和 r_i 计算得到 α^R 作为被选中的用户群组通信的组密钥.

对于 n 个用户中没有被选中的 $n-m$ 个用户来讲,验证过程与被选中的用户非常类似. 首先他也要验证叶结点 v 中存储的信息 $h_v = h^{e_v}$还是 $h_v = g^{e_v}$,很显然此时 $h_v = g^{e_v}$. 用户知道自己没有被选中,但是这时用户也要继续验证下去以保证 Prover 进行了公正的筛选和正确的承诺,并没有进行欺骗. 此时该用户的验证路径 $P(v)$ 上既有虚结点又有实结点. 假设该用户为 M_j, Prover 在发送 α_j^{R} 的同时,发送代表该成员的叶结点 v 中存储的 h_v 和相应的 r_v,以及在路径 $P(v)$ 上除叶结点和根结点之外每一个结点 $w(w \in S_v)$ 存储的 m_w, c_w, h_w, e_w, r_w 和每一个结点 w 的兄弟结点 $l(l \in FS_v)$ 中存储的 c_l. 用户对于中间结点和根结点的验证过程和前面描述的被选中的群组成员的验证过程是相同的,这里不再赘述.

(注意:实际上对于验证路径上的每一个虚结点 v 来说, $m_v = \alpha$. 依照我们在前面承诺过程中描述的, Prover 对于 e_v, r_v 的选择是遵循一定的规则而并非随机的,因此通过适当的选取可以保证依照上述验证算法计算的 $m_v = \alpha$.)

同样的未被选中的用户递归验证到根结点公开的承诺值,相等则意味着 Prover 没有进行欺骗. 反之则证明有欺骗存在,这次分配的密钥无效.

下面我们介绍这种基于零知识集合的群组密钥分配方案的成员动态变化情况. 当有 n 个成员以外的用户要求加入密钥交换时, Prover 首先根据开始选取群组成员的标准确定是否允许新的用户加入已经选定的大小为 m 的群组当中. 此时已经构造的 Merkle 树的叶结点中还有 $2_k - n$ 个空结点, Prover 可以随机选取任意一个空结点 v 表示该新用户. 如果不满足该标准,也就是说该新用户不能加入已有的群组当中,此时对于新结点 v 的操作和前面介绍的对于虚结点的操作相同,构造 (m_v, c_v, h_v),使得 $h_v = g^{e_v}$. 因为该结点原来是空结点,因此结点中存储的值为 $m_v = 0$, $h_v = g^{e_v}$, $c_v = g^{m_v} h_v^{r_v} \bmod p = g^{e_v r_v} \bmod p$. 当该结点变为新的第 $n+1$ 时, $m_v = \alpha_{n+1}$, $e_v = \alpha_{n+1}$, $c_v = g^{\alpha_{n+1} + e_v r_v}$. 我们知道 r_v 是由 Prover 选取的,他可以重新选择 r_v 使得 c_v 和 h_v 的值不变,这样由该结点向上的结点中的承诺值都不需要改变,只是该结点中的相应的 $(m_v, c_v, h_v, e_v, r_v)$ 发生了变化,而对于其它结点的计算没有任何影响. 这也就是我们前面介绍的对于虚结点的 e_v, r_v 的选择遵循一定规则的原因所在. 同样的,当一个群组以外的用户退出时,承诺值也不需要进行改变,只需要将 Merkle 树上代表该用户的叶结点变为空结点,重新选择和计算相应的值即可,对于其它结点不产生影响.

但是当被选定的大小为 m 的群组当中有成员离开或者有新的用户经过 Prover 筛选可以加入到群组中的时候, Prover 必须重新运行方案计算新的承诺. 因为此时对于 Merkle 树添加和删除的是实结点,这对于该结点以上直到根结点的承诺值都会产生影响,因此必须重新承诺. 还有一种情况当用户增加到大于 2^k 的时候,也就是说 Merkle 树没有多余的空结点来表示用户的时候, Prover 需要重新建立新的深度为 $k+1$ 的 Merkle 树并重新进行承诺.

4 安全性分析

这里我们简单分析一下上述方案的安全性:

秘密性 协议中传输的包含会话密钥 α^R 的信息只有被选中的用户(假设为 M_i)接收到的 α_i^{R}. 任意的未被选中的用户不知道相应的 r_i,因此他获得会话密钥的难度等价于求解离散对数问题. 由我们的前提假设离散对数问题是困难的,因此方案的秘密性可以保证.

认证性 基于我们的假设,方案中传输的关于计算会话密钥及验证承诺的消息都是通过一个安全通道实现的. 由安全通道的存在可以保证方案的认证性和数据完整性.

完备性 通过前面我们描述的承诺和验证的方案,很显然任意一个用户都可以有效的验证他是否是群组成员,并且被选中的群组成员可以实现安全的密钥分配,因此完备性可以保证.

零知识性 在该方案中任意一个成员都可以验证其是否为群组成员,但是对于其它被选中的成员的身份以及被选中成员的个数一无所知. 对于树结构中一个成员结点 v(叶结点)来说,在验证过程中路径 $P(v)$ 上的任意结点得到的其兄弟结点 u 信息只包括 c_u,他并不知道 h_u, m_u, e_u, r_u,因此无法判断该兄弟结点是实结点还是虚结点,也就无法得到关于兄弟分支上群组成员个数的情况. 因此零知识性可以保证.

合理性 如果 Prover 进行欺骗以达到被选中者验证结果表现为未被选中或者未被选中者被验证为群组成员,那么在构造叶结点 v 的承诺值时由于 e_v 是按照方案规定产生,为了构造 h_v 欺骗验证者 Prover 必须知道 $\log_g h$. 但是我们已经假设 $\log_g h$ 对于每一个用户和 Prover 来说都是不可知的,因此这种欺骗无法实现. 此外,如果 Prover 在验证过程中对用户提供的验证资料作假同时保证承诺值不变,那么或者他知道 $\log_g h$,或者他发现了哈希函数的一个碰撞. 由前提假设知这些都是计算上不可行的. 因此,合理性可以保证.

5 结论

本文设计了一个新的群组密钥分配方案,在群组密钥分配中引入了零知识集合的概念. 与一般的群组密钥交换方案相比,这种方案不仅可以对群组成员进行密钥的分配,而且可以同时实现群组成员的零知识性,也就是对于成员个数及身份进行隐藏. 这种具有零知识性的群组密钥分配协议,在网络通信中某些需要进行匿名群组密钥交换的领域具有一定的应用价值. 本文的重点在于为动态群组密钥交换协议加入零知识性,与一般类型的群组协议相比,效率较低,如何在保证更多安全属性的同时提供效率是我们进一步的工作. 同时,本文给出的这个方案仍然是集中式的密钥分配方案,如何实现其他拓扑结构的群组密钥交换协议的零知识性,以及如何实现密钥协商和零知识性的结合,也是我们下一步的研究工作.

参考文献:

[1] S Micali, M Rabin, J Kilian. Zero-Knowledge sets [A]. 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS '03)

[C]. IEEE Computer Society ,Cambridge ,2003 ,11 :80 - 91.

[2] T Pedersen. Non-interactive and information - theoretic secure verifiable secret sharing[J]. Leture Notes in Computer Science 576. London UK: Springer Verlag ,1991. 129 - 140.

[3] M Blum ,A De Santis ,S Micali ,G Persiano. Noninteractive zero-knowledge[J]. SIAM Journal of Compute. 1991 ,20(6) :1084 - 1118.

[4] S Halevi ,S Micali. Practical and provably-secure commitment schemes from collisionn-free hashing[A]. Proceedings of 16th international Advances in Cryptology Conference-Crypto '96[C]. LectureNotes in Computer Science ,vol. 1109 ,Santa Barbara ,California ,1996 :201 - 215.

[5] FIPS PUB 180 - 1 ,Secure Hash Standard. National Institute for Standards and Technology[S]. Gaithersburg ,MD ,USA ,April 1995.

[6] Y Kim ,A Perrig ,G Tsudik. Simple and fault-tolerant key agreement for dynamic collaborative groups[A]. Proceedings of 7th ACM Conference on Computer and Communications Security[C]. Athens ,Greece :Jajodia ,editor ,2000 ,11 :235 - 244.

作者简介:

孙海波　男,1977 年 3 月生于辽宁省盘锦市,中科院软件所博士生,研究方向为网络安全,安全协议分析. E-mail :hsun @is. isacs. ac. cn.

林东岱　男,1964 年出生,中科院软件所研究员,博士生导师,研究方向为密码学与网络安全.