一种基于密钥矩阵的动态秘密分享方案*)

颜 浩 陈克非
(上海交通大学计算机科学与工程系 上海200030)

A Dynamic Secret Sharing Scheme Based on Key Matrix

YAN Hao CHEN Ke-Fei
(Department of Computer Science and Engineering, Shanghai Jiaotong University, Shanghai 200030)

Abstract In this paper we introduce a dynamic secret sharing scheme. This scheme not only can be used unrestricted times to share and resume different secret without regenerating the information in the users hands, satisfy the (k,n) threshold demand, but also can realize the asymmetric user right secret sharing and dynamically add and delete the relevant users.

Keywords Secret sharing, Secret resuming, Symmetric encryption

1 引言

秘密分享是密码学中的一个很重要的研究方向，它要解决的问题是把一个秘密在多个人之间分享，单个的一个人无法得到秘密的内容，必须通过多个人的合作才能恢复秘密。

已有的秘密分享方案见文[1～8]。一般对秘密分享方案要求能够实现(k,n)门限，即秘密在n个人之间分享，其中k个人以上合作才可以恢复秘密[1～6]。也有方案是针对(n,n)情况的，即必须组中全体用户的合作才能恢复秘密[7,8]。文[1～3]的方案的一个缺点是秘密分享是一次性的：一次秘密恢复后，用户手中的信息就公开了，即无秘密可言了，下次在进行秘密分享必须重新生成用户手中的信息。文[4～8]的方案改进了这个问题，我们称之为动态秘密分享。一般来说，动态秘密分享要求可以在不改变用户们手中信息的条件下多次甚至是无限次地分享和恢复不同的秘密，而不用每次分享秘密都要通过秘密信道为每个用户分配子秘密。动态秘密分享为每个用户一次性分配固定的信息，每当进行一次秘密分享时，系统中心只是在公告板上公布一些仅和这次秘密分享有关的信息，用户们通过合作，利用公告信息和各自手中的信息来恢复秘密。而下次如果分享不同的秘密，仍然只是在公告板上公布信息，用户手中的信息不用改变。任意单个用户都不能从公告信息中得到秘密，也不能从恢复秘密过程中得到其他用户手中的信息，或是从一次秘密分享恢复中得到关于另一次秘密分享的秘密。

在一般的秘密分享机制中，各分享秘密的用户之间是对等的，在实际应用中，还可能存在这样的情况：用户所能恢复秘密的权限是不对称的，如A用户可以单独恢复秘密，而B和C合作，或B和D、E合作能恢复秘密，C、D、E合作不能恢复秘密等等。如果权限的规则比较复杂，上述方案要实现这种不对称权限秘密分享比较困难。

在动态秘密分享中，还可能要求可以动态地增加删除参与秘密恢复的用户，而不要重新为每个用户分发信息。

本文借用广播节目加密协议[9]中的密钥矩阵的概念，构造一种新的秘密分享方案。本方案不但可以实现动态的秘密分享，实现(k,n)门限的秘密分享，还可以实现不对称权限的秘密分享，并且可以动态地增加或删除参与秘密恢复的用户。

2 基于密钥矩阵的动态秘密分享方案

2.1 系统初始化

系统中心SC设两个参数r和t，构造一个大小为r×t的矩阵M，选择一种对称加密算法，矩阵的每个元素为一个随机选择的对称加密密钥k_{ij}，$i=1\cdots r$，$j=1\cdots t$，即总共有r×t个不同的密钥。可以看成有r行密钥集合，每个集合大小为t。

假设参与秘密恢复的用户群为$u_1,u_2,\cdots,u_n$。根据秘密恢复的要求为每一个用户u_i选择一个合适的集合$Set_{u_i}\subseteq\{1\cdots r\}$，只要对于任何一个可以恢复秘密的用户集合T(T中元素为这个用户集合中的用户的下标)，有$(\bigcup_{i\in T}Set_{u_i})\supseteq\{1\cdots r\}$，而对于任何一个不可以进行秘密恢复的用户集合T'(T'中元素为这个用户集合中的用户的下标)，有$\{1\cdots r\}\not\subset(\bigcup_{i\in T'}Set_{u_i})$，任意两个用户的$Set_u$可以有交集。然后，对于每一个$Set_{ui}$中的数字$m$，从矩阵$M$的第$m$行中随机选择一个密钥$k_{mj}$，这样为每个用户生成一个大小为$|Set_{ui}|$的密钥集合$K_{ui}$，分发给$u_i$保密。后面结合详细的例子来说明这一步骤的意义。

2.2 秘密分享

假设要分享的秘密为S，S是一个长为q的比特串，$S\in\{0,1\}^q$。系统中心随机选择$\{S_i\}_{i=1}^r$，满足$\oplus_{i=1}^r S_i=S$，即S等于S_i集合的异或操作。对每一个$S_i(i=1\cdots r)$，用密钥矩阵M的第i行的t个密钥分别进行对称加密，在公告板上公布所有的$r\times t$个加密数据。

2.3 秘密恢复

对于单个用户u，他只能解密K_u中对应行所在的若干个S_i，只要他的K_u不覆盖所有的r行，他就不能得到所有的S_i，从而不能恢复秘密。同理，对于一个用户集合u_T，只要他们手中K_u的并集不能覆盖所有的r行，他们就不能恢复秘密。反

*)本文由国家自然科学基金项目(No. 69973031, No. 90104005)，国家863项目(2001AA140437)资助。颜 浩 硕士研究生。陈克非 博士，教授，博士生导师。

之，如果一个用户集合手中 K_w 的并集覆盖了所有的 r 行，那么他们每人分别解密自己所能知道的几个 S_i，用户集合中所有得到的 S_i 必然覆盖了整个$\{S_i\}_{i=1}^{t}$，从而通过运算 $S=\oplus_{i=1}^{t} S_i$ 得到秘密。

2.4 最简单的例子

考虑最简单的情况，两个用户 u_1 和 u_2，取 $r=2,t=1$，系统初始化的密钥矩阵为$\{k_1,k_2\}$，把 k_1 分给 u_1，k_2 分给 u_2。

对于秘密 S，系统中心随机选择 S_1 和 S_2 使 $S=S_1\oplus S_2$，用 k_1 对 S_1 加密，k_2 对 S_2 加密，并公布加密值，因为 u_1 不能解密 S_2，u_2 不能解密 S_1，所以他们各自都无法单独得到秘密，只有 u_1 解密 S_1，u_2 解密 S_2 再合作后才能得到秘密 S。秘密分享恢复的运算是非常简单的。

3 (k,n)门限的秘密分享

设 k=2,n=3，即有3个用户，任意2个以上用户的合作才能恢复秘密。取 r=3,t=1，随机选择3个密钥 k_1、k_2、k_3，将 $\{k_2,k_3\}$给 u_1，$\{k_1,k_3\}$给 u_2，$\{k_1,k_2\}$给 u_3。这样，任意两个以上用户都可以合作得到秘密，而单独一个用户无法得到秘密。同理对于更大的 k 和 n，只是增加 r 的值并合理分配密钥集合给每个用户就可以实现(k,n)门限的秘密分享。

对于一般的(k,n)情况的密钥的分配，我们把它转化为这样的数学问题(t 都取1)：对于正整数 n 和 k,k≤n。要求确定 r 的数值(r 即为 S 被分成的子秘密数量或者说是密钥矩阵的行数)，构造一个 n×r 的分配矩阵 M(k,n)(不是第2节所说的密钥矩阵，之所以称为分配矩阵是因为这个矩阵表明了如何为 n 个用户分配集合{1,2,…,r}的一个子集)。矩阵的每一行代表 n 个用户中的一个，每一列代表 r 个密钥中的一个。矩阵的每个行列交点(i,j)，如果等于1表示第 j 个密钥被分配给第 i 个用户，等于0表示第 j 个密钥不配给第 i 个用户。这样第 i 行等于1的位置的集合就相当于前面第2节所说的 Set_w。

分配矩阵需要满足如下性质：

(1)任意 k 行向量的行或运算后等于一个1×r 的全1行向量。(即覆盖{1…r}这个集合)

(2)任意 k−1行向量的行或运算后的1×r 的行向量中至少包含一个0。

对于本节开始 k=2,n=3的例子，其分配矩阵 M(2,3)为：

$$\begin{bmatrix}0&1&1\\1&0&1\\1&1&0\end{bmatrix}$$

为了构造任意(k,n)的分配矩阵 M(k,n)，我们在这里给出一个递归的构造方法。对于某个特定的(k,n)，这种方法构造的分配矩阵在空间复杂度上未必是最好的，但是我们借此来说明这种构造的可行性。

首先证明 M(1,n)(只要一个人就可以恢复秘密)、M(2,n)和 M(n,n)(需要全体 n 个人来恢复秘密)的存在性：显然对于 M(1,n)使用一个 n×1的全1的列向量(即 r=1)，对于 M(n,n)使用一个 n×n 的单位矩阵(即 r=n)就可以满足要求(一般地，也可以是任意置换矩阵)。对于 M(2,n)我们使用如下形式的分配矩阵(n×n)：

$$\begin{bmatrix}0&1&\cdots&1\\1&0&\ddots&\vdots\\\vdots&\ddots&0&1\\1&\cdots&1&0\end{bmatrix}$$，即单位矩阵每一个元素 I_{ij} 都以 $1-I_{ij}$ 代替而形成的矩阵。

现在假设我们已经知道 M(k,n−1)(n−1×r 维，k>2,k≤n−1)的构造方法，并且，以此为基础来构造 M(k,n)。

M(k,n)=

$$\begin{bmatrix}M(k,n-1) & \overbrace{0\ldots0}^{C_{n-2}^{k-3}} & \overbrace{1\ldots1}^{r}\\ & D(k,n) & \\ \underbrace{1\ldots1}_{r} & \underbrace{0\ldots0}_{C_{n-2}^{k-3}} & M(k,n-1)^T\end{bmatrix},$$

(n,2*r+C_{n-2}^{k-3})维。

其中 D(k,n)是一个(n−2,C_{n-2}^{k-3})维的矩阵，这个矩阵的每一列有 k−3个0，其余 n−k+1个为1，每一列0的位置是 n−2中取 k−3的一种组合，而 C_{n-2}^{k-3}列使 k−3个0的位置遍历了可取的位置组合。如$D(4,5)=\begin{bmatrix}0&1&1\\1&0&1\\1&1&0\end{bmatrix}$，

$$D(5,6)=\begin{bmatrix}0&1&1&0&1&0\\0&0&1&1&0&1\\1&0&0&0&1&1\\1&1&0&1&0&0\end{bmatrix}.$$

定理1 所构造的 M(k,n)仍然满足性质(1)和(2)。

以 M(k,n)最中间的第 r+1列到第 r+C_{n-2}^{k-3}列为分界线，我们称左面 r 列为左半矩阵，右面 r 列为右半矩阵。

(1)在 M(k,n)中任意取 k 行，证这 k 行进行行或运算后等于一个2r+C_{n-2}^{k-3}列的全1行向量。因为矩阵是中心对称的，所以只需讨论左半矩阵，分两种情况：1)包含第 n 行(1到 r 列全1)，那么这 k 行在左半矩阵的 r 列进行行或操作后必为1×r 列全1行向量；2)不包含第 n 行，则这 k 行在左半矩阵的 r 列都在 M(k,n−1)中取得，由于 M(k,n−1)满足性质(1)，这 k 行在左半矩阵的 r 列进行行或操作后仍然为1×r 列全1行向量。所以这 k 行在左半矩阵的 r 列进行行或操作后必为1×r 列全1行向量，同理右半矩阵的 r 列进行行或操作后也为1×r 列全1行向量。而中间第 r+1列到第 r+C_{n-2}^{k-3}列每一列只有 k−1个0，取 k(k>2)行后至少取到一个1，这样进行行或操作后也得到一个1×C_{n-2}^{k-3}维全1行向量。由1)和2)得证在 M(k,n)中任意取 k 行后，进行行或运算后等于一个2r+C_{n-2}^{k-3}列的全1行向量。满足性质(1)。

(2)在 M(k,n)中任意取 k−1行，证这 k−1行进行行或运算后的向量中至少包含一个0。分两种情况讨论，1)如果这 k−1行同时包含第1行和第 n 行，那么就将在 D(k,n)中取到 k−3行。而 D(k,n)中每列有 k−3个0，并且 D(k,n)中所有列构成 k−3个0在位置上的组合，所以这 k−3行必在 D(k,n)中取到一列为 k−3个0，设这一列为 M(k,n)中的第 p 列(C_{n-2}^{k-3}≥p>r)，在加上第 p 列的第1行和第 n 行都为0，所以这 k−1行在第 p 列取到的元素为全0，即在进行行或运算后为0。2)如果这 k−1行不同时包含第1行和第 n 行，假设不包括第 n 行，那么这 k−1行在左半矩阵所取的 r 列都在 M(k,n−1)中取得，由于 M(k,n−1)满足性质(2)，所以这 k−1行在左半矩阵所取的 r 列进行行或运算后的向量中至少包含一个0。同理如果是不包括第1行，则会在右半矩阵产生至少一个0。由1)和2)得证在 M(k,n)中任意取 k−1行，这 k−1行进行行或运算后

的向量中至少包含一个0。满足性质(2)。

至此我们得到了从M(k,n-1)(k>2)构造M(k,n)的方法并证明了正确性，因为M(k,k)是已知的，所以由此出发我们可以构造任意(k,n)的分配矩阵M(k,n)，继而实现任意(k,n)门限的秘密分享。

4 更一般的秘密分享问题

4.1 不对称权限的秘密分享

假设有这样5个用户，A、B、C、D、E。A是老板，权利最大，可以单独恢复秘密；B是老板助理，虽然不能单独恢复秘密，但是只要和另外任何一个人合作就可以恢复秘密；C、D、E级别最低，并且同级，他们3个一起合作才能恢复秘密，当然，他们3个任何一个和A或B中的一个合作也能恢复秘密。

这里取r=4，t=1，密钥分配为SetA={1,2,3,4}，SetB={1,2,3}，SetC={1,4}，SetD={2,4}，SetE={3,4}。如此分配即可符合前面的不对称权限要求。比如C、D、E的三个密钥集合的并集可以覆盖所有4行，而他们任意两个的密钥集合都不能恢复秘密。

对于如何对一般性的不对称权限的秘密分享进行构造是比较困难的，本文仅给出对于不对称权限的一种描述方法，对于如何一般化地构造还是有待于进一步解决的问题。

对于不对称权限的描述即是指出哪些用户进行合作可以恢复秘密，Stinson[10]提出了使用访问结构(Access structure)来描述一般性的不对称权限的秘密分享，每一个允许恢复秘密的用户组合称为一个授权子集，访问结构即为所有授权子集的集合，而构造访问结构可以由所有最小授权子集的集合的闭包形成。

对于本方案来说，由于为每个用户分配信息关键在于确定每个用户u_i的Set_{u_i}，因此我们认为从单个用户的角度来描述访问结构可能更利于分配方法的一般化构造，所以对于每个用户X，建立一个集合SS_X，集合中每个元素都是一个用户集合S_X，这个用户集合S_X中的所有用户同X一起合作应能够恢复秘密，而每个这样的集合都应该是最小集合，即不存在$S'_X \subset S_X$，而S'_X也能和X进行秘密恢复。其中有两个特例，如果用户X可以单独恢复秘密，则$SS_X=\{\{X\}\}$，而如果X不能和任何用户集合合作恢复秘密也不能单独恢复秘密，则$SS_X=\phi$。

这样对于上面的例子每个用户的SS_X如下：

A：{{A}}

B：{{A},{C},{D},{E}}

C：{{A},{B},{D,E}}

D：{{A},{B},{C,E}}

E：{{A},{B},{C,D}}

而这个例子用Stinson的访问结构来描述，则其最小授权子集集合(即访问结构的基)为{{A},{B,C},{B,D},{B,E},{C,D,E}}。

事实上，本方案对用户权限的描述方法和Stinson的访问结构是等价的。从本方案的SS_X出发，把每个用户的符号加入其对应SS_X中的所有S_X中，在把所生成的所有用户的S_X合并为一个集合SS'_X，删除重复的S_X，并对于如果存在形如{X}的S_X，则删除以下S_X：$\{S_x | X \in S_x, |S_x| \geq 2\}$，最后得到的$SS'_X$就是对应的Stinson访问结构中的最小授权子集集合(访问结构的基)。同样，从Stinson访问结构的基出发，对于某个用户X，抽取出包含X的所有最小授权子集以及所有只有一个元素的最小授权子集，去除这些最小授权子集中的符号X，也能够得到本方案的所有SS_X。

4.2 可以增加删除用户的秘密分享

前面的几个例子中一旦某个用户得到一个密钥集合，他就可以一直参与秘密的恢复。而实际当中可能需要删除某个用户的秘密恢复权限，如一个职员离职后需要取消他的秘密知情权。我们的方案可以通过设定t的值来实现删除用户。

仍然以前面(2,3)门限为例。现在我们取r=3，t=2，即需要选择6个密钥$\begin{bmatrix} k_{1,1} & k_{1,2} \\ k_{2,1} & k_{2,2} \\ k_{3,1} & k_{3,2} \end{bmatrix}$(密钥矩阵)，将$\{k_{1,1},k_{2,1}\}$给$u_1$，$\{k_{2,2},k_{3,2}\}$给$u_2$，$\{k_{3,1},k_{1,2}\}$给$u_3$。这样虽然任意两个用户的$Set_u$都有交集，但是他们的$K_u$无交集。当需要删除某个用户如$u_2$时，只要系统中心分享秘密时不再用$k_{2,2},k_{3,2}$进行加密或将这两个密钥用新的密钥替换掉即可使$u_2$失效，以后生成的秘密分享$u_2$都不能参与恢复，而$u_1$和$u_3$的功能不受影响。如果要进一步保证在取消$u_2$权限之前的秘密也对$u_2$失效，则可以重新对这些秘密生成$\{S_i\}$集合，并用新的密钥矩阵加密并更新公告栏上的信息。如果将来有一个新用户u_4需要担任u_2的角色，则只要将新的$k'_{2,2},k'_{3,2}$分发给他即可。

这种做法比较节省空间，但是如果用户的增加删除比较频繁，则对于管理就非常复杂。这时可以取t=n，即为每个用户对应密钥矩阵中的一列(不是把整个列都发给用户，而只是取其中的一个子集)，列中没有分配给这个用户的密钥则不使用。这样做对于用户的增加删除则是对密钥矩阵列的增加删除，管理方便，当然需要的空间就增加了，因为每一列都有不用的元素。

可见，通过设定合适的t值，只要分发给每个用户的K_u无交集，即可实现动态增删用户的秘密分享。

5 新方案的安全性、特点、性能分析和同其他方案的比较

5.1 安全性

1. 由于秘密S需要所有子秘密S_i的异或操作才能恢复，而一个没有全部r行密钥的用户不能解密出所有的子秘密，从而保证了秘密的分享。这里有一种小概率的例外不安全情况，因为$\{S_i\}_{i=1}^{r}$是随机选择的，所以存在这样的可能：对某个S的分享过程中，不但所有r个S_i的异或等于S，S_i的一个子集的异或也等于S。即存在r'<r，使$\oplus_{i=1}^{r'} S_i = S$成立(或者说$\oplus_{i=r'+1}^{r} S_i = \{0\}^q$成立)。这样从表面上看，就存在可能不需要所有r个$S_i$也能恢复秘密S。但是仔细分析，即使有一个用户集合得到了前r'个S_i，由于他们还不知道S和$\oplus_{i=r'+1}^{r} S_i \{0\}^q$，所以他们无法猜测$\oplus_{i=1}^{r} S_i = S$，而如果他们知道了$\oplus_{i=r'+1}^{r} S_i = \{0\}^q$，也就相当于他们已经得到了所有的$S_i$，也就理所当然可以恢复S。而且，由于每个$S_i$的q个位都是随机选择的，随着q的增长，出现在这种不安全情况的概率也会指数级地减小(q个比特全零的概率同q成负指数关系)。当然，为了一定要避免单个用户尝试用自己可以知道的S_i来当作S进行攻击，系统中心可以在分享S时对r个S_i进行排列，并检测每个排列的前r/2个S_i的异或是否等于S或等于全零。如果是，则重新随机选择S_i并重新检测，直到消除这种不安全情况。(由于这种不安全情况出现的概率是负指数级的，所以需要重新选择的概率也是微乎其微的)。

(下转第175页)

增多而提高，因为节点数越多，并发性越高。

以当前节点数为6对不同大小的文件进行读写试验，测试结果如图8，可以看出 FTDSS 读写性能比单机读写性能要高，并且文件越大，读写性能增幅越大，说明并发读写对大文件更有利。

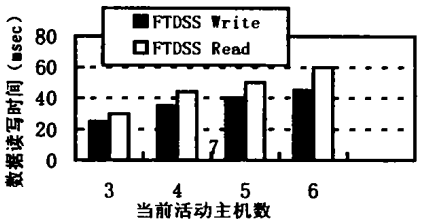

图7 FTDSS 读写随当前活动主机数的变化情况

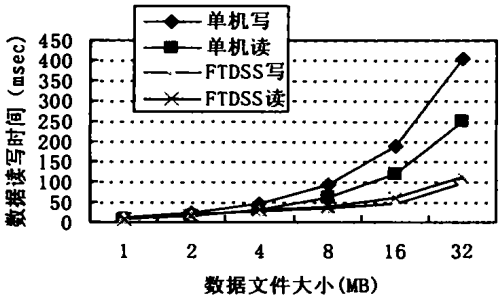

图8 FTDSS 的读写随文件大小的变化情况

结束语 本文提出了基于网络的分布式容错共享存储 FTDSS。FTDSS 将网络中各节点的磁盘空间组织成全局共享的存储空间，FTDSS 将文件以数据分片和冗余的校验分片形式分布式存储，只要一定数目的节点在线，就可以从全局空间完整读出文件，获得高的容错性，FTDSS 通过网络存储实现了数据远程容灾，FTDSS 将文件分片并行读写，获得比单机高的读写性能。因此，FTDSS 有很大的实用价值，FTDSS 可以在不添加任何硬件设备的基础上，在网络中实现高容错的共享存储。

参 考 文 献

1 Plank J S. A tutorial for Fault-Tolerate in RAID-like systems: [Technical Report UT-CS-96-332]. Department of Computer Science University of Tennessee, July 1996. 10～22

2 Mplero X, Silla F, Santonia V. Performance analysis of storage area networks using high-speed LAN interconnects. In: Proc. of the 8th ISPAN Conf. IEEE Computer Society. Sept. 2000. 5～9

3 Atchley S, et al. Fault-Tolerance in the Network Storage Stack. In: Proc. of the FAST 2002 Conf. Monterey, California, USA, Jan. 2002. 3～5

4 Tierney B L, Lee J, Crowley B, Holding M. A Network-Aware Distributed Storage Cache for Data Intensive Environments. In: Proc. of the FAST 2002 Conf. Monterey, California, USA, Jan. 2002. 15～18

5 Sobti S, et al. PersonalRAID: Mobile Storage for distributed and disconnected computers. In: Proc. of the FAST 2002 Conf. Monterey, California, USA, Jan. 2002. 5～8

6 Gibson G. Cost-effective high-bandwidth storage architecture. In: Proc. of ACM ASPLOS, Oct. 1998. 4～5

（上接第58页）

2. 秘密恢复时，一个用户只能知道其他合作用户的 Set_u，而不能知道其他人的 K_u，因此一个用户不能从一次秘密恢复中得到关于其他人手中的密钥信息，也不能从一次秘密恢复中得到有关另一次秘密分享中的秘密。

5.2 特点

1. 每个用户手中持有的是固定的 Set_u 和 K_u，因此进行多次秘密分享时不需要重新生成用户手中的信息，从而实现了动态的秘密分享。

2. 方案可以实现(k,n)门限的秘密分享，并且对最常见的(n,n)门限的情况的实现非常简单(使用单位矩阵形式的分配矩阵)。

3. 提出了一种同 Stinson 访问结构等价的，但更适合本方案的描述一般性不对称权限秘密分享的方法。

4. 方案可以对参与秘密恢复的用户动态地增加和删除，而不影响其他用户的信息。

5.3 性能分析

算法的时空复杂度是线性的，在秘密分享阶段需要 O(r×t)的时间和空间要求，在秘密恢复阶段每个用户最多进行 r 次解密运算，而恢复秘密只进行 r 次异或运算。

5.4 和其他秘密分享方案的比较

1. 文[1～3]的方案是一次性的秘密分享，本文的方案是可以反复使用而不需要更改用户手中的信息的，即所谓动态秘密分享。

2. 文[4～6]的方案可以实现(k,n)门限秘密分享，但是进行(n,n)形式是太过复杂，而本文的方案在(n,n)的情况下非常简单，使用的是单位矩阵型的分配矩阵(见第4节的分析)。

3. 文[7～8]的方案可以实现动态秘密分享，但是只适合于(n,n)形式，本文的方案可以同时满足这两种要求，并且文[7]中的幂指数运算计算量太大，而本文使用异或操作进行秘密的分享和恢复，计算量更易接受。

结论 本文提出的基于密钥矩阵的秘密分享方案，不但实现了秘密分享中各种基本要求，包括动态秘密分享，(k,n)门限，还提出并实现了不对称权限的秘密分享，可以动态增加删除参与秘密恢复的用户，并且要求比较小的时空复杂度。根据实际应用的需要，可以设计出更满足某种需要的秘密分享方案。

参 考 文 献

1 Shamir A. How to share a secret. Comm. ACM, 1979, 22(11): 612～613

2 Karnin E D, Greene J W. On secret sharing systems. IEEE Trans Inform Theory, 1983, IT-29(1): 35～41

3 Tompa M, Woll H. How to share a secret with cheaters. J Crypto, 1988, 1: 133～138

4 Tompa M, Blakley G R. Threshold schemes with disenrollment. In: Proc. Crypto 1992, Santa Barbara, CA, Aug. 1992. 13-1-13-4

5 He J, Dawson E. Multistage secret sharing based on one-way function. Electron Lett, 1994, 30(19): 1591～1592

6 Sun H M, Shieh S P. Construction of dynamic threshold schemes. Electron Lett, 1994, 30(24): 2023～2024

7 Pinch R G E. Online multiple secret sharing. Electron Letts, 1996, 32(12): 1087～1088

8 谭凯军，诸鸿文. 基于单向函数的动态秘密分享机制. 通信学报，1999. 7

9 Chor B, Fiat A, Naor M. Tracing Traitors. In: Proc. Advances in Cryptology-Crypto '94, Springr-Verlag LNCS 839, 1994. 257～270

10 Stinson D R. Cryptography: Theory and Practice, CRC Press, Boca Raton, 1995