Invited Survey Paper

A Survey of Key-to-Key User Hierarchic Representation Mechanisms

CHIN-CHEN CHANG*

This paper discusses some methods of key-to-key user hierarchic representation. For each mechanism, we describe (1) a model for determining the relationship between two users, (2) the memory storage required for keys, and (3) the construction of keys. The common drawbacks of the key-to-key user hierarchic representation strategy are also described. Despite great progress, three important problems remain open.

1. Introduction

Various methods have been reported for implementing user hierarchy structure in information protection systems [Graham and Denning 1972, Saltzer 1974, Saltzer and Schroeder 1975, Wu and Hwang 1984, Chang and Chen 1985, Chang and Hung 1986, Chang 1987, Chang and Chang 1989]. Graham and Denning [1972] proposed a subject tree hierarchy, Saltzer [1974] presented the access-controller hierarchy of the accessor-list system, and Saltzer and Schroeder [1975] introduced a MULTICS protection ring-a linear hierarchy of access privileges. They are beyond the scope of this paper.

Here, we shall review all of the work relevant to key-to-key user-hierarchic representation done by other researchers. By a user-hierarchy, we mean a hierarchy structure that represents the seniority of users in a protection system.

A simple example of user-hierarchy is shown in Fig. 1, where users are represented by their *ID* numbers.

In both Fig. 1 and Fig. 2, we see the relations among four users. These relations allow user 2 to read the files constructed by user 4, since user 2's position of "father" is superior to that of user 4. Because user 4 is the son of user 2, a request from user 2 to change his privilege with respect to user 4's files from execute to read will be accepted. If h_{ij}, the relation of user i to user j, is "Father of," then we can conclude that h_{ji} means "Son of." Thus, only the lower triangle is sufficient to represent the whole matrix. Throughout this paper, we shall use a lower triangle matrix to represent a user hierarchy matrix system. We call this lower triangle matrix the H-matrix. The H-matrix corresponding to Fig. 2 is as follows:

Since H-matrices are always very sparse and easy to expose, there have been many reports of key-to-key user hierarchic mechanisms for implementing them. A key-to-key user hierarchic mechanism can also be seen as a mechanism that encodes columns of H-matrices in such a way that decoding would require a knowledge of keys.

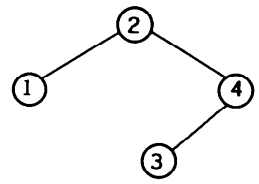

Fig. 1

user i \ user j	1	2	3	4
1	0	4	1	2
2	3	0	5	3
3	1	6	0	4
4	2	4	3	0

0: Don't care
1: No relation
2: Brother
3: Father of
4: Son of
5: Grandfather of
6: Grandson of

Fig. 2 A matrix structure.

For those unfamiliar with the term, a key-to-key user hierarchic mechanism assigns a set of keys to each user in such a way that an operation on any two keys from two users would reveal the relationship of the two key owners. The "relation" information can be used in deciding how to handle requests to change access attributes.

The organization of the paper is as follows: In Section 2, we shall discuss the key-to-key method, which is based on the Galois field algebra devised by Wu and

*Institute of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan 62107, R.O.C.

$$H=\begin{bmatrix}0&0&0&0\\3&0&0&0\\1&6&0&0\\2&4&3&0\end{bmatrix}$$

Fig. 3

Hwang [1984]. The work done by Chang and Chen [1988], which relates to the generalized Chinese remainder theorem, is discussed in Section 3. In Section 4, we shall introduce the key-to-key method proposed by Chang and Hung [1986], which is based on Euler's theorem. The key-to-key method based on the Chinese remainder theorem [Chang 1987] will be discussed in Section 5. A single-key user hierarchic representation scheme proposed by Chang and Chang [1989] will be discussed in Section 6. Finally, our conclusions are stated in Section 7.

2. Key-to-Key User Hierarchic Representation Mechanism Based on Galois Field Algebra

In this section, we shall introduce a very important concept proposed by Wu and Hwang [1982]. They considered the following problem: Given an H-matrix, assign a vector key to each user in such a way that a multiplication operation on any two keys will reveal the relationship of the two key owners.

Example 2.1

Consider an H-matrix with four users, as shown in Fig. 3. The original user hierarchy structure is as shown in Fig. 2. In order to find all the users' keys, a key matrix is first constructed

$$K=\begin{pmatrix}k_{11}&k_{12}&k_{13}&k_{14}\\k_{21}&k_{22}&k_{23}&k_{24}\\k_{31}&k_{32}&k_{33}&k_{34}\\k_{41}&k_{42}&k_{43}&k_{44}\end{pmatrix}$$

such that $K_i * K_j = h_{ij}$, where $1 \le j < i \le 4$, K_i represents the i-th row vector of K and $*$ is the inner product computation over $GP(7)$, where 7 is the smallest prime number larger than all elements in the H-matrix.

By reserving six variables in the lower triangle, we can arbitrarily assign values to other elements in matrix K. We have

$$K=\begin{pmatrix}3&1&5&1\\k_{21}&4&2&3\\k_{31}&k_{32}&1&6\\k_{41}&k_{42}&k_{43}&1\end{pmatrix}.$$

Thus, all keys can be obtained by solving the following six equations:

$$k_{21}\times 3+4\times 1+2\times 5+3\times 1=h_{21}=3,$$
$$k_{31}\times 3+k_{32}\times 1+1\times 5+6\times 1=h_{31}=1,$$
$$k_{31}\times k_{21}+k_{32}\times 4+1\times 2+6\times 3=h_{32}=6,$$
$$k_{41}\times 3+k_{42}\times 1+k_{43}\times 5+1\times 1=h_{41}=2,$$
$$k_{41}\times k_{21}+k_{42}\times 4+k_{43}\times 2+1\times 3=h_{42}=4,$$

and

$$k_{41}\times k_{31}+k_{42}\times k_{32}+k_{43}\times 1+1\times 6=h_{43}=3.$$

In the Galois field $GF(7)$, all the user key values are found:

user	key value
1	(3, 1, 5, 1)
2	(0, 4, 2, 3)
3	(6, 0, 1, 6)
4	(4, 5, 1, 1)

Fig. 4 Users and key values.

Now, if user 4 requests permission to update a file constructed by user 2, the correspondence relationship can be computed as $K_4 * K_2 = (4, 5, 1, 1) * (0, 4, 2, 3) = 4 = h_{42}$. On the other hand, if we want to know that relation between user 1 and user 4, we have to compute the relation for user 4 and user 1 first, and then invert this relation. Thus we have $K_4 * K_1 = (4, 5, 1, 1) * (3, 1, 5, 1) = 2$, and the inverse of "2" = 2, which gives the relation "brother of."

Here we observe that the length of each key is $O(n)$, where n is the total number of users. Thus, the size of the required storage of keys actually exceeds that of the original H-matrix.

3. Key-to-Key User Hierarchic Representation Mechanism Based on the Generalized Chinese Remainder Theorem

The key-to-key user hierarchic representation mechanism based on the generalized Chinese remainder theorem was proposed by Chang and Chen [1985].

Theorem 1: Generalized Chinese Remainder Theorem

Let $r_1, r_2, \cdots, r_t$ be t positive integers and $m_1, m_2, \cdots, m_t$ be t positive integers. If $(m_i, m_j) \mid r_i - r_j$, $i \ne j$ and $1 \le i, j \le t$, then the system of equations $C \equiv r_i \pmod{m_i}$, $i = 1, 2, \cdots, t$ has a common solution C. Again, let n be the number of users. Chang and chen's method was inspired by Wu and Hwang's approach. They proposed three modifications:

1. They proposed the following formula for h_{ij}:

$$h_{ij}=\begin{cases}\left\lfloor \dfrac{K_i}{p_j} \right\rfloor \bmod n, & \text{if } i>j;\\ Inverse\left(\left\lfloor \dfrac{K_j}{p_j} \right\rfloor \bmod n\right) & \text{otherwise}\end{cases}$$

instead of

$$h_{ij}=\begin{cases}K_i * K_j \text{ over } GF(p), & \text{if } i>j;\\ Inverse(K_j * K_i) \text{ over } GF(p), & \text{otherwise}\end{cases}$$

as proposed by Wu and Hwang [1984]. Here $\lfloor x \rfloor$

denotes the largest integer equal to or less than x.

2. They proposed a simpler method of calculating all K_i's.

3. They proposed a simple method of assigning p_j's to all users [respectively].

Let (K_i, p_i) be a key pair of user i and let h_{ij} be the (i, j)th element of an H-matrix. Let the number of users be n. Chang and Chen's method can be described as follows:

(1) Choose n coprime integers $p_1, p_2, \cdots, p_n$ for n users, respectively, where $\min\{p_i\}_{i=1,2,\cdots,n} > n > \max\{h_{ij}\}_{1\le i,j\le n}$.

(2) Compute $K_i = (\sum_{j=1}^{n} b_j \cdot n \cdot (\Pi_{l\ne j} p_l) \cdot N_j) \bmod (n \cdot \Pi_{l=1}^{n} p_l)$ such that $h_{ij} = \lfloor K_i/p_j \rfloor \bmod n$, where b_j satisfies $(n \cdot \Pi_{l\ne j} p_l) \cdot b_j \equiv n \pmod{n \cdot p_j}$ and $N_j = \lceil h_{ij} \cdot p_j/n \rceil$, $i = 1, 2, \cdots, n$. Here $\lceil x \rceil$ denotes the smallest integer equal to or greater than x.

(3) Associate each (K_i, p_i) with user U_i, where $i = 1, 2, \cdots, n$.

(4) The relation between user U_i and user U_j is

$$h_{ij} = \begin{cases} \left\lfloor \dfrac{K_i}{p_j} \right\rfloor \bmod n, & \text{if } i > j; \\ \textit{Inverse}\left(\left\lfloor \dfrac{K_j}{p_i} \right\rfloor \bmod n \right), & \text{otherwise.} \end{cases}$$

Theoretically, in Chang and Chen's method, $O(n)$ locations are required to store all key values if the "overflow" problem is ignored. The number of divisions is needed only twice, since we compute the relation between two users. The K_i's of Chang and Chen's method come from the integer domain of $[0, (n \cdot \Pi_{l=1}^{n} p_l) - 1]$. Therefore, when n is large, each K_i will grow very rapidly.

4. Key-to-Key User Hierarchic Representation Mechanism Based on Euler's Theorem

Chang and Hung [1986] proposed a method based on Euler's theorem to represent a user hierarchy structure.

Assume that there are n users and that h_{ij} is the (i, j) the attribute of an H-matrix. Let $p(x)$ denote a coprime numbers transformation function that transforms the set $\{1, 2, \cdots, n\}$ to another set of n coprime numbers. Let $p(j)$ be greater than h_{ij} for $1 \le i, j \le n$. Let $(K_i, p(i))$ be a key pair of user i. The proposed relation of user i to user j is

$$h_{ij} = \begin{cases} K_i \bmod p(j), & \text{for } 1 \le j < i \le n; \\ \textit{Inverse}(K_j \bmod p(i)), & \text{for } 1 \le i < j \le n. \end{cases}$$

Chang and Hung's method can be described as follows:

(1) Compute a coprime numbers transformation function $p(x) = Dx + E$ by using the algorithm, called Algorithm DE, proposed by Jaeschke [1981] with the set $\{1, 2, \cdots, n\}$ as its input to find two appropriate integers D and E.

(2) Compute

$$K_i = \left(\sum_{j=1}^{n} h_{ij} \left(\prod_{\substack{i=1 \\ i\ne j}}^{n} p(i)^{\phi(p(j))} \right) \right) \bmod \left(\prod_{j=1}^{n} p(j) \right), \quad i = 1, 2, \cdots, n,$$

where $\phi(x)$ means the number of positive integers less than x that are relatively prime to x

(3) Associate each $(K_i, p(i))$ to user U_i, $i = 1, 2, \cdots, n$.

(4) The relation of user i to user j is

$$h_{ij} = \begin{cases} K_i \bmod p(j), & \text{if } 1 \le j < i \le n; \\ \textit{Inverse}\,(K_j \bmod p(i)), & \text{otherwise.} \end{cases}$$

The keys of this method come from the integer domain of $\{x \mid 0 \le x \le \Pi_{j=1}^{n} (p(j) - 1$, where $p(j)$'s are coprime integers and x is an integer$\}$. Therefore, we see that each K_i will inevitably be large when n becomes large.

5. Key-to-Key User Hierarchic Representation Mechanism Based on the Chinese Remainder Theorem

The key-to-key user hierarchic representation mechanism based on the Chinese remainder theorem was proposed by Chang [1987]. Let K_i be the key of user i and let h_{ij} be the (i, j)th element of an H-matrix. Let (K_i, p_i) be a key pair of user i and let the number of users be n. The relation of user i to user j proposed by Chang [1987] is

$$h_{ij} = \begin{cases} K_i \bmod p_j, & \text{if } i > j; \\ \textit{Inverse}(K_j \bmod p_i), & \text{otherwise.} \end{cases}$$

His method used the Chinese remainder theorem, and can be stated as follows:

(1) Choose n coprime integers $p_1, p_2, \cdots, p_n$ for n users, respectively, where $p_j > \max\{h_{ij}\}_{i=1,2,\cdots,n; j=1,2,\cdots,n}$.

(2) Compute $K_i = (\sum_{j=1}^{m} b_j D_j h_{ij}) \bmod (\Pi_{j=1}^{n} p_j)$, where $D_j = \Pi_{i\ne j} p_i$ and b_j satisfies $(\Pi_{i\ne j} p_i) b_j \equiv 1 \pmod{p_j}$, where $j = 1, 2, \cdots, n$.

(3) Associate each $(K_i, p(i))$ with user U_i, where $i = 1, 2, \cdots, n$.

(4) The relation of user i to user j is

$$h_{ij} = \begin{cases} K_i \bmod p(j), & \text{if } 1 \le j < i \le n; \\ \textit{Inverse}(K_j \bmod p(i)), & \text{otherwise.} \end{cases}$$

Example 5.1

Consider the H-matrix in fig. 3 again. Let $p_1 = 7$, $p_2 = 8$, $p_3 = 9$, and $p_4 = 11$. $K_1 = 0$, $K_2 = 2376$, $K_3 = 4950$, and $K_4 = 660$ satisfy $h_{ij} = K_i \bmod p_j$, for $1 \le j < i \le 4$.

Let us consider "h_{32}". K_3 and p_2 are selected. $h_{32} = K_3 \bmod p_2 = 4950 \bmod 8 = 6$, which is correct.

Though each key value of Chang's method will not exceed the multiplication of all coprime integers used, some keys will inevitably be large when the total number of users becomes large.

6. Single Key User Hierarchic Representation Scheme

Recently, Chang and Chang [1989] considered a scheme in which each user is associated with a key and an *ID* number. Through operations on the key of one user and the *ID* number of another, the relation between two users can be found.

Their method was inspired by Chang's method [1987], for which they proposed two modifications:

1. They proposed a simpler method for obtaining the relation between two different users by using the key of one user and the *ID* number of another.
2. They used a very straightforward formula to find the key values. No recursive calculation is needed in their formula as it is in Chang's.

In the single key user hierarchic representation scheme, Chang and Chang computed h_{ij}, the relation of user i to user j, as $\lceil K_i/t^{j-1}\rceil \bmod t$ if $i>j$ and $Inverse(\lceil K_j/t^{i-1}\rceil \bmod t)$ otherwise, where K_i represents the key of user i and t is the total number of distinct relations. That is, given an H-matrix, they computed K_i's for n users so that h_{ij} can be found by $\lceil K_i/t^{j-1}\rceil \bmod t$ in case of $i>j$. A straightforward formula $K_i=\sum_{s=1}^{i-1} h_{is}t^{s-1}-\sum_{s=1}^{i-2} t^s$ was also presented by Chang and Chang for computing each key value.

Example 6.1

Consider the H-matrix as depicted in Fig. 3 again. Chang and Chang's mechanism would first compute all the keys by using the following formula

$$K_i=\sum_{s=1}^{i-1} h_{is}t^{s-1}-\sum_{s=1}^{i-2} t^s$$
$$=\sum_{s=1}^{i-1} h_{is}\cdot 7^{s-1}-\sum_{s=1}^{i-2} 7^s.$$

Thus, $K_2=3$, $K_3=36$, and $K_4=121$. Note that user 1 has the smallest user's ID number, so the key of user 1 is not needed. We then compute some h_{ij}'s.

$$h_{42}=\left\lceil \frac{K_4}{7^{2-1}}\right\rceil \bmod 7$$
$$=\left\lceil \frac{121}{7}\right\rceil \bmod 7$$
$$=18 \bmod 7$$
$$=4, \text{ which is correct.}$$

Similarly,

$$h_{23}=Inverse\left(\left\lceil \frac{K_3}{7^{2-1}}\right\rceil \bmod 7\right)$$
$$=Inverse\left(\left\lceil \frac{36}{7}\right\rceil \bmod 7\right)$$
$$=Inverse\,(6)$$
$$=5, \text{ which is correct.}$$

Readers may be interested to know whether the upper bound of Chang and chang's keys is smaller than that generated by using Chang's approach [1987].

Since $K_i=\sum_{s=1}^{i-1} h_{is}t^{s-1}-\sum_{s=1}^{i-2} t^s$ implies that $K_2<K_3<\cdots<K_n$ and $K_n<t^{n-1}$ can be shown, the upper bound of our keys is identical to t^{n-1}. Recalling Chang's Key-to-Coprime mechanism, we see that each user's key is smaller than $\prod_{i=1}^{n} p_i$, where $p_1, p_2, \cdots$ and p_n are coprime integers that are geater than or equal to t.

Thus the upper bound of Chng and Chang's key values is smaller than that of Chang's key values.

7. Conclusions

We have surveyed some key-to-key user hierarchic representation mechanisms and introduced some techniques for computing all the users' key-values. We also analyzed the memory storage for keys, and found that three severe problems still remain open. (1) When the number of users becomes large, the users' key values usually severely over-flow, possible beyond the integer word in a 32-bit computer. How to modify these mechanisms to fit 32-bit computers will be an interesting research task. (2) Key-to-key user hierarchic representation mechanisms implement H-matrices but it is observed that the size of required storage (because of the length of their keys) actually exceeds that of the original H-matrices. Therefore, a new mechanism with shorter or simpler representations of keys should be designed. (3) When a new user is added to the system, all keys should be reconstructed. Is it possible to have a new key-to-key user hierarchic representation scheme such that inserting a new user is simple and straightforward? our next task is to develop the user hierarchic representation strategy to a more practical level.

References

1. CHANG, C. C. (1987) On the Implementation of User Hierarchy Structure in Information Systems, *Proceedings of the International Conference on Computer Software and Applications*, *IEEE*, Tokyo, Japan (Oct. 1987), 412-415.
2. CHANG, C. C. and CHANG, C. H. (1989) A Single Key User Hierarchic Representation Mechanism, *Proceedings of the International Workshop on Discrete Algorithms and Complexity*, Fukuoka, Japan, IPSJ, 89-AL-12 (Nov. 1989), 145-147.
3. CHANG, C. C. and CHEN, C. P. (1985) The User-Hierarchy Structure in the Information Protection System, *IMC'S 1985 Asia-Pacific Regional Information and Micrographic Management Congress*, Taipei, Taiwan (July 1985), 441-449.
4. CHANG, C. C. and HUNG, M. F. (1986) A User Hierarchy Mechanism for Computer System Resources, *Journal of Science and Engineering of NCHU*, **22** (Nov. 1985), 183-194.
5. GRAHAM, G. S. and DENNING, P. L. (1972) Protection-Principles and Practice, *Proc. Spring Jt. Computer Conference*, **40**, AFIPS Press, Montvale, N. J. (1972), 417-429.
6. JAESCHKE, G. (1981) Reciprocal Hashing: A Method for Generating Minimal Perfect Hashing Functions, *Comm. ACM*, **24**, 12 (Dec. 1981), 829-833.
7. SALTZER, J. H. (1974) Protection and Control of Information Sharing in Multics, *Commun. Ass. Comput. Mach.*, **17**, (July 1974), 388-402.
8. SALTZER, J. H. and SCHROEDER, M. D. (1975) The Protection of Information in Computer Systems, *Proc. IEEE*, **63**, (Sep. 1975), 1278-1308.
9. WU, M. L. and HWANG, T. Y. (1984) Access Control with Single-Key-Lock, *IEEE Trans. Softw. Eng.*, SE-**10**, 2, (Mar. 1984), 185-191.

(Received March 12, 1990)