Vol. 35 No. 10 Transactions of Information Processing Society of Japan Oct. 1994

Regular Paper

Assigning Digital Keys to Determine Relationships in a User Hierarchy Structure

CHIN-CHEN CHANG † and HORNG-TWU LIAW ††

In this paper, a new hierarchy mechanism based on digital keys is proposed. The objective of the mechanism is that the relationships, such as Brother, Father-of, Son-of, and so on, between any two users can be revealed conveniently. This helps in determining whether a file owned by a user may be read, written, or executed by the other users. In our scheme, each user is given a 4-tuple of digital keys. Through careful design of keys, we can get economic keys for each user. Furthermore, the relationship between two users can be easily revealed by performing a simple algorithm. In addition, whenever a new user is appended into the user hierarchy system, the corresponding keys can be determined quickly without changing any existing keys.

1. Introduction

Recently, computer systems, the greatest invention in the 20th century, have come from research institutes into offices and homes. The popularity and advanced progress of computer networks have made multiuser sharing of expensive computer resources a reality. However, sharing may cause some undesired phenomena such as unauthorized accesses, inconsistent status of shared resources. Therefore, an important issue in a multiuser computing environment is the question of how to control access to computer resources and/or identify whether a user has enough privilege to alter the access right of another user. To date, many papers[1)-9)] have addressed this problem.

In a computer protection system, the user hierarchy mechanism is always used to represent the relationships among users. The information on these relationships can be used to handle the requests for changing access attributes. For example, changing the read, write or execution rights, deleting old users, and inserting new users are typical requests of changing privileges. **Figure 1** shows a user hierarchy structure. In that figure, each user is represented by its ID number. In addition, each branch connecting the adjacent users represents the father-son relation between these two users.

Recently, researchers have devoted themselves to finding efficient schemes for implementing the user hierarchy structures. Many approaches have been proposed such as the ownership hierarchy of subjects by Graham et al.,[4)] the access hierarchy of users by Gudes,[5)] the access control hierarchy by Saltzer et al.,[8)] the key-key pair mechanism by Wu et al.,[9)] the user hierarchy mechanism based on Chinese remainder theorem by Chang.[2)] The overall schemes mentioned above cannot solve the dynamic insertion/deletion problem. In this paper, we propose a dynamic user hierarchy mechanism which assigns each user an economic key pair, so that the relationship between any two users can be derived by performing a simple formula to their corresponding key pairs. The rest of this paper is organized as follows: In Section 2, we discuss the related research. Our efficient scheme for determining relationships in information systems is described in Section 3. The performance comparisons of previous works and ours are described in Section 4. Finally, concluding remarks are given in Section 5.

2. A Review of Past Research

Among the schemes[2),4),5),8),9)] mentioned in the previous section, only the methods proposed by Ref. 2), 9) adopt the implicit methods to design. They assign each user a key/key pair and through a simple mathematical operation, the relationship between the associated users is

† Institute of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan 621, R. O. C.

†† Department of Information Management, The World College of Journalism and Communications, Taipei, Taiwan 116, R. O. C.

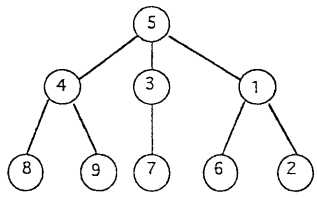

Fig. 1 The user hierarchy structure.

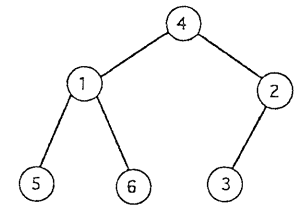

Fig. 2(a) The user hierarchy structure.

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

0: No relation
1: Brother of
2: Father of
3: Grandfather of
4: Son of
5: Grandson of

Fig. 2(b) The matrix representation.

deduced. In the following, we illustrate these two methods through two simple examples, respectively.

Example 1

Consider a user hierarchy structure with 6 users as shown in **Fig. 2(a)**. **Figure 2(b)** shows the matrix representation.

It can be seen that the matrix is in essence symmetric by close inspection of the matrix representation.

This interpretation is that the relation for user i to user j is the reverse of the relation for user j to user i. That is, if h_{ij}, the relationship of user i to user j, reveals "father of", then h_{ji}, the relationship of user j to user i, means "son of". In addition, each element in main diagonal, which represents one's relationship to himself, is useless. Therefore, the lower triangle can be used to represent the whole matrix. This lower triangle matrix is called the matrix H as shown in **Fig. 3**.

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

0: No relation
1: Brother of
2: Father of
3: Grandfather of
4: Son of
5: Grandson of

Fig. 3 The matrix H.

By applying the mechanism of Ref. 9, we then construct a key matrix in order to find all the users' keys. The matrix K is constructed as follows:

$$K=\begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} & k_{15} & k_{16} \\ k_{21} & k_{22} & k_{23} & k_{24} & k_{25} & k_{26} \\ k_{31} & k_{32} & k_{33} & k_{34} & k_{35} & k_{36} \\ k_{41} & k_{42} & k_{43} & k_{44} & k_{45} & k_{46} \\ k_{51} & k_{52} & k_{53} & k_{54} & k_{55} & k_{56} \\ k_{61} & k_{62} & k_{63} & k_{64} & k_{65} & k_{66} \end{bmatrix}$$

such that $K_i * K_j = h_{ij}$, where $1 \le j < i \le 6$, K_i represents the i-th row vector of K and $*$ is the inner product computation of GF(7), here 7 is the smallest prime number larger than all elements in the matrix H and chosen as the module number of the Galois field. Then we reserve fifteen lower triangle variables and randomly assign values to the other twenty-one elements in the matrix K. Thus, the matrix K becomes

$$K=\begin{bmatrix} 2 & 4 & 0 & 3 & 1 & 0 \\ k_{21} & 2 & 0 & 0 & 0 & 0 \\ k_{31} & k_{32} & 1 & 0 & 0 & 1 \\ k_{41} & k_{42} & k_{43} & 1 & 0 & 3 \\ k_{51} & k_{52} & k_{53} & k_{54} & 1 & 0 \\ k_{61} & k_{62} & k_{63} & k_{64} & k_{65} & 1 \end{bmatrix}$$

Furthermore, we obtain all keys by solving the fifteen equations:

$$\begin{aligned}
&k_{21} * 2+2 * 4+0 * 0+0 * 3+0 * 1+0 * 0 && = h_{21}=1 \\
&k_{31} * 2+k_{32} * 4+1 * 0+0 * 3+0 * 1+1 * 0 && = h_{31}=0 \\
&k_{31} * k_{21}+k_{32} * 2+1 * 0+0 * 0+0 * 0+1 * 0 && = h_{32}=4 \\
&k_{41} * 2+k_{42} * 4+k_{43} * 0+1 * 3+0 * 1+3 * 0 && = h_{41}=2 \\
&k_{41} * k_{21}+k_{42} * 2+k_{43} * 0+1 * 0+0 * 0+3 * 0 && = h_{42}=2
\end{aligned}$$

$$
\begin{aligned}
&k_{41} * k_{31}+k_{42} * k_{32}+k_{43} * 1+1 * 0+0 * 0+3 * 1 && =h_{43}=3\\
&k_{51} * 2+k_{52} * 4+k_{53} * 0+k_{54} * 3+1 * 1+0 * 0 && =h_{51}=4\\
&k_{51} * k_{21}+k_{52} * 2+k_{53} * 0+k_{54} * 0+1 * 0+0 * 0 && =h_{52}=0\\
&k_{51} * k_{31}+k_{52} * k_{32}+k_{53} * 1+k_{54} * 0+1 * 0+0 * 1 && =h_{53}=0\\
&k_{51} * k_{41}+k_{52} * k_{42}+k_{53} * k_{43}+k_{54} * 1+1 * 0+0 * 3 && =h_{54}=5\\
&k_{61} * 2+k_{62} * 4+k_{63} * 0+k_{64} * 3+k_{65} * 1+1 * 0 && =h_{61}=4\\
&k_{61} * k_{21}+k_{62} * 2+k_{63} * 0+k_{64} * 0+k_{65} * 0+1 * 0 && =h_{62}=0\\
&k_{61} * k_{31}+k_{62} * k_{32}+k_{63} * 1+k_{64} * 0+k_{65} * 0+1 * 1 && =h_{63}=0\\
&k_{61} * k_{41}+k_{62} * k_{42}+k_{63} * k_{43}+k_{64} * 1+k_{65} * 0+1 * 3 && =h_{64}=5\\
&k_{61} * k_{51}+k_{62} * k_{52}+k_{63} * k_{53}+k_{64} * k_{54}+k_{65} * 1+1 * 0 && =h_{65}=1.
\end{aligned}
$$

After the fifteen equations are solved in the Galois field GF(7), we obtain

$$
K=\begin{bmatrix}
2 & 4 & 0 & 3 & 1 & 0\\
0 & 2 & 0 & 0 & 0 & 0\\
3 & 2 & 1 & 0 & 0 & 1\\
1 & 1 & 2 & 1 & 0 & 3\\
10 & 0 & 5 & 6 & 1 & 0\\
4 & 0 & 1 & 3 & 1 & 1
\end{bmatrix}
$$

That is, the key vectors of all users are $K_1=(2, 4, 0, 3, 1, 0)$, $K_2=(0, 2, 0, 0, 0, 0)$, $K_3=(3, 2, 1, 0, 0, 1)$, $K_4=(1, 1, 2, 1, 0, 3)$, $K_5=(10, 0, 5, 6, 1, 0)$, and $K_6=(4, 0, 1, 3, 1, 1)$, respectively.

Here, the determinant D of the matrix K is not zero. One thing should be noticed is if $D=0$, we must reassign the main diagonal and the upper triangle variables in the original matrix K again and renewedly find the remaining fifteen variables until the resulting matrix K is nonsingular.

Now, if we want to identify whether user 3 has the privilege to access the files constructed by user 2 or not, we can simply compute the expression $K_3 * K_2=(3, 2, 1, 0, 0, 1) * (0, 2, 0, 0, 0, 0) =4=h_{32}$ and obtain that user 3 is a son of user 2. On the other hand, if we would like to find the relation for user 3 to user 4, we must evaluate the relation for user 4 to user 3 first, and then reverse the relation. That is, $K_4 * K_3=(1, 1, 2, 1, 0, 3) * (3, 2, 1, 0, 0, 1) =3=h_{43}$ and reverse(3) $=5$.

The most straightforward advantage of this mechanism is its simplicity. The relationship between two users can be derived by performing a single multiplication operation on two vectors. However, the time complexity for constructing the key vectors is $O(n^4)$ and the required space for recording the key vectors in fact exceeds that of the corresponding H-matrix. Besides, when a user is inserted into or deleted from the system, all the key vectors must be reconstructed.

Later, Chang[2] proposed a user hierarchy mechanism based on Chinese remainder theorem. In a computer protection system of n users, user i possesses a key K_i and a coprime integer q_i. H_{n*n} is a predefined matrix H and h_{ij} is the (i, j)th element of the matrix H. Given a finite set $S=\{q_1, q_2, \cdots, q_n\}$ of coprime integers for n users, where $q_j>\max\{h_{ij}\}_{1<i,j<n}$, the relation for user i to user j is revealed by

$$
h_{ij}=\begin{cases} K_i \bmod q_j, & \text{if } i>j\\ \text{reverse}(K_j \bmod q_i), & \text{otherwise.}\end{cases}
$$

In Ref. 2) Chang proposed an algorithm to compute K_i's which are smaller than $q_1 \cdot q_2 \cdot \cdots \cdot q_n$. The interested readers are referred to Ref. 2 for more details.

Example 2

Let us consider the matrix H in Example 1 again. Assume $q_1=7$, $q_2=8$, $q_3=9$, $q_4=11$, $q_5=13$, $q_6=17$. By using the Chinese remainder theorem, there exists $K_1=0$, $K_2=350064$, $K_3=612612$, $K_4=598026$, $K_5=4184856$ and $K_6=4750344$ satisfying $h_{ij}=K_i \bmod q_j$, for $1\leq j<i\leq 6$.

Let us consider "h_{42}" for example. $K_4=598026$ and $q_2=8$ are selected and then the following operation is performed:

$$h_{42}=K_4 \bmod q_2=598026 \bmod 8=2.$$

The main drawback of Chang's method is that keys grow very rapidly when the total number of users becomes large. In addition, this scheme still cannot solve the dynamic insertion/deletion problem.

3. Our Approach

In this section, a dynamic scheme is proposed that fulfills the requirement of user hierarchy system. In our scheme, without loss of generality, let the user hierarchy be represented by a directed acyclic graph. Besides, assume P_1, P_2,

$\cdots$, P_k are k distinct primes, where k is large enough to use in our assignment. In the following, an algorithm for assigning the key for each user is presented.

Algorithm ASSIGNMENT

Input: A user hierarchy U.

Output: A set of t_i's, U_i's being assigned to the users of C, respectively.

Step 1: for root user C_1 do
Let $t_1=P_1$ and $U_1=1$.

Step 2: for each non-root user C_i, which is the child of all directed ancestors C_{j1}, C_{j2}, $\cdots$, C_{jh}, do
if in-degree of C_i is 1 then $[t_i=t_{j1}\cdot P_b,\ U_i=1]$
else $[t_i=\mathrm{lcm}(t_{j1}, t_{j2}, \cdots, t_{jh})\cdot P_b,\ U_i=P_{j1}\cdot P_{j2}\cdots\cdot P_{jh}]$,
where P_b, $1\le b\le k$, is the smallest prime which is larger than all primes used previously.

The following example illustrates how t_i's and U_i's are assigned by the algorithm proposed above.

Example 3

Consider a hierarchy structure having 6 nodes in it, as depicted in **Fig. 4**.

By the algorithm proposed above, we can derive the following equations:

$t_1=P_1,\ U_1=1$
$t_2=t_1\cdot P_2,\ U_2=1$
$t_3=t_1\cdot P_3,\ U_3=1$
$t_4=t_2\cdot P_4,\ U_4=1$
$t_5=\mathrm{lcm}(t_1, t_2, t_3)\cdot P_5,\ U_5=P_1\cdot P_2\cdot P_3$
$t_6=t_3\cdot P_6,\ U_6=1$.

In addition, let m be a large number and larger than the total number of users and possesses h decimal digits. We assign $S=10^h$ and each user a fraction B, $0<B<1$, to represent one's brother/sister relationship with others in a user hierarchy. Here, the brother/sister nodes have the same B except the "common" brother/sister node. The "common" brother/sister node implies that it has more than one father/mother node and its associated B is different from the other brother/sister nodes, but the difference is less than $1/S$. However, the difference of B for the no relation nodes of the same level must equal to or more than $1/S$. That is, user i and user j are brothers/sisters if and only if they are on the same level and $|B_i-B_j|<1/S$. Let $R_i=d_1+B_i$, where d_i denotes the maximum level number of user i from the top. Thus, user i possesses a 4-tuple key (t_i, R_i, U_i, P_i) for $i=1, 2, \cdots, n$. Finally, we can evaluate the relationship between two users by performing the following algorithm.

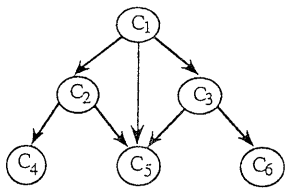

Fig. 4 The hierarchy structure.

Algorithm RELATIONSHIP

Input: The ID numbers i and j.

Output: The relationship between user i and user j.

Step 1: flag=0; relation=0

Step 2: if $|R_i-R_j|<1/S$ then [relation=1; return]
if $t_i \nmid t_j$[☆] then [interchange (t_i, t_j);
interchange (U_i, U_j)
interchange(P_i, P_j);
flag=1]
if $P_i | U_j$ then relation=2
else [if $t_i | t_j$ then relation
$=|\lfloor R_j\rfloor-\lfloor R_i\rfloor|+1$
else relation $=0$]

Step 3: if flag=1 then relation = reverse (relation)

In the following, we prove the correctness of the proposed method generally.

Theorem 3. 1: Let C_i and C_j be two nodes, and (t_i, R_i, U_i, R_i), (t_j, R_j, U_j, P_j) be their corresponding keys. The relationship between them can be derived by the algorithm RELATIONSHIP mentioned above.

Proof: We divide the proof into the following two cases.

Case 1: $C_i \ngeq C_j$

From the algorithm ASSIGNMENT, we know that each t is the product of lcm of his directed ancestors' t values and one new prime number. Thus, if $C_i \ngeq C_j$, t_j does not contain the factor of t_i. That is, $t_i \nmid t_j$.

☆ $t_i \nmid t_j$ means t_i is not the factor of t_j.
$t_i \mid t_j$ means t_i is the factor of t_j.

Case 2: $C_i \geq C_j$

With the same reason of Case 1, if $C_i \geq C_j$, t_j contains the factor of t_i and $t_i | t_j$. In the following, we divide the proof into three subcases.

Case 2. 1: The relationship of C_i and C_j is "brother of".

In our scheme, let m be a large number and larger than the total number of users and possesses h decimal digits. We assign $S=10^h$ and each user a fraction B, $0<B<1$, to represent one's brother/sister relationship with others in a user hierarchy. Here, the brother/sister nodes have the same B except the "common" brother/sister node. The "common" brother/sister node implies that it has more than one father/mother node and its associated B is different from the other brother/sister nodes, but the difference is less than $1/S$. However, the difference of B for the no relation nodes of the same level must equal to or more than $1/S$. Thus, user i and user j are brothers/sisters if and only if they are on the same level and $|B_i - B_j| < 1/S$.

Case 2. 2: The relationship of C_i and C_j is "father of".

From the step 2 in algorithm RELATIONSHIP, we can easily see that if C_i is the father node of C_j, then $P_i | U_j$.

Case 2. 3: The relationship of C_i and C_j is the other.

In our scheme, $R_i = d_i + B_i$, where d_i denotes the maximum level number of user i from the top. Thus, if the relationship of C_i and C_j is neither "father of" nor "brother of", the relationship can be derived by the equation "relation $=\lfloor R_i \rfloor - \lfloor R_j \rfloor + 1$".

Q. E. D.

Now, let us again consider the user hierarchy structure in Example 3. Without loss of generality, we have $P_1<P_2<P_3<P_4<P_5<P_6$. Thus, the "optimal" assignments of these primes are $P_1=2$, $P_2=3$, $P_3=5$, $P_4=7$, $P_5=11$ and $P_6=13$. That is, the t_i's and U_i's assigned to all users are $t_1=2$, $t_2=6$, $t_3=10$, $t_4=42$, $t_5=330$, $t_0=130$, and $U_1=U_2=U_3=U_4=U_6=1$, $U_5=30$.

Let $m=50$ which possesses 2 decimal digits. Thus, we can assign $S=10^2=100$. Furthermore, since user 1 is in level 1, we assign $d_1=1$. Similarly, since user 2 and user 3 are on level 2, we assign $d_2=d_3=2$. For the same reason, $d_4=d_6=3$. In addition, the maximum level number of user 5 from the top is 3, thus, $d_5=3$. In order to represent the brother relationship between user 2 and user 3, we randomly assign a "distinct" fraction which is different from other B_i's on the same level, say 0.01, to B_2 and B_3. Thus, $B_2=B_3=0.01$. However, user 5 is the "common" brother/sister with user 4 and user 6. So, we assign user 5 a different value, say $B_5=0.011$. In order to satisfy the condition $|B_i - B_j| \geq 1/S$ for any no relation nodes i and j, we assign $B_4=0.01$ and $B_6=0.02$. Consequently, user 1 has no brothers, we assign $B_1=0.00$. At last, by adding d_i to B_i, we have a relational parameter R_i which is associated to user i for $i=1, 2, \cdots, 6$. **Figure 5** shows all associate relational parameters R_i's.

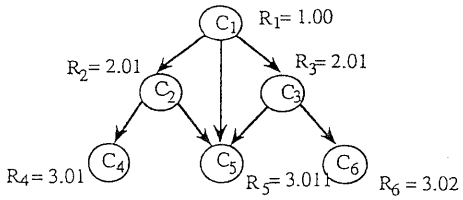

Fig. 5 The assignment of associated parameters.

Thus, user i possesses an integer t_i and a relational parameter R_i, for $i=1, 2, \cdots, 6$. That is, we have six 4-tuple keys

$(t_1, R_1, U_1, P_1) = (2, 1.00, 1, 2)$,
$(t_2, R_2, U_2, P_2) = (6, 2.01, 1, 3)$,
$(t_3, R_3, U_3, P_3) = (10, 2.01, 1, 5)$,
$(t_4, R_4, U_4, P_4) = (42, 3.01, 1, 7)$,
$(t_5, R_5, U_5, P_5) = (330, 3.011, 30, 11)$,
$(t_6, R_6, U_6, P_6) = (130, 3.02, 1, 13)$.

Now, let us consider h_{53}. Since $|R_5 - R_3| > 0.01$, user 5 and user 3 are not brothers. By performing algorithm Relationship, where $i=5$ and $j=3$, the relationship of user 5 and user 3 is 4, that is, the relationship of user 5 and user 3 is "son of". Again, let us consider h_{45}. By examing $|R_4 - R_5| = 0.001 < 0.01$, we obtain the relationship of h_{45} is "brother of", which is correct.

Now, we append a new user, say C_7, to the hierarchy structure of Example 3 as shown in **Fig. 6**.

In our scheme, whenever a new user is appended into the user hierarchy system, the corresponding keys will be determined immediately without changing any previously defined keys. That is, by employing the algorithm Assignment, $t_7 = \mathrm{lcm}(t_2 \cdot t_3) \cdot P_7 = 510$, $R_7 = 3.012$, $U_7 = P_2 \cdot P_3 = 6$, and $P_7 = 17$ is derived. Further-

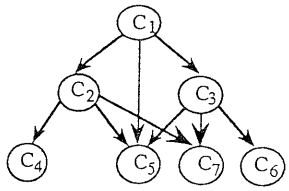

Fig. 6 The new user C_7 is appended into the system.

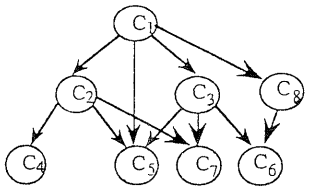

Fig. 7 The new user C_8 is inserted into the system.

more, a new user, say C_8, is inserted into the hierarchy structure as shown in **Fig. 7**.

In our scheme, only some relevant keys came from his descendant users are needed to be modified. That is, the keys of C_8 is constructed and C_6 is modified as follows:

$t_8 = t_1 \cdot P_8 = 38$,
$R_8 = 2.01$, $U_8 = 1$,
$P_8 = 19$
$t_6 = \mathrm{lcm}(t_3, t_8) \cdot P_6 = 2470$,
$R_6 = 3.02$, $U_6 = P_3 \cdot P_8 = 95$,
$P_6 = 13$.

4. The Performance Comparisons of Wu et al.'s, Chang's and Ours

The method proposed by Wu et al.[9] adopts the key-key pair mechanism to determine the relationships in a user hierarchy structure; however, there still exists some drawbacks by employing their mechanism. Their method needs to record a set of key vectors. Thus, when the number of users becomes larger and larger, the key vectors will also gradually be increased. In other words, their method needs spend $O(n^2)$ space to store all the key vectors, where n is the total number of users. Another, their mechanism must solve a large set of linear equations for obtaining the corresponding key vectors. If the determinant of the obtained matrix of the set of key vectors is zero, we must reassign the variables in the matrix until the resulted matrix is nonsingular. The overall time complexity for constructing the key vectors is $O(n^4)$ and the required space for recording the key vectors in fact exceeds that of the corresponding H-matrix.

Later, Chang[2] proposed another user hierarchy mechanism based on Chinese remainder theorem for n users, in which all keys K_i's are upper bounded by $q_1 \cdot q_2 \cdot \cdots \cdot q_n$, where q_i is a coprime integer associated to user i. Since each number can be factored into a sequence of prime numbers, we can immediately derive the equation

$$q_i = P_{i_1}^{e_{i_1}} \cdot P_{i_2}^{e_{i_2}} \cdot \cdots \cdot P_{i_r}^{e_{i_r}},$$

where P_{i_j} is a prime number and e_{i_j} is a natural number, for $1 \le i \le n$ and $1 \le j \le r$. Furthermore, q_1, q_2, $\cdots$ and q_n are coprime integers, there exists n distinct prime numbers which are smaller than or equal to q_i's, respectively. For convenience, let $P_i \le q_i$ for $i = 1, 2, \cdots, n$. Thus, each K_i is smaller than $P_1 \cdot P_2 \cdot \cdots \cdot P_n$, where P_1, P_2, $\cdots$ and P_n are n distinct prime numbers. It can be seen that the main disadvantage of Chang's method is that some keys will grow very rapidly when the total number of users becomes large.

As to our scheme, each $t_i = \mathrm{lcm}(t_{j_1}, t_{j_2}, \cdots, t_{j_h}) \cdot P_b$, where P_b, $1 \le b \le k$, is the smallest prime which is larger than all primes used previously. Thus, t_i is upper bounded by the multiplication of n prime numbers. Owing to the n-th prime number is as large as $n \log n$, thus requiring $\log(n \log n)$ bits of storage. The products of n primes requires therefore is less than or equal to $O(n \log(n \log n))$ bits.

Furthermore, our method uses some simple arithmetic and assignment operations to compute the key pairs for all users. But when deal with very large numbers, the critical division operation of our procedure cannot be operated on in constant time. Because each t_i may be as large as $O(n \log(n \log n))$ bits, the division may need $O(n^2 \log^2(n \log n))$ time in the worst case with long division method in our algorithm. For the same reason, Chang's method use many multiplication operations of large prime numbers. Thus, it faces the same trouble. Comparing to Wu et al.'s method, our approach avoids computing complicated equations and reduces the storage size from $O(n^2)$ to $O(n \log(n \log n))$.

5. Concluding Remarks

A dynamic user hierarchy mechanism based on digital keys is proposed in this paper. We

can easily reveal the relationship between two users by performing a simple algorithm. When the number of users becomes large, the implementation of our scheme is easier than those of Wu et al.'s and Chang's schemes.

In our scheme, no modification of the keys is needed when a new user is appended into the system, while almost all past user hierarchy schemes need. Besides, only minor modification is needed when a new user is inserted into the system. Hence, our scheme is suitable for practical implementation. Up-to-date, all proposed approaches could not directly handle other relationships, such as cousin, niece/nephew, uncle/aunt except use the "common" brother to access, but this is still very hard. Thus, how to modify our method to handle the above mentioned problem? This remains for inquiry.

Acknowledgment The authors would like to express his gratitude for the referees' excellent suggestions.

References

1) Akl, S. G. and Taylor, P. D.: Cryptographic Solution to a Problem of Access Control in a Hierarchy, *ACM Trans. Computer System*, Vol. 1, No. 3, pp. 239-247 (1983).
2) Chang, C. C.: On the Implementation of User Hierarchy Structure in Information System, *Proceedings of International Conference on Computer Software and Applications*, pp. 412-415, IEEE, Tokyo, Japan (Oct. 1987).
3) Chang, C. C. and Liaw, H. T.: A New Approach to Assign Cryptographic Keys in a Tree Structure for Access Control, *Journal of Electrical Engineering*, Vol. 32, No. 3, pp. 185-191 (1989).
4) Graham, G. S. and Denning, P. L.: Protection-Principles and Practices, *Proc. Spring Jt. Computer Conference*, Vol. 40, pp. 417-429, AFIPS Press, Montvale, N. J. (1972).
5) Gudes, E.: The Design of a Cryptography-based Secure File System, *IEEE Trans. Softw. Eng.*, Vol. SE-6, No. 5, pp. 411-419 (1980).
6) Liaw, H. T., Wang, S. J. and Lei, C. L.: A Dynamic Cryptographic Key Assignment Scheme in a Tree Structure, *Computers Math. Applic.*, Vol. 25, No. 6, pp. 109-114 (1993).
7) Mackinnon, S. T., Taylor, P. D., Meijer, H. and Akl, S. G.: An Optimal Algorithm for Assigning Cryptographic Keys to Control Access in a Hierarchy, *IEEE Trans. Comput.*, Vol. C-34, No. 9, pp. 797-802 (1985).
8) Saltzer, J. H. and Schroeder, M. D.: The Protection of Information in Computer Systems, *Proc. IEEE*, Vol. 63, pp. 1278-1308 (1975).
9) Wu, M. L. and Hwang, T. Y.: Access Control with Single-Key-Lock, *IEEE Trans. Softw. Eng.*, Vol. SE-10, No. 2, pp. 185-191 (1984).

(Received July 1, 1991)
(Accepted June 20, 1994)

Chin-Chen Chang was born in Taichung, Taiwan, Republic of China, on November 12, 1954. He received his B.S. degree in Applied Mathematics in 1977 and his M. S. degree in Computer and Decision Sciences in 1979 from National Tsing Hua University, Hsinchu, Taiwan. He received his Ph. D. degree in Computer Engineering in 1982 from National Chiao Tung University, Hsinchu, Taiwan. During the academic years 1980-83, he was on the faculty at the Department of Computer Engineering at National Chiao Tung University. From 1983 to 1989, he was on the faculty at the Institute of Applied Mathematics, National Chung Hsing University, Taichung, Taiwan. From August 1989 to July 1922, he was the head and professor of the Institute of Computer Science and Information Engineering at National Chung Cheng University, Chiayi, Taiwan. Since August 1992, he has been the Dean of the College of Engineering at National Chung Cheng University.

In additions, he has served as a consultant to several research institutes and government departments. His current research interests include database design, computer cryptography, coding theory, and data structures.

Horng-Twu Liaw was born in Taichung, Taiwan, Republic of China, on February 2, 1964. He received the B. S. degree in Computer Engineering from National Chiao Tung University, Hsinchu, Taiwan, in 1986, the M. S. degree in Applied Mathematics from National Chung Hsing University, Taichung, Taiwan, in 1989, and the Ph. D. degree in Electrical Engineering from National Taiwan University, Taipei, Taiwan, in 1992, respectively. He is currently an associate professor of the Department of Information Management at The World College of Journalism and Communications, Taipei, Taiwan. He is also the Director of the Computer Center at The World College of Journalism and Communications. His research interests include information security, algorithms design and analysis.