ELSEVIER

Information Sciences 130 (2000) 23–40

INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL

www.elsevier.com/locate/ins

A novel access control method using Morton number and prime factorization

Henry Ker-Chang Chang [a,*], Jing-Jang Hwang [b], Hsing-Hua Liu [b]

[a] *Department of Information Management, Chang Gung University, Taoyuan, Taiwan, ROC*
[b] *Institute of Information Management, National Chiao Tung University, Hsinchu, Taiwan, ROC*

Received 3 March 1999; received in revised form 1 March 2000; accepted 15 July 2000

Abstract

A novel scheme used for controlling access requests in security information system is proposed. In the proposed method, the system administrator chooses distinct prime numbers representing each atomic access right as well as four large prime numbers for encryption. By setting these representative prime numbers as input parameters, the proposed method applies a one-way function combining the Morton number theory transferring into a single value to derive the encrypted compound privilege (ECP). With ECP, verification of right of access can be achieved easily and secretly. Meanwhile, the proposed scheme provides the following advantages: (1) the verification of right of access can be effectively implemented using the Morton sequence with coordinate transformation; (2) the problem of dynamic access control also can be effectively implemented; (3) integrity and confidentiality while controlling system resources can be ensured; (4) the proposed method can decrease the redundancy of the access matrix in some specific circumstances.

1. Introduction

The Internet and Intranet established a foundation for the global community and remote access to resources in networks are very popular today. In

* Corresponding author. Address: P.O. Box 7-12, Chung-Ho, Taipei, Taiwan, ROC; Tel: +886-3-3960947; fax: +886-2-22232876.

E-mail addresses: changher@mail.cgu.edu.tw (H.K.-C. Chang), u8434803@cc.nctu.edu.tw (H.-H. Liu).

0020-0255/00/$ - see front matter
PII: S0020-0255(00)00073-6

open environments, security systems must have ability to authorize of requested operations; this is the security problem of *access control*. More specifically, access control is a core function for information system security. The access control model offers a framework for describing the protection mechanism. The initial model was introduced by Graham and Denning [1]. In this model, the state of an information protection mechanism is defined by a triple (S,O,A), where S is the set of subjects which are active entities of the model, O is the set of protected objects and A is an access control matrix, in which each column consists of subjects representing human or programs, and each row consists of objects representing files or records. An entry a_{ij} for $A[S,O]$ describes the right of subject S_i to access object O_j. The access right defines the kind of authorized access to the object where r(read), w(write), and e(execute), o(owner), a(append) etc. All these rights are generic and can be combined together for a subject. For example, an object may be applied for right of access in r, w and e separately, or various combinations may be used. A simple access control matrix as shown in Fig. 1 is used to specify rights of subjects to access objects. For example, the subjects S_1, and S_2 have 'read' and 'write' access rights to object O_1 while S_1 is the owner of object O_3.

A_{ij}	O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8
S_1	r, w	r	own		r, w		w	
S_2	r, w			r		e	w	w
S_3	w		app			e	r, e	r, w
S_4	r	r			r	r, w	w	r
S_5	r	r	r	r				
S_6	r	r	r	r				
S_7	r	r	r	r				
S_8	r	r	r	r				

Fig. 1. An access control matrix $[a_{ij}]_{8\times 8}$.

Based on Graham and Denning's abstraction protection model, Wu and Hwang [2] proposed a single-key-lock (SKL) mechanism in which there is only one key for each subject and one lock for a each object. To derive an access right a_{ij} for subject to an object, a function f of a key and lock is used; mathematically, $f(K_i, L_j) = a_{ij}$.

Several relevant methods have appeared in the literature which are based on SKL work. Chang [3,4] proposed two methods based on the Chinese remainder theorem and Euler's theorem, respectively. Laih et al. [5] used a Newton interpolating polynomial to design another method in 1989 while Chang and Jiang [6] presented a binary version of Wu and Hwang's method. Hwang et al. [7] proposed a new SKL scheme using prime factorization. In Hwang's scheme, each subject S_i is assigned a distinct prime as the key K_i, and a lock L_j is produced as $L_j = \prod_{i=1}^{m} (K_i)^{a_{ij}}$ for the object O_j, where a_{ij} is the right of subject S_i to access object O_j. Since a lock is the product of some prime powers, it can easily exceed the limited range of the largest integer allowed in a computer system. Hwang et al. used to decompose each lock value into an X-based representation to solve this problem, where X can be any integer. Chang et al. [8] proposed a scheme based on binary coding and prime factorization. In Chang's method, each access right a_{ij} can be represented by a binary form. Again, each user U_i is assigned a distinct prime as the key K_i. The lock vector L_j is produced as $(L_j(b), L_j(b-1), \ldots, L_j(1))$, where $L_{j(x)} = \prod_{i=1}^{m} (K_i)^{a_{ij}^{(x)}}$. There is a problem of overflow, which is inevitable $a_{ij}^{(x)}$'s $\in \{1\}$.

In order to evaluate the effectiveness and efficiency of an SKL scheme, the following six criteria are considered [7]:

1. the effort involved in initializing keys and locks;
2. the effort involved in computing an access right from a lock and key;
3. the effort involved in revising keys and locks when an access right is modified;
4. the effort involved in appending and updating keys and locks when a new user or file is added;
5. the effort involved in removing and updating keys and locks when a user or file is deleted;
6. the space needed for storing keys and locks.

These criteria are, therefore, generally applied in performance evaluation of and comparison among various schemes. In this paper, we intend to develop a new method based on the Morton sequence and prime factorization to improve schemes derived by Hwang and Shao [7] and Chang and Lou [8]. According to the six criteria, our method has better performance than the works in [7] and [8]. In particular, the proposed method has a compression effect on the matrix in some specific circumstances, which decreases redundancy in the access matrix.

The rest of this paper is organized as follows. The Morton sequence is introduced first in Section 2; it helps the proposed method work effectively. In

Section 3, we describe the proposed method first, and then algorithms are developed for physical application. In Section 4, the performance of the proposed method is analyzed and compared with that of other schemes. Conclusions and directions for future research are given in Section 5.

2. Morton sequence

Morton [9] proposed the addressing scheme commonly referred to as Morton sequencing. Morton sequencing is created by interleaving the bits of the binary representations of the x and y coordinates (each represented by a fixed number of digits) of a specific position in the matrix. Fig. 2 using 3-bit binary representation gives a simple example. Each sequence of an element is formed by the y-axis and x-axis, such that the y-axis bit is prior to the x-axis bit. The sequence at (3,2) is 13 since $2 = (010)_2$ and $3 = (011)_2$, so the interleaving 001101 is $(13)_{10}$. On the other hand, the sequence 55 has the binary form 110111, in which the odd-numbered bits 101 are the y-axis and the even-numbered bits 111 are the x-axis, so the coordinates are (7,5). From Fig. 2, it is clear that the Morton sequence in a square matrix which follows a scanning order like the character 'Z'.

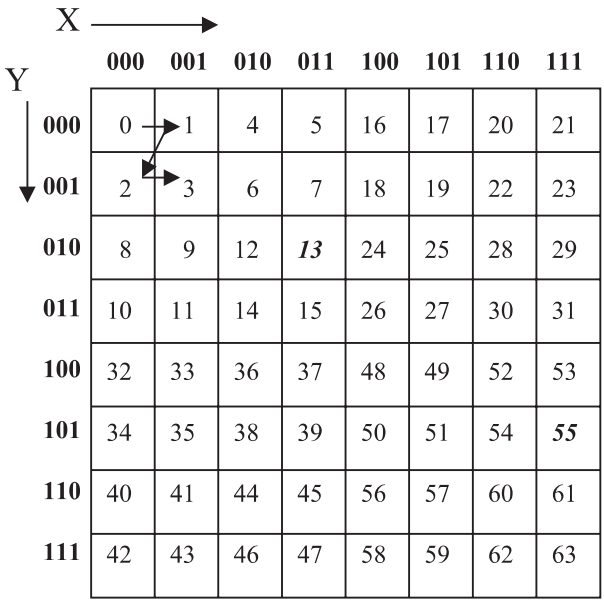

Fig. 2. The Morton sequence in the access control matrix.

3. The proposed new method

This section shows our design for access control in a security environment. The server workstation resides a security manager whose job is to monitor and maintain the access control mechanism. A user from a client node has to deliver a request to find an opportunity to access the resources in the system. Section 3 comprises three parts. Firstly, the design for the access control mechanism will be developed. Secondly, the verification procedure will be proposed. Finally, the design for dynamic access control will be presented.

3.1. The design for the access control mechanism

3.1.1. Basic mechanism

The following will explain how a one-way function can be designed in which the relationship of the Morton sequence and associated access rights within access matrix are embedded.

Step 1: *Creating compound access rights.* Consider $A_{m\times n}$ as an access control matrix, where m is the number of subjects, n is the number of objects, and the access right a_{ij} is the (i,j)th element of $A_{m\times n}$ for the subject S_i to the object O_j, which is composed of atomic access rights, such as read, write, execute, append etc. The system manager chooses distinct prime numbers p for each atomic access right. Thus, the scope of a subject's authorized operations, a_ij, can be considered as a *compound access right*, which is derived through multiplication of a series of prime numbers

$$a_{ij} = \prod p. \tag{1}$$

Step 2: *Morton sequence transformation*. In this step, we assign each a_{ij} as a distinguishable Morton number z. According to the Morton sequence, the sequence can be derived as described in Section 2. In other words, the sequence a_z has an access value a_{ij}, which represents the access right for a subject S_i to objects O_j.

Step 3: *One-way function transformation*. The security manager chooses four prime numbers, q_1, q_2, q_3, and q_4, for key identification. Thus, a one-way function can be used to transform a_z and q_t to derive encrypted compound privilege (ECP). It can be represented in the following form:

$$\mathrm{ECP}_s = \prod_{z=4(s-1)}^{z+3} q_t^{a_z}, \quad \text{where } t = 1,2,3,4,$$
$$s = 1,2,3,\ldots,n \text{ and } a_z \text{ represents the access value of } a_{ij} \text{ at Morton sequence } z. \tag{2}$$

Using Eq. (2), we can see that ECP applies four keys to encrypt subject–object specific access control information. The purpose of ECP is to prevent

from improper authorization Ti access system resources. Once the system administrator computes all the ECP values from the access matrix, the administrator can store and maintain the ECP values locally on a server workstation.

For example, consider an access control matrix $A[S, O]$ with access rights a_{ij} from eight subjects and objects $A_{8\times8}$ as shown in Fig. 3. The contents of Fig. 3 come from the mapping of Fig. 1. First, the system manager chooses distinct prime numbers for each atomic access right, such as read = 2, write = 3, execute = 5, owner = 7, and append = 11. The compound access rights in the access matrix can be derived as $a_{00} = 6(2 \times 3)$, $a_{01} = 2$, $a_{02} = 7, \ldots,$ $a_{26} = 10(2 \times 5)$. Second, given each a_{ij} in the access matrix as a distinct Morton number z, such as the Morton sequence a_0 has value of access right a_{00}, a_1 has value a_{01}, a_2 has a value a_{10}, etc. Finally, the security manager chooses four prime numbers, such as $q_1 = 2$, $q_2 = 3$, $q_3 = 5$ and $q_4 = 7$, as keys of identification. Using Eq. (2), the corresponding ECP values can be computed as follows:

$$\mathrm{ECP}_1 = 2^6 \times 3^2 \times 5^6 \times 7^0 = 9\,000\,000 \qquad \mathrm{ECP}_5 = 2^6 \times 3^0 \times 5^0 \times 7^5 = 1\,075\,648$$
$$\mathrm{ECP}_2 = 2^7 \times 3^0 \times 5^0 \times 7^2 = 6272 \qquad \mathrm{ECP}_6 = 2^3 \times 3^1 \times 5^3 \times 7^3 = 1\,029\,000$$
$$\mathrm{ECP}_3 = 2^3 \times 3^0 \times 5^2 \times 7^2 = 9800 \qquad \mathrm{ECP}_7 = 2^0 \times 3^5 \times 5^2 \times 7^6 = 714\,717\,675$$
$$\mathrm{ECP}_4 = 2^{11} \times 3^0 \times 5^0 \times 7^0 = 2048 \qquad \mathrm{ECP}_8 = 2^{10} \times 3^6 \times 5^3 \times 7^2 = 4\,572\,288\,000$$

Once the system manager computes the ECP values, the matrix can be discarded. Only the ECP values are stored and maintained.

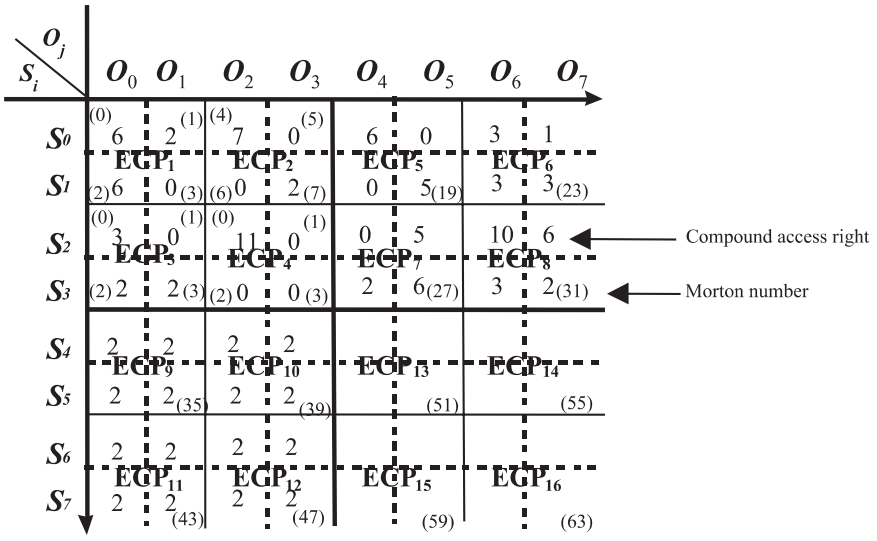

Fig. 3. Access control matrix with ECP values.

Once the content of the access matrix is determined and computed as ECP values by the security manager, the procedures for verifying access requests is developed. In addition, dynamic access control methods are also provided.

3.2. Verification of access requests

Suppose that the subject S_i wants to access the object O_j using the access request r. He/she issues the request triple (S_i, O_j, r). Two procedures need to be performed:

1. Find the Morton sequence z, in which the value a_{ij} corresponding to S_i and O_j in the access matrix can be found.
2. Perform Algorithm A (Authorization–Validation procedure) below using a one-way function $f(q_t, \mathrm{ECP}_s)$ to compute the compound access rights a_{ij}, where theq_t and ECP values can be derived from the Morton number a_z. If the derived result r is a factor of the ECP, then the access request is accepted. In other words, if the request is granted, then the request r should be a factor for the ECP values.

Algorithm A (*Authorization–Validation procedure* (q_t, ECP_s)).
Input (S_i, O_j, r), **where**
(1). Find a_z from procedure (1)
(2). Derive ECP_s, q_t where $t = [z \bmod 4] + 1,\ s = [z/4] + 1$
Set: var T, Q, R, X: integer; $X = 1$
Step 1: $T \leftarrow \mathrm{ECP}_s$;
Step 2: $Q \leftarrow T/q_t; R \leftarrow T \bmod q_t$;
// Set Q be the quotient, and set R to be the remainder.//
Step 3: if $(Q > 1)$ and $(R = 0)$ then $\{X = X + 1;\ T \leftarrow Q$; goto Step 2;$\}$
Step 4: output X; //compound access right a_{ij}.//
Step 5: if $r|X$ exist, then accept the request r, otherwise reject it.

After the iteration of Step 3 and Step 4, the compound access right of a_{ij} is kept in X. For example, when the subject S_0 requests a read operation $(r = 2)$ on the object O_0, described as $(S_i, O_j, r) = (S_0, O_0, 2)$, we derive a Morton sequence for a_{00} is $0(z = 0)$. Then, via Algorithm A, the system administrator gets $q_t = q_1 = 2$, and ECP value $=$ ECP_1 $=$ 9 000 000. Finally, the output X is computed as $f(q_t, \mathrm{ECP}_s) = 6$, in which the request $r =$ '2' is a factor of '6'; then, the access request is accepted.

3.3. Dynamic access control

In order for the administrator to be able to maintain the access matrix, dynamic access control, including modification of access rights, insertion of subjects/objects, and deletion of subjects/objects, is required. The following sections describe this control details.

3.3.1. Modification of access rights

Consider the situation in which an access right is changed from a_{ij} to a'_{ij}. Here, only the corresponding ECP values for a'_{ij} should be updated; other ECP values remain to the same

$$\mathrm{ECP}'_s = \mathrm{ECP}_s \times q_t^{(a'_{ij}-a_{ij})}, \text{ where } t = (z \bmod 4) + 1 \text{ and } s = (z/4) + 1.$$

For example, suppose the compound access right $a_{00} = 6$ (*read* and *write*) is changed to 10 (*read* and *execute*). First, we derive a Morton sequence, where a_{00} is 0; then, the system administrator gets the identification key $q_t = q_1 = 2$ and its corresponding ECP values $= \mathrm{ECP}_1 = 9\,000\,000$, where $t = (0 \bmod 4) + 1 = 1$, $s = (0/4) + 1 = 1$. The new ECP value for the modified access right is computed as follows:

$$\mathrm{ECP}'_1 = 9\,000\,000 \times 2^{(10-6)} = 9\,000\,000 \times 2^4 = 144\,000\,000.$$

3.3.2. Insertion of subjects/objects

There are two possibilities to consider: the first one is to add a subject to the access matrix; the second one is to add an object. To insert a new subject S_{m+1} into the access matrix, where the corresponding access rights are $a_{(m+1),j}$ for $j = 1, 2, \ldots, n$, the proposed scheme needs to calculate the new ECP values only, without modifying all of the existing ECP values. Due to the characteristic of the Morton sequence within the access matrix, only q_t and q_{t+1} are used to insert a new subject.

The algorithm for inserting the new subject is shown below. When a new subject is added, the administrator gets q_t and q_{t+1} first and then sets them as inputs for Algorithm B.

Algorithm B (*//Inserting subjects //*).

Input $(a_{(m+1),j}, q_t, q_{t+1})$
Output (ECP'_s)
Begin
  for $j = 1$ to n //* n is the number of objects in the access matrix*//
    Find the Morton number z for each input $a_{(m+1),j}$ and $a_{(m+1),j+1}$
    Derive $t = [z \bmod 4] + 1$ $s = [z/4] + 1$
  $\mathrm{ECP}'_s = q_t^{a_z} \times q_{t+1}^{a_{z+1}}$
  $j = j + 2$ /* each time, two access rights are computed at the same time */
end.

For example, suppose a new subject S_8 is added to the system for which the corresponding access rights are $a_{80} = 5, a_{81} = 2$, $a_{82} = 3$, $a_{83} = 3$, $a_{84} = 5$, $a_{85} = 7$, $a_{86} = 7$, and $a_{87} = 7$ by performing Algorithm B, such that the new ECP values are computed as follows:

$$\mathrm{ECP}'_{33} = 2^5 \times 3^2 = 288,$$

$$\mathrm{ECP}'_{34} = 2^3 \times 3^3 = 216,$$

$$\mathrm{ECP}'_{37} = 2^5 \times 3^7 = 69\,984,$$

$$\mathrm{ECP}'_{38} = 2^7 \times 3^7 = 279\,936.$$

To insert a new object O_{n+1} into the access matrix, and the access rights are $a_{i,(n+1)}$ for $i = 1, 2, \ldots, m$, and the proposed method needs to calculate the new ECP values only. The computed Morton sequence z for each added a_{ij} is derived as a_z first, and only q_t and q_{t+2} are used to insert a new object. The algorithm for inserting a new object is shown below. When a new object is added, the administrator gets q_t and q_{t+2} first, and then sets them as inputs for Algorithm C.

Algorithm C (*//Inserting objects//*).
Input $(a_{i,(n+1)}, q_t, q_{t+2})$
Output (ECP'_s)
Begin
for $i = 1$ to m //* m is the number of subjects in the access matrix * //
Find the Morton number z for each input $a_{i,(n+1)}$ and $a_{i+1,(n+1)}$
Derive $t = [z \bmod 4] + 1$ $s[z/4] + 1$
$\mathrm{ECP}'_s = q_t^{a_z} \times q_{t+2}^{a_{z+2}}$
$i = i + 2$ /* each time, two access rights are computed at the same time */
end.

For example, suppose a new object O_8 is added to the system for which the corresponding access rights are $a_{08} = 5$, $a_{18} = 2$, $a_{28} = 3$, $a_{38} = 3$, $a_{48} = 5$, $a_{58} = 7$, $a_{68} = 7$, and $a_{78} = 7$ by performing Algorithm C, such that the new ECP values are computed as follows:

$$\mathrm{ECP}'_{17} = 2^5 \times 5^2 = 500,$$

$$\mathrm{ECP}'_{19} = 2^3 \times 5^3 = 1000,$$

$$\mathrm{ECP}'_{25} = 2^5 \times 5^7 = 2\,500\,000,$$

$$\mathrm{ECP}'_{27} = 2^7 \times 5^7 = 10\,000\,000.$$

Fig. 4 gives an example which helps to explain how corresponding ECP values can be calculated when new subjects/objects are inserted into access control matrix.

3.3.3. Deleting subjects/objects

To delete a subject S_r from the access matrix of a system and remove the corresponding access rights, such as $a_{r,j}$ for $j = 1, 2, \ldots, n$, the proposed

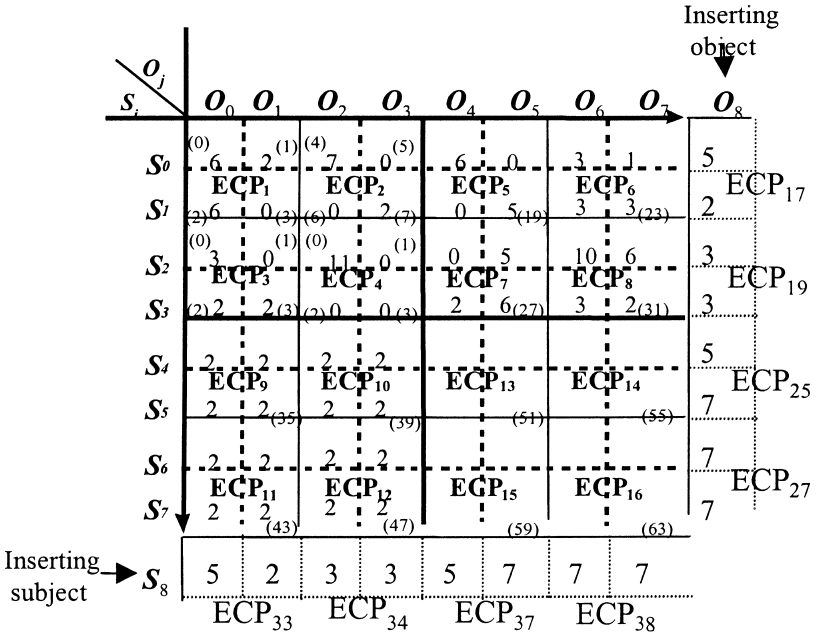

Fig. 4. The results of inserting a subject/object into the access control matrix.

method requires that ECP values corresponding to each $a_{r,j}$ be recomputed. The algorithm for deleting a subject is shown below. When a subject is deleted, the administrator gets q_t and q_{t+1} first, and then sets them as inputs for Algorithm D.

Algorithm D (*//Deleting Subjects.//*).

Input (S_r, O_j, q_t, q_{t+1})

Output (ECP'_s)

for $j = 1$ to n // **n is the number of objects in the access matrix** //

Find the Morton number z for each input a_{rj} with S_r, O_j

Derive $t = [z \bmod 4] + 1$ $s = [z/4] + 1$

Step 1: $T \leftarrow \mathrm{ECP}_s$; $W \leftarrow q_t$

Step 2: $Q \leftarrow T/W$; $R \leftarrow T \bmod W$

Step 3: if $(R = 0)$ then $\{T \leftarrow Q$; Goto Step 2;$\}$

else $\{T \leftarrow Q; W \leftarrow q_{t+1}$; Goto Step 2;$\}$

until {R $\neq$ 0;}//output Q as ECP'_s.//

$j = j + 2$ /* each time, two access rights are computed at the same time */

end.

For example, assume that the subject S_2 is deleted from the system. In Fig. 5, we see that the ECP values for S_2 corresponding to O_j are ECP_3, ECP_4, ECP_7,

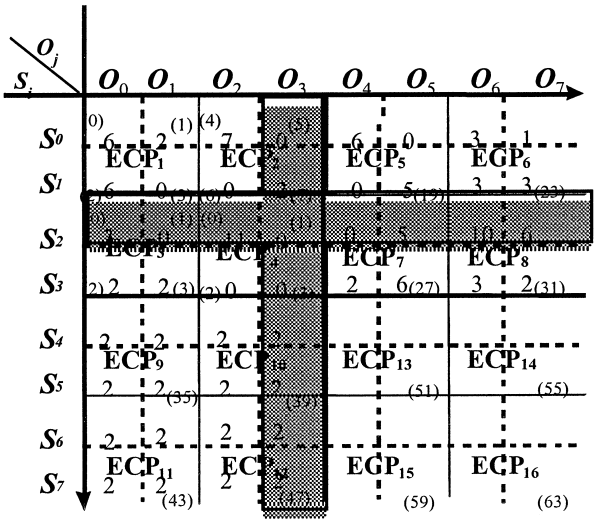

Fig. 5. The results of deleting a subject/object in the access control matrix.

and ECP_8. These ECP values will be modified. Using Algorithm D, the modified values of the ECP values are recomputed as follows:

$$ECP'_3 = ((9800/2^3)/3^0) = 1225,$$
$$ECP'_4 = ((2048/2^{11})/3^0) = 1,$$
$$ECP'_7 = ((714\,717\,675/2^0)/3^5) = 2941225,$$
$$ECP'_8 = ((4\,572\,288\,000/2^{10})/3^6 = 6125.$$

Now, an object O_r is to be deleted from the access matrix, so the corresponding access rights, such as $a_{i,r}$ for $i = 1, 2, \ldots, n$, have to be deleted. Again, the proposed method requires that the ECP values for the corresponding access right of $a_{i,r}$ be recomputed. The algorithm for deleting an object is shown below. When an object is deleted, the administrator gets q_t and q_{t+2} first, and then sets them as inputs for the Algorithm E.

Algorithm E (*// Deleting Objects.//*).
Input (S_i, O_r, q_t, q_{t+2})
Output (ECP'_s)
for $i =$ 1 to $m//m$ **is the number of subjects in the access matrix** //
Find the Morton number z for each input a_{ir} with S_i, O_r
Derive $t = [z \bmod 4] + 1$ $s = [z/4] + 1$
Step 1: $T \leftarrow ECP_s$; $W \leftarrow q_t$
Step 2: $Q \leftarrow T/W$; $R \leftarrow T \bmod W$

Step 3: if ($R = 0$) then $\{T \leftarrow Q$; Goto Step2; $\}$
else $\{T \leftarrow Q$; $W \leftarrow q_{t+2}$; goto Step 2;$\}$
until $\{R \neq 0;\}$//output Q as ECP'_s.//
$i = i + 2$ /* each time, two access rights are computed at the same time */
end.

For example, assume that the object O_3 is deleted from the system. In Fig. 5, we see that the ECP values corresponding to O_3 and the subjects S_i are ECP_2, ECP_4, ECP_{10}, and ECP_{12}. These ECP values have to be modified. Using Algorithm E, the modified values of ECP are recomputed as follows:

$$\mathrm{ECP}'_2 = ((6272/3^0)/7^2) = 128,$$

$$\mathrm{ECP}'_4 = ((2048/3^0)/7^0) = 2048,$$

$$\mathrm{ECP}'_{10} = ((44100/3^2)/7^2) = 100,$$

$$\mathrm{ECP}'_{12} = ((44100/3^2)/7^2) = 100.$$

4. Performance analysis and comparison

There is always a serious problem of storing the access matrix for any access control mechanism. The problem appears either as a sparse matrix or occupation of a large amount of storage space. Compression of the access matrix is desirable if it is possible. The proposed scheme has the advantage of being able to compress the matrix. The reduction of the amount of required is due to the compression of ECP values. In this section, storage compaction will be first described. The comparative advantages from the six criteria for the proposed method will also be illustrated.

4.1. Storage space compression

Situation may occur in which the identical level of access rights for a group of subjects may be compressed; e.g., all the students in a group have a common access authorization and are assigned to a specific directory. Fig. 6 shows this situation for subjects from S_4 to S_7 who have a common access right '2' to O_2 and O_3; as a result, we can compress these data from ECP_9 to ECP_{12} into $\mathrm{ECP}_{9(C)}$, in which (C) represents a compressed right. The value of the new ECP is identical to that of the previously computed 44 100. Meanwhile, the validation of access requests from S_4 to S_7 follows the same procedure for verifying authorization described above. In this case, the security manager does not require to manage each access right separately; the new $\mathrm{ECP}_{(C)}$ will be used to certify access rights.

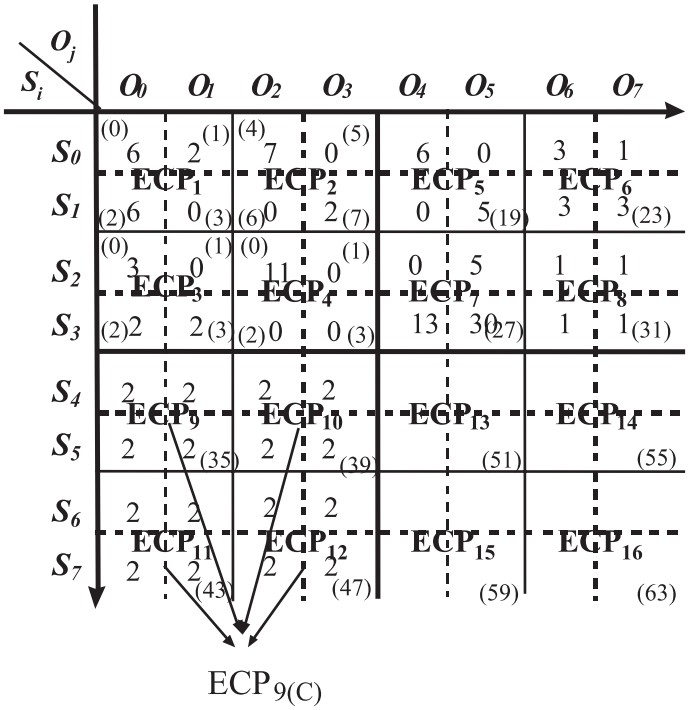

Fig. 6. Compression of ECP values in the access control matrix.

4.2. Performance comparisons

In this section, the performance of the proposed method will be analyzed and compared with that of other SKL schemes based on the set of six criteria [7]. Previous methods will be briefly discussed. They will be summarized in six tables below.

4.2.1. Effort involved in initializing keys and locks

Table 1 deals with initialization of m keys and n locks. Solving sets of linear equations is time-consuming in the schemes developed by Wu and Hwang [2] and Chang and Jiang [2,6]. Two methods developed by Chang [3,4] have to solve the same overflow problem previously mentioned and may require application of the decomposition technique. Although the scheme in [8] based on binary coding and prime factorization can avoid the overflow problem, the overflow problem while the Lock value was calculated will occur in the worst case when the binary value $x = 1$. However, our method requires only four

Table 1
Initialization of the keys and locks

SKL schemes	Effort involved in initializing keys and locks
Wu and Hwang [2]	Given m keys, solve n sets of m linear equations for n lock vectors
Chang, 1986 [3]	Given n locks, compute $K_i = \sum_{j=1}^{n}(L/L_j) \times x_j \times a_{ij} \bmod L$ for m keys
Chang, 1987 [4]	Given n locks, compute $K_i = \sum_{j=1}^{n}\lceil a_{ij} \times L_j/n \rceil \times n \times M_j$ for m keys
Laih et al. [5]	Given m keys, compute $L_j(x) = \sum_{i=1}^{m} G_i^z \prod_{s=1}^{i-1}(x - K_s)$ for n lock vectors
Chang and Jiang [6]	Given m keys, solve n sets of bm 0–1 linear equations for n lock matrices
Hwang et al. [7]	Given m keys, compute $L_j = \prod_{i=1}^{m} K_i^{a_{ij}}$ for n locks in the X-based form
Chang, 1997 [8]	Given m keys, compute $L_{j(x)} = \prod_{i=1}^{m} K_i^{a_{ij}^{(x)}}$ for $x = 1, 2, \ldots, b$ and $j = 1, 2, \ldots, n$
Our method	Given 4 keys, compute ECP values $= \prod_{z=4(s-1)}^{z+3} q_t^{a_z}$, where $t = 1, 2, 3, 4$

keys to generate ECP values. The overflow problem can be avoided no matter whether or not the subjects and objects propagate.

4.2.2. Effort involved in computing an access right from a lock and key

The numbers of operations needed to find access rights for various schemes are listed and compared in Table 2. Hwang et al.'s method [7], Chang's method [8] and our method need only a constant number of operations to compute access rights while others require a number of operations which is proportional to m (i.e., the number of users).

4.2.3. Effort involved in revising keys and locks when an access right is modified

For modification of access rights, Table 3 shows that Hwang and Shao's [7] and Chang's [8] methods can modify the original key or lock value to obtain a new one. Our method only needs to modify the original ECP value; hence, the recomputation effort is smaller than that required by the other methods. Modification in two methods by Chang's [3,4] are, however, more complex due to computations of x_j, M_j and L.

Table 2
Computation of the access rights

SKL schemes	Operations needed to compute the access right a_{ij}
Wu and Hwang [2]	m multiplications, $(m-1)$ additions and one division
Chang, 1986 [3]	One division
Chang, 1987 [4]	Two divisions and one subtraction
Laih et al. [5]	$(i-1)$ multiplications, $(i-1)$ additions and one division
Chang and Jiang [6]	bm ANDs and $b(m-1)$ XORs
Hwang et al. [7]	$\leqslant a_{\max}$ (X-based) divisions ($a_{\max}$: maximal value of access right)
Chang's, 1996 [8]	b divisions
Our method	X divisions (X: maximal value of compound access right)

Table 3
Modification for access rights

SKL schemes	Efforts involved in changing the access right a_{ij} to a'_{ij}
Wu and Hwang [2]	Solve a new set of m linear equations for new locks L'_j
Chang, 1986 [3]	Recompute $K_i = K_i + (L/L_j) \times x_j \times (a_{ij} - f(K_i, L_j)) \bmod L$
Chang, 1987 [4]	Recompute $K_I = K_i + (\lceil a_{ij} \times L_j/n \rceil - \lceil f(K_i, L_j) \times L_j/n \rceil) \times n \times M_j$
Laih et al. [5]	Recompute the $(m - i + 1)$ coefficients G'_i for L_j
Chang and Jiang [6]	Solve a new set of bm 0–1 linear equations for L_j
Hwang et al. [7]	Recompute $L_j = L_j \times (K_i)^{(a'_{ij} - a_{ij})}$
Chang, 1997 [8]	Recompute $L_{j(x)} = L_{j(x)} \times (K_i)^{\left(a'^{(x)}_{ij} - a^{(x)}_{ij}\right)}$ for $x = 1, 2, \ldots, b$
Our method	Recompute $\mathrm{ECP}'_s = \mathrm{ECP}_s \times q_t^{(a'_{ij} - a_{ij})}$ for $t = (z \bmod 4) + 1,\ s = (z/4) + 1$

4.2.4. *Effort involved in appending and updating keys and locks when a new user or file is added*

The appendability and removability properties listed in Tables 4 and 5 might be critical issues for dynamic access control in practical applications. For appendability, Laih et al.'s scheme [5] is best because it satisfies both user and file appendability. In our method, when a new subject is added, it recomputes only the ECP values of corresponding accessible objects instead of all the ECP values; when a new object is added, it recomputes only the ECP values of the corresponding accessible subjects, so the proposed method is still easy to implement.

Table 4
Appendability

SKL schemes	User appendability	File appendability
Wu and Hwang [2]	Recompute all lock vectors	Yes
Chang, 1986 [3]	Yes	Recompute all keys
Chang, 1987 [4]	Yes	Recompute all keys
Laih et al. [5]	Yes (add a coefficient to each lock vector)	Yes
Chang and Jiang [6]	Recompute all lock matrices	Yes
Hwang et al. [7]	Recompute the locks of accessible files only	Yes
Chang, 1997 [8]	To add user U_{m+1}, recompute the elements of lock vector for $a^{(x)}_{(m+1)j} = 1$ only	Yes
Our method	To add subject $S_{(m+1)}$ recompute the elements of ECP values for $s = 1$ to $n/2$	To add object $O_{(n+1)}$ recompute the elements of ECP values for $s = 1$ to $m/2$

Table 5
Removability

SKL schemes	Subject removability	Object removability
Wu and Hwang [2]	Recompute all lock vectors	Yes
Chang's, 1986 [3]	Yes	Recompute all keys
Chang's, 1987 [4]	Yes	Recompute all keys
Laih et al. [5]	To delete U_i, recompute $(m-i)$ coefficients of all for deleting	Yes
Chang and Jiang [6]	Recompute all lock matrices	Yes
Hwang et al. [7]	Recompute the locks of accessible files only	Yes
Chang, 1997 [8]	To delete user U_{m+1}, recompute the elements of lock vector for $a^{(x)}_{(m+1)j}=1$ only	Yes
Our method	To delete subject $S_{(m+1)}$, recompute the elements of the ECP values for $s=1$ to $n/2$	To delete object $O_{(n+1)}$, recompute the elements of the ECP values for $s=1$ to $m/2$

4.2.5. Effort involved in removing and updating keys and locks when a user or file is deleted

For removability, as a subject S_i is deleted, Wu and Hwang's [2], Chang's [3] and Jiang's [5] methods need to recompute all the locks while our method only needs to recompute the ECP values of the corresponding accessible objects instead of all the ECP values when a new object is deleted. In short, the recomputation effort required by our method is relatively small when a subject or object is added to or removed from the system.

4.2.6. Space for storing keys and locks

The required storage space for keys and locks is compared in Table 6. Note that $O(m+n)$ in Chang's [3] method is obtained by ignoring the overflow issue.

Table 6
Storage requirement

SKL schemes	The complex of the required
Wu and Hwang [2]	$O(m^2+mn)$
Chang, 1986 [3]	$O(m+n)$
Chang, 1987 [4]	$O(m+n)$
Laih et al. [5]	$O(mn)$
Chang and Jiang [6]	$O(m^2+bmn)$
Hwang et al. [7]	$O(mn)$
Chang, 1997 [8]	$O(mn)$
Our method	$\max(O(m),O(n))$

The storage requirement required by Hwang's [7] and Chang's [8] methods is not less than O(mn). In our method, the keys are bounded by four large prime numbers. Let l be the longest among all the elements of the ECP values, and let $q_{\max}$ be the maximal key value. ECP values $= \prod_{z=4(s-1)}^{z+3} q_l^{a_z} \leqslant 2^w$, where w is the bit-length of an integer allowed in a computer system. Since the amount of ECP is bounded by access matrix $A_{m\times n}$ such as (m/4) or (n/4) a matrix of max(m,n). Storage for ECP values is hence max $(\mathrm{O}(m), \mathrm{O}(n))$.

5. Conclusions

We have proposed a novel scheme for controlling access requests in a secure information system. Based on prime factorization and the Morton sequence, we have presented an improvement of Hwang et al.'s [7] and Chang et al.'s [8] SKL methods. Different from the conventional SKL scheme, the overflow problem while computed ECP does not occur in our scheme. Based on six criteria, the proposed scheme is considerably better for access control than most of the other comparable schemes. The convenient way in which it modifies ECP values while adding or removing objects/subjects is also impressive. Furthermore, with our compression method, the proposed scheme is suitable for implementing a large access control matrix in a distributed computer system.

Future work may include an attempt to extend the proposed idea to integrate both authentication and authorization in a security system. We could choose a distinct prime number to represent each entity-identification and access right, then computed as ECP. An ECP would be assigned to each entity's identification and stored in a place where the user who must access to information resources. The proposed method implies entity identifiers and authorization operations. On receiving the subject requested, the systems manager would verify the subject's identity and its authorization operations by referring to the corresponding ECP value. This would greatly improve integrity and confidentiality in access control systems.

References

[1] D.E.R. Denning, Cryptography and Data Security, Addison-Wesley, Reading, MA, 1982.
[2] M.L. Wu, T.Y. Hwang, Access control with single-key-lock, IEEE Trans. Software Eng. 10 (2) (1994) 185–191.
[3] C.C. Chang, On the design of a key-lock-pair mechanism in information protection systems, BIT 26 (4) (1986) 410–417.
[4] C.C. Chang, An information protection scheme based upon number theory, The Comput. J. 30 (3) (1987) 249–253.

[5] C.S. Laih, L. Harn, J.Y. Lee, On the design of a single-key-lock mechanism based on Newton's interpolating polynomial, IEEE Trans. Software Eng. 15 (9) (1989) 1135–1137.
[6] C.K. Chang, T.M. Jiang, A binary single-key-lock system for access control, IEEE Trans. Comput. 38 (10) (1989) 1462–1466.
[7] J.J. Hwang, B.M. Shao, P.C. Wang, A new access control method using prime factorization, The Comput. J. 35 (1) (1992) 16–22.
[8] C.C. Chang, D.C. Lou, A binary access control method using prime factorization, Informatics Comput. Sci. 96 (1997) 15–26.
[9] G.M. Morton, A computer oriented geodetic database, and a new technique in file sequencing, IBM Canada Ltd, March 1 (1966).