An Information Protection Scheme Based upon Number Theory

C. C. CHANG

Institute of Applied Mathematics, College of Science and Engineering, National Chung Hsing University, Taichung, Taiwan 400, Republic of China

We propose a new mechanism that fulfils the requirement of a single key–lock (SKL for short) information protection system. Using our method, each user is given a key, each file a lock, and an operation on the key of a user with the lock of a file yields the user's access privilege on the secured file.

Received October 1985

1. INTRODUCTION

Recently, time-sharing computer systems have permitted large numbers of users to share common databases. Because of this large number of users, people have begun to be concerned with the rising importance of information security.[2,3]

It is generally agreed that some kind of information protection measure is required to prevent disclosures to unauthorized persons. An access control mechanism grants a user in a system the privilege to access information resources in the system. For instance, users may be able to access files via READ, WRITE, EXECUTE, DELETE or APPEND commands, but different users will be given different access rights to individual files. Traditionally, this can be achieved by using an access control matrix which specifies who has what access privileges to system resources.[1] Currently, there are several methods that implement the access control matrix such as the accessor-list method, the capability-list method and the key–lock matching method.[4] The main disadvantage of these methods is that they all use lists with variable number of entries.

In this paper we propose a new access control mechanism based upon Euler's theorem. In our method, every legal user is given a digital key and every secured resource is given a digital lock. Through a simple operation, the access privilege that the ith accessor possesses on the jth resource can immediately be determined. In general, as compared with the other methods, our method uses less storage space and is less time-consuming.

In Section 2 we describe Graham and Denning's abstract protection model.[4] Section 3 introduces a key–lock pair mechanism which was proposed by Wu and Hwang.[6] A new protection system based upon number theory, which implements the access control matrix, is proposed in Section 4. Conclusions and an open research problem are given in Section 5.

2. AN ABSTRACT MODEL BASED UPON ACCESS CONTROL MATRIX

In this section, we first describe the role that an access control matrix plays in access control and then introduce an abstract model based upon access control matrix proposed by Graham and Denning.[4]

Sharing segments of data or programs becomes more and more important in today's computing systems. For example, one might let others use a data segment that he has stored or a routine that he has developed. Therefore, as sharing is inevitable, it is important to let the system know which users are allowed to what degree of access on which segments of data or programs, so that data with certain privacy would be at the disposal of right users. An access control matrix can represent privacy decisions on the relationships of users to files. Let us consider the matrix shown in Figure 1.

	File				
User	1	2	3	4	5
Chang	Read	Write	Own		Own
Du	Write		Read	Own	
Lee	Read	Own		Own	Read
Shen	Read		Read Write		

Figure 1.

Fig. 1 illustrates the access control matrix of a simple information protection system with four users and five files (i.e. segments of data or programs). In this case we can see that Chang owns file 3, which Du can read and Shen can both read and write; and file 4 is owned by both Du and Lee. Later, Chang might grant Lee the right to write on file 3, or Chang might request to read on file 4, which is owned by both Du and Lee. The process of changing the privileges in an access control matrix is called dynamic access control or dynamic use of the access control matrix.

The concept and the implementation of access control matrix seem to be simple and easy. But we cannot straightforwardly store the access control matrix because it will tend to be sparse, when the system grows large. Several implementation techniques are proposed in Refs 4–6.

In the next section we shall describe the key–lock pair mechanism which is proposed by Wu and Hwang.[6] This mechanism can be used to store the access control matrix. That is, instead of storing the access control matrix directly, we shall store keys and locks. Then the access control matrix can be found from these key-lock pairs.

3. A KEY–LOCK PAIR MECHANISM

The exact organization of the protection system introduced in this section is as shown in Fig. 2.

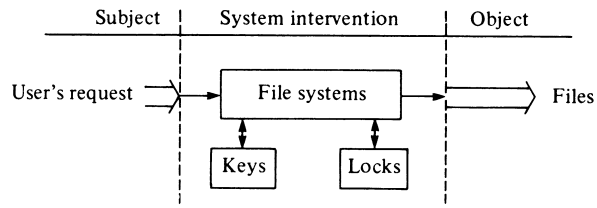

Figure 2.

For a user, every attempt to access a file is intercepted and validated by the file system. The dashed lines represent that the file system's intervention is invisible to the user. We assume that there are a fixed number of n files to be protected. The file system assigns a lock for each file. When a new user joins the system, his access rights to these n files are decided, and the file system generates a key for each user according to the locks and his access rights.

We now review Wu and Hwang's key–lock pair mechanism that fulfils the requirement of the single key–lock (SKL for short) system.[6] Again, each user is given a key and each file a lock by the file system. An operation on the key of user i with the lock of file j yields the attribute value in the (i, j)th entry of the invisible access control matrix. The construction procedures of keys and locks and the operations of keys and locks are better illustrated through a simple example.

Example 3.1

We assume that the state of a simple system having five users and six files is shown in Fig. 3.

	File j					
User i	1	2	3	4	5	6
1	2	4	1	4	4	0
2	1	2	2	1	2	1
3	2	0	3	3	3	0
4	1	1	2	1	2	3
5	3	0	4	2	2	1

0, No access
1, Execute
2, Read
3, Write
4, Own

Figure 3.

In this case access by user i to file j is allowed only when the request of the privilege matches the attribute value a_{ij}, which is the (i, j)th entry in the access control matrix as shown in Fig. 3. Here, we note that a linear hierarchy of access privileges may optionally be implied, as is the case in our example. That is, the access is allowed only if the request privilege is smaller than or equal to a_{ij}. Therefore, the right to read implies the right to execute, the right to own implies the rights for all and so on.

Wu and Hwang[6] deviced a method to assign each user U_i a key K_i of n-digit sequence and each file F_j a lock L_j of n-digit sequence too, where n is the total number of files. In this method the attribute value a_{ij} can be evaluated as $a_{ij} = K_i * L_j$, where the operator $*$ means the inner product over Galois Field $GF(t)$, and t is chosen as the smallest prime number that is larger than all attribute members of the access control matrix considered.

In our case, since all a_{ij}s have values less than 7, we may assume that the access control matrix is over finite field $GF(7)$. Now a 5×5 non-singular matrix with pseudo-random entries over $GF(7)$ is chosen as follows:

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

At this moment we choose the rows of matrix K as the keys for the users. That is,

$$K_1 = (4, 2, 1, 3, 0)$$
$$K_2 = (5, 0, 1, 2, 0)$$
$$K_3 = (3, 4, 0, 1, 3)$$
$$K_4 = (2, 0, 1, 3, 4)$$
$$K_5 = (2, 1, 1, 1, 4)$$

For evaluating the lock of F_1, we assume that $L_1 = (x_1, x_2, x_3, x_4, x_5)$. Thus we have the following five equations:

$$2 = 4x_1 + 2x_2 + 1x_3 + 3x_4 + 0x_5$$
$$1 = 5x_1 + 0x_2 + 1x_3 + 2x_4 + 0x_5$$
$$2 = 3x_1 + 4x_2 + 0x_3 + 1x_4 + 3x_5$$
$$1 = 2x_1 + 0x_2 + 1x_3 + 3x_4 + 4x_5$$
$$3 = 2x_1 + 1x_2 + 1x_3 + 1x_4 + 4x_5$$

over $GF(7)$.

Solving these equations we have $L_1 = (x_1, x_2, x_3, x_4, x_5) = (4, 2, 3, 0, 1)$. Similarly, we determine

$$L_2 = (1, 1, 2, 1, 2)$$
$$L_3 = (6, 1, 1, 3, 2)$$
$$L_4 = (2, 5, 1, 2, 1)$$
$$L_5 = (3, 2, 6, 1, 2)$$
$$L_6 = (2, 0, 3, 1, 0).$$

To check its correctness, let us consider $a_{45} = K_4 * L_5 = (2, 0, 1, 3, 4) * (3, 2, 6, 1, 2)$

$$= 23$$
$$= 2 \quad \text{over } GF(7),$$

which is correct.

Three big disadvantages of Wu and Hwang's[6] method are (1) the size of required storage (due to the keys and locks) actually exceeds that of the original access control matrix; (2) the operations of keys and locks are tedious; (3) the constructions of keys and locks are not simple. Therefore we intend to develop a new key–lock pair mechanism as described in the next section.

4. THE KEY–LOCK PAIR MECHANISM BASED UPON EULER'S THEOREM

In this section we present a mechanism that fulfils the requirement of the SKL protection system. Again, we suppose that each user U_i is given a key K_i, each file F_j a lock L_j, and an operation on the K_i with L_j yields the attribute a_{ij} in the (i, j)th entry of the access control matrix. We propose the access right value $a_{ij} = \lfloor K_i / L_j \rfloor \bmod n$, where n is the total number of files. Our method will use Euler's Theorem.

Theorem 4.1 (Euler's Theorem)

For every a and b such that $(a, b) = 1$, $a^{\phi(b)} \bmod b = 1$, where $\phi(b)$ is the number of values among 0, 1, 2, ..., $b-1$ that are relatively prime to b.

Example 4.1

Let $a = 4$ and $b = 9$. Since $(4, 9) = 1$ and $\phi(9) = 6$, we have $a^{\phi(b)} \bmod b = 4^{\phi(9)} \bmod 9 = 4^6 \bmod 9 = 4096 \bmod 9 = 1$.

Generally, for an arbitrary b,

$$\phi(b) = b \cdot \prod_{i=1}^{r} \left(1 - \frac{1}{p_i}\right),$$

where $b = p_1^{e_1} p_2^{e_2} \ldots p_r^{e_r}$ is the prime factorization of b (that is, the p_is are distinct primes, and e_i is the number of occurrences of p_i). Thus, for $b = 12 = 2^2 \cdot 3^1$, $\phi(12) = 12 \cdot (1-\frac{1}{2})(1-\frac{1}{3}) = 12 \times \frac{1}{2} \times \frac{2}{3} = 4$.

Theorem 4.2

Let $A_{m \times n}$ be an access control matrix. The (i, j)th element a_{ij} in $A_{m \times n}$ denotes the access right value of user i for file j, where $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$. Let $L = \{L_1, L_2, \ldots, L_n\}$ be a finite set of pairwise relatively prime locks for n files, where

$$\text{Min}\{L_j\}_{j=1,2,\ldots,n} \geqslant n > \text{Max}\{a_{ij}\}_{\substack{i=1,2,\ldots,m \\ j=1,2,\ldots,n}},$$

the access right value $a_{ij} = \lfloor K_i / L_j \rfloor \bmod n$ if the integer key

$$K_i = \sum_{j=1}^{n} N_{ij} \cdot n \cdot \left(\prod_{\substack{i=1 \\ i \neq j}}^{n} L_i\right)^{\phi(L_j)}, \quad \text{where } N_{ij} = \left\lceil \frac{a_{ij} \cdot L_j}{n} \right\rceil.$$

Proof

Let $$M_j = \left(\prod_{\substack{i=1 \\ i \neq j}}^{n} L_i\right)^{\phi(L_j)}.$$

Since $$\left(\prod_{\substack{i=1 \\ i \neq j}}^{n} L_i, L_j\right) = 1,$$

by Theorem 4.1, we have $M_j \bmod L_j = 1$ for $j = 1, 2, \ldots, n$. It is obvious that $M_j \bmod L_i = 0$ for $i \neq j$. Therefore we have $nM_j \bmod nL_j = n$ and $nM_j \bmod nL_i = 0$ for $i \neq j$.

Let $$K_i = \sum_{j=1}^{n} N_{ij} \cdot n \cdot M_j.$$

Then
$$\begin{aligned} K_i \bmod (nL_j) &= \left(\sum_{j=1}^{n} N_{ij} \cdot n \cdot M_j\right) \bmod (nL_j) \\ &= N_{ij} \cdot n \bmod nL_j \\ &= n(N_{ij} \bmod L_j) \\ &= n\left(\left\lceil \frac{a_{ij} \cdot L_j}{n} \right\rceil \bmod L_j\right) \\ &= n \cdot \left\lceil \frac{a_{ij} \cdot L_j}{n} \right\rceil \\ &= n \cdot N_{ij}. \end{aligned}$$

In other words, $K_i = (nL_j)x_j + nN_{ij}$. Therefore $(k_i / L_j) = n \cdot x_j + (n \cdot N_{ij} / L_j)$. Moreover,

$$\lfloor K_i / L_j \rfloor = n \cdot x_j + \lfloor n \cdot N_{ij} / L_j \rfloor = n \cdot x_j + \left\lfloor \frac{n\lceil (a_{ij} \cdot L_j)/n \rceil}{L_j} \right\rfloor.$$

Since

$$\left\lfloor \frac{n \cdot a_{ij} \cdot L_j / n}{L_j} \right\rfloor \leqslant \left\lfloor \frac{n \cdot \lceil a_{ij} \cdot L_j / n \rceil}{L_j} \right\rfloor < \frac{n(a_{ij} \cdot L_j / n) + 1}{L_j},$$

we have

$$\lfloor a_{ij} \rfloor \leqslant \left\lfloor \frac{n\lceil a_{ij} \cdot L_j / n \rceil}{L_j} \right\rfloor < a_{ij} + \frac{n}{L_j}.$$

Because $L_j \geqslant n$ for all $j = 1, 2, \ldots, n$, we have

$$a_{ij} \leqslant \left\lfloor \frac{n \cdot \lceil a_{ij} \cdot L_j / n \rceil}{L_j} \right\rfloor < a_{ij} + \frac{n}{L_j} \leqslant a_{ij} + 1.$$

In other words, $\left\lfloor \dfrac{n \cdot \lceil a_{ij} \cdot L_j / n \rceil}{L_j} \right\rfloor = a_{ij}$.

Moreover,

$$\left\lfloor \frac{K_i}{L_j} \right\rfloor = nx_j + a_{ij}.$$

This implies that $\lfloor K_i / L_j \rfloor \bmod n = a_{ij}$. Q.E.D.

Example 4.2

Fig. 4 illustrates the state of a simple system having four users and three files.

	File j		
User i	1	2	3
1	2	1	2
2	1	0	1
3	0	1	2
4	1	0	2

Figure 4.

Let $L_1 = 3$, $L_2 = 4$ and $L_3 = 5$. By using Theorem 4.2, we can calculate

$$\begin{aligned} K_1 &= \sum_{j=1}^{3} N_{1j} \cdot 3 \cdot \left(\prod_{\substack{i=1 \\ i \neq j}}^{3} L_i\right)^{\phi(L_j)} = \sum_{j=1}^{3} \left\lceil \frac{a_{1j} \cdot L_j}{3} \right\rceil \cdot 3 \cdot \left(\prod_{\substack{i=1 \\ i \neq j}}^{3} L_i\right)^{\phi(L_j)} \\ &= \lceil 2 \times 3/3 \rceil \times 3 \times (4 \times 5)^{\phi(3)} + \lceil 1 \times 4/3 \rceil \times 3 \times (3 \times 5)^{\phi(4)} \\ &\quad + \lceil 2 \times 5/3 \rceil \times 3 \times (3 \times 4)^{\phi(5)} \\ &= 2 \times 3 \times (20)^2 + 2 \times 3 \times (15)^2 + 4 \times 3 \times (12)^4 \\ &= 2 \times 3 \times 400 + 2 \times 3 \times 225 + 4 \times 3 \times 20737 \\ &= 2400 + 1350 + 248832 \\ &= 252582 \end{aligned}$$

Moreover,

$a_{11} = \lfloor K_1 / L_1 \rfloor \bmod n = \lfloor 252582/3 \rfloor \bmod 3 = 84194 \bmod 3 = 2,$

$a_{12} = \lfloor K_1 / L_2 \rfloor \bmod n = \lfloor 252582/4 \rfloor \bmod 3 = 63145 \bmod 3 = 1,$

$a_{13} = \lfloor K_1 / L_3 \rfloor \bmod n = \lfloor 252582/5 \rfloor \bmod 3 = 50516 \bmod 3 = 2.$

Similarly, we have $K_2 = 2064$, $K_3 = 250182$ and $K_4 = 250032$. The reader can verify that, in this case, each $a_{ij} = \lfloor K_i / L_j \rfloor \bmod 3$.

The K_is obtained by using Theorem 4.2 are usually large integers. In the following theorem we shall show that we can reduce the values of K_is.

Theorem 4.3

Let $A_{m\times n}$ be a matrix where each (i,j)th element a_{ij} is a non-negative integer, $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$. Let $L = \{L_1, L_2, \ldots, L_n\}$ be a set of pairwise relatively prime numbers.

$$K_i = \sum_{j=1}^{n} \lfloor a_{ij}\cdot L_j/n\rfloor \cdot n\cdot \left(\prod_{\substack{i=1\\ i\neq j}}^{n} L_i\right)^{\phi(L_j)} \bmod \left(n\cdot \prod_{i=1}^{n} L_i\right)$$

is the smallest positive integer such that $\lfloor K_i/L_j\rfloor \bmod n = a_{ij}$ for $i = 1, 2, \ldots, m$ and

$j = 1, 2, \ldots, n$ if $\operatorname{Min}\{L_j\}_{j=1,2,\ldots,n} \geqslant n > \operatorname{Max}\{a_{ij}\}_{\substack{i=1,2,\ldots,m\\ j=1,2,\ldots,n}}$.

Proof

Since $(L_i, L_j) = 1$ for $i \neq j$ and $1 \leqslant i, j \leqslant n$, by Theorem 4.2, we have

$$K_i = \sum_{i=1}^{n} \left\lceil \frac{a_{ij}\cdot L_j}{n}\right\rceil \cdot n\cdot \left(\prod_{\substack{i=1\\ i\neq j}}^{n} L_i\right)^{\phi(L_j)}$$

which satisfies that $\lfloor K_i/L_j\rfloor \bmod n = a_{ij}$, for $i = 1, 2, \ldots\ldots, m$ and $j = 1, 2, \ldots, n$. Moreover, if $\lfloor K_i/L_j\rfloor \equiv \lfloor K'_i/L_j\rfloor \bmod n$ for $j = 1, 2, \ldots, n$, then $\lfloor K_i/L_j\rfloor - \lfloor K'_i/L_j\rfloor = x_j\cdot n$, where x_j is an integer. This implies that $K_i = K'_i + x_j\cdot n\cdot L_j$ for all j. So we have $K_i - K'_i = x_j\cdot n\cdot L_j$ for all j. Therefore,

$$K_i - K'_i = x\cdot \left(n\prod_{j=1}^{n} L_j\right),$$

where x is also an integer. This argument shows that there exists at most one solution such that

$$0 \leqslant K_i < n\cdot \prod_{j=1}^{n} L_j.$$

Hence

$$K_i = \sum_{i=1}^{n} \lfloor a_{ij}\cdot L_j/n\rfloor \cdot n\cdot \left(\prod_{\substack{i=1\\ i\neq j}}^{n} L_i\right)^{\phi(L_j)} \bmod \left(n\cdot \prod_{j=1}^{n} L_j\right)$$

is the smallest positive integer such that $a_{ij} = \lfloor K_i/L_j\rfloor \bmod n$ for $i = 1, 2, \ldots\ldots, m$ and $j = 1, 2, \ldots\ldots, n$.

Q.E.D.

Example 4.3

Let us consider Example 4.2 again.

$$n\cdot \prod_{j=1}^{n} L_j = 3\cdot \prod_{j=1}^{3} L_j = 3\times(3\times 4\times 5) = 3\times 60 = 180.$$

Let K'_i be an integer satisfying that

$$K'_1 = K_1 \bmod \left(3\cdot \prod_{j=1}^{3} L_j\right).$$

We have $K'_1 = 252\,582 \bmod 180 = 42$. Now,
$a_{11} = \lfloor K'_1/L_j\rfloor \bmod 3 = \lfloor 42/3\rfloor \bmod 3 = 14 \bmod 3 = 2$.
$a_{12} = \lfloor K'_1/L_2\rfloor \bmod 3 = \lfloor 42/4\rfloor \bmod 3 = 10 \bmod 3 = 1$
and
$a_{13} = \lfloor K'_1/L_3\rfloor \bmod 3 = \lfloor 42/5\rfloor \bmod 3 = 8 \bmod 3 = 2$.
Similarly, we calculate $K'_2 = 2064 \bmod 180 = 84$,
$K'_3 = 250\,182 \bmod 180 = 162$ and
$K'_4 = 250\,032 \bmod 180 = 12$.

The reader can see that the K'_is are smaller than those found in Example 4.2.

In the following, we shall report a very attractive property of the Fermat numbers. By a Fermat number, we mean a number of the form $2^{2^k}+1$, where k is a non-negative integer.

Theorem 4.4

Any two different Fermat numbers are relatively prime. Using Theorem 4.4, we may have the following theorem.

Theorem 4.5

Let $A_{m\times n}$ be an access control matrix. The (i,j)th element a_{ij} in $A_{m\times n}$ denotes the access right value of user i for file j, where $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$. Let $L_j = 2^{2^{j+e}}+1$, where $e \geqslant (\log_2\log_2(n-1))-1$ and e is an integer, be a lock of the jth file. $a_{ij} = \lfloor K_i/L_j\rfloor \bmod n$ if the integer key

$$K_i = \sum_{j=1}^{n} N_{ij}\cdot n\cdot \left(\prod_{\substack{i=1\\ i\neq j}}^{n} L_i\right)^{\phi(L_j)}$$

where $N_{ij} = \lfloor a_{ij}\cdot L_j/n\rfloor$ and $n > \operatorname{Max}\{a_{ij}\}_{\substack{i=1,2,\ldots,m\\ j=1,2,\ldots,n}}$.

Proof

Since L_js are distinct Fermat numbers, by Theorem 4.4 we have $(L_i, L_j) = 1$ for $i \neq j$ and $1 \leqslant i, j \leqslant n$. Because $e \geqslant (\log_2\log_2(n\text{-}1))-1$, we have $2^{2^{1+e}}+1 \geqslant n$. Therefore, $L_j \geqslant n$ for $j = 1, 2, \ldots, n$. Moreover, since $n > \operatorname{Max}\{a_{ij}\}_{\substack{i=1,2,\ldots,m\\ j=1,2,\ldots,n}}$, we can conclude that $\operatorname{Min}\{L_j\}_{j=1,2,\ldots,n} \geqslant n > \operatorname{Max}\{a_{ij}\}_{\substack{i=1,2,\ldots,m\\ j=1,2,\ldots,n}}$ in this case.

Because $K_i = \sum_{j=1}^{n} \left\lceil \frac{a_{ij}\cdot L_j}{n}\right\rceil \cdot n\cdot \left(\prod_{\substack{i=1\\ i\neq j}}^{n} L_i\right)^{\phi(L_j)}$,

by Theorem 4.2, $a_{ij} = \lfloor K_i/L_j\rfloor \bmod n$, for $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$. We have the proof. Q.E.D.

According to the above theorem, given an access control matrix we can have all the key values. Now we have to answer a very eristic question. If the access control matrix is changed, do we have to recompute all K_is and L_is?

The following discussions answer our question.

Let the a_{ir} in an access control matrix $A_{m\times n}$ be changed into a'_{ir}. Then the original key K_i of the ith user is changed into K'_i, where

$$K'_i = \sum_{\substack{j=1\\ j\neq r}}^{n} N_{ij}\cdot n\cdot \left(\prod_{\substack{i=1\\ i\neq j}}^{n} L_i\right)^{\phi(L_j)} + N'_{ir}\cdot n\cdot \left(\prod_{\substack{i=1\\ i\neq r}}^{n} L_i\right)^{\phi(L_r)}.$$

This K'_i can be rewritten as

$$K'_i = K_i - N_{ir}\cdot n\cdot \left(\prod_{\substack{i=1\\ i\neq r}}^{n} L_i\right)^{\phi(L_r)} + N'_{ir}\cdot n\cdot \left(\prod_{\substack{i=1\\ i\neq r}}^{n} L_i\right)^{\phi(L_r)},$$

where $N'_{ir} = \lfloor a'_{ir}\cdot L_r/n\rfloor$. That is

$$K'_i = K_i - (N_{ir} - N'_{ir})\cdot n\cdot \left(\prod_{\substack{i=1\\ i\neq r}}^{n} L_i\right)^{\phi(L_r)}$$

$$= K_i - \left(\left\lceil \frac{a_{ir} \cdot L_r}{n} \right\rceil - \left\lceil \frac{a'_{ir} \cdot L_r}{n} \right\rceil\right) \cdot n \cdot \left(\prod_{\substack{i=1 \\ i \neq r}}^{n} L_i\right)^{\phi(L_r)}.$$

Let $\pi_{i=1}^{n} L_i$ be a constant L. Our new key K'_i is actually equal to

$$K_i - \left(\left\lceil \frac{a_{ir} \cdot L_r}{n} \right\rceil - \left\lceil \frac{a'_{ir} \cdot L_r}{n} \right\rceil\right) \cdot n \cdot \left(\frac{L}{L_i}\right)^{\phi(L_r)}.$$

It can be easily seen that K'_i is correct.

Thus, if a_{ij} is changed, the new key K_i can be easily recomputed.

5. CONCLUDING REMARKS

In this paper we have presented a new key–lock pair mechanism. Our method is based upon the following idea. Let there be a set of n pairwise relatively prime locks $L = \{L_1, L_2, \ldots\ldots, L_n\}$. Then the access right

$$a_{ij} = \lfloor K_i / L_j \rfloor \bmod n \text{ if } K_i = \sum_{j=1}^{n} [a_{ij} \cdot L_j / n] \cdot n \cdot M_j,$$

where $M_j \bmod L_i = 0$ if $j \neq i$ and is equal to 1 if $j = i$. Thus we believe that an attractive research problem has been opened. Is it possible to have the most appropriate M_js such that the representation of each key may not extend beyond 32 bits?

REFERENCES

1. R. W. Conway, W. L. Maxwell and H. C. Morgan, On the implementation of security measures in information systems. *Communications of the ACM* **15** (4) 221–220 (1972).
2. D. E. Denning and P. J. Denning, Data security. *Computing Survey* **11** (3) 227–249 (1979).
3. D. E. Denning, P. J. Denning and M. D. Schwartz, The tracker: a threat to statistical database security. *Transactions on Database Systems of the ACM* **4** (1) 76–79 (1979).
4. G. S. Graham and P. J. Denning, Protection–principles and practice, *Proceedings of AFIPS 1972 SJCC* **40**, 417–429 (1972).
5. T. Y. Hwang and J. C. Ton, An access control mechanism for computer system resources. *Proceedings of International Computer Symposium, 1980, Taipei, R.O.C.*, pp. 1243–1252.
6. M. L. Wu and T. Y. Hwang, Access control with single-key–lock. *IEEE Transactions on Software Engineering* **SE-10** (2) 185–191 (1984).

Announcement

20–22 April 1988

International Conference on Electronic Publishing, Document Manipulation and Typography, Nice, France. *Call for papers*

An international conference on Electronic Publishing, Document Manipulation and Typography will be held at Nice, France, 20–22 April, 1988. The Conference is being organised by INRIA, France in association with a number of sponsors. This conference may be considered as a successor to the EP 86 conference organised at the University of Nottingham, England in April 1986 by the British Computer Society. An associated exhibition will provide an opportunity for participants to see systems in action or at a prototype stage.

Topics

The conference will cover all aspects of computer document preparation, text processing and printing. It will include topics such as document design, authoring systems, electronic publishing and digital typography, and it will be orientated specifically towards new ideas and techniques in these fields. Papers – which should present original research work or give a comprehensive survey of a particular area – are invited on any new topic related to document processing, including the following.

- Document structures (analysis and recognition).
- Document editors or formatters, integration of text, graphics and images.
- Markup languages and translation from one to another.
- Computer-based and dynamic documents.
- Procedural page description languages.
- Interfaces with other software.
- Expert systems for editing.
- Specific documents (mathematics, chemistry, humanities, music, exotic languages, etc.).
- Font design and use, visual issues.
- Electronic publishing, applications and techniques.
- Linguistic approaches and semantic structures of texts.

Main deadlines

31 July, 1987 Papers to be received by the Program Committee Chairman.

31 October, 1987 Notification of acceptance and mailing of instructions for preparation of the final paper.

31 January, 1988 Final paper received by the Proceedings Editor so that the Conference Proceedings can be available at the Conference.

Further details

To be placed on the mailing list for this conference, please contact:

Jacques André, IRISA/INRIA EP 88, Campus de Beaulieu, F-35042 Rennes Cedex, France; or send relevant information by electronic mail to *Usenet*:...*mcvax*!*inria*!*irisa*!*jandre*.