Reducing Space for Storing Keys in a Single Key Access Control System

J. K. Jan¹, J. C. Lee², C. H. Lin³, Chin-Chen Chang⁴

¹Department of Applied Mathematics
National Chung Hsing University
Taichung, Taiwan, R.O.C.

²Institute of Information Science
National Chung Hsing University
Taichung, Taiwan, R.O.C.

³Department of Information Science
Tunghai University
Taichung, Taiwan, R.O.C.

⁴Department of Computer Science and Information Engineering
National Chung Cheng University
Chaiyi, Taiwan, R.O.C.

Abstract

Jan proposed a mechanism that fulfils the requirement of a single key-lock (SKL for short) information protection system. Using Jan’s SKL method, each user is given a key, each file a sequence number, and an operating on the key of a user with the sequence number of a file yields the user’s corresponding access privilege on the file. In Jan’s literature, a formula to compute each user’s key for a given set of assigned sequence numbers and a set of corresponding access privileges is also provided. However, different assignments of sequence numbers to the secured files will yield different keys. In this paper, an efficient way to assign a set of sequence numbers to the set of files such that the maximal key value is minimized is presented.

Key Words: file protection, access control, single-key scheme, access control matrix

1. Introduction

Recently, conventional time-sharing computer systems have permitted large numbers of users to share common databases. So information protection in a computer system becomes more and more important. To achieve the goal of information protection, many access control methods have been proposed [1, 2, 3, 4, 6, 7, 8, 9]. In 1972, Graham and Denning developed an abstract information protection model [4]. Their model is based upon the concept of states of a protection system, which are represented by an access control matrix. In the access control matrix, a triple (Si, Oj, aij) is used to indicate a state of a protection. Here Si is the ith subject such as user, processor, or utility program, Oj is the jth object such as file, memory segment, or information, and aij which is the (i, j)th entry of the access control matrix represents the access right (or access privilege) of Si to Oj.

Since an access control matrix is always very sparse, it is not cost effective to implement such a matrix directly. Varieties of methods have been developed for representing these access control matrices efficiently from the point of views of retrieval time required and data storage used. Traditionally, three practical implementation
methods, the capability-list method, the accessor-list method, and the key lock matching method, were introduced in [4]. However, the capability-list method and the accessor-list method have to do an exhaustive search whenever an accessing or updating request occurs. The key lock matching method has a capability-list for search subject, and a lock-list for each object. Thus the problem of wasting a lot of time in search a list still remains.

In 1984, Wu and Hwang [9] used two vectors \( K_i \) and \( L_j \), as the key of user \( i \) and the lock of file \( j \) to find \( a_{ij} \) through \( f(K_i, L_j) = K_i \cdot L_j = a_{ij} \), where the operator \( \cdot \) denotes the inner product operation in Galois Field algebra \( GF(p) \), and \( p \) is the smallest prime number greater than the largest access privilege of the access control matrix \( A \). Key vectors, \( K_i \)'s, must be linearly independent, and lock vectors, \( L_j \)'s, are constructed by solving sets of linear equations. Throughout the paper, let \( m \) represent the number of users and \( n \) the number of files. Then the storage required in [21] is \( n(n + m)\log(p) \) bits and the time complexity for computing keys and locks is \( O(nm^2) \). However, if a new file is inserted, all of the keys have to be reconstructed.

Chang [2] proposed single-key-lock system based upon the Chinese remainder theorem. In Chang’s single-key-lock system, all the locks are chosen to be pairwise relatively prime integers and key values are obtained by

\[
K_i = \sum_{j=1}^{n} (L / L_j) x_j \mod L_j,
\]

where \( L = \prod_{j=1}^{n} L_j \), and \( (L / L_j) x_j \mod L_j = 1 \)

If the number of bits required to represent the product of the first \( n \) prime numbers is proportional to \( n(\log n) \), then the storage required in Chang’s scheme is \( \Omega(n(m + 1)(\log n)) \). The time complexity for computing all of the keys is \( O(mn^2(\log R)) \).

In 1990, Jan [5] proposed the single key method. Using that method, each user \( i \) is assigned a key \( K_i \) which is defined as \( \sum_{j=1}^{n-1} a_{ij} B^{n-j} \) in an access control matrix \( A_{m \times n} \), where \( B \) represents an integer greater than any \( a_{ij} \). The access privilege for the user \( U_i \) to the file \( F_j \), can be revealed by computing \( a_{ij} = \left\lfloor \frac{K_i}{B^{n-j}} \right\rfloor \mod B \), where \( j \) is the sequence number of \( F_j \). The number of bits needed to represent a key is proportional to \( R \), where \( R \) is the number of privileges. The time complexity for computing all the keys is \( O(mnt(\log n)) \).

Now, let us give an example to illustrate Jan’s single key method.

**Example 1:**

Suppose \( A = \begin{bmatrix} 3 & 0 & 1 & 4 \\ 2 & 4 & 3 & 1 \end{bmatrix} \) is the access matrix, and \( B = 5 \), then

\[
K_1 = 3 \times 5^3 + 0 \times 5^2 + 1 \times 5^1 + 4 \times 5^0 = 384
\]

\[
K_2 = 2 \times 5^3 + 4 \times 5^2 + 3 \times 5^1 + 1 \times 5^0 = 366
\]

and \( K_3 = 4 \times 5^3 + 4 \times 5^2 + 2 \times 5^1 + 3 \times 5^0 = 613 \)

Here we let the sequence number of \( F_1 \) be 1, \( F_2 \) be 2, \( F_3 \) be 3, and \( F_4 \) be 4.

If we permute the four columns \( (A^1, A^2, A^3, A^4) \) of the access control matrix \( A \) to \( (A^1, A^2, A^4, A^3) \) then we will get a new access control matrix \( A' \), that is

\[
A' = \begin{bmatrix} 3 & 1 & 4 & 0 \\ 2 & 3 & 1 & 4 \\ 4 & 2 & 3 & 4 \end{bmatrix}
\]

**Example 2:**

Consider the access control matrix \( A' \), we assign the sequence number 1 to \( F_1' \), the sequence number 2 to \( F_2' \), the sequence number 3 to \( F_3' \), and the sequence number 4 to \( F_4' \), where \( F_1' = F_1 \), \( F_2' = F_3 \), \( F_3' = F_4 \), \( F_4' = F_2 \), of the previous example. Then

\[
K_1 = 3 \times 5^3 + 1 \times 5^2 + 4 \times 5^1 + 0 \times 5^0 = 420
\]

\[
K_2 = 2 \times 5^3 + 3 \times 5^2 + 1 \times 5^1 + 4 \times 5^0 = 459
\]

and \( K_3 = 4 \times 5^3 + 2 \times 5^2 + 3 \times 5^1 + 4 \times 5^0 = 469 \)

Comparing the resulted keys in Examples 1 and 2, we find that the maximal key in Example 2 is less than that in Example 1. So, we have the input that different assignments of sequence numbers to the different files will result in different key values. Thus, how to assign the set of sequence numbers to the set of files such that the maximal key value yielded is minimized is the main concern of this research. In next section, firstly, we will formally describe the problem that we want to solve. Then some important theorems which will be used in later section will be shown. In Section 3, an algorithm for assigning the most appropriate sequence numbers to the files will be presented. Section 4 will conclude this paper.
2. Our Problem and Some Theorems

This section will show some theorems which can be used to determine the most appropriate permutation of columns of the given access control matrix such that the maximal yielded key by Jan’s single key scheme to the resulted access control matrix is minimized.

2.1 A Description of Our Problem

Consider an access control matrix $A_{m \times n} = (a_{ij})_{m \times n}$. The minimax problem to the access control matrix $A_{m \times n}$ is defined as finding an integer $k = \min \max \{\sum_{j=1}^{n} f_p(a_{ij})B^{n-j} \}$, where $a_{ij} \in N \cup \{0\}$, $B > a_{ij}$, $f_p$ is a permutation function defined on $\{a_{ij}, a_{j2}, \ldots, a_{jn}\}$, $1 \leq i \leq m$, for $p = 1, 2, \ldots, n!$. The permutation function $f_p$ is a reordering of each column in the matrix $A_{m \times n}$. The value of $\sum_{j=1}^{n} f_p(a_{ij})B^{n-j}$ is the key value of the row $A_i$ in the $A_{m \times n}$ under $f_p$. The vector of $(\sum_{j=1}^{n} f_p(a_{ij})B^{n-j}, \sum_{j=1}^{n} f_p(a_{j2})B^{n-j}, \ldots, \sum_{j=1}^{n} f_p(a_{jn})B^{n-j})$ is the key vector of the matrix $A$ under a permutation function $F_p$. The value $\max_{i=1}^{m}\{\sum_{j=1}^{n} f_p(a_{ij})B^{n-j}\}$ is the maximum of a key vector of $A$ under $f_p$. The number $n!$ means all $n!$ different permutations on each row in $A$. Our goal is to find the minimum key among all maximum keys in the key vector under $n!$ different permutation functions $f_p$’s.

2.2 Preliminaries

In this subsection, we will give some definitions on the access control matrix $A_{m \times n} = (a_{ij})_{m \times n}$. For convenience, we suppose that all the rows in the matrix are distinct. Throughout this section, $\cong$ means “is defined as”.

Definition 1:

$$ A_j \cong [a_{i1}, a_{i2}, \ldots, a_{in}] $$

and

$$ \mathbf{A}^j \cong \begin{bmatrix} a_{ij} \\ a_{aj} \\ \vdots \\ a_{mj} \end{bmatrix} $$

Definition 2:

$$ [a_{i1}, a_{i2}, \ldots, a_{in}] \oplus [a_{j1}] \cong [a_{i1}, a_{i2}, \ldots, a_{ij1}, a_{ij2}, \ldots, a_{in}] $$

$$ [a_{i1}, a_{i2}, \ldots, a_{in}] \oplus [a_{j1}] \cong [a_{i2}, a_{i3}, \ldots, a_{jn}] $$

Definition 3:

$$ \max(A_j) \cong \max\{a_{i1}, a_{i2}, \ldots, a_{in}\} $$

$$ \max(A_j') \cong \max\{a_{i1}, a_{i2}, \ldots, a_{in}\} $$

Definition 4:

$$ [a_{i1}, a_{i2}, \ldots, a_{in}] \cong [b_{i1}, b_{i2}, \ldots, b_{in}] $$

such that $a_{i1} = b_{i1}$, $a_{i2} = b_{i2}, \ldots, a_{ij1} = b_{ij1}$, and $a_{ij} > b_{ij}$, where $1 \leq j \leq n$.

Definition 5:

$$ \text{MaxR}(A) \cong A_j \text{ such that } A_j > A_y, \text{ if } i \neq y. $$

Definition 6:

$$ f[A_j] \cong [f(a_{i1}), f(a_{i2}), \ldots, f(a_{in})], \text{ where } f \text{ is a permutation function.} $$

$$ f[A_j] \cong [f(A_{i1}) \quad f(A_{i2}) \quad \vdots \quad f(A_{in})], \text{ where } f \text{ is a permutation function defined on } A_j, \ 1 \leq i \leq m. $$

Definition 7:

$$ s(A_j) \cong \sum_{j=1}^{n} a_{ij}B^{n-j}, \text{ where } s(A_j) \text{ represents the single key of } A_j \text{ and the base } B \geq a_{ij} \text{ defined on } A_j, j = 1, 2, \ldots, n. $$

Definition 8:

$$ F_j \cong \{f_{(j-1)(n-1)+1}, f_{(j-1)(n-1)+2}, \ldots, f_{j(n-1)+1}\}, \text{ where } fr\text{'s are permutation functions.} $$

Definition 9:

$$ \text{MinF}_{j}[A_j] \cong [a_{ij}'], \text{ where } [a_{i2}', a_{i3}', \ldots, a_{in}'] \text{ is a nonincreasing sequence that is a reorder of}$$
Definition 10:
Let there be \( t \) maximum attributes \( a_{1,1}, a_{2,1}, \ldots, a_{n,1} \) in the \( j \)th column of the matrix, where \( a_{i,1} = a_{i,2} = \ldots = a_{n,1} = a \) = the maximum attribute and \( i_1 < i_2 < \ldots < i_t \). Then \( a_{i,1} = a_{i,2} = \ldots = a_{i,t} \), \( A_{i,1} \equiv A_{i,2} \equiv \ldots \equiv A_{i,t} \), \( A_{i,1} \equiv A_{i,2} \equiv \ldots \equiv A_{i,t} \equiv \) the row in which \( a_{i,1} \) is located.

Definition 11:
\[
A^{<j>} \equiv \left[ \begin{array}{c}
a_{1,j} \\
a_{2,j} \\
\vdots \\
a_{r,j}
\end{array} \right], \text{ where } \{a_{1,j}, a_{2,j}, \ldots, a_{r,j}\}
\]
is the set of all maximums of \( A^j \).

Definition 12:
\( A^{(k)} \equiv A^{<k>} \) and \( \max(A^i) \leq \max(A^j) \) if \( k \in \{1, 2, \ldots, n\} \). We call \( A^{(k)} \) the minimax column in \( A \).

Definition 13:
\( \text{MinMax}(A) \equiv \min \max_{i=1}^{n} s(f_p[A_i]) \).

Example 2.1:
Let \( A = \begin{bmatrix} 3 & 0 & 1 & 4 \\ 2 & 4 & 3 & 1 \\ 4 & 4 & 2 & 3 \end{bmatrix} \), \( f = (1, 2) \). We have \( f[A_1] = [0 \ 3 \ 1 \ 4], \ f[A_2] = [4 \ 2 \ 3 \ 1], \ f[A_3] = [4 \ 4 \ 2 \ 3] \).

MaxR[\( f[A] \)] = \( \{f[A_1], f[A_2], f[A_3]\} \) = \( [4 \ 4 \ 2 \ 3] \).

\( a_{1,1} = a_{1,3} = 4, \ a_{1,1} = a_{1,3} = 3 \).

\( A_{1,1} = [4 \ 4 \ 2 \ 3], \ A_{1,3} = [2 \ 4 \ 3 \ 1] \).

\( A^{<1>} = [4], \ A^{<2>} = [4 \ 4], \ A^{<3>} = [3], \ A^{<4>} = [4] \).

In the following, we are going to show some basic theorems.

**Theorem 2.1:**
If \( a_1 > b_1 \),
then \( s([a_1 a_2 \ldots a_n]) > s([b_1 b_2 \ldots b_n]) \)

**Proof:**
\[
s([a_1 a_2 \ldots a_n]) - s([b_1 b_2 \ldots b_n]) = (a_1 B_n + a_2 B_{n-1} + \ldots + 1) - (b_1 B_n + b_2 B_{n-1} + \ldots + 1) = (a_1 - b_1)B_n + (a_2 - b_2)B_{n-1} + \ldots + 1 + (a_1 - b_1) + (a_2 - b_2) + \ldots + 1 \geq 0.
\]

**Theorem 2.2:**
If \( [a_1 a_2 \ldots a_n] \geq [b_1 b_2 \ldots b_n] \),
then \( s([a_1 a_2 \ldots a_n]) > s([b_1 b_2 \ldots b_n]) \).

**Proof:**
From Theorem 2.1, this theorem can be concluded by mathematical induction.

**Corollary 2.1:**
\[
s(\text{MaxR}[f[A]]) = \max \left\{ s(f[A_1]), s(f[A_2]), \ldots, s(f[A_n]) \right\}
\]

**Theorem 2.3:**
\[
s(A_j) = a_{i,j}B_n + s(A_j \ominus [a_{i,j}])
\]

**Proof:**
\[
s(A_j) = \sum_{j=1}^{n} a_{j,j}B_n = \sum_{j=1}^{n} a_{j,j}B_n + s(A_j \ominus [a_{i,j}])
\]

**Corollary 2.2:**
\[
s(f[A_j]) = a_{i,j}B_n + s(f[A_j] \ominus [a_{i,j}]), \quad \text{if } f \in F_j.
\]

The following four theorems can be used in the algorithm. Which will be introduced in next section.
Theorem 2.4:

There is one and only one ordered pair \((i, j)\) such that \(\max(A^j) < \max(A^y)\) if \(j \neq y\) and \(a_{ij} < a_{ij}\) if \(i \neq x\). that is \(\text{MinMax}(A) = s(\text{MinF}_j[A_j])\).

proof:

\[
\min_{p=1}^{n!} \max_{i=1}^{m} \{s(f_p[A_i])\} = \min_{p=1}^{n!} \max \left( \begin{array}{c}
\left\{ \begin{array}{c}
s(f_1[A_1]) \\
s(f_1[A_2]) \\
\vdots \\
s(f_1[A_m])
\end{array} \right\}, \\
\left\{ \begin{array}{c}
s(f_2[A_1]) \\
s(f_2[A_2]) \\
\vdots \\
s(f_2[A_m])
\end{array} \right\}, \\
\vdots \\
\left\{ \begin{array}{c}
s(f_m[A_1]) \\
s(f_m[A_2]) \\
\vdots \\
s(f_m[A_m])
\end{array} \right\}
\right)
\]

That is, from Corollary 2.1, we have

\[
\min_{p=1}^{n!} \max_{i=1}^{m} \{s(f_p[A_i])\} = \min \{s(\text{MaxR}[f_1[A]]) \text{, } s(\text{MaxR}[f_2[A]]) \text{, } \ldots \text{, } s(\text{MaxR}[f_{(n-1)}[A]]), s(\text{MaxR}[f_{(n-1)}[A]]) \text{, } s(\text{MaxR}[f_{(n-1)}[A]]) \text{, } \ldots \text{, } s(\text{MaxR}[f_{(n-1)}[A]]) \text{, } s(\text{MaxR}[f_{(n-1)}[A]]) \text{, } \ldots \text{, } s(\text{MaxR}[f_{(n-1)}[A]]) \}\}
\]

From Definition 8 and Theorem 2.1, if \(\max(A^j) < \max(A^y)\) for \(j \neq y\), we get

\[
\min_{p=1}^{n!} \max_{i=1}^{m} \{s(f_p[A_i])\} = \min \{s(\text{MaxR}[f_{(j-1)(n+1)}[A]]) \text{, } s(\text{MaxR}[f_{(j-1)(n+1)}[A]]) \text{, } \ldots \text{, } s(\text{MaxR}[f_{(j-1)(n+1)}[A]]) \}\}
\]

[Note: \(a_{ij} = f(a_{ij})\) if \(f \in F_j\)]
Consequently, from Theorem 2.2, we have
\[
\begin{align*}
\min_{p=1}^{n!} \max_{i=1}^{m} \{ s(f_p[L_i]) \} \\
&= \min \{ s(\max[R[f_{(j-1)n+1}[A_i]]) \} , s(\max[R[f_{(j-1)n+2}[A_i]]) \}, ..., s(\max[R[f_{(n-1)n+1}[A_i]]) \}, \\
&\text{if } a_{ij} > a_{jk} \text{ for } i \neq x.
\end{align*}
\]

\[ = s(MinF^{i}[A_i]).\]

Example 2.2:

Let \( A = \begin{bmatrix} 3 & 0 & 1 & 4 \\ 2 & 4 & 3 & 1 \\ 4 & 4 & 2 & 3 \end{bmatrix} \) and \( B = 5. \) Now we want to find \( MinMax(A). \)

Solution:

There exists one and only one ordered pair \((2, 3)\) such that \( \max(A^2) < \max(A^3) \) for \( y \neq 3, \) and \( a_{23} > a_{33} \) for \( x \neq 2. \)

So \( MinMax(A) = s(MinF^3[A_2]) = s([3 \ 1 \ 2 \ 4]) = 3 \times 5^3 + 1 \times 5^2 + 2 \times 5 + 4 \times 5^0 = 414. \)

Theorem 2.5:

There is one and only one such that \( A^1 = ([a_{1,i}] [a_{2,i}] \ldots [a_{r,i}]) \) and \( \max(A^1) < \max(A^r) \) if \( j \neq y. \)

That is,

\[
MinMax(A) = a_{1,i}B^{n-1} + \min_{p=1}^{n!} \max_{i=1}^{m} \{ s(f_p[A_{i,r}] \Theta a_{1,i}) \}. 
\]

Proof:

\[
\begin{align*}
&\min_{p=1}^{n!} \max_{i=1}^{m} \{ s(f_p[A_i]) \} \\
&\quad = \min_{p=1}^{n!} \max \{ s(f_1[A_i]) , s(f_2[A_i]) , ..., s(f_m[A_i]) \} \\
&\quad = \min \max \{ s(f_1[A_i]) , s(f_2[A_i]) , ..., s(f_m[A_i]) \}, \max \{ s(f_1[A_i]) , s(f_2[A_i]) , ..., s(f_m[A_i]) \}, \max \{ s(f_1[A_i]) , s(f_2[A_i]) , ..., s(f_m[A_i]) \}, \\
&\quad = \max \{ s(f_1[A_i]) , s(f_2[A_i]) , ..., s(f_m[A_i]) \}, \max \{ s(f_1[A_i]) , s(f_2[A_i]) , ..., s(f_m[A_i]) \}, \max \{ s(f_1[A_i]) , s(f_2[A_i]) , ..., s(f_m[A_i]) \}. 
\end{align*}
\]
Reducing Space for Storing Keys in a Single Key Access Control System

\[
\max\left(\begin{array}{c}
s(f_{(n-1)(n-1)+2}[A_1]) \\
s(f_{(n-1)(n-1)+2}[A_2]) \\
\vdots \\
s(f_{(n-1)(n-1)+2}[A_m])
\end{array}\right), \ldots, \max\left(\begin{array}{c}
s(f_{n}[A_1]) \\
s(f_{n}[A_2]) \\
\vdots \\
s(f_{n}[A_m])
\end{array}\right)
\]

\[= \min\{s(MaxR[f_1(A)]) , s(MaxR[f_2(A)]) , \ldots , s(MaxR[f_{(n-1)+1}(A)]) , s(MaxR[f_{(n-1)+1}(A)]) , s(MaxR[f_{(n-1)+1}(A)]) , \ldots , s(MaxR[f_{(n-1)+1}(A)]) , \ldots , s(MaxR[f_{(n-1)+1}(A)]) , s(MaxR[f_{(n-1)+1}(A)])\}.
\]

(From Corollary 2.1)

\[= \min\{s(MaxR[f_{(j-1)(j-1)+1}(A)]) , s(MaxR[f_{(j-1)(j-1)+1}(A)]) , \ldots , s(MaxR[f_{(j-1)(j-1)+1}(A)])\} , \ldots , s(MaxR[f_{(j-1)(j-1)+1}(A)]) , s(MaxR[f_{(j-1)(j-1)+1}(A)])\}.
\]

(From Definition 8 and Theorem 2.1) [Note: \(a_{ij} = f(a_{ij})\) or \(f \in F_j\)]

\[= \min \max\left(\begin{array}{c}
s(f_{(j-1)(j-1)+1}[A_1]) \\
s(f_{(j-1)(j-1)+1}[A_2]) \\
\vdots \\
s(f_{(j-1)(j-1)+1}[A_r])
\end{array}\right), \ldots, \max\left(\begin{array}{c}
s(f_{(j-1)(j-1)+1}[A_1]) \\
s(f_{(j-1)(j-1)+1}[A_2]) \\
\vdots \\
s(f_{(j-1)(j-1)+1}[A_r])
\end{array}\right)
\]

\[= a_{i,j} + \min \max\left(\begin{array}{c}
s(f_{(j-1)(j-1)+1}[A_1] \Theta [a_{1,j}]) \\
s(f_{(j-1)(j-1)+1}[A_2] \Theta [a_{2,j}]) \\
\vdots \\
s(f_{(j-1)(j-1)+1}[A_r] \Theta [a_{r,j}])
\end{array}\right), \ldots, \max\left(\begin{array}{c}
s(f_{(j-1)(j-1)+1}[A_1] \Theta [a_{1,j}]) \\
s(f_{(j-1)(j-1)+1}[A_2] \Theta [a_{2,j}]) \\
\vdots \\
s(f_{(j-1)(j-1)+1}[A_r] \Theta [a_{r,j}])
\end{array}\right)
\]

(From Corollary 2.2)

\[= a_{i,j}B^{n-1} + \min_{p=\frac{(j-1)(j-1)+1}{n}} \max_{r=1}^r \max\{s(f_p[A_1, \Theta a_{i,j}])\} . \quad Q.E.D.
\]

Example 2.3:

Let \(A = \begin{bmatrix} 3 & 0 & 1 & 4 \\ 2 & 4 & 3 & 1 \\ 4 & 4 & 3 & 3 \end{bmatrix}\) and \(B = 5\). Now we want to find \(MinMax(A)\).

Solution:

There exists one and only one \(j = 3\), such that \(A(3) = \begin{bmatrix} a_{1,3} \\ a_{2,3} \\ a_{3,3} \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \\ 3 \end{bmatrix}\) and \(\max(A^3) < \max(A^r)\) if \(y \neq 3\).

So \(MinMax(A) = a_{2,3}B^{4-1} + \min_{p=\frac{(3)(3)+1}{n}} \max_{r=1}^r \max\{s(f_p[A_{1,3} \Theta a_{i,3}])\} = 3 \times 3^3 + \begin{bmatrix} 2 & 4 & 1 \\ 4 & 4 & 3 \end{bmatrix}\)

From Theorem 2.4, we have...
\[ \text{MinMax}(A) = 3 \times 5^3 + s(\text{MinF}^3[R_2]) \], if \( R = \begin{pmatrix} 2 & 4 & 1 \\ 4 & 4 & 3 \end{pmatrix} \)
\[ = 3 \times 5^3 + 3 \times 5^2 + 4 \times 5^1 + 4 \times 5^0 \]
\[ = 474 \]

**Theorem 2.6:**

There exist two ordered pair \((i_1, j_1)\) and \((i_2, j_2)\), where \( i_1 \neq i_2 \) and \( i_2 \neq i_2 \) such that 
\[ \text{max}(A_i) = \text{max}(A_i) < \text{max}(A'_i) \] if \( y \neq j_1 \) and \( y \neq j_2 \), \( a_{ij_1} > a_{ij_2} \) if \( x \neq i_1 \), \( a_{i_2j_2} > a_{i_2j_2} \) if \( x \neq i_2 \);
and \( \text{MinF}^{i_1} [A_{i_1}] > \text{MinF}^{i_2} [A_{i_2}] \), that is \( \text{MinMax}(A) = s(\text{MinF}^{i_2} [A_{i_2}]) \).

**proof:**

\[
\min_{p=1}^{n!} \max_{i=1}^{m} \{s(f_p[A_i])\}
\]
\[
= \min_{p=1}^{n!} \max_{i=1}^{m} \left( \begin{array}{c}
\frac{s(f_p[A_i])}{s(f_p[A_1])} \\
\vdots \\
\frac{s(f_p[A_m])}{s(f_p[A_m])}
\end{array} \right)
\]
\[
= \max_{p=1}^{n!} \left( \begin{array}{c}
\frac{s(f_1[A_1])}{s(f_1[A_1])} \\
\vdots \\
\frac{s(f_1[A_m])}{s(f_1[A_m])}
\end{array} \right), \max_{p=1}^{n!} \left( \begin{array}{c}
\frac{s(f_2[A_1])}{s(f_2[A_1])} \\
\vdots \\
\frac{s(f_2[A_m])}{s(f_2[A_m])}
\end{array} \right), \ldots, \max_{p=1}^{n!} \left( \begin{array}{c}
\frac{s(f_{(n-1)}[A_1])}{s(f_{(n-1)}[A_1])} \\
\vdots \\
\frac{s(f_{(n-1)}[A_m])}{s(f_{(n-1)}[A_m])}
\end{array} \right), \max_{p=1}^{n!} \left( \begin{array}{c}
\frac{s(f_{(n-1)+1}[A_1])}{s(f_{(n-1)+1}[A_1])} \\
\vdots \\
\frac{s(f_{(n-1)+1}[A_m])}{s(f_{(n-1)+1}[A_m])}
\end{array} \right),
\]
\[
= \min \{s(\text{MaxR}[f_1[A]])\}, s(\text{MaxR}[f_2[A]]) \ldots, s(\text{MaxR}[f_{(n-1)}[A]])\}, s(\text{MaxR}[(f_{(n-1)+1}[A]))\}, \ldots, s(\text{MaxR}[f_{(n-1)+1}[A]])\}, \ldots, s(\text{MaxR}[f_{(n-1)+1}[A]])\}.
\]
\[
= \min \{s(\text{MaxR}[f_{(j-1)}(n-1)+1][A])]\}, s(\text{MaxR}[f_{(j-1)(n-1)+2}[A]]) \ldots, s(\text{MaxR}[f_{(j-1)(n-1)+1}[A]])\}, \ldots, s(\text{MaxR}[f_{(j-1)(n-1)+1}[A]])\}.
\]
\[
= \min \{s(\text{MaxR}[f_{(j-1)(n-1)+1}[A])]\}, s(\text{MaxR}[f_{(j-1)(n-1)+2}[A]]) \ldots, s(\text{MaxR}[f_{(j-1)(n-1)+1}[A]])\}, \ldots, s(\text{MaxR}[f_{(j-1)(n-1)+1}[A]])\}.
\]
if \( \max(A^h) = \max(A^{j_h}) \), \( \max(A^h) < \max(A^j) \) for \( j_1 \neq y \) and \( \max(A^h) < \max(A^y) \) for \( j_2 \neq y \)  
(From Definition 8 and Theorem 2.1) [Note: \( a_{ij} = f(a_{ij}) \) if \( f \in F_j \)]

\[
= \min \{ s(f_{j_1(1)(n-1)+1}[A]), s(f_{j_2(1)(n-1)+2}[A]) \ldots, s(f_{j_1(n-1)}[A]) \ldots, s(f_{j_2(n-1)}[A]) \}, \text{ if } a_{i,j_i} > a_{i,j} \text{ for } x \neq i, a_{i_2,j_2} > a_{x,j} \text{ for } x \neq i
\]

\[
= \min \{ s(Min^{F^h}[A_{j_1}]), s(Min^{F^h}[A_{j_2}]) \} \quad \text{ (From Theorem 2.2)}
\]

\[
= s(Min^{F^h}[A_{j_2}]), \text{ if } Min^{F^h}[A_{j_1}] > Min^{F^h}[A_{j_2}] .
\]

\[Q.E.D.\]

**Example 2.4:**

Let \( A = \begin{bmatrix} 3 & 0 & 1 & 4 \\ 2 & 4 & 3 & 1 \\ 2 & 4 & 2 & 3 \end{bmatrix} \) and \( B = 5 \). Now we want to find \( MinMax(A) \).

**Solution:**

There exists two ordered pairs \((1, 1)\) and \((2, 3)\), such that \( \max(A^1) = \max(A^3) < \max(A^2) \). if \( y \neq 1 \), \( y \neq 3 \), and \( a_{11} > a_{x1} \) if \( x \neq 1 \), \( a_{23} > a_{x3} \) if \( x \neq 2 \), and \( Min^{F^1}[A_i] = [3 \ 0 \ 1 \ 4] < [3 \ 1 \ 2 \ 4] = Min^{F^3}[A_2] \).

So \( MinMax(A) = s(Min^{F^1}[A_i]) = s([3 \ 0 \ 1 \ 4]) = 384 \).

**Theorem 2.7:**

There exist two integers \( j_1 \) and \( j_2 \), such that \( A^{(j_1)} = ([a_{1,1}^{(j_1)}]_{a_{1,1}^{(j_1)}} \ldots a_{r,r}^{(j_1)})^{(i)} \), \( A^{(j_2)} = ([a_{1,1}^{(j_2)}]_{a_{1,1}^{(j_2)}} \ldots a_{r,r}^{(j_2)})^{(i)} \) and \( \max(A^{(j_1)}) = \max(A^{(j_2)}) < \max(A^i) \) if \( y \neq j_1 \) and \( y \neq j_2 \). That is \( MinMax(A) = a_{1,1}^{(j_1)}B^{a_{1,1}^{(j_1)}} + \min \left( \min_{p=(j_1-n+1)+1}^{j_2(n-1)!} \max_{i=1}^{n-1} \right) s(f_p[A_{(i,j)}] \ominus a_{(i,j)})) \), \( \min_{p=(j_2-n+1)+1}^{j_2(n-1)!} \max_{i=1}^{n-1} \right) s(f_p[A_{(i,j)}] \ominus a_{(i,j)})) \).

\[
\text{proof:}
\]

\[
= \min \max_{p=1}^{n!} \left( \begin{array}{ccc}
\left( s(f_{p}[A_{1}]) \right) \\
\vdots \\
\left( s(f_{p}[A_{m}]) \right)
\end{array} \right)
\]

\[
= \min \max \left( \begin{array}{c}
\left( s(f_{1}[A_{1}]) \right) \\
\vdots \\
\left( s(f_{1}[A_{m}]) \right)
\end{array}, \max \left( \begin{array}{c}
\left( s(f_{2}[A_{1}]) \right) \\
\vdots \\
\left( s(f_{2}[A_{m}]) \right)
\end{array}, \ldots, \max \left( \begin{array}{c}
\left( s(f_{m}[A_{1}]) \right) \\
\vdots \\
\left( s(f_{m}[A_{m}]) \right)
\end{array} \right) \right)
\]

\[
= \min \max \left( \begin{array}{c}
\left( s(f_{(n-1)+1}[A_{1}]) \right) \\
\vdots \\
\left( s(f_{(n-1)+1}[A_{m}]) \right)
\end{array}, \max \left( \begin{array}{c}
\left( s(f_{(n-1)+2}[A_{1}]) \right) \\
\vdots \\
\left( s(f_{(n-1)+2}[A_{m}]) \right)
\end{array}, \ldots, \max \left( \begin{array}{c}
\left( s(f_{2}[A_{1}]) \right) \\
\vdots \\
\left( s(f_{2}[A_{m}]) \right)
\end{array}, \ldots, \max \left( \begin{array}{c}
\left( s(f_{1}[A_{1}]) \right) \\
\vdots \\
\left( s(f_{1}[A_{m}]) \right)
\end{array} \right) \right) \right), \ldots, \max \left( \begin{array}{c}
\left( s(f_{m}[A_{1}]) \right) \\
\vdots \\
\left( s(f_{m}[A_{m}]) \right)
\end{array} \right) \right)
\]

\[
= \min \max \left( \begin{array}{c}
\left( s(f_{(n-1)+1}[A_{1}]) \right) \\
\vdots \\
\left( s(f_{(n-1)+1}[A_{m}]) \right)
\end{array}, \max \left( \begin{array}{c}
\left( s(f_{(n-1)+2}[A_{1}]) \right) \\
\vdots \\
\left( s(f_{(n-1)+2}[A_{m}]) \right)
\end{array}, \ldots, \max \left( \begin{array}{c}
\left( s(f_{2}[A_{1}]) \right) \\
\vdots \\
\left( s(f_{2}[A_{m}]) \right)
\end{array}, \ldots, \max \left( \begin{array}{c}
\left( s(f_{1}[A_{1}]) \right) \\
\vdots \\
\left( s(f_{1}[A_{m}]) \right)
\end{array} \right) \right) \right) \ldots, \max \left( \begin{array}{c}
\left( s(f_{m}[A_{1}]) \right) \\
\vdots \\
\left( s(f_{m}[A_{m}]) \right)
\end{array} \right) \right)
\]

\[\ldots, \max \left( \begin{array}{c}
\left( s(f_{1}[A_{1}]) \right) \\
\vdots \\
\left( s(f_{1}[A_{m}]) \right)
\end{array} \right) \right)
\]

}
\[
\begin{align*}
\max \left( \frac{s(f_j((n-1)t+2)[A_i])}{s(f_j((n-1)t+1)[A_i])} \right), \ldots, \max \left( \frac{s(\mathcal{P}(f_{(n-1)t+1}[A_i]))}{s(f_{(n-1)t+1}[A_i])} \right), \ldots, \max \left( \frac{s(f_j((n-1)t+1)[A_i])}{s(f_j((n-1)t+1)[A_i])} \right),
\end{align*}
\]

\[
= \min \left( s(\mathcal{R}(f_{(n-1)t+1}[A])) \right), s(\mathcal{R}(f_{(n-1)t+1}[A])), \ldots, s(\mathcal{R}(f_{(n-1)t+1}[A])), s(\mathcal{R}(f_{(n-1)t+1}[A])), \ldots,
\]

\[
= \min \left( s(\mathcal{R}(f_{(n-1)t+1}[A])) \right), s(\mathcal{R}(f_{(n-1)t+1}[A])), \ldots, s(\mathcal{R}(f_{(n-1)t+1}[A])), \ldots,
\]

\[
\text{if } \max(A^n) = \max(A^k) < \max(A^r) \text{ for } j_1 \neq y \text{ and } j_2 \neq y
\]

(From Corollary 2.1)

\[
= \min \left( s(\mathcal{R}(f_{(n-1)t+1}[A])) \right), s(\mathcal{R}(f_{(n-1)t+1}[A])), \ldots, s(\mathcal{R}(f_{(n-1)t+1}[A])), \ldots,
\]

(From Definition 8 and Theorem 2.1) \[\text{[Note: } a_{ij} = f(a_{ij}) \text{ if } f \in F_j \]}

\[
\begin{align*}
\max \left( \frac{s(f_j((n-1)t+2)[A_i])}{s(f_j((n-1)t+1)[A_i])} \right), \ldots, \max \left( \frac{s(\mathcal{P}(f_{(n-1)t+1}[A_i]))}{s(f_{(n-1)t+1}[A_i])} \right), \ldots, \max \left( \frac{s(f_j((n-1)t+1)[A_i])}{s(f_j((n-1)t+1)[A_i])} \right),
\end{align*}
\]

\[
= a_{1n}B^{n+1} + \min_{j_1} \left( \min_{j_1} \max_{j_1} \left( s(\mathcal{P}(A_{[i,j]} \bigtriangleup a[i,j])) \right) \right), \ldots, \min_{j_1} \left( \min_{j_1} \max_{j_1} \left( s(\mathcal{P}(A_{[i,j]} \bigtriangleup a[i,j])) \right) \right).
\]

Example 2.5:

\[
\begin{bmatrix}
3 & 0 & 1 & 4 \\
2 & 4 & 3 & 1 \\
3 & 4 & 2 & 3
\end{bmatrix}
\]

Let \( A = \begin{bmatrix} 3 & 0 & 1 & 4 \\ 2 & 4 & 3 & 1 \\ 3 & 4 & 2 & 3 \end{bmatrix} \) and \( B = 5 \). Now we want to find \( \text{MinMax}(A) \).

Solution:

There exists \( j_1 = 1 \) and \( j_2 = 3 \), such that \( A^{(1)} = \begin{bmatrix} 3 \\ 3 \end{bmatrix} \), \( A^{(3)} = \begin{bmatrix} 3 \end{bmatrix} \) and \( \max(A^1) = \max(A^3) < \max(A^r) \)

if \( y \neq 1 \) and \( y \neq 3 \). So,

\( Q.E.D. \)
\[
\text{MinMax}(A) = a_{i,j} B^{n-1} + \min_{p=\{j_1, \ldots, j_{n-1}\}} \max_{i=1}^{n-1} \{s(f_p[A(i,j) \ominus a_{i,j}])\},
\]
\[
= 3 \times 5^3 + \min \left\{ \text{MinMax} \left[ \begin{array}{ccc} 0 & 1 & 4 \\ 1 & 2 & 3 \end{array} \right], \text{MinMax}\left[ \begin{array}{cc} 2 & 4 \\ 1 & 2 \end{array} \right] \right\}.
\]
\[
= 3 \times 5^3 + \min\{s([2 & 3 & 4]), s([1 & 2 & 4])\} = 3 \times 5^3 + 1 \times 5^2 + 2 \times 5^1 + 4 \times 5^0 = 414.
\]

3. An Algorithm for Minimizing the Maximum Key

In this section, we will give a recursive algorithm for finding the most appropriate permutation of column vectors of the given access control matrix such that the maximal key yielded by Jan’s single key scheme is minimized.

The following algorithm will call the function MinMax(), which is a recursive function to return a vector \( R \) such that \( s(R) \) is minimaximal key value.

**Algorithm FMMS(A)**

**Input:** A matrix \( A \).

**Output:** A sequence \( J \) of columns such that the maximal key value computed according to \( J \) is minimized.

**Step 1:** \( R = \text{MinMax}(A) \);

**Step 2:** \( J = \text{Getcolumnindex}(R) \);

**Step 3:** \( \text{Return}(J) \);

**END FMMS.**

Note: After the most appropriate sequence \( J \) of columns is obtained, the key value for each user in the system can be computed by Jan’s single key access control method. For instance, suppose that \( \text{Depth 1:} \)

\[
K_i = \sum_{k=1}^{n} a_{i,k} B^{n-k}.
\]

It can be seen that the maximal value of all \( K_i \)'s is among minimized.

**Function Minimax(A)**

**Input:** A matrix \( A \).

**Output:** A vector \( R \) such that \( s(R) \) is the minimaximal key value.

**Step 1:** Find \( A^{(1)} \), \( A^{(2)} \), ..., \( A^{(c)} \).

**Step 2:** Find all minimax columns

**Algorithm FMMS(A)**

**Input:** A matrix \( A \).

**Output:** A sequence \( J \) of columns such that the maximal key value computed according to \( J \) is minimized.

**Step 1:** \( R = \text{MinMax}(A) \);

**Step 2:** \( J = \text{Getcolumnindex}(R) \);

**Step 3:** \( \text{Return}(J) \);

**END FMMS.**

Note: After the most appropriate sequence \( J \) of columns is obtained, the key value for each user in the system can be computed by Jan’s single key access control method. For instance, suppose that \( \text{Depth 1:} \)

\[
K_i = \sum_{k=1}^{n} a_{i,k} B^{n-k}.
\]

It can be seen that the maximal value of all \( K_i \)'s is among minimized.

**Function Minimax(A)**

**Input:** A matrix \( A \).

**Output:** A vector \( R \) such that \( s(R) \) is the minimaximal key value.

**Step 1:** Find \( A^{(1)} \), \( A^{(2)} \), ..., \( A^{(c)} \).

**Step 2:** Find all minimax columns

Consider the access control matrix

\[
A = \begin{bmatrix} 3 & 0 & 1 & 4 \\ 3 & 4 & 3 & 1 \end{bmatrix}
\]

again. We use FMMS algorithm to compute each key for each user. From Step 1 of FMMS, we have

Depth 1:
Combining (1), (2) and (3), we get

\[ = \text{Get\_Column\_Index}([a_{11} \ a_{23} \ a_{34} \ a_{32}]) \]

\[ = (1, 3, 4, 2) \]

From Step 3 of FMMS, we have

\[ \text{Return}(1, 3, 4, 2); \]

Since \( J = (j_1, j_2, j_3, j_4) = (1, 3, 4, 2) \), we can compute each users’ key through

\[ K_j = \sum_{k=1}^{n} a_{jk} B^{n-k}. \]

Thus,

\[ K_1 = s([a_{11} \ a_{13} \ a_{14} \ a_{12}]) = s([3 \ 1 \ 4 \ 0]) = 420, \]

\[ K_2 = s([a_{21} \ a_{23} \ a_{24} \ a_{22}]) = s([3 \ 3 \ 1 \ 4]) = 459, \]

\[ K_3 = s([a_{31} \ a_{33} \ a_{34} \ a_{32}]) = s([3 \ 3 \ 3 \ 4]) = 469. \]

4. Conclusions

In this paper, firstly, we give a brief review of Jan’s single key access control scheme. We have proposed an efficient way to assign a set of sequence numbers to the set of files such that the maximal key value is minimized. Using our approach, the memory for storing the keys yielded by Jan’s method will be significantly reduced.

References


Manuscript Received: Jun. 8, 2000
Accepted: Jun. 29, 2000