Controlling FD and MVD Inferences in Multilevel Relational Database Systems

Tzong-An Su, *Member, IEEE*, and Gultekin Ozsoyoglu, *Member, IEEE*

Abstract—**We investigate the inference problems due to functional dependencies (FD) and multivalued dependencies (MVD) in a multilevel relational database (MDB) with attribute and record classification schemes, respectively. For FD's, we first determine the set of functional dependencies to be taken into account in order to prevent FD-compromises. Then we prove that incurring minimum information loss to prevent compromises is an NP-complete problem. Finally, we give an exact algorithm to adjust the attribute levels so that no compromise due to functional dependencies occurs. For MVD's, we determine some necessary and sufficient conditions for MVD-compromises. We then determine the set of MVD's to be taken into account for controlling inferences. Finally, we give an algorithm to prevent MVD-compromises in a relation with conflict-free MVD's.**

Index Terms— **Compromise, data classification, database security, functional dependency, inference control, information loss, multilevel relational database, multivalued dependency, NP-complete.**

I. Introduction

A *multilevel relational database system (MDB)* is a relational database system which stores data of different security classifications and provides these data to users with different clearances. In a multilevel relational database system, data d is assigned a *classification level* $L(d)$, and each user u is assigned a *clearance level* $L(u)$. Only when $L(u) \geq L(d)$, where $\geq$ represents a partial ordering relation, the user u can access data d. There are three main data classification schemes, namely, *classification by records (tuples)*, *classification by attributes*, and *classification by elementary items (tuple components)* [5].

Research about multilevel relational database systems was first initiated in 1982 by the group of Woods Hole Summer Study on Multilevel Database Management Security. Since then, the topic has attracted researchers' attention, and some important results have been derived [4], [5], [9], [10], [6], [7], [12], [14], [15]. In Denning's paper [5], four types of attacks to multilevel database systems are discussed, and one approach, called commutative filters, is proposed to overcome inference problems. But there are still some open problems. One important issue is the inference problem due to integrity constraints (i.e., constraints that enforce the correctness of data in the database). Consider the following examples.

Example 1.1: Assume employee salaries in a company are based on an evaluation factor, called rank, of every employee. That is, employees who have the same rank get the same salary. Let R be a relation scheme in the company database, where R contains *EMPLOYEE-ID*, ..., *RANK*, and *SALARY* attributes. Assume the *SALARY* attribute is classified at TOP-SECRET level and the *RANK* attribute is at SECRET level. Also the following functional dependency holds in R:

$$RANK \rightarrow SALARY$$

which denotes the relationship that, within tuple t of R, for a given *RANK* value (e.g., t[*RANK*]) there is always a unique *SALARY* value (e.g., t[*SALARY*]). In other words, *RANK functionally determines SALARY* in R. Now, a user u with clearance SECRET can infer the salary information (which is unauthorized to him) if he also knows the mapping between *RANK* and *SALARY* attribute values. Note that under this classification scheme, the user u can always compromise the database since he certainly knows the mapping from his rank to his salary, and thus, he can use this information to infer salaries of other people with the same rank. □

Example 1.2: Let $R(SID, G_1, G_2, \cdots, G_m, GPA)$ be a relation scheme of a university database, where *SID* is student id number, $G_i, 1 \leq i \leq m$, represents the grade of the student with student id number *SID* in course C_i, and *GPA* represents his cumulative grade-point-average. It is clear that the functional dependency $G_1 G_2 \cdots G_m \rightarrow GPA$ holds in the database. If *SID*, and $G_i, 1 \leq i \leq m$ are classified as SECRET and *GPA* is classified as TOP-SECRET, then a user u with clearance level SECRET can infer the TOP-SECRET information *GPA* of any student. □

Example 1.3: Let $R = (M, S, W)$ be a relation scheme in a military database with record classification scheme, where M, S, W represent the name of the mission, the name of the warships involved in the mission, and weapons used in the mission. Assume every warship involved carries the same weapons, and in mission m_1 there are three warships and three types of weapons involved. Then, we have the relation r in the database shown in Fig. 1.

Now, if a user u with clearance level 4 knows the fact that every warship carries the same set of weapons, then u can infer

Manuscript received July 7, 1989; revised May 11, 1990. This work was supported by the National Science Fundation under Grant IRI-8910294 and was done when T.-A. Su was with The University of Toledo. The work of G. Ozsoyoglu was supported by the National Science Foundation under Grants DCR-8306616 and DCR-8605554. The preliminary form of this paper appeared in the Proceeding of the 1987 IEEE Symposium on Security and Privacy.

T.-A. Su was with Department of Computer Science and Engineering, The University of Toledo, Toledo, OH 43606. He is now with AT&T Bell Laboratories, Naperville, IL 60566.

G. Ozsoyoglu is with Department of Computer Engineering and Science, Case Western Reserve University, Cleveland, OH 44106.

IEEE Log Number 9101532.

Level	M	S	W
1	m_1	s_1	w_1
2	m_1	s_1	w_2
1	m_1	s_1	w_3
3	m_1	s_2	w_1
4	m_1	s_2	w_2
5	m_1	s_2	w_3
4	m_1	s_3	w_1
5	m_1	s_3	w_2
6	m_1	s_3	w_3

Fig. 1. A multilevel relation.

the tuple (m_1, s_2, w_3), (m_1, s_3, w_2), and (m_1, s_3, w_3) which are all unauthorized to him. □

The above examples illustrate that if the integrity constraints on the data are not properly reflected to the classification levels, users can infer unauthorized information from a multilevel relational database.

In this paper, we investigate the inference problems due to integrity constraints in an MDB. Specifically, we consider the security problems caused by two major types of data dependencies, i.e., functional dependencies and multivalued dependencies [17], [13]. For each type of data dependency, we show the inference mechanism (attacking method) and a level assignment method to prevent such kind of inferences.

Section II gives the definitions used in the paper. Section III describes the inferences arising from functional dependencies. We determine the number of functional dependencies to be taken into account in order to prevent the so-called FD-compromise. Also, we prove that preventing the FD-compromise, and at the same time, achieving the minimum information loss by changing the minimum number of attribute levels is NP-complete. Finally, we propose an algorithm to adjust attribute levels to prevent FD-compromises so that the minimum information loss goal is obtained. In Section IV, we first give necessary and sufficient conditions to prevent MVD-compromises due to a single MVD. We then give a necessary and sufficient condition to prevent MVD-compromises when there are multiple (conflict-free [16], [11]) MVD's. Finally, we present an algorithm to adjust the tuple levels in a relation so that MVD-compromises are eliminated. Section V concludes.

II. Definitions and Terminology

A. The Relational Data Model

An *attribute* is a property of some entity. As a convention, we use $A, B, C, \cdots$ for single attributes and $Z, Y, W, \cdots$ for sets of attributes. Corresponding to each attribute A is a set of possible values for the attribute. This set is called the *domain* of the attribute A, denoted by $DOM(A)$. A *relation scheme* R is a finite set of attributes $\{A_1, A_2, \cdots, A_n\}$, denoted by $R(A_1, A_2, \cdots, A_n)$. Let $DOM(R) = DOM(A_1) \times DOM(A_2) \times \cdots \times DOM(A_n)$, where $\times$ is the Cartesian product. A *relation instance* r over the relation scheme R is a finite set of mappings $\{t_1, t_2, \cdots, t_n\}$ from R to $DOM(R)$ with the restriction that for each mapping $t \in r$, $t(A_i)$ must be in $DOM(A_i)$, $1 \leq i \leq n$. The mappings are called *tuples* (or *records*). Given a set of attributes X, an X-value x is an assignment of values to the attributes in X from their respective domains. Given a relation r over the relation scheme R, $t[X]$ represents the value of the set of attributes X in the tuple t of relation r. The notations XY and xy represent the union of two sets of attributes X and Y and an XY-value, respectively.

Two important operations on relations are *projection* and *join*. Let r be a relation over R, and $X \subseteq R$. The projection of r on X, denoted by $\pi_X(r)$, is the relation $\{t[X] \mid t \in r\}$ over X, where $t[X]$ denotes the X-value of tuple t. The join of two relations r over $R(XY)$ and s over $S(XZ)$, where X, Y, and Z are mutually disjoint sets of attributes, is a relation q over $Q(XYZ)$. The relation q, denoted by $r \bowtie s$, is defined as $\{xyz \mid xy \in r, xz \in s\}$.

B. The Multilevel Relational Database

MDB denotes a multilevel relational database. Consider an MDB which uses the attribute classification scheme; that is, each attribute A_i in the MDB is assigned a classification level $L(A_i)$, $L(A_i) \in N$, and N is the set of natural numbers. Let $L(u)$, $L(u) \in N$, represent the clearance level of the user u. Then, the user u can access attribute A_i values of tuples iff $L(u) \geq L(A_i)$. On the other hand, if the MDB uses the record classification scheme, then each tuple (record) t in a relation r is assigned a classification level $L(t)$, $L(t) \in N$. The user u can access tuple t iff $L(u) \geq L(t)$.

C. FD-Compromises

Let R be a relation scheme in the MDB, X and Y be subsets of the attributes in R, and r be any instance of R. We say $X \rightarrow Y$, read "X *functionally determines* Y," if it is not possible that r has two tuples that agree in the components for all attributes in the set X, yet disagree in one or more components for attributes in the set Y [17]. For example, we have *RANK* $\rightarrow$ *SALARY* in Example 1.1 and $G_1G_2 \cdots G_m \rightarrow$ *GPA* in Example 1.2.

As shown in Examples 1.1 and 1.2 in Section I, unauthorized information in an MDB can be inferred using functional dependencies if the classification levels are not properly assigned. Consider a functional dependency (FD) $X \rightarrow Y$ in some relation R, where $X, Y \subseteq Attr(R)$, $Attr(R)$ denotes the attribute set of R. We say that there exists an *FD-compromise* due to the FD $X \rightarrow Y$ and the mapping between X and Y attribute values if, for any user u with clearance level $L(u)$, for all X_i in X, for some Y_j in Y, and $L(Y_j) > L(u) \geq L(X_i)$, u can infer the unauthorized information $t[Y_j]$ of some tuple t in R, by querying the authorized information $t[X_i]$ and by utilizing the associated mapping between X and Y.

Note that if a user knows the FD $X \rightarrow Y$, but not the associated mapping between X and Y attribute values then the knowledge of the FD $X \rightarrow Y$ is not sufficient for compromise. We assume that for any FD mentioned, the user also knows the related mapping.

D. MVD-Compromises

Let R, r, X, and Y be defined as in Section II-C. We say that there is a *multivalued dependency* (MVD) of Y on X, denoted

by $X \rightarrow\rightarrow Y$, if whenever t and s are two distinct tuples in r with $t[X] = s[X]$ (that is, t and s agree on the attributes in X), then r also contains tuples u and v where $u[X] = v[X] = t[X] = s[X]$, $u[Y] = t[Y]$ and $u[R-X-Y] = s[R-X-Y]$, and $v[Y] = s[Y]$ and $v[R-X-Y] = t[R-X-Y]$, where "$-$" denotes set difference.

In Example 1.3, we have $M \rightarrow\rightarrow S$ (or $M \rightarrow\rightarrow W$).

Given a multivalued dependency (MVD) $X \rightarrow\rightarrow Y$ in some relation R of an MDB with the record classification scheme, we say that there exists an *MVD-compromise* due to the MVD $X \rightarrow\rightarrow Y$, if, for any user u with clearance level $L(u)$, u can infer the unauthorized information of some tuple t in r (i.e., $L(t) > L(u)$) by querying the authorized tuple information and using the rules given in the MVD definition.

III. Inferences Due to Functional Dependencies and a Protection Methodology

Given a set F of FD's on a relation R with $U = Attr(R)$, let F^+, the closure of F, be the set of FD's logically implied by F. F^+ can be computed by Armstrong's Axioms [1], [17], [13]:

Reflexivity: If $Y \subseteq X \subseteq U$ then $X \rightarrow Y$.

Transitivity: If $X \rightarrow Y$ and $Y \rightarrow Z$, $X, Y, Z \subseteq U$, then $X \rightarrow Z$.

Union: If $X \rightarrow Y$ and$X \rightarrow Z$,$X, Y, Z \subseteq U$, then $X \rightarrow YZ$.

We say that two sets of FD's F and G are equivalent if $F^+ = G^+$.

In this section, we investigate the problem of inferences due to the users' knowledge of the set F of FD's and the associated mappings, and propose a method to prevent the FD-compromise. We assume that all attributes stored in the MDB have been assigned classification levels according to some "real-world" requirements. Due to some FD's existing in the MDB, FD-compromises may occur. Our main goal here is to transform the inference control problem due to FD's into an access control problem by introducing a classification level adjustment scheme for those attributes involved in the FD's.

A. A Sufficient Condition for Preventing FD-Compromises

In general, there may be several FD's existing in an MDB, and this in turn introduces some problems. The first problem is how many FD's we should take into account, and the second one is the effects of the interaction among FD's.

Given a set F of FD's, one can use Armstrong's Axioms to derive the closure F^+, and consider F^+ to prevent FD-compromises. However, the size of F^+ may be exponential in the size of F. The following lemma shows that if there is no FD-compromise caused by each FD in F then there is no FD-compromise in F^+.

Lemma 3.1: Consider the set F of FD's and the associated mappings. If every FD $X \rightarrow Y$ in F has the property that $L(X_i) \geq L(Y_j)$, for all $Y_j \in Y$, where X_i is some attribute in X then there is no FD-compromise in F^+.

Proof: Considering any FD in F alone, there is no FD-compromise. Moreover, any (repetitive) application of an Armstrong Axiom to FD's with no FD-compromise will give an FD without any FD-compromise. Q.E.D.

The above result shows how to prevent an FD-compromise by assigning classification levels to attributes involved in the FD's. We now consider the following example.

Example 3.1: Consider a set F of FD's in an MDB: $\{AB \rightarrow C, C \rightarrow A, BC \rightarrow D, ACD \rightarrow B\}$. Let the original classification assignments for attributes A, B, C, and D be such that $L(B) > L(C) > L(A) > L(D)$. Thus, FD's $AB \rightarrow C$, $C \rightarrow A$ and $BC \rightarrow D$ contain no FD-compromise, but $ACD \rightarrow B$ does. In order to avoid FD-compromises in F^+, we must adjust the classification levels such that all four FD's in F contain no FD-compromise. One way to do this is by adjusting the levels of A and C and letting them be equal to the level of B. As an alternative, we can let $L(B) = L(C)$ and have all four FD's contain no FD-compromise. □

Example 3.1 shows that we can adjust the classification levels of attributes in FD's to avoid FD-compromises. Since there are many ways to do the adjustments, we face with the problem of choosing the best one. Due to the real-world requirements, we cannot do the adjustments by lowering the classification levels; in that case, we reveal some unauthorized data. Therefore, the only way is to raise the levels. As a result of raising classification levels, some authorized data are then restricted, and we have the so-called "information loss." For the sake of information richness, we aim at finding a way which results in the minimum information loss.

B. The Minimum Information Loss Classification Adjustment Scheme

To reflect the actual information loss, we formulate the classification level adjustment problem as follows. Each attribute A_i at each allowable classification level m is associated with a weight w_{im}, where w_{im} is a positive integer, based on the usage and the importance of the attribute to the application. Under this scheme, we have $w_{im} \leq w_{in}$ if $m > n$, since the information about A_i at level m is at least as restricted as that at the level n. Thus, each time we raise the level of A_i, we will have a nonincreased weight. The information loss caused by the classification level adjustment can be defined as follows.

Information Loss: The difference between the total weight of attributes in the MDB before and after the adjustment.

Example 3.2: Consider Example 3.1 with the original weights 7, 1, 6, and 10 for the attributes A, B, C, and D, respectively. To prevent the FD-compromise, we can raise the level of C to $L(B)$. Assume the weight of attribute C at level $L(B)$ is 3. As the result of the adjustment, we have the information loss of 3. □

Our goal is to find a level adjustment method which eliminates FD-compromises and incurs the minimum weight (information) loss. Now we can state the problem as below.

Problem CLA: Given a set F of FD's existing in an MDB with the attribute classification scheme, does there exist a level adjustment scheme with a weight (information) loss $\leq K$ so that, after the adjustments, for any FD $X \rightarrow Y$ in F, we have $L(X_i) \geq L(Y_j)$, for all $Y_j \in Y$ and some $X_i \in X$?

The CLA problem turns out to be an NP-complete problem, and thus, there is no known efficient algorithm to solve it. Here, we prove it to be NP-complete by transforming a known

NP-complete problem, i.e., the hitting set problem, to it. The hitting set problem [8] is defined below.

Problem HS: Consider a collection C of subsets of a finite set S, and a positive integer $K \leq |S|$. Is there a subset $S' \subseteq S$ with $|S'| \leq K$ such that S' contains at least one element from each subset in C?

Theorem 3.1: Problem CLA is NP-complete.

Proof: First we transform HS to CLA. The transformation is as follows.

Let $S = \{E_1, E_2, \cdots, E_n\}, C = \{C_1, C_2, \cdots, C_m\}$, and the universal attribute set $U = S \cup \{X_1, X_2, \cdots, X_m\}$. Also, let $L(X_1) = L(X_2) = \cdots = L(X_m)$, $L(E_1) = L(E_2) = \cdots = L(E_n)$, $L(X_i) > L(E_j)$, $1 \leq i \leq m$, $1 \leq j \leq n$, and the weight of each attribute E_j, $E_j \in S$, at the level $L(E_j)$ be 2, and that at level $L(X_i)$ be 1. The set F of FD's is formed by

$$\begin{cases} C_1 \rightarrow X_1 \\ C_2 \rightarrow X_2 \\ . \\ . \\ . \\ C_m \rightarrow X_m. \end{cases}$$

Clearly this transformation takes polynomial time.

Next we prove that there exists a subset $S' \subseteq S$ with $|S'| \leq K$ such that S' contains at least one element from each subset in C iff we can find an adjustment scheme with a weight loss $\leq K$ so that there is no FD-compromise due to FD's in F.

($\Leftarrow$) Assume such a subset S' exists. Our adjustment scheme is as follows. For any C_i, $1 \leq i \leq m$, if there is only one E_j such that $E_j \in C_i$ and $E_j \in S'$, then we raise the level of E_j to $L(X_i)$. If there is more than one such E_j, then we choose one, say E_j, and raise its level to $L(X_i)$. Clearly, this adjustment has the weight loss $\leq K$ and from Lemma 3.1, F contains no FD-compromise.

($\Leftarrow$) Assume we have an adjustment scheme with the weight loss $\leq K$ so that there is no FD-compromise. Since $L(X_i) > L(E_j)$, for all E_j in C_i, $1 \leq i \leq m$, every FD in F must be adjusted. Therefore, we can let S' be the set of the attributes whose levels are adjusted. Since the total weight loss is $\leq K$, we know that the number of attributes which get adjusted is also $\leq K$ (Note that the weight loss for adjusting one attribute is at least 1.) Therefore, $|S'| \leq K$. Also, S' contains at least one element from each subset in C. Q.E.D.

C. Level Adjustment Algorithm to Prevent FD-Compromises

Although the CLA problem is NP-complete, and it seems that we should resort to an approximation algorithm to solve it, we, instead, choose to use an exact algorithm since both the number of FD's in F and the number of attributes involved are, presumably, not large in most applications.

Below we give an algorithm to adjust the classification levels of attributes so that no FD-compromise exists due to a given set of FD's. In algorithm CLA, a procedure, called FD-ADJUST, which actually does the adjusting work is called. For the efficiency of the algorithm, in steps 3 and 4 of the algorithm, we first delete those irrelevant FD's, i.e., those that do not have to be adjusted. Also, in procedure FD-ADJUST, we use the branch-and-bound method by keeping a current optimal value of the weight loss (OPT). When an attribute is adjusted, we compare the current total value of the weight loss (CNT) to the current optimal one (OPT). If the former is larger than the latter, the procedure need not continue; instead, we can begin to test another case.

Algorithm CLA:

Input:

- A set F of FD's.
- A partition UIN of the universal attribute set, where each block in UIN contains one attribute and its associated classification level.
- Weighting scheme of attributes in the MDB.

Output:

- A partition $UOUT$ of the universal attribute set, where each block in $UOUT$ contains attributes which have the same classification level and the associated classification level. The new classification level assignment represented by $UOUT$ eliminates the FD-compromises, and it incurs the minimum information loss.

1) Initialize. $OPT := \infty$;
2) Right Decomposition. For each FD in F with a composite right-hand side, i.e., the right-hand side is not a single attribute, decompose it as follows: Let $X \rightarrow A_1A_2 \cdots A_n \in F$, then decompose it into $\{X \rightarrow A_1, X \rightarrow A_2, \cdots, X \rightarrow A_n\}$. Let the result of the right decomposition of all FD's be F_r.
3) Find the largest classification level L_l in UIN.
4) Delete those FD's from F_r with the left-hand sides containing an attribute A with $L(A) = L_l$. Let F_t denote the resulting set.
5) Left Decomposition. Decompose each FD in ft in the following manner: Let $B_{i1}B_{i2} \cdots B_{il} \rightarrow A_i \in F_t$, then $F_i = \{(B_{i1}, A_i), (B_{i2}, A_i), \cdots, (B_{il}, A_i)\}$, $1 \leq i \leq m$, where $|F_t| = m$.
6) Form all the combinations of two-tuples in the set of F_i sets, $1 \leq i \leq m$, by taking one two-tuple from each F_i. Let *OPEN* denote the set of all combinations.
7) **repeat**
   Choose one combination C of two-tuples from *OPEN*;
   OPEN := *OPEN* $- \{C\}$;
   CALL FD-ADJUST (C); **until** *OPEN* $= \emptyset$
8) Terminate.

procedure FD-ADJUST (C);
begin
$CNT := 0$;
$U' := UIN$;
Let M be the list of attributes M_i appearing in the left-hand sides of all two-tuples in C, with $L(M_1) \geq L(M_2) \geq \cdots \geq L(M_n)$, where n is the number of distinct attributes appearing in the left-hand sides of all two-tuples in C;
for $i := n$ **down to** 1 **do**
  begin
  Let C_i be the subset of C with M_i on the left-hand

sides of two-tuples in C.
if there is any two-tuple (M_i, A) in C_i with $L(M_i) < L(A)$ **then**
begin
Choose the attribute K on the right-hand side of two-tuples in C_i with the largest classification level;
$B_k := B_k \cup B_{M_i}$
(* B_k is the block in U' containing attribute K *);
$U' := U' - B_{M_i}$;
$CNT := CNT + (w_{M_i,L(M_i)} - w_{M_i,L(K)})$;
(* $w_{M_i,L(M_i)}$ is the weight of attribute mi at level $L(M_i)$ *)
if $CNT \geq OPT$ **then** return
end
end;
$OPT := CNT$;
$UOUT := U'$
end;

D. Time Complexity of the Level Adjustment Algorithm

We now consider the time complexity of the adjustment algorithm CLA. In the procedure FD-ADJUST, there are two major parts contributing to the time complexity. The first one sets up the ordered list M, which takes $O(m \log m)$ time, where m is the number of two-tuples in C. The second part is the execution of the *for* loop, which takes $O(m)$ time to finish in the worst case. The worst case, $n = m$, occurs when each two-tuple in C has a distinct left-hand side.

In the main procedure, step 6 dominates the complexity, which produces $\prod_{i=1}^{m} |F_i|$ combinations. Although $|F_i|$ varies for different FD's of F, in the worst case, we can assume $|F_i| = |UIN|$. Thus, we have the total worst-case complexity $O(m \log m |UIN|^m)$.

IV. Inferences Due to Multivalued Dependencies

In this section, we discuss the security problems caused by the MVD inferences. We assume the MDB uses the record classification scheme i.e., each tuple in a relation is assigned a classification level. Under this assumption, we first examine the inferences from a single MVD, and then extend our results into the case of multiple MVD's.

A. Single MVD Inferences

In this section, we discuss the inference problems in an MDB due to a single MVD. We first define some notations used in the discussion. Let r be a relation over a relation scheme R, t be any tuple in r, and u represent a user of the MDB. We define

$T_u = \{t \mid t \in r \text{ and } L(t) \leq L(u)\}$
$T(X = x) = \{t \mid t \in r \text{ and } t[X] = x\}$
$T_{low}(X = x; u) = \{t \mid t \in T(X = x) \text{ and } L(t) \leq L(u)\}$
$T_{high}(X = x; u) = \{t \mid t \in T(X = x) \text{ and } L(t) > L(u)\}$.

The basic result is stated in Lemma 4.1 below.

Lemma 4.1: Consider an MDB with the record classification scheme. Let R be a relation scheme in the MDB and $X \rightarrow\rightarrow Y$ exist in R. Let u represent a user of the MDB with clearance level $L(u)$ and r be a relation over R. There is no MVD-compromise due to $X \rightarrow\rightarrow Y$ in r iff for any x in π_X (r) and any t_2 in $T_{high}(X = x; u)$ either a) there does not exist t_1 in $T_{low}(X = x; u)$ such that $t_1[Y] = t_2[Y]$ or b) there does not exist t_1 in $T_{low}(X = x; u)$ such that $t_1[R - X - Y] = t_2[R - X - Y]$.

Proof:

($\Rightarrow$) Assume there exist x in $DOM(X)$ and t_2 in $T_{high}(X = x; u)$ such that $t_2[Y] = t_i[Y]$ for some $t_i \in T_{low}(X = x; u)$ and $t_2[R - X - Y] = t_j[R - X - Y]$ for some $t_j \in T_{low}(X = x; u)$, then, by the definition of MVD, the user u can use t_i and t_j (which are authorized to him) to infer t_2. Thus, an MVD-compromise occurs.

($\Leftarrow$) First, assume a) is satisfied. Since a) is satisfied, for any t_2 in $T_{high}(X = x; u), t_2[Y]$ cannot be inferred by the user u using the rules in the MVD definition, and thus t_2 can not be inferred. Therefore, no MVD-compromise will occur. Now assume b) is satisfied. Using the same argument, one can also prove that no MVD-compromise will occur. Q.E.D.

Lemma 4.1 describes a procedural technique to prevent MVD-compromises by assigning proper classification levels to tuples in a given relation. Consider the following example.

Example 4.1: Consider the Example 1.3 with the MVD $M \rightarrow\rightarrow S$ (or $M \rightarrow\rightarrow W$). We can change the classification levels of tuples in r such that the condition in Lemma 4.1 is satisfied and thus, there is no MVD-compromise. Note that in Fig. 2, the levels of tuples (m_1, s_1, w_3), (m_1, s_2, w_1), (m_1, s_3, w_1) and (m_1, s_3, w_2) have now been changed. □

Two variations of Lemma 4.1 which use the notations from relational algebra are listed in Lemma 4.2 and 4.3.

Lemma 4.2: Consider an MDB with the record classification scheme. Let R be a relation scheme in the MDB, r be a relation over R, and $X \rightarrow\rightarrow Y$ exist in R. Also let $P = \pi_Y(T_{low}(X = x; u))$ and $Q = \pi_{R-X-Y}(T_{low}(X = x; u))$, where π denotes the projection operation of a relation. There is no MVD-compromise due to $X \rightarrow\rightarrow Y$ iff for all x, $x \in \pi_X(r)$ and for all u, $\pi_{R-X}(T_{low}(X = x; u)) = P \times Q$, where $\times$ denotes the Cartesian product.

Lemma 4.3: Consider an MDB with the record classification scheme. Let R be a relation scheme in the MDB, $X \rightarrow\rightarrow Y$ exist in R, $P = \pi_{XY}(T_u)$ and $Q = \pi_{X(R-X-Y)}(T_u)$, where π denotes the projection operation of a relation. There is no MVD-compromise due to $X \rightarrow\rightarrow Y$ iff for all u, $T_u = P \bowtie Q$, where $\bowtie$ denotes the join operation.

B. Compromises Due to a Set of MVD's

In Section IV-A, we have considered the inferences due to a single MVD. In general, there may be several MVD's in a given relation. Therefore, we have the same problem as in the case of functional dependencies, namely, dealing with a set of MVD's and still preventing the MVD-compromises.

Given a set of MVD's, one can use inference rules to derive new MVD's. Since these new MVD's are implied by the old ones, we should consider them together. Ref. [3] describes a complete set of inference rules for MVD's as follows.

1) Complementation: Let X, Y, and Z be sets such that their union is R, and $Y \cap Z \subseteq X$. Then $X \rightarrow\rightarrow Y$ iff $X \rightarrow\rightarrow Z$.

Level	M	S	W
1	m_1	s_1	w_1
2	m_1	s_1	w_2
5	m_1	s_1	w_3
4	m_1	s_2	w_1
4	m_1	s_2	w_2
5	m_1	s_2	w_3
6	m_1	s_3	w_1
6	m_1	s_3	w_2
6	m_1	s_3	w_3

Fig. 2. A multilevel relation without any MVD-compromises.

2) Reflexivity: If $Y \subseteq X$, then $X \rightarrow\rightarrow Y$.
3) Augmentation: If $Z \subseteq W$ and $X \rightarrow\rightarrow Y$ then $XW \rightarrow\rightarrow YZ$.
4) Transitivity: If $X \rightarrow\rightarrow Y$ and $Y \rightarrow\rightarrow Z$ then $X \rightarrow\rightarrow Z - Y$.

Given a set M of MVD's, let M^+, the closure of M, denote the set of MVD's which can be derived from M using the inference rules 1–4. Investigating all the MVD's in M^+ for MVD-compromises is a time-consuming task. However, the following theorem gives us a short-cut.

Theorem 4.1: Given a set M of MVD's, if there is no MVD-compromise due to any MVD in M then there exists no MVD-compromise due to M^+.

Proof: See the Appendix.

Theorem 4.1 tells us that to prevent MVD-compromises, we can consider only MVD's in the given set M instead of its closure M^+.

Theoretically, given a relation scheme, any possible set of MVD's can exist in the relation scheme. It is claimed that [16] some of these sets of MVD's are unrealistic, and do not exist in real-world examples. Therefore, in this paper, we investigate only those sets of MVD's which are claimed to exist in the real world i.e., *conflict-free* MVD's [11], [16]. In [16], it is shown that in real-world applications the only sets of MVD's that need ever be considered are conflict-free.

Before we give the definition of conflict-free MVD's, we need some more terminology. Let M be a set of MVD's in relation R, the set $\{Y_1, \cdots, Y_n\}$ is a *dependency basis* for X, (X, $Y_i \subseteq R$) if $X \rightarrow\rightarrow Y_i$ is in M^+, $Y_i \neq \emptyset$, $X \cap Y_i = \emptyset$, $1 \leq i \leq n$, and for any MVD $X \rightarrow\rightarrow Z$ implied by M, $Z - X$ is the union of some of the Y_i's. We use $DEP(X) = \{Y_1, \cdots, Y_n\}$ or $X \rightarrow\rightarrow Y_1|\cdots|Y_n$ to denote the dependency basis of X. It is known [2] that a dependency basis must exist for any X and M. Given a set M of MVD's, X is a *key* of M if there exists an MVD $X \rightarrow\rightarrow Y$ in M. Each Y in $DEP(X)$ is called an *essential dependent* of X if $X \rightarrow\rightarrow Y$ cannot be derived from M without using an MVD having the left-hand side X. A key is *essential* if it has an essential dependent.

We are now in a position to define conflict-free MVD's. Let M be a set of MVD's. M is *conflict-free* if for any two essential keys X and Y of M, the following condition holds:

$$DEP(X) = \{V_1, \cdots, V_k, X_1, \cdots, X_i, (Z_x Y_1 \cdots Y_j)\}$$

and

$$DEP(Y) = \{V_1, \cdots, V_k, Y_1, \cdots, Y_j, (Z_y X_1 \cdots X_i)\}$$

where

$$\{V_1, \cdots, V_k\} \subseteq DEP(X \cap Y) \text{ and } Z_x X = Z_y Y.$$

Example 4.2: Consider the relation $R(P, U, S, M, W)$ in an MDB, which represents a Person in some Mission, the Unit where the person comes from, the Speciality of the person, and the Weapons used in the mission, respectively. We have the following MVD's existing in R :

$$P \rightarrow\rightarrow U \qquad P \rightarrow\rightarrow S \qquad P \rightarrow\rightarrow MW$$
$$M \rightarrow\rightarrow W \qquad M \rightarrow\rightarrow PUS$$

It is easy to see that $DEP(P) = \{U, S, MW\}$ and $DEP(M) = \{W, PUS\}$. Moreover, the above set of MVD's is conflict-free since one can verify that the following substitutions on the conflict-free MVD definition are correct:

$$i = 2\ j = 1\ k = 0$$
$$X = P\ X_1 = U\ X_2 = S\ Z_x = M$$
$$Y = M\ Y_1 = W\ Z_y = P.$$ □

Another data dependency related to multivalued dependency is called the join dependency (JD). Let R be a relation scheme. A decomposition of R is a set $D = \{R_1, R_2, \cdots, R_k\}$ of subsets of R such that $R = R_1 \cup R_2 \cup \cdots \cup R_k$. It is not necessary that the R_i's be disjoint. Let $D = \{R_1, R_2, \cdots, R_k\}$ be a decomposition of R. A relation r over R satisfies the join dependency $\bowtie [R_1, R_2, \cdots, R_k]$ (or $\bowtie [D]$) if $r = \pi_{R_1}(r) \bowtie \pi_{R_2}(r) \bowtie \cdots \bowtie \pi_{R_k}(r)$. Given a relation scheme R with a set M of MVD's and a decomposition D on R, we say that M is equivalent to the join dependency $\bowtie [D]$ iff, for any relation instance r over R, r satisfies M iff r satisfies $\bowtie [D]$.

An important property of a conflict-free set of MVD's is stated below [16].

Lemma 4.4: Given a relation scheme R and a conflict-free set M of MVD's over R, there is a unique join dependency $J = \bowtie [X_1, X_2, \cdots, X_n]$ over R such that M is equivalent to J.

Using this property, for conflict-free MVD's, we can derive a necessary and sufficient condition for preventing MVD-compromises as stated in Theorem 4.2.

Theorem 4.2: Consider a relation scheme R, a conflict-free set M of MVD's over R, and a relation instance r over R. Let $J = \bowtie [X_1, X_2, \cdots, X_n]$ be the equivalent join dependency of M, and $P_i = \pi_{X_i}(T_u)$, $1 \leq i \leq n$. There is no MVD-compromise in r due to M iff for all u, $T_u = P_1 \bowtie P_2 \bowtie \cdots \bowtie P_n$.

Proof:

($\Rightarrow$) We first prove that for all u, T_u satisfies M. Let $X \rightarrow\rightarrow Y$ be an arbitrary MVD in M, and $s, t \in T_u$ with $s[X] = t[X]$. By the definition of the MVD, we can generate two different tuples v and w with $v[X] = w[X] = s[X] = t[X]$ in r, because r satisfies $X \rightarrow\rightarrow Y$. Since there is no MVD-compromise due to M (and thus due to $X \rightarrow\rightarrow Y$), v and w should be in T_u. Therefore, T_u satisfies $X \rightarrow\rightarrow Y$. Same

arguments can apply to other MVD's in M, and we know that T_u satisfies M. Because, for conflict-free set M of MVD's, $\bowtie [X_1, X_2, \cdots, X_n]$ is the equivalent join dependency of M, and T_u, for all u, satisfies M, therefore, for all u, $T_u = P_1 \bowtie P_2 \bowtie \cdots \bowtie P_n$.

($\Leftarrow$) Since for all u, $T_u = P_1 \bowtie P_2 \bowtie \cdots \bowtie P_n$, T_u satisfies the join dependency $\bowtie [X_1, X_2, \cdots, X_n]$. Because $\bowtie [X_1, X_2, \cdots, X_n]$ is the equivalent join dependency of M, therefore, for all u, T_u must also satisfy M. Thus, for any user u, the tuples which u can infer are always in T_u, and there is no MVD-compromise. Q.E.D.

Theorem 4.2 is the general case of Lemma 4.1–4.3. A simple explanation is as follows. T_u represents the set of tuples authorized to be accessible to the user u. If, for each user u, T_u satisfies J (and thus, M), then all the tuples which can be inferred by the user u using the MVD definition (or JD definition) are always in T_u. Thus, u cannot infer unauthorized tuples and therefore there is no MVD-compromise. Theorem 4.2 gives us a guideline to prevent MVD-compromises due to a set of conflict-free MVD's. In the next section, we transform the MVD inference problem into an access control problem by raising levels of some tuples so that the MDB satisfies Theorem 4.2, and thus there exists no MVD-compromises.

C. Level Adjustment Algorithm to Prevent MVD-Compromises

1) Level Adjustment Algorithm: Assume all the data stored in the MDB have been assigned classification levels by using the record classification scheme. Due to some MVD's existing in the MDB, there may be MVD-compromises. Our goal is to find an algorithm which adjusts the classification levels of records (tuples) in the MDB so that there is no MVD-compromise. Based on Theorem 4.2, we propose the following algorithm to adjust the security levels of tuples and prevent MVD-compromises. We first find the equivalent join dependency of the conflict-free set of MVD's, and then perform the actual adjustment work. Thus, given a set M of conflict-free MVD's existing in a relation scheme R, our algorithm can be divided into three phases. The first phase finds the dependency bases of keys of all MVD's in M. Based on these dependency bases, phase two finds the database scheme S where the join dependency $\bowtie [S]$ is equivalent to M. The third phase does the actual adjustment work. Note that for the sake of completeness we have included below some known results (i.e., procedure DEP and DECOM). For the relevant theory, we refer readers to [11] and [16].

Phase 1: Finding dependency bases. The following procedure is a modified version of the one in [2]. Similar algorithms can be found in [17] and [13]. The procedure DEP takes a relation scheme R and a set M of conflict-free MVD's in R as input. The output of the procedure DEP consists of dependency bases of all the keys in M.

```
procedure DEP
begin
for each key X in M do
      begin
      BASIS(X) := {{A}|A ∈ X} ∪ {R − X};
      FLAG : = "T";
      while FLAG do
            begin
            FLAG : = "F";
            for each MVD W →→ Z in M do
                  begin
                  Y := ∪{V|V ∈ BASIS(X) and V∩W ≠ ∅};
                  Z1 := Z − Y;
                  if Z1 is not empty and is not a union of
                  elements of BASIS(X) then
                        begin
                        FLAG : = "T";
                        BASIS(X) := BASIS(X) ∪ {Z1};
                        repeat
                        find a pair of sets Z2 and Z3 in
                        BASIS(X) that are not disjoint and
                        replace them by the sets Z2 − Z3,
                        Z3 − Z2, and Z2 ∩ Z3,
                        throwing away the empty set
                        until BASIS(X) consists of a
                        disjoint collection of sets
                        end
                  end
            end
      BASIS(X) := BASIS(X) - {{A} | A ∈ X};
      return BASIS(X)
      end
end;
```

Phase 2: Finding the set of relation schemes S from R such that $\bowtie [S]$ is equivalent to M. The following algorithm is from [11]. The procedure DECOM accepts the relation scheme R, the set M of conflict-free MVD's described in phase 1, and the output of procedure DEP (i.e., dependency bases of keys in M) as its input, and generates a set S of relation schemes from R such that $\bowtie [S]$ is equivalent to M.

```
procedure DECOM;
begin
S := {R};
for each key X in M do
      begin
      C1 := ∅;
      for each Y in S such that Y contains X do
            begin
            S := S − {Y};
            C1 := C1 ∪ {X(Y ∩ W)|Y ∩ W ≠ ∅, for
            some W in DEP(X)};
            end;
      S := S ∪ C1
      end
end;
```

Example 4.3: Consider the relation scheme R and its associated conflict-free set M of MVD's in Example 4.2. Applying procedure DEP on R, we obtain $DEP(P) = \{U, S, MW\}$ and $DEP(M) = \{W, PUS\}$. Based on these dependency bases, we can use procedure DECOM to derive the equivalent join dependency of R which is $\bowtie [PU, PS, PM, MW]$. □

Phase 3: Adjusting the levels of records. The procedure MVD-ADJUST adjusts the security levels of tuples in a

relation instance r of R and eliminates MVD-compromises.

The basic idea of the procedure MVD-ADJUST is based on Theorem 4.2. Let $S = \{S_1, S_2, \cdots, S_m\}$ be the set of relation schemes derived from Phase 2, and the clearance levels of the MDB users range from 1 to LNO with the level at LNO having the highest authority. The procedure takes each clearance level i starting from LNO and performs the following work in the descending order of clearance levels.

For each clearance level i, let T_i be the set of tuples which are authorized to be accessible to users with level i (i.e., $T_i = \{t \mid t \in r, L(t) \leq i\}$). Because of the existence of M in r, there are MVD-compromises in T_i. Part of the compromises may arise from the inferences of tuples with levels i by those users with levels less than i. To eliminate this type of MVD-compromises, we first divide T_i into two parts R_i and R_{i-1} where R_i is the set of tuples with classification levels equal to i and R_{i-1} is the set of tuples with levels less than i. Next, we raise the levels of some tuples in R_{i-1} to level i. Let the resulting R_{i-1} (after the adjustment) be R'_{i-1}. Now, if R'_{i-1} satisfies $\bowtie [S]$ (or M) then all the tuples which can be inferred by users with levels less than i are always in R'_{i-1}. Thus, the part of MVD-compromises described above can be eliminated.

So far, we have only taken care of one type of MVD-compromises in T_i, i.e., compromises caused by the inferences of tuples with level i. Other MVD-compromises may still exist in T_i, e.g., the inferences of tuples with level $i-1$ by users with levels less than $i-1$. Since, after the above adjustment, users with levels less than i cannot compromise tuples with level i, to eliminate the inferences of tuples with level $i-1$ by users with levels less than $i-1$, we can consider the current set of tuples with levels equal to or less than $i-1$ (i.e., R'_{i-1}) and repeat the process described above. To completely eliminate MVD-compromises, the above process must be repeated until i is equal to level 2.

Example 4.4: Consider the relation scheme R and the associated conflict-free set M of MVD's in Example 4.2. From Example 4.3, we know that the equivalent join dependency of M is $\bowtie [S] = \bowtie [PU, PS, PM, MW]$. Let r be a relation instance of R. Fig. 3(a) shows the relation r and the classification levels of tuples in r. As shown in r, the classification levels for tuples are 1, 2, and 3, i.e., LNO = 3. For simplicity of the discussion, we refer to a tuple using its identification number which is stored under the attribute Id.

It is easy to see that MVD-compromises exist in r (e.g., tuple 11 can be inferred from tuples 12 and 13). Using the process described above, we first divide r (or, in this example, T_3) into $R_3 = \{4, 11\}$, and R_2 contains the remaining tuples with levels less than 3. We then raise the levels of tuples 3, 7, 8, and 12 in R_2 to level 3. This adjustment will prevent users with levels less than 3 from inferring tuples with level 3. The resulting R'_2 is shown in Fig. 3(b). As shown in Fig. 3(b), R'_2 satisfies $\bowtie [S]$.

Although we have eliminated the inferences of tuples with level 3, we still need to consider the inferences of tuples with level 2 from users with levels less than 2. Therefore, we divide T_2 (or R'_2) into $R_2 = \{13\}$ and $R_1 = \{1, 2, 5, 6, 9, 10, 14\}$ and try to raise the levels of tuples in R_1 to level 2. After carefully checking on R_1, we know that R_1 has already satisfied $\bowtie [S]$,

Id	Level	P	U	S	M	W
1	1	p_1	u_1	s_1	m_1	w_1
2	1	p_1	u_1	s_1	m_1	w_2
3	2	p_1	u_1	s_1	m_2	w_3
4	3	p_1	u_1	s_1	m_2	w_4
5	1	p_1	u_1	s_3	m_1	w_1
6	1	p_1	u_1	s_3	m_1	w_2
7	2	p_1	u_1	s_3	m_2	w_3
8	1	p_1	u_1	s_3	m_2	w_4
9	1	p_2	u_1	s_2	m_1	w_1
10	1	p_2	u_1	s_2	m_1	w_2
11	3	p_3	u_2	s_3	m_2	w_3
12	1	p_3	u_2	s_3	m_2	w_4
13	2	p_3	u_2	s_4	m_2	w_3
14	1	p_3	u_2	s_4	m_2	w_4

(a)

Id	Level	P	U	S	M	W
1	1	p_1	u_1	s_1	m_1	w_1
2	1	p_1	u_1	s_1	m_1	w_2
5	1	p_1	u_1	s_3	m_1	w_1
6	1	p_1	u_1	s_3	m_1	w_2
9	1	p_2	u_1	s_2	m_1	w_1
10	1	p_2	u_1	s_2	m_1	w_2
13	2	p_3	u_2	s_4	m_2	w_3
14	1	p_3	u_2	s_4	m_2	w_4

(b)

Id	Level	P	U	S	M	W
1	1	p_1	u_1	s_1	m_1	w_1
2	1	p_1	u_1	s_1	m_1	w_2
5	1	p_1	u_1	s_3	m_1	w_1
6	1	p_1	u_1	s_3	m_1	w_2
9	11	p_2	u_1	s_2	m_1	w_1
10	1	p_2	u_1	s_2	m_1	w_2
14	1	p_3	u_2	s_4	m_2	w_4
13	2	p_3	u_2	s_4	m_2	w_3
3	3	p_1	u_1	s_1	m_2	w_3
4	3	p_1	u_1	s_1	m_2	w_4
7	3	p_1	u_1	s_3	m_2	w_3
8	3	p_1	u_1	s_3	m_2	w_4
11	3	p_3	u_2	s_3	m_2	w_3
12	3	p_3	u_2	s_3	m_2	w_4

(c)

Fig. 3 Illustration of Procedure MVD-ADJUST. (a) r before level adjustments. (b) R'_2 ($=T_2$). (c) r after level adjustments.

i.e., tuples with levels greater than or equal to 2 cannot be inferred by users with level 1. Thus, there is no need to do any level adjustment on R_1. Also, since the level 1 is the lowest level in the MDB (or in terms of the notation described above, i = 2), we can terminate the process and have the relation r without any MVD-compromise. The final r is shown in Fig. 3(c). □

In general, when we raise levels of tuples to prevent MVD-compromises, there can be several ways to do the adjustment. In Example 4.4, instead of raising the levels of tuples 3, 7, 8,

and 12, we can raise the levels of tuples 8, 12, 13, and 14 to 3 and prevent the compromises of tuples 4 and 11. Therefore, we have the same problem as in the case of the FD-compromise, namely, choosing the best adjustment scheme which incurs minimum information loss. The information loss in the MVD-compromise case is similar to that in the FD-compromise case and can be defined as follows.

Each tuple t_i at each allowable classification level m is associated with a weight w_{im}. Under this scheme, we have $w_{im} \leq w_{il}$ if $m > l$. Thus, each time we raise the level of a tuple t_i, we will have a nonincreased weight. We can define the information loss in the case of MVD-compromise as the difference between the total weight of tuples in the relation r after and before the level adjustment. Based on this definition, we now describe the method used in the procedure MVD-ADJUST for raising the levels of tuples in a relation. This method not only prevents MVD-compromises but, at the same time, reduces the information loss caused by the level adjustment.

The method is implemented from statements 4 to 9 in the procedure MVD-ADJUST. We consider a specific T_i and its partitions into R_i and R_{i-1} as described in previous paragraphs. To prevent the inferences of tuples in R_i (i.e., tuples with levels equal to i) from users with levels less than i, we must raise levels of some tuples in R_{i-1}. Our method to raise levels is based on the following corollary of Theorem 4.2.

Corollary 4.1: Let S, R_i, R_{i-1} be defined as before, and t_j be a tuple in R_i. If there exists an S_k, $S_k \in S$, $1 \leq k \leq m$, such that $t_j[S_k] = v_{jk}$ and v_{jk} is not in $\pi_{S_k}(R_{i-1})$ then users with clearance level less than i cannot infer the tuple t_j.

Proof: Since v_{jk} is not in $\pi_{S_k}(R_{i-1})$, by using the join dependency $\bowtie [S]$ (and thus the MVD inference rules in Section II-D), users with levels less than i cannot generate the tuple t_j. Q.E.D.

Example 4.5: Consider the relation T_2 shown in Fig. 3(b). We can divide T_2 into R_2 = {13} and R_1 = {1, 2, 5, 6, 9, 10, 14}. Since $t_{13}[MW](= m_2 w_3)$ does not belong to $\pi_{MW}(R_1)$, users with level 1 cannot infer t_{13}. □

Before we raise the levels of tuples in R_{i-1} to level i, we first check R_i to see whether there exist tuples which cannot be inferred. If there are, we just remove them from R_i and store them in the final resulting set, i.e., RESULT. These actions are performed in statement 5.

To reduce the information loss, we define some parameters to determine which tuples in R_{i-1} need to have their levels raised. First consider R_{i-1}. For each v_{jk} in $\pi_{S_k}(R_{i-1})$, $1 \leq k \leq m$, $1 \leq j \leq |\pi_{S_k}(R_{i-1})|$, we compute the total information (weight) loss due to raising levels of tuples in R_{i-1} to the level i, where the values of S_k in these tuples are equal to v_{jk}. Let this total weight loss be wl_{jk}. A similar process applies to R_i. For each v_{jk} in $\pi_{S_k}(R_i)$, $1 \leq j \leq |\pi_{S_k}(R_i)|, 1 \leq k \leq m$, we count the total number of tuples in R_i where the values of S_k in these tuples are equal to v_{jk}. Let this number be N_{jk}.

To measure the information loss and thus, to determine which tuples in R_{i-1} should be adjusted, for each wl_{jk} defined above, we define the parameter p_{jk} as the ratio of wl_{jk} to N_{jk} ($p_{jk} = N_{jk}/wl_{jk}$), where N_{jk} is the number of tuples in R_i with the S_k value equal to v_{jk}. The meaning of p_{jk} can be explained as follows. If we raise the levels of tuples in R_{i-1} having the S_k value of v_{jk} then we can eliminate inferences of p_{jk} tuples in R_i with one unit weight loss. It is clear that we should choose the largest p_{jk} and raise the levels of tuples with the corresponding value v_{jk} in S_k, since choosing the largest p_{jk} means that we can eliminate the maximum number of inferences with one unit weight loss.

The procedure MVD-ADJUST below accepts the relation r and the relation scheme S generated in phase 2 as its input, and adjusts the classification levels of tuples in r to eliminate MVD-compromises. The output of the algorithm is the adjusted relation r which is stored in RESULT.

```
procedure MVD-ADJUST;
 begin
 RESULT: = ∅ ;
 T := r;
1 for i := LNO downto 2 do (* LNO is the number of
   security levels in the MDB *)
            begin
2 R_i := {t | t ∈ T, L(t) = i};
3 R_{i-1} := {t | t ∈ T, L(t) < i};
4 while R_i ≠ ∅ do
            begin
5    for each S_k in S do
           for each v_jk in π_{S_k}(R_i) do
                 if v_jk not in π_{S_k}(R_{i-1}) then
                 begin
                 MOVE := {t | t ∈ R_i, t[S_k] = v_jk};
                 RESULT : = RESULT ∪ MOVE;
                 R_i := R_i − MOVE;
                 end;
6    for each S_k in S do
          for each v_jk in π_{S_k}(R_{i-1}) do
            wl_jk := total weight loss due to raising tuples
                        in R_{i-1}
            with S_k value of v_jk to level i;
7    for each S_k in S do
           for each v_jk in π_{S_k}(R_i) do
               N_jk : = total number of tuples with S_k value
                        of v_jk;
8    for each S_k in S do
           for each v_jk in π_{S_k}(R_i) do
               p_jk := N_jk/wl_jk;
9    let v_jk be the value corresponding to the maximum p_jk
     (break the tie arbitrarily);
     RAISE : = {t | t ∈ R_{i-1}, t[S_k] = v_jk};
     for each t in RAISE do
            L(t) := i;
     R_{i-1} := R_{i-1} − RAISE;
     MOVE := {t | t ∈ R_i, t[S_k] = v_jk};
     RESULT := RESULT ∪ RAISE ∪ MOVE;
     R_i := R_i − MOVE
     end;
     T := R_{i-1}
   end
end;
```

PU				PS				PM				MW			
v	wl	N	p	v	wl	N	p	v	wl	N	p	v	wl	N	p
p_1u_1	12	1	0.08	p_1s_1	5	1	0.2	p_1m_1	8	0	1	m_1w_1	6	0	0
p_2u_1	4	0	0	p_1s_3	7	0	0	p_1m_2	4	1	0.25	m_1w_2	6	0	0
p_3u_2	5	1	0.2	p_2s_2	4	0	0	p_2m_1	4	0	0	m_2w_3	3	1	0.33
				p_3s_3	2	1	0.5	p_3m_2	5	1	0.2	m_2w_4	6	1	0.16
				p_3s_4	3	0	0								

Fig. 4. Computation of parameters v, wl, N, and p.

Finally we illustrate the procedure MVD-ADJUST by using the following example.

Example 4.6: Let R, r, M, S, and LNO be the same as defined in Example 4.4. Also, assume the weights of tuples at level 3, 2, and 1 are 1, 2, and 3, respectively. We now perform the procedure MVD-ADJUST on r. From Example 4.4, we split r into R_3 and R_2. R_3 = {4, 11} and R_2 contains those remaining tuples with levels less than 3 in r. At this point, all the values in π_{S_k} (R_3), $1 \leq k \leq m$, can be found in the corresponding π_{S_k} (R_2). Therefore, both of the two tuples in R_3 can be inferred using the tuples in R_2 and thus statement 5 can be skipped.

Now we must determine which tuples in R_2 need to have their levels raised in order to prevent compromises on the tuples in R_3. We do this by first calculating the parameter p_{jk} for each value v_{jk} in $\pi_{S_k}(R_2)$, $1 \leq j \leq |\pi_{S_k}(R_2)|$, $1 \leq k \leq m$. Fig. 4 shows the values of each p_{jk}. For example, adjusting the tuples in R_2 with $PU = p_1u_1$, i.e., raising their levels to 3, will eliminate the inference of the first tuple in R_3 and the p value for p_1u_1 is 1/12, where 12 is the total weight loss due to raising the seven tuples in R_2 to level 3, and 1 indicates that there is only one tuple in R_3 which can be protected from the MVD-inference due to this adjustment.

As a result of the exhaustive calculation of all possible p values, we find that p_3s_3 has the maximum p value of 0.5. We thus raise the levels of tuples in R_2 with PS value equal to p_3s_3 to 3 and this avoid the inference of the second tuple in R_3. We implement that by changing the level of $p_3u_2s_3m_2w_4$ of R_2 and then moving it with the second tuple of R_3 to the RESULT set. This process (while loop) is repeated until all the tuples in R_3 are moved to RESULT set. The current R_2 is shown in Fig. 3(b). In fact, R_2 is the subrelation of r which can be accessed by users with clearance level 2 without any MVD-compromise on tuples with level 3. Although users with levels less than or equal to 2 cannot infer tuples with level 3 at this point, still, level 1 users may infer level 2 tuples in R_2. Therefore, we consider the current R_2 as a new relation which may have MVD-compromises and apply the entire process described above again (This will be the second iteration of statement 1.) The final result is shown in Fig. 3(c). □

The correctness of procedure MVD-ADJUST is stated and proved in the following theorem.

Theorem 4.3: Given a relation scheme R, a conflict-free set M of MVD's over R and a relation instance r of R, let $S = \{S_1, S_2, \cdots, S_m\}$ be a decomposition over R, and $\bowtie [S]$ be the equivalent join dependency of M. After performing the procedure MVD-ADJUST on r, there is no MVD-compromise in r.

Proof: We first show that the procedure terminates. Since the for loop in statement 1 iterates LNO − 1 times (limited number of iterations), what we need to show is the termination of the while loop in statement 4. For each i (and thus R_i), we remove some tuples from R_i in each iteration of the while loop. In each iteration, two types of tuples in R_i will be removed (if any). One type is those tuples which can be inferred by using tuples in R_{i-1} and the other type is those which cannot. Because tuples in R_i always belong to one of these two categories, as the iterations continue, R_i will become empty, and the while loop will terminate.

Now we show that MVD-ADJUST correctly adjusts the classification levels and prevents MVD-compromises. For each i, $2 \leq i \leq$ LNO, we show that R_{i-1} satisfies the join dependency $\bowtie [S]$ when the while loop terminates. Given R_{i-1}, $2 \leq i \leq$ LNO, after the termination of the while loop, let $P_j = \pi_{S_j}(R_{i-1}), 1 \leq j \leq m$ and $JP = P_1 \bowtie P_2 \bowtie \cdots \bowtie P_m$. Let $t \in R_{i-1}$. For each j, $1 \leq j \leq m$, $t[S_j]$ is in P_j. By the definition of the join operation, t is in JP, i.e., $R_{i-1} \subseteq JP$. Let $s \in JP$. s must be in r because $\bowtie [S]$ is the equivalent join dependency of M. Furthermore, s must be in R_{i-1}, since otherwise, we can still generate a tuple in R_i, which contradicts with the assumption that $R_i = \emptyset$, and the termination of the while loop. Thus, $JP \subseteq R_{i-1}$. From the two results shown above, we know $R_{i-1} = JP$ and there is no MVD-compromise in r. Q.E.D.

2) Time Complexity of the Level Adjustment Algorithm: Since the entire adjustment algorithm consists of three consecutive phases, to determine the time complexity, we first determine the time complexity of each phase. The phase that dominates the algorithm gives the time complexity of the entire algorithm.

From [2], we know that the time complexity of the procedure DEP is $O(|M| \times |R|^3)$, where $|M|$ is the number of MVD's in the relation R, and $|R|$ is the number of attributes in R. For phase 2, the procedure DECOM takes $O(|M|^2 \times |R|)$ [11]. We now determine the time complexity of the procedure MVD-ADJUST.

Let the number of tuples in the original relation r be n. Consider the first iteration of the first for loop (statement 1). In the first iteration, it takes $O(n)$ to form R_i and R_{i-1} (statements 2 and 3). Now consider the statements inside the while loop. Statement 5 can be implemented using multilists, i.e., each S_k has a linked list storing the values in π_{S_k} (r). Assume inserting a value v_{jk} of π_{S_k} (r) into the corresponding list takes $|\pi_{S_k}(r)|/2$ operations (comparisons) (average search time to determine whether v_{jk} is already in the list). To create all these lists for R_{i-1} in the worst case takes $p \sum_{i=1}^{m} |\pi_{S_i}(r)|/2$, where p is the number of tuples in R_{i-1}.

To remove those tuples in R_i which cannot be inferred using tuples in R_{i-1} (with a v_{jk} value which is not in $\pi_{S_k}(R_{i-1})$), we can, for each tuple in R_i, check the value of each S_k. The entire process in the worst case takes $q\sum_{i=1}^{m}|\pi_{S_i}(r)|/2$, where q is the number of tuples in R_i. Thus, to complete statement 5 (in the worst case) takes operations (comparisons) $(p+q)\sum_{i=1}^{m}|\pi_{S_i}(r)|/2$ time, which is $O(n\sum_{i=1}^{m}|\pi_{S_i}(r)|)$. The time complexity for performing statements 6 and 7 is also $O(n\sum_{i=1}^{m}|\pi_{S_i}(r)|)$. Performing statements 8 and 9 takes $O(\sum_{i=1}^{m}|\pi_{S_i}(r)|)$ time. The statements after statement 9 can be implemented in $O(n)$.

Assume the security levels are evenly distributed to tuples in R. For the first iteration of statement 1, $|R_i|$, the number of tuples in R_i, is equal to $n/$ LNO. Now consider the while loop. In the worst case, for each iteration of the while loop, we may remove only one tuple from R_i. Thus the time complexity for the first iteration of statement 1 is $(n/\mathrm{LNO})(n\sum_{i=1}^{m}|\pi_{S_i}(r)|)$. Note that as the iterations of the while loop go on, the size of R_i changes. Thus, the time complexity of statements inside the while loop will also change. Here we consider only the worst case, and assume the time complexity for each iteration of the while loop is the same which is $n\sum_{i=1}^{m}|\pi_{S_i}(r)|$. Now we add the effect of the loop in statement 1. The number of iterations of statement 1 is LNO. Thus, the time complexity of the procedure MVD-ADJUST is $O(n^2\sum_{i=1}^{m}|\pi_{S_i}(r)|)$.

Now we can determine the time complexity of the entire algorithm. Comparing the time complexity of the three procedures, we find that the procedure MVD-ADJUST dominates those of the other two. Therefore, we conclude that the time complexity of the entire algorithm is $O(n^2\sum_{i=1}^{m}|\pi_{S_i}(r)|)$.

V. Conclusions

In this paper, we first examine the inference problem due to functional dependencies under the attribute classification scheme. We decide the number of functional dependencies needed to be taken into account, and then prove that incurring minimum information loss to prevent FD-compromises is an NP-complete problem. Despite the NP-completeness result, we propose an exact algorithm to adjust attribute levels so that FD-compromises are avoided, since, presumably, the number of attributes involved in the algorithm is small. This algorithm will incur minimum information loss.

Under the record classification scheme, we also study the inference problem due to multivalued dependencies. We derive several necessary and sufficient conditions for MVD-compromises. For a set M of MVD's, we prove that it is sufficient to consider only M itself, instead of M^+, in order to prevent MVD-compromises caused by M. It is argued that in the real world, the only sets of MVD's need to be considered are conflict-free. Based on this observation, we assume that M is conflict-free, and develop an algorithm to adjust the tuple levels in a relation so that MVD-compromises are eliminated.

Although the scope of the MDB we investigate in this paper is restricted to a single relation, one may (at least theortically) extend the results to the entire MDB since the MDB can be considered as a universal relation. Also, our results consider only attribute and record classification schemes. However, we believe that they are useful in further studies for the construction of a secure multilevel relational database system.

Appendix
Proof of Theorem 4.1

Proof: We show that any MVD obtained from M using the inference rules 1–4 does not have any MVD-compromise.

[rule 1]: We prove the claim for the case that if $X \rightarrow\!\!\!\!\rightarrow Y$ then $X \rightarrow\!\!\!\!\rightarrow Z$. The other case is symmetric and is omitted. Since $X \rightarrow\!\!\!\!\rightarrow Y$ does not have any MVD-compromise, we have : for all $x \in \pi_X(r)$ and for all u, for any $t_2 \in T_{high}(X = x; u)$, either $t_2[Y] \neq t_1[Y]$ or $t_2[R-X-Y] \neq t_1[R-X-Y]$, for all $t_1 \in T_{low}(X = x; u)$. i) $t_2[R-X-Y] \neq t_1[R-X-Y] \Rightarrow t_2[(X \cup Y \cup Z)-X-Y] \neq t_1[(X \cup Y \cup Z)-X-Y] \Rightarrow t_2[Z-X-Y] \neq t_1[Z-X-Y] \Rightarrow t_2[Z] \neq t_1[Z]$. ii) $t_2[Y] \neq t_1[Y] \rightarrow t_2[Y-X-(Y \cap Z)] \neq t_1[Y-X-(Y \cap Z)] \Rightarrow t_2[R-X-Z] \neq t_1[R-X-Z]$. From the above result and Lemma 4.1, $X \rightarrow\!\!\!\!\rightarrow Z$ does not cause MVD-compromise.

[rule 2]: Using Lemma 4.1, we prove the following : for all $x \in \pi_X(r)$ and for all u, for any $t_2 \in T_{high}(X = x; u)$, $t_2[R-X-Y] \neq t_1[R-X-Y]$, for all $t_1 \in T_{low}(X = x; u)$. Since $Y \subseteq X$, $t_2[R-X-Y] = t_2[R-X]$ and $t_1[R-X-Y] = t_1[R-X]$. Now, it is clear that $t_2[R-X-Y] \neq t_1[R-X-Y]$, otherwise, $t_2 = t_1$.

[rule 3]: Using Lemma 4.1, we prove the following: Assume $X \rightarrow\!\!\!\!\rightarrow Y$ does not have any MVD-compromise. Then for all $xw \in \pi_{XW}(r)$ and for all u, for any $t_2 \in T_{high}(XW = xw; u)$ either $t_2[YZ] \neq t_1[YZ]$ or $t_2[R-XW-YZ] \neq t_1[R-XW-YZ]$, for all $t_1 \in T_{low}(XW = xw; u)$. Since $t_1[X] = x$ and $t_2[X] = x$, therefore, $t_2 \in T_{high}(X = x; u)$, for all $t_2 \in T_{high}(XW = xw; u)$ and $t_1 \in T_{low}(X = x; u)$, for all $t_1 \in T_{low}(XW = xw; u)$. Because $X \rightarrow\!\!\!\!\rightarrow Y$ does not cause MVD-compromise, we have for all $x \in \pi_X(r)$ and for all u, for any $t_2 \in T_{high}(X = x; u)$, either $t_2[Y] \neq t_1[Y]$ or $t_2[R-X-Y] \neq t_1[R-X-Y]$, for all $t_1 \in T_{low}(X = x; u)$. (Case 1): $t_2[Y] \neq t_1[Y]$. We then have $t_2[YZ] \neq t_1[YZ]$, for all $t_2 \in T_{high}(XW = xw; u)$, for all $t_1 \in T_{low}(XW = xw; u)$, and thus, the MVD $XW \rightarrow\!\!\!\!\rightarrow YZ$ does not cause any MVD-compromise for this case. (Case 2): $t_2[R-X-Y] \neq t_1[R-X-Y]$. Since $t_2[XW] = t_1[XW]$, for all $t_2 \in T_{high}(XW = xw; u)$, for all $t_1 \in T_{low}(XW = xw; u)$, we have $t_2[R-XW-Y] \neq t_1[R-XW-Y]$. Also, $Z \subseteq W$, therefore, $t_2[R-XW-YZ] \neq t_1[R-XW-YZ]$ and thus, the MVD $XW \rightarrow\!\!\!\!\rightarrow YZ$ does not cause any MVD-compromise in this case.

[rule 4]: Using Lemma 4.1, we prove the following: Assume $X \rightarrow\!\!\!\!\rightarrow Y$ and $Y \rightarrow\!\!\!\!\rightarrow Z$ do not cause any MVD-compromise. Then for all $x \in \pi_X(r)$ and for all u, for all $t_2 \in T_{high}(X = x; u)$ either $t_2[Z-Y] \neq t_1[Z-Y]$ or $t_2[R-X-(Z-Y)] \neq t_1[R-X-(Z-Y)]$, for all $t_1 \in T_{low}(X = x; u)$. Since $X \rightarrow\!\!\!\!\rightarrow Y$ does not cause any MVD-compromise, we have for all $x \in \pi_X(r)$ and for all u, for any $t_2 \in T_{high}(X = x; u)$, either $t_2[Y] \neq t_1[Y]$ or $t_2[R-X-Y] \neq t_1[R-X-Y]$, for all $t_1 \in T_{low}(X = x; u)$. (Case 1): $t_2[Y] \neq t_1[Y]$. Since $t_2[X] = t_1[X]$, we have $t_2[Y-X] \neq t_1[Y-X]$. Furthermore, we have $t_2[(Y-X)(R-X-Y-Z)] \neq t_1[(Y-X)(R-X-Y-Z)]$. That is, $t_2[R-X-(Z-Y)] \neq t_1[R-X-(Z-Y)]$. thus, the

MVD $X \rightarrow\rightarrow Z - Y$ does not cause any MVD-compromise for this case. (Case 2) : $t_2[R-X-Y] \neq t_1[R-X-Y]$. Since $t_2[R-X-Y] \neq t_1[R-X-Y]$, for all $t_1 \in T_{low}(X = x; u)$, there must be some tuples s_i in $T_{low}(X = x; u)$, $1 \leq i \leq m$, such that $t_2[Y] = s_i[Y]$ (Case 1 is not satisfied). Also, $Y \rightarrow\rightarrow Z$ does not cause MVD-compromise, thus, we can consider the two subcases below.

i) $t_2[R - Y - Z] \neq s_i[R - Y - Z]$, $1 \leq i \leq m$, $s_i \in T_{low}(X = x; u)$. Since $t_2[X] = s_i[X]$, $1 \leq i \leq m$, therefore, $t_2[R - X - Y - Z] \neq s_i[R - X - Y - Z]$, $1 \leq i \leq m$. Thus, we have $t_2[(R - X - Y - Z)(Y - X)] \neq s_i[(R - X - Y - Z)(Y - X)]$, $1 \leq i \leq m$, i.e., $t_2[R-X-(Z-Y)] \neq s_i[R-X-(Z-Y)]$, $1 \leq i \leq m$. Since $t_2[Y] \neq s_j[Y]$, $s_j \in T_{low}(X = x; u)$, $s_j \neq s_i$, $1 \leq i \leq m$, and $t_2[X] = s_j[X]$, therefore, $t_2[Y - X] \neq s_j[Y - X]$, and thus, $t_2[(Y-X)(R-X-Y-Z)] \neq s_j[(Y-X)(R-X-Y-Z)]$. That is, $t_2[R - X - (Z - Y)] \neq s_j[R - X - (Z - Y)]$, for all $s_j \in T_{low}(X = x; u)$, $s_j \neq s_i$, $1 \leq i \leq m$. Combining the two cases together, we have $t_2[R - X - (Z - Y)] \neq s[R - X - (Z - Y)]$, for all $s \in T_{low}(X = x; u)$. Thus, $X \rightarrow\rightarrow Z - Y$ does not cause MVD-compromise for this case.

ii) $t_2[Z] \neq s_i[Z]$, $1 \leq i \leq m$, $s_i \in T_{low}(X = x; u)$. We aim at proving that $t_2[Z - Y] \neq s[Z - Y]$, for all $s \in T_{low}(X = x; u)$. Assume this is not true. Since $t_2[Z] \neq s_i[Z]$, i.e., $t_2[Z-Y] \neq s_i[Z-Y]$, $1 \leq i \leq m$, we only need to check those s_j's, $s_j \in T_{low}(X = x; u)$, $s_j \neq s_i$, $1 \leq i \leq m$. Let $s_j \in T_{low}(X = x; u)$, $s_j[Z - Y] = t_2[Z - Y]$ and $s_j[Y] \neq t_2[Y]$. Since $X \rightarrow\rightarrow Y$, we should have $s'_j \in r$ such that $s'_j[R-X-Y] = s_j[R-X-Y]$ (or $s'_j[Z-Y] = s_j[Z-Y]$), and $s'_j[Y] = t_2[Y]$. Because $s_i[Y] = t_2[Y] = s'_j[Y]$, $1 \leq i \leq m$, $s_i \in T_{low}(X = x; u)$, and $s'_j[R - X - Y] = s_j[R - X - Y]$, $s_j \in T_{low}(X = x; u)$, also, $X \rightarrow\rightarrow Y$ does not cause any MVD-compromise, therefore, $s'_j \in T_{low}(X = x; u)$. Thus, s'_j is one of the s_i's, $1 \leq i \leq m$. Let this s'_j be represented by s_i. Now we have $s_i \in T_{low}(X = x; u)$, $s_i[Y] = t_2[Y]$ and $s_i[Z - Y] = s_j[Z - Y] = t_2[Z - Y]$, i.e., $s_i[Z] = t_2[Z]$. This contradicts our initial assumption that $t_2[Z] \neq s_i[Z]$. Thus, $X \rightarrow\rightarrow Z - Y$ does not cause any MVD-compromise in this case. Q.E.D.

ACKNOWLEDGMENT

We are grateful to the National Science Fundation for the support of this research. We would also like to thank the referees for their suggestions in improving the paper.

REFERENCES

[1] W. Armstrong, "Dependency structure of database relationship," *Proc. IFIP*, pp. 580–583, 1974.

[2] C. Beeri, "On the membership problem for functional and multivalued dependencies in relational databases," *ACM Trans. Database Syst.*, vol. 5, no. 3, pp. 241–259, Sept. 1980.

[3] C. Beeri, R. Fagin, and J. H. Howard, "A complete axiomatization for functional and multivalued dependencies," in *Proc. ACM SIGMOD Conf.*, 1977, pp. 47–61.

[4] D. E. Denning, "Cryptographic checksums for multilevel data security," in *Proc. 1984 Symp. Security and Privacy*, IEEE Computer Society, 1984, pp. 52–61.

[5] ——, "Commutative filters for reducing inference threats in multilevel database systems," in *Proc. 1985 Symp. Security and Privacy*, IEEE Computer Society, 1985, pp. 134–146.

[6] D. E. Denning *et al.*, "A multilevel relational data model," in *Proc. 1987 Symp. Security and Privacy*, IEEE Computer Society, 1987, pp. 220–234.

[7] D. E. Denning *et al.*, "The SeaView security model," in *Proc. 1988 Symp. Security and Privacy*, IEEE Computer Society, 1988, pp. 218–233.

[8] M. R. Garey, and D. S. Johnson, *Computers and Intractability, A Guide to The Theory of NP-Completeness*. San Francisco, CA: Freeman, 1979.

[9] R. D. Graubart, "The integrity-lock approach to secure database management," in *Proc. 1984 Symp. Security and Privacy*, IEEE Computer Society, 1984, pp. 62–74.

[10] R. D. Graubart and K. J. Duffy, "Design overview for retrofitting integrity-lock architecture onto a commercial DBMS," in *Proc. 1985 Symp. Security and Privacy*, IEEE Computer Society, 1985, pp. 147–159.

[11] Y. E. Lien, "On the equivalence of database models," *J. ACM*, vol. 29, no. 2, pp. 333–362, 1982.

[12] T. F. Lunt *et al.*, "A near-term design for the SeaView multilevel database system," in *Proc. 1988 Symp. Security and Privacy*, IEEE Computer Society, 1988, pp. 234–244.

[13] D. Maier, *The Theory of Relational Databases*. Rockville, MD: Computer Science Press, 1983.

[14] M. Morgenstern, "Security and inference in multilevel database and knowledge-base systems," in *Proc. 1987 SIGMOD Conf.*, ACM, 1987.

[15] ——, "Controlling logical inference in multilevel database systems," in *Proc. 1988 Symp. Security and Privacy*, IEEE Computer Society, 1988, pp. 245–255.

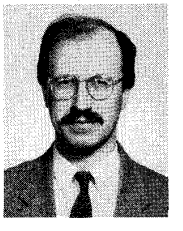
[16] E. Scior, "Real-world MVD's," in *Proc. ACM SIGMOD Conf.*, 1981, pp. 121–132.

[17] J. D. Ullman, *Principles of Database Systems*. Rockville, MD: Computer Science Press, 1988.

Tzong-An Su (S'85–M'86) received the B.S. and M.S. degrees in computer science from National Chiao-Tung University, Hsinchu, Taiwan, in 1977 and 1979, respectively, and the Ph.D. degree in computer engineering and science from Case Western Reserve University, Cleveland, OH, in 1986.

He worked from 1986 to 1989 as an Assistant Professor in the Department of Computer Science and Engineering, The University of Toledo, Toledo, OH. He is currently a member of technical staff in AT&T Bell Laboratories, Naperville, IL, where he is involved in the design of a real-time distributed DBMS. His research interests are in the areas of database security, inference control, secure data model, and secure database design.

Dr. Su is a member of IEEE Computer Society and the Association of Computing Machinery.



Gultekin Ozsoyoglu (S'79–M'80) received the B.S. degree in electrical engineering and the M.S. degree in computer science from the Middle East Technical University, Ankara, Turkey, in 1972 and 1974, respectively, and the Ph.D. degree in computing science from the University of Alberta, Edmonton, Alta., Canada, in 1980.

He is currently an Associate Professor of Computer Engineering and Science, Case Western Reserve University, Cleveland, OH. His research interests include databases and security.