AN ANALYSIS OF TWO MECHANISMS FOR PROTECTING THE SECURITY OF DATABASES*

by

Ernst Leiss

Research Report CS-77-33

Department of Computer Science

University of Waterloo
Waterloo, Ontario, Canada

October 1977

* This research was supported by the National Research Council of Canada under Grant No. A-1617.

An Analysis of Two Mechanisms for Protecting the Security of Databases

by

Ernst Leiss

Abstract

Recently, problems of security in databases have been studied where queries involved medians. The results imply that it is almost impossible to provide security of data within such a framework. In this note we extend this conclusion to queries involving averages, using entirely different methods,namely linear algebra. In particular, in the case of averages there does not exist a "president-janitor" paradox. We then analyze the method of restricted overlap of queries, and determine bounds which imply that is is not practicable. We conclude by giving a method using random access which guarantees the security of the database at virtually no additional cost while providing the required information.

Introduction and Notation

Recently, several papers have dealt with the database security problem from the viewpoint of computational complexity ([1]; for further references see there). For our purposes this problem can be formulated as follows. A database is a set of N elements $x_1, x_2, \ldots, x_N$, where access is by specifying an appropriate index. For example, x_i might be the salary of person i. A query is a function of zero or more arguments i.e. indices in the range from 1 to N. In the following discussion we will always assume that the x_i are (real) numbers. Typical examples in this case are $M_k(i_1, \ldots, i_k)$ which returns the median of the k elements $x_{i_1}, \ldots, x_{i_k}$, $A_k(i_1, \ldots, i_k)$ which returns the average of the k elements $x_{i_1}, \ldots, x_{i_k}$, and the random selector function S. We will also consider $G_k(i_1, \ldots, i_k)$, the geometric mean defined by $\left(\prod_{j=1}^{k} x_{i_j}\right)^{1/k}$. In many databases some information is considered to be "classified", i.e. is not accessible to the user. However, it is not uncommon that the user is allowed to apply M_k, A_k, G_k, or S to such classified information since the feeling seems to be prevalent that this does not violate the security. For example, while individual salaries might be classified, medians or averages of k ($k \geq 2$) can be computed. - Clearly, it must be assumed that the arguments to M_k, A_k, and G_k are pairwise different.

In [1] the case was studied where the query function is M_k. It was shown that $O(k^{1/2})$ queries are sufficient to compromise the database i.e. to provide the - unauthorized - user with access to some classified information (although not necessarily that of the user's choice).

Furthermore it was shown that not all classified information can be accessed in this way (the "president-janitor" problem).

In the present note we will assume that the query function is A_k. In this case we show that $k+1$ queries are sufficient to compromise the database. This is optimal, i.e. any set of k or fewer queries does not compromise the database. The given method to determine x_j for given j requires $O(k)$ arithmetical operations. It immediately gives rise to an optimal method to determine m elements which requires $\max(m, k+1)$ queries and $O(\max(m, k))$ arithmetical operations. In particular $k+1$ queries are sufficient to determine $k+1$ elements! Similar results hold for queries based on G_k under the assumption that all x_i are positive. We then analyze the method of restricted overlap of queries, suggested in [2]. We derive tight bounds for this method by reducing the problem to a question of symmetric block designs. These bounds imply that this method should be avoided for practical purposes. We conclude the paper by giving a method involving random access which guarantees the confidentiality of the information stored in the database at the cost of two random accesses for each query.

Queries Involving Averages: The Optimal Case

We assume that all queries are of the form

$$q \leftarrow A_k(i_1, \ldots, i_k)$$

where $A_k(i_1, \ldots, i_k) = \frac{1}{k} \sum_{j=1}^{k} x_{i_j}$, $k \geq 2$ is arbitrary and fixed throughout, and q is the result of the query. For brevity we denote a query by $(i_1, \ldots, i_k)$. We show that $k+1$ queries are necessary and sufficient to compromise the database. The proof is then extended to give a fast method to determine the values $x_{s_1}, \ldots, x_{s_m}$ for given $s_1, \ldots, s_m$.

Consider the following $k+1$ queries:

$$\begin{aligned}
q_1 &\leftarrow (i_1, i_2, \ldots, i_{k-1}, i_k) \\
q_2 &\leftarrow (i_2, i_3, \ldots, i_k, i_{k+1}) \\
q_3 &\leftarrow (i_3, i_4, \ldots, i_{k+1}, i_1) \\
&\vdots \\
q_k &\leftarrow (i_k, i_{k+1}, \ldots, i_{k-3}, i_{k-2}) \\
q_{k+1} &\leftarrow (i_{k+1}, i_1, \ldots, i_{k-2}, i_{k-1}) \quad .
\end{aligned}$$

In other words we cyclicly permute $(i_1, \ldots, i_{k+1})$ and then take the k-tuple consisting of the first k elements. This can be rewritten as a system of $k+1$ linear equations

$$B_k \cdot x = k \cdot q$$

where $x = \begin{pmatrix} x_1 \\ \vdots \\ x_{k+1} \end{pmatrix}$ $(x_j = x_{i_j})$, $q = \begin{pmatrix} q_1 \\ \vdots \\ q_{k+1} \end{pmatrix}$, and $B_k = (b_{\ell j})$ is a

$(k+1\ ,\ k+1)$-matrix such that $b_{\ell j} = 1$ if i_j is involved in the ℓ-th query, otherwise $b_{\ell j} = 0$. For instance if $k = 3$ we have the following system

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = 3 \cdot \begin{pmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{pmatrix} .$$

In order to show that we can uniquely determine any x_ℓ it is necessary and sufficient that B_k be nonsingular, i.e. that the determinant $\det B_k$ be different from zero.

Let $F_k = (f_{ij})$ be a $(k+1, k+1)$-matrix, $f_{ij} = \begin{cases} 0 & i = j+1 \\ 1 & \text{otherwise.} \end{cases}$

We show by induction on k that $\det F_k = 1$ for $k \geq 1$. Clearly $\det F_1 = \det F_2 = 1$. Assume $\det F_{k-1} = 1$; we have to show $\det F_k = 1$:

$$\det \begin{pmatrix} 1 & 1 & 1 & \dots & 1 & 1 \\ 0 & 1 & 1 & \dots & 1 & 1 \\ 1 & 0 & 1 & \dots & 1 & 1 \\ \vdots & & & & & \\ 1 & 1 & 1 & \dots & 0 & 1 \end{pmatrix} = \det \left(\begin{array}{c|ccccc} 1 & 0 & 0 & \dots & 0 & 0 \\ \hline 0 & 1 & 1 & \dots & 1 & 1 \\ 1 & 0 & 1 & \dots & 1 & 1 \\ \vdots & & & & & \\ 1 & 1 & 1 & \dots & 0 & 1 \end{array}\right)$$

$$= (-1)^{1+1} \cdot \det F_{k-1} = 1 .$$

Similarly we show that $\det B_k = k$ for $k \geq 1$. Clearly $\det B_1 = 1$, $\det B_2 = 2$. Assume $\det B_{k-1} = k-1$; we have to show $\det B_k = k$:

$$\det\begin{pmatrix} 1 & 1 & 1 & \dots & 1 & 0 \\ 0 & 1 & 1 & \dots & 1 & 1 \\ 1 & 0 & 1 & \dots & 1 & 1 \\ \vdots & & & & & \\ 1 & 1 & 1 & \dots & 0 & 1 \end{pmatrix} = \det\begin{pmatrix} 1 & 1 & 1 & \dots & 1 & 0 \\ -1 & 1 & 1 & \dots & 1 & 1 \\ 0 & 0 & 1 & \dots & 1 & 1 \\ \vdots & & & & & \\ 0 & 1 & 1 & \dots & 0 & 1 \end{pmatrix}$$

$$= (-1)^{1+1}\cdot\det F_{k-1} + (-1)\cdot(-1)^{2+1}\cdot\det B_{k-1} = 1 + k - 1 = k .$$

This shows that the information obtained in $k+1$ queries is sufficient to compromise the database. In fact, it shows considerably more, for we can determine <u>all</u> the x_j's using this information since the latter problem is equivalent to solving the system of linear equations $B_k\cdot x = k\cdot q$ in x. Furthermore, we now can easily determine further elements $x_{i_{k+2}}, \dots, x_{i_m}$. This can be achieved by $m - k - 1$ further queries:

$q_s \leftarrow (i_{s-k+1}, i_{s-k+2}, \dots, i_s)$ for $s = k+2, k+3, \dots, m$ (in this order!).

Since all but the last element (x_{i_s}) in these queries are already known we obtain the formula

$$x_{i_s} = k\cdot q_s - \sum_{j=s-k+1}^{s-1} x_{i_j} , \quad s = k+2, \dots, m .$$

We summarize: Let $i_1, \dots, i_m$ indices be given. To determine the values $x_{i_1}, \dots, x_{i_m}$ $\max(k+1, m)$ queries are sufficient.

Clearly solving a system of $k+1$ linear equations in $k+1$ unknowns is not a very attractive method. However, due to the special form of the matrix B_k we can give explicit formulae for the x_j in

terms of the q_ℓ . We have

$$x_i = \sum_{j=1}^{k+1} q_j - k \cdot q_{|i+1|}$$

where

$$|s| = \begin{cases} s & \text{if } s \in \{1, \ldots, k+1\} \\ k+1 & \text{if } s \equiv 0 \mod k+1 \\ s \bmod k+1 & \text{otherwise} \end{cases} .$$

For the proof we first observe that $B_k \cdot x = k \cdot q$ is equivalent to

$$k \cdot q_i = \sum_{j=1}^{k+1} x_j - x_{|i-1|} \quad \text{(by inspection)} .$$

Substitution yields

$$\sum_{s=1}^{k+1} x_s - x_{|i-1|} = \left[\sum_{s=1}^{k+1} \left(\sum_{j=1}^{k+1} q_j - k \cdot q_{|s+1|} \right)\right] - \left(\sum_{j=1}^{k+1} q_j - k \cdot q_{||i+1|-1|} \right)$$

$$= \left[(k+1) \cdot \sum_{j=1}^{k+1} q_j - k \cdot \sum_{j=1}^{k+1} q_j \right] - \left(\sum_{j=1}^{k+1} q_j - k \cdot q_{|i|} \right)$$

$$= \sum_{j=1}^{k+1} q_j - \sum_{j=1}^{k+1} q_j + k\, q_i = k q_i \quad .$$

This holds for all $i = 1, \ldots, k+1$, and shows that

$$x_i = \sum_{j=1}^{k+1} q_j - k \cdot q_{|i+1|}$$

is, in fact, a solution. Since B_k is nonsingular, it is the only one.

This proves:

(a) To determine one element, $k+1$ queries and $O(k)$ arithmetic operations suffice.

(b) To determine $m \leq k+1$ elements, $k+1$ queries and $O(k + m) = O(k)$ arithmetic operations suffice. Note that we compute $\sum_{j=1}^{k+1} q_j$ only once.

(c) To determine $m' > k+1$ elements, m' queries suffice. As for the number of arithmetic operations we first apply (b) $\lfloor \frac{m'}{k+1} \rfloor$ times with $m = k+1$ to pairwise disjoint sets, and for the remaining elements we apply (b) with $m = m' - (k + 1) \cdot \lfloor \frac{m'}{k+1} \rfloor$. Altogether this requires $O(m')$ arithmetic operations.

It is easy to see that this method is optimal in the number of queries. Any set of queries is equivalent to a system of linear equations. It follows that $k+1$ queries are necessary to determine at most $k+1$ elements. This is obvious if one bears in mind the following facts: Any query involves k different elements; two queries are identical iff they operate on the same set of elements. Thus any reasonable set of queries must involve at least $k+1$ elements. If we have only k' ($< k+1$) equations in at least $k+1$ unknowns this leads to an underdetermined system of linear equations, which has either no solution or infinitely many solutions. It is easily verified that every x_i depends on all $k+1$ queries (see for example the proof that B_k is nonsingular). Hence none of the x_i's can be uniquely determined because there does exist a solution by assumption. Since all the x_i depend on $k+1$ queries $O(k)$ arithmetic operations are necessary to calculate one x_i. This consequently implies that the above method is optimal with respect to the number of arithmetic operations, as well.

The reader should note that any error in a response q will affect only $k+1$ elements x_{i_j}, i.e. there is no indefinite propagation of errors. Also, round-off errors in the computations should not present any problems.

From the viewpoint of data security the most alarming aspect is the following: If a malicious user wants to access all the classified information it does not make any difference whatsoever how large k is $(k \geq 3)$.

We conclude this section with a remark on the geometric mean G_k. If we assume that all the x_i are positive the above results also hold for G_k in place of A_k, since the two models are isomorphic by virtue of the functions log and exp. If we also allow negative numbers $k+1$ queries do not compromise a database in general since we might be able to change signs. For example, if we have the following system

$$\sqrt[2]{x_1 \cdot x_2} = 1 \ , \quad \sqrt[2]{x_2 \cdot x_3} = 2 \ , \quad \sqrt[2]{x_1 \cdot x_3} = 3$$

and if we assume $x_1, x_2, x_3 > 0$ then we get the unique solution

$$x_1 = 3/2 \ , \quad x_2 = 2/3 \ , \quad x_3 = 6 \ .$$

If we drop this assumption we also have the solution

$$x_1 = -3/2 \ , \quad x_2 = -2/3 \ , \quad x_6 = -6 \ .$$

Ensuring Security of Databases: Restricted Overlap

In the preceding section we saw that $k + 1$ queries are necessary and sufficient to compromise a database when we allow queries of the type $A_k(i_1, \ldots, i_k)$. Similar results hold for queries of the type M_k ([1],[2]) or G_k. Thus we have to conclude that allowing queries of these types and aiming at security of databases are conflicting objectives.

In [2] restriction of the overlap of queries was suggested to ensure the security of classified information. More precisely, given a set of queries the overlap r is the maximal number of common indices in any two queries $(i_1, \ldots, i_k)$ and $(j_1, \ldots, j_k)$ in the set. Clearly, r can be restricted to $\{1, \ldots, k-1\}$, since $r = 0$ (no overlap) and $r = k$ (repetition of a query) are of no interest; it should be obvious that a set of queries of any type with overlap $r = 0$ can not compromise a database. Examples show that restricting the overlap r in conjunction with allowing only a limited number of queries does indeed guarantee privacy of data. For instance, if $k = 4$ the minimal number of queries of type A_4 to compromise the database is 7 for $r = 2$ and 13 for $r = 1$ (versus 5 for $r = 3$), for $k = 3$ and $r = 1$ this number is 7 (versus 4 for $r = 2$). However, the claim in [2] that their "theorem 1 points out that there exist mechanisms (bounding the overlap and the number of queries) that can protect a database" is somewhat misleading because the lower bound given in this theorem in [2] on the number of queries necessary to compromise a database has the maximum k.

In this section we will give an analysis of the method of restricted overlap. We will obtain tight bounds by reducing it to symmetric block designs. These bounds imply that the method is of little practical use to ensure confidentiality of data.

If we restrict our attention to queries of type A_k, one can easily verify that the problem of finding a compromising set of queries is equivalent to the following problem:

> Find an integer t, $t = t(r,k)$, and a (t,t) - matrix $D_{k,r}$, $D_{k,r} = (d_{ij})$ $1 \le i, j \le t$, such that each row $d_{i.}$ consists of k ones and $t - k$ zeros, any two rows $d_{i.}$ and $d_{\ell.}$ have at most r ones in the same position, and $D_{k,r}$ is nonsingular.

Let us analyze this method in order to assess its value for practical applications. Most of this analysis will be done by appealing to some theorems in combinatorial theory. In particular, all the theorems cited in this paragraph are from [3]. Since we can formulate our results without making reference to block designs, we will not give their definition. Note, however, that the theorems from [3] all belong to the theory of block designs. In the following assume $k \ge 3$.

First we observe that we are only interested in the case where t is minimal, i.e. if we have matrices $D_{k,r}$, $D'_{k,r}$ of dimension t, t', respectively, we need consider only the smaller one. Furthermore, the following should be obvious: Let $D_{k,r}$ be a matrix such that any two rows have precisely overlap r, and suppose that $D_{k,r}$ is of minimal dimension t with respect to this property. Also, let $D'_{k,r}$ be a matrix such that not any two rows have overlap r, and let t' be its dimension. Then $t' > t$. More general, if $D_{k,r}$ has more rows than $D'_{k,r}$ with overlap precisely r, and if $D_{k,r}$ is of minimal dimension, the dimension of $D_{k,r}$ is smaller than that of $D'_{k,r}$.

Therefore, we restrict our attention to matrices $E_{k,r}$ of dimension t in which any two rows have exactly overlap r. This implies:

1. $E_{k,r} \cdot E_{k,r}^T = (k-r)\cdot I + r\cdot J$, where I is the identity matrix of dimension t, and J is the matrix consisting of ones only. This is easily verified by considering the inner product of a row of $E_{k,r}$ with itself.
2. $\det (E_{k,r} \cdot E_{k,r}^T) = (k-r)^{t-1} \cdot (tr-r+k)$ ([3], p.103). Therefore $\det (E_{k,r}) = \sqrt{(k-r)^{t-1}\cdot(tr-r+k)}$, and $t > k > r$ implies $E_{k,r}$ is nonsingular.
3. $E_{k,r} \cdot J = k \cdot J$, where J is as in 1.. Again, this follows by considering inner products.

We now use a theorem by Ryser ([3], p.104, theorem 10.2.3) which states that 1., 2., and 3. imply $t = \frac{k^2-k}{r} + 1$. We therefore conclude that matrices $E_{k,r}$ exist only if r is a divisor of k^2-k. We summarize these results as follows:

Let r and k be given, $1 \le r < k$, and assume there exists a matrix $E_{k,r}$. Then any matrix $D_{k,r}$ satisfying the requirements set out above has at least dimension

$$\frac{k^2-k}{r} + 1.$$

Before we continue we remark that the Bruck-Ryser-Chowla theorem in our case provides a necessory and sufficient condition for the existence of matrices $E_{k,r}$ for given k and r. In fact, letting $t = \frac{k^2-k}{r} + 1$ such a matrix exists iff

t is even and $k-r$ is a square

or

t is odd and $z^2 = (k-r)x^2 + (-1)^{(t-1)/2} r\cdot y^2$ has a solution in integers x, y, z different from $(0, 0, 0)$.

For a proof we refer to [3],theorem 16.4.1 (p.282) and sections 10.3 and 10.4.

The above lower bound on the dimension of $D_{k,r}$ implies that any number of queries less than $\frac{k^2-k}{r} + 1$ will not compromise the database. In particular, choosing the smallest possible value for r, namely 1, the best this method can give is an upper bound of $k^2 - k + 1$ queries. Since it can be shown (see [3], p.111) that for any k and $r = 1$ there exists a matrix $E_{k,1}$, this is an absolute bound; clearly, even if $r > 1$ we can assume $r = 1$ and use $E_{k,1}$ which is of dimension $k^2 - k + 1$. Therefore under no circumstances is it possible to allow more than $k^2 - k$ queries with common overlap if the database is not to be compromised. This alone would suggest that the method of restricted overlap is not feasible in practice. Moreover, in order to be able to test whether the overlap condition is satisfied, all queries of a user must be stored (and there can be arbitrarily many since disjoint queries do not affect anything) for as long as the information in the database remains unchanged. In the case of salaries this may well be a month, in the case of financial contributions to political parties even longer. Yet, even this would not guarantee that the database cannot be compromised because two malicious users each one having issued less than the maximal number of queries with common overlap can combine their results and can thereby compromise the database. In summary, the method of restricted overlap is totally infeasible.

Ensuring Security of Databases: Random Selector Functions

The results of the last section suggest that restrictions imposed on the "form" of the queries are not very feasible. On the other hand, the first section amply demonstrates that there is a need for some mechanism to protect a database. In order to achieve this we propose the following method. Rather than using queries of the types A_k, G_k, or M_k we suggest the following types of queries, where $0 \leq v \leq k-1$ (again $k \geq 3$):

$$A_k^v\ (i_1, \ldots, i_v, S, \underbrace{\ldots}_{k-v}, S),$$

$$G_k^v\ (i_1, \ldots, i_v, S, \underbrace{\ldots}_{k-v}, S), \quad \text{and}$$

$$M_k^v\ (i_1, \ldots, i_v, S, \underbrace{\ldots}_{k-v}, S).$$

This is to be understood as follows: The user provides v indices $i_1, \ldots, i_v$, and the remaining $k-v$ values are determined by the random selector function S. For instance

$$q \leftarrow A_k^v\ (i_1, \ldots, i_v, S, \ldots, S)$$

means

$$q = \frac{1}{k} \cdot \sum_{j=1}^{v} x_{i_j} + \frac{1}{k} \cdot \sum_{j=1}^{k-v} s_j$$

where s_j is a result of S. In the following discussion we will concentrate on queries of the type A_k^{k-1}. However, all the results can be applied to A_k^v for arbitrary $v < k$, and some of them also to G_k^v and M_k^v.

We claim that it is impossible to compromise a database with a set of queries of type A_k^{k-1}, even for very large k. Clearly, if k is not too small, the results of a query $(i_1, \ldots, i_{k-1},.)$ will be a good approximation to the desired (exact) value

$$\frac{1}{k-1} \cdot \sum_{j=1}^{k-1} x_{i_j}.$$

Thus, as far as statistical data is concerned this modification does not make much of a difference.

Let us first assume, that the results of the queries form the right-hand side of a system of linear equations. Since so far we were concerned with lower bounds only, this assumption was not necessary. However, now we allow arbitrarily many queries and have to be more careful.

To substantiate our claim under this assumption we first observe that our best "guess" for the values s_j returned by S is the average

$$s^* = \frac{1}{N} \cdot \sum_{i=1}^{N} x_i$$

taken over the whole database, or - if we don't know s^* - the average over all responses to the queries issued. Therefore, rather than dealing with a system of equations

$$D_k \cdot x = k \cdot q$$

where $D_k = D_{k,k-1}$ (i.e. any overlap is permitted) we must solve the modified system

$$D_k \cdot x = \hat{q}$$

where $\hat{q} = ((k+1)q_1 - s_1, \ldots, (k+1) \cdot q_k - s_k)^T$.

However, all what we can solve is the system

$$D_k \cdot x = \hat{\hat{q}}$$

where $\hat{\hat{q}} = ((k+1)\cdot q_1 - s^*, \ldots, (k+1)\cdot q_k - s^*)^T$. Another way to look at this problem is to assume that $\hat{q}$ contains errors, and is given by $\hat{\hat{q}}$. The sensibility of a system of linear equations $M\cdot x = c$ to errors in c is usually measured by the condition number

$$\text{cond}(M) = ||M|| \cdot ||M^{-1}||$$

where $||\ ||$ is some matrix norm, and M^{-1} denotes the inverse matrix to M. We pick the following norm $||\ ||$:

$$||M|| = \left(\sum_{i,j=1}^{n} |m_{ij}|^2 \right)^{\frac{1}{2}}, \quad M = (m_{ij})_{1\le i,\, j\le n}.$$

Let us determine a lower bound on $\text{cond}(D_k)$. Clearly,

$$||D_k|| = \sqrt[2]{k\cdot t}$$

t being the dimension of D_k. We claim that $||D_k^{-1}|| \ge \sqrt[2]{t/k}$. Let $D_k = (d_{ij})$, $D_k^{-1} = (d_{ij}')$. Since $D_k \cdot D_k^{-1} = I$ we have

$$\sum_{\ell=1}^{t} d_{i\ell}\, d'_{\ell i} = 1.$$

We want to make $\left(\sum_{\ell=1}^{t} |d'_i|^2 \right)^{\frac{1}{2}}$ as small as possible, for all $i = 1, \ldots, t$. So, let $d'_{\ell i} = 0$ whenever $d_{i\ell} = 0$. Therefore, at most k of the $d'_{\ell i}$ are nonzero. Furthermore, these must add up to 1; note that $d_{i\ell} \in \{0,1\}$. Now, consider

$$\left(\sum_{j=1}^{k} |z_j|^2 \right)^{\frac{1}{2}}$$

as a function in the k variables $z_1, \ldots, z_k$, and determine a minimum subject to the constraint $\sum_{j=1}^{k} z_j = 1$. This minimum is attained for

$$z_1 = \ldots = z_k = 1/k,$$

and is $k^{-\frac{1}{2}}$. This holds for all $i = 1, \ldots, t$, hence we get

$$||D_k^{-1}|| \geq (\sum_{i=1}^{t} \sum_{j=1}^{k} \frac{1}{k^2})^{\frac{1}{2}} = \sqrt{t/k} .$$

Therefore we obtain

$$\text{cond } D_k \geq t.$$

This implies that all matrices D_k are quite sensitive to errors; note that $t > k$. Furthermore, the errors will be considerable since on the average $s_j - s^*$ will be of the same order of magnitude as s^*. Therefore, it is not possible to obtain useful results when solving this system of equations, the database cannot be compromised.

D.J. Taylor pointed out that there exists a way to compromise a database if we do not assume that the results of the queries form the right-hand side of a system of linear equations, provided the random selector function S satisfies a certain property. This method can be described as follows: Let $\{q_j(i_1^{(j)},\ldots,i_k^{(j)}, S) \mid j = 1, \ldots, t\}$ be a set of queries of type A_{k+1}^k corresponding to a matrix $D_{k,k-1}$ of dimension t such that the following holds: If q_j' is q_j as query of type A_k (i.e. no S appears) then by solving $D_{k,k-1} \cdot x = q'$ we can determine all $x_{i_1}, \ldots, x_{i_k}$. Now define $q_{j,m}$ to be the mth repetition of query

q_j; note that, in general, $q_{j,m} \neq q_{j,m'}$ since the result s_m of the random selector function S in the mth repetition can be different from $s_{m'}$. However,

$$\frac{1}{N^*} \sum_{m=1}^{N^*} q_{j,m}$$

might converge to a certain value with increasing N^*, say q_j^*, i.e.

$$q_j^* = \sum_{\ell=1}^{t} d_{j,\ell} x_{i_\ell} + s_j^*$$

where $D_{k,k-1} = (d_{j,\ell})$. Assume this holds for all $j = 1, \ldots, t$. Furthermore assume that

$$s_1^* = \ldots = s_t^* = s^*$$

i.e. the value to which S converges does not depend upon the particular query it is used in. We will assume that s^* is known; usually it can be obtained by averaging. Under these circumstances it is possible to compromise the database since this simply amounts to solving the following system of linear equations

$$D_{k,k-1} \cdot x = \begin{pmatrix} q_1^* - s^* \\ \vdots \\ q_t^* - s^* \end{pmatrix} .$$

This suggests several methods to prevent a breach of security. One is to do away with the assumption that s^* is known; one might for instance change it from time to time. Another one is to make sure that N^*, the number for which $\frac{1}{N^*} \cdot \sum_{m=1}^{N^*} s_m$ is a reasonable approximation of s^*,

is very large, preferably $N^* = O(N)$. A third possibility is to drop the assumption $s_1^* = \ldots = s_t^*$. The first possibility is rather simple-minded; to change s^* from time to time will introduce a time dependency which might be undesirable for reasons outside of the scope of the questions we deal with here. We also discard the second strategy because in order to implement it queries must be counted. This would lead to problems similar to those in connection with restricted overlap, such as the problem of storing information about queries for a long period of time, or the problem of users combining their results. We therefore have a look at the third proposal, dropping the assumption

$$s_1^* = \ldots = s_t^* .$$

By applying the above arguments from linear algebra it is not difficult to see that the database cannot be compromised if there is a possibility that $s_j^* \neq s_{j'}^*$ for $j \neq j'$. Of course, we assume that no s_j^* is known; note that this information cannot be retrieved from the queries if one uses the following scheme. Let T be a random selector function; we do not assume anything else about T. Whenever a query $(i_1, \ldots, i_{k-1}, S)$ is to be executed two calls to T are made, yielding t_1 and t_2. If $x_{i_1} < x_{i_{k-1}}$ then the maximum of t_1 and t_2 is the value of S returns, otherwise S returns the minimum. Note, that in this way no information about the range of the x_i's is required, no time dependency is introduced, and no previous values of T must be stored. Furthermore it can be implemented at virtually no cost. We do realize that a slight bias in the statistical information might be introduced by this method but we feel this is negligible.

In conclusion, queries of the types A_k^{k-1}, G_k^{k-1}, or M_k^{k-1}, for not too small k and a suitable choice of the random selector function S, provide all the statistical information the user is supposed to obtain while on the other hand they protect the security of databases at practically no additional cost i.e. they do not require extensive storing of previous queries and costly checking for conditions (such as overlap).

Acknowledgement

I am indebted to L.J. Cummings and D.J. Taylor with whom I discussed some aspects of the problem. In particular, Larry Cummings drew my attention to the reduction to symmetric block designs and to the book by M. Hall, while David Taylor pointed out a tacit assumption I made in an earlier version of the last section.

[1] Dobkin, D. and Lipton, R. and Reiss, S.P. Aspects of the Database Security Problem, Proceedings of a Conference on Theoretical Computer Science, University of Waterloo, August 15-17, 1977, Waterloo, Ontario, Canada.

[2] Dobkin, D. and Jones, A.K. and Lipton, R. Secure Data Bases: Protection Against User Inference, Yale Computer Science Research Report #65, April 1976.

[3] Hall, M. Jr. Combinatorial Theory, Blaisdell Publishing Company, Waltham, Mass., 1967.