A Model Formulation of User Interface for Statistical Databases with Perturbed Responses

Csaba Egyhazy

TR 86-33

A Model Formulation of User Inference for Statistical Databases with Perturbed Responses

1. INTRODUCTION

A statistical database is defined as a collection of N records used to produce summary information only. Each record contains confidential category and data fields of the same fixed length. Category fields are used to identify and select records, while data fields hold other information, mostly numerical.

A statistical database is said to be secure if enough protective mechanisms exist to guard against the disclosure of confidential information about a single record. Although statistical databases are vulnerable to loss of confidentiality (i.e., they are compromisable) by all the same means as other databases, they can be compromised even when protective measures have been taken against those general threats. Forbidding access to confidential information about a single record is a difficult proposition indeed, since the security problem is not just to protect against returning information about a single record but answering a sequence of queries from which individual confidential information can be inferred.

All statistical databases must in some way protect against the accurate disclosure of confidential attribute values for individuals. Denning [5] distinguishes between category and data fiels. For the purposes of this paper, category fields are those fields (attributes) one is allowed to use in formulating queries. The selection formula for a query is called a characteristic formula. A query set consists of those records that match a given characteristic formula. Data fields are those whose values are to be protected against user inference; Denning [5] is less restrictive in its use of the term. Data fields contain numeric values, while category fields need not. Unique identifier, or key, fields are a third kind of field. These three field types are not necessarily disjoint. A key may consist of four fields, two of which may be category fields. Similarly, category fields may overlap data fields, i.e., some category fields may contain protected information. In some of the literature, key fields and category fields are considered synonomous in that queries may be directed at individual records.

The literature is rich in papers dealing with the security of statistical databases. Many have demonstrated how vulnerable statistical databases can be to various threats. A smaller number of papers have attempted to develop mechanisms to counter such threats. Some classes of protection schemes, as given in [10], are: 1) place cardinality restrictions on the query sets; 2) partition the database, so that query sets are required to contain one or more blocks ("kernels") of the partition; and 3) perturb responses to the queries. Each protection scheme attempts to prevent accurate inferential disclosure of key values, i.e., values corresponding to an individual. Some additionally define compromise as the ability to infer values corresponding to any "protected" query set.

The "tracker" literature (e.g. [2], [4], [5], [6], [7]) considers threats to protection schemes of the first class. The general conclusion of these papers is that the more diverse the database in categorizing individuals, the more likely the database is to have a general tracker, a characteristic formula that can be applied for every person an interrogator desires to investigate. Attempts to restrict the query set size tend to limit the usefulness of the database without stopping the threat.

The method of partitioning the database into kernels will not be discussed at depth in this paper, since Ozsoyoglu [10] contains a good discussion of such techniques. How restrictive this method is depends on how the kernels are defined. If one restricts the kernels to correspond to truth sets of certain kinds of predicates involving only category, fields, subject to a minimal cardinality requirement, an argument could be made that all "legitimate" responses are still possible. This seems an inferior method to the "random sample query" technique. Both methods are most appropriate for large data bases where legitimate query sets are never very small.

Several algorithms for perturbing responses are discussed by Beck [1]. The paper advocates for an unbiased perturbation of responses, with the perturbation dependent on the amount by which the true response for the query set differs from the mean response across all individuals in the data base. One of the objectives of this paper is to put Beck's work in the context of basic questions that are addressed throughout the literature: what kinds of queries are allowed, what it means to compromise the database, and the implications of the first two responses on the accuracy and reliability of answers. In addition, this paper proposes a generic definition of compromisability and a new model of user inference for databases with perturbed responses.

THE TRACKER LITERATURE

The first model of a statistical database appeared in the work of Kam [8]. It is comprised of records with a key of k bits which specifies its attributes. Every query has a specification of values for $s \leq k$ of these bits. The model handels sums over records whose keys match with its specified bits and averages, only when the database happens to consist of exactly one record for each key. This last limitation is circumvented in [2] by means of a model which takes the possibility of nonexisting records into account, and gives information on sums and averages of a subset of records. Chin [2] illustrates the threat to the security of the statistical database of queries involving fewer than two records. Under those conditions he shows how easily we can deduce the existence of all records if we confirm the existance of any particular record in the database. Similarly, we can also deduce the information about the existing records if we have the knowledge about the information of any particular record. The above can be stated in the form of theorems, as done in [2], and proved by defining the Hamming distance between a known existing key and any other key and using that distance to show by induction that the existance of any other record can be deduced. The paper by Chin [2] concludes by establishing a set of preliminary guidelines about the maximum amount of information which the system can provide without threatening the security of the statistical database.

The tracker literature has demonstrated that suppressing queries based on cardinality is ineffective. Moreover Chin and Ozsoyoglu [3] offer valid criticisms of the kind of databases discussed in the tracker literature. Allowable query set construction is often unrealistically generous and devoid of semantic meaning ([7], [2]). Limiting execessive overlap between queries has the added drawback of requiring log trails and introduces the possibility that a valid request might be shut out because of the sequence of queries taken to achieve the response.

In general, it is assumed throughout the tracker literature that query responses are accurate. That is, the database returns the stored value that has not been perturbed beyond normal roundoff. Category fields may include a candidate key. If so the user is capable of specifying an individual record. The mechanism for controlling queries is thus not primarily the query syntax, but the cardinality restrictions on query sets. The user may ask questions that are syntactically valid, but the system may refuse to answer when the query set is determined. The system may refuse because the cardinality of a query set is too small or large [5] or because there is excessive overlap between query sets [7]. The latter not only assumes a restricted overlap cardinality but also requires all query sets to be of the same cardinality.

Since an identical response must be given each time a query is resubmitted, responses are completely reliable. Compromise is thus defined as knowledge of a sample statistic (e.g. count, mean,) of a forbidden query set, where "knowledge" is taken to mean "exact value."

Furthermore, the tracker literature has demonstrated that the number of queries required to compromise a database can be quite small if the set of queries is formulated in a special way. The main notion introduced by the tracker literature has been that of characteristic formulas.

A characteristic formula C is an arbitrary logical formula using category values as terms connected by Boolean operators. Each query for a database uses a characteristic formula. The query set X_C is the set of records whose category fields match C. The size of the query set is denoted by COUNT (C). The query that returns the sum of the values in the j^th data field for records in X_C is given by SUM (C;j). While the query that returns the mean of the values in the j^th data field for records in X_C is given by MEAN (C;j). Furthermore, assume that the number of records is N and the size of the smallest allowable query set is k. Then, using # to denote unanswerable query, the family of queries computing raw statistics are of the form:

$$q\ (C) = \begin{cases} \sum\limits_{i \in x_C} v_{ij}^m & k \leq \text{COUNT}(C) \leq N-k \\ \# & \text{otherwise} \end{cases}$$

where v_{ij} is the value in the date field j of record i, and m is an integer.

For m = 0 the query returns COUNT (C), when m = 1 the query returns SUM (C;j) and for m = 2 we get MEAN (C;j), which translates into:

$$\text{COUNT}(C) = \begin{cases} |X_C| & \text{if } k \leq |X_C| \leq N-k \\ \# & \text{otherwise} \end{cases}$$

$$\text{SUM}\ (c_{ij}) = \begin{cases} \sum\limits_{i \in X_C} V_{ij} & \text{if } k \leq |X_C| \leq N-k \\ \# & \text{otherwise} \end{cases}$$

$$\text{MEAN}\ (C) = \begin{cases} \dfrac{\sum\limits_{i \in X_C} V_{ij}}{X_C} & \text{if } k \leq |X_C| \leq N-k \\ \# & \text{otherwise} \end{cases}$$

Trackers are special characteristic formulas which can be used to calculate indirectly the values of unanswerable queries.

The individual tracker compromise requires that a new individual tracker be found for each person because it is based on something that is known. The general tracker is any characteristic formula T whose query size is in the restricted subrange 2k,N-2k , that is $2k \leq \text{COUNT}(T) \leq N-2k$.

The general tracker is not guaranteed to work when $k > N/4$; that is, when more than half the range of query set sizes is disallowed. However, this does not insure that the database is secure, because for a given C, there may exist a formula T for which the general tracker equations do work. The database may be vulnerable to compromise by the double tracker method using two characteristic formulas together. A double tracker is a pair of characteristic formulas (T,U) for which

$$X_T \subseteq X_U,$$
$$2k \leq \text{COUNT}(T) \leq N-2k,$$
$$2k \leq \text{COUNT}(U) \leq N-k.$$

The first paper on trackers [2] did not explore whether or not trackers of multiplicity greater than 2 exist for answering the unanswerable queries when $N/3 \leq k \leq N/2$. Since then the security problem for statistical databases has been extensivelly reported in the literature using Denning's work as a base. Recently Michalewicz [9] proposed a method that goes around the condition $k \leq |X_C| \leq N-k$, enabling calculation of $|X_C|$ (and q(c)) for all but two cases. Although the discussion in [9] is restricted to queries of the type COUNT(C), the author states that it can easily be expanded to arbitrary queries q(C) satisfying a given condition. The sequence P may be non-compromisable only in the following two cases:

(a) $|A| = 1$ and $|B| > 1$

(b) $|A| = 0$ and $k>N/3$.

Case (a) means that the sizes of the query sets for elementary conjunctions cannot be determined if there exists only one allowable elementary conjunction query q_i and there are no less than two elementary conjunction queries q_j such that the query $q = q_i + q_j$ is not allowed.

Case (b) means that the sizes of the query sets for elementary conjunctions cannot be determined if there is no allowable elementary conjunction query and the minimum size of an allowable query set is greater than one third the size of the database. In all other cases, the database can be compromised--even when the existence of trackers cannot be guaranteed.

Most of the work on threats to the security of statistical databases respond to suitable count or sum queries. However, mean values can be used equally effectively in compromising statistical databases, as demonstrated in [4].

A predicate for describing a subgroup, that is, for specifying which records in the database are used in answering the query is called a characteristic (chc). Characteristics correspond to simple predicates of the form category field=data value, or expressions built up from simple predicates and the Boolean operators v, x and ~. DeYonge [4] demonstrated that although one can determine the mean of a characteristic if one knows its sum and count, the converse is not true; from the mean of a characteristic one can not directly determine its corresponding sum or count.

To find a compromise that goes around the condition $k \leq |X_C| \leq N-k$, Michalewicz [9] considers elementary conjunctions; an elementary conjunction being a unique combination of category--values. If there are m category fields, and the i-th category field can take n_i values $v_{i_1}, \ldots, v_{i_{n_i}}$, the number of elementary conjunctions that can be formed is

$$N = \prod_{i=1}^{m} n_i .$$

Any expression C can be transformed into an equivalent disjunction $C_1 + \ldots + C_r$ of some number of elementary conjunctions with mutually disjoint sets X_{C_r}.

Let $C_1, C_2, \ldots, C_s$ be all the elementary conjunctions with a non-empty query set. Consider the sequence $P = (p_1, p_2, \ldots, p_s)$, where

$$p_j = |X_{C_j}| \text{ for } j = 1,2,\ldots,s.$$

For a fixed k and a given subset J of $\{1, 2, \ldots, s\}$, the sum

$$S(J,P) = \sum_{j \in J} p_j$$

is said to be allowable if $k \leq S(J,P) \leq N-k$. The threat to the security of a statistical database can then be stated as follows: If, given all allowable sums S(J,P), the values

P1,P2,...,Ps can be inferred, then the database can be compromised.

THE PERTURBATION LITERATURE

Chin and Ozsoyoglu [3], while advocating security constraints at the conceptual model level (which is beyond the scope of this paper), concede that perturbing or altering data records may be more effective than the security mechanisms suggested in the tracker literature. One possible approach at perturbing data is given by Beck [1]. He assumes the system will not refuse response to any query, including data concerning individual records. This does not necessarily mean the user is capable of specifying individual records. The user may still be restricted to characteristic formulas involving a limited set of category fields. Since he does not rule out requests for individual records, returned values must not be both accurate and reliable or the database could be compromised by any definition of compromisability. Two possibilities are discussed. The data could be altered when stored, and thereafter treated as true values, or the data could be stored accurately with distortion taking place when responses are reported to the user. Beck [1] correctly points out a problem with storing distorted data, that it is difficult to know by what amount to distort individual values when one does not know how queries will be combined. For example, if two individual values are each distorted by five, the difference of those values may be distorted by as much as ten. This creates the situation where a response with query set containing two individuals is more distorted than a response with query set containing one individual, which goes against the general principle that smaller query sets need more protection than larger query sets. Another problem is that changes in database policy cannot be implemented as easily. For example if a database administrator (DBA) decides more protection is needed, it may be difficult to determine what additional distortions must be made to stored data that are already distorted.

For these reasons a better solution is to perform the distortion at response time, based on accurate stored data. There are again two options. A query might always produce the same (distorted) response each time it is posed, or the response might vary at each request. A constant response has two advantages. First, it gives the appearance of an accurate response (one tends to identify accuracy and reliability). Any user who was not aware of the query processing mechanisms might therefore incorrectly apply threat mechanisms such as trackers to the database, thereby obtaining the illusion of knowledge. This might be a potentially fruitful area for further investigation. The other advantage of a constant response is that compromise may be more difficult. Beck's [1] argument that compromise is equally difficult whether the responses are distorted once or repeatedly perturbed is based on very generous assumptions concerning allowable characteristic formulas: either one must be able to construct arbitrary query sets of known composition, or arbitrary functions of category fields must be allowed. These assumptions are probably unrealistic and make his conclusions more restrictive, not less so. This is analagous to the nature of mathematical structures. More axioms may make proofs easier, but the results are less applicable. For example, all theorems concerning rings apply to fields, but not vice versa. The extra axioms applied to fields limit the applicability of the resulting theorems.

There are some problems returning a constant response one would not encounter with a variable response. In order to return a constant response with proper statistical characteristics, one must apply the same seed number each time to the pseudo-random number generator. One must determine whether the seed is a function of the semantics of the request, the query set, the value requested or some combination of the above.

All possibilities of data inference must be considered before one can determine whether there is any security advantage in returning a constant response. For instance, could the same request be stated in several ways, or even as the resultant of two separate requests. The chief advantage of repeated perturbations is therefore that it is easier to implemment than a constant distorted response.

Beck's [1] definition of compromisability assumes responses are perturbed at each request. The database is said to be compromisable if "it is possible to estimate a value y_i with $\hat{y}_i$ such that

$$\sigma(\hat{y}_i) < c \mid y_i - \bar{y} \mid$$

where c is a constant (for the entire database) which may be specified by the person having responsibility for database security." With this definition extreme values are afforded greater protection than values close to the mean.

One notes that for any given estimator $\hat{y}_i$, $\sigma(\hat{y}_i)$ is a population parameter and thus is constant. The question of compromisability therefore is whether the user can find a new estimator which is better than a single response, which is a point estimate. If one has many point estimates (several different query responses) the average $\overline{x}$ of those responses is an estimator with smaller standard deviation than that of any point estimator. One is tempted to interpret $\sigma(\hat{y}_i)$ as either the sample standard deviation of a particular set of query responses, or as the standard deviation of the population of all possible query responses. The latter is correct only if $\hat{y}_i$ is the estimator obtained from the response to a single query. If the average $\overline{X}$ of a set of n responses is used as an estimator, $\sigma(\hat{y}_i) = \sigma(\overline{X})$ represents the standard deviation of the population of all possible means of query response samples of size n.

A MODEL OF USER INFERENCE FOR A DATABASE WITH PERTURBED RESPONSES

The user is interested in inferring a numerical value F(I) associated with some individual I, where $I \in D$ (the set of individuals with records in the database). F usually represents the value of a field, but it may be a function of one or more fields. The user may be able to request F(I) directly, or the characteristic formulas available to him may not allow this. If Q is any query set, let $F(Q)$ denote the average value of $F(I)$ over the query set, i.e., $F(Q) = \text{avg}(\{F(I); I \varepsilon Q\}$ (Notationally we will not distinguish between I and $\{I\}$, so that $F(I)$ is the special case of $F(Q)$ where $Q = \{I\}$.) Let $\overline{F}$ denote $F(D)$, the mean value of F over the database.

One wishes to protect against user inference by perturbing responses to queries. The database does this by responding with an estimate $\hat{F}(Q)$ of $F(Q)$. Denote a response $\hat{F}(Q) = F(Q) + Ferr(Q)$, where $F(Q)$ is constant and $Ferr(Q)$ is a random variable with mean zero. Thus $\hat{F}(Q)$ is itself a random variable with mean $F(Q)$. Any given response $\hat{F}(Q)$ may be viewed as a mean $\overline{X}$ of a sample of size one ($n=1$) drawn from the set of possible responses to the same query. If the query is posed many times and independent responses are obtained ($n > 1$) then $\overline{X}$ is a point estimator that is likely to estimate $F(Q)$ closely for large n. Note that unless the error distribution is the same for each query, $\overline{X}$ cannot be considered a sample from a single population; this is the case with Beck's [1] "improved procedure."

A cautionary word concerning notation is in order at this point. $F(Q)$ denotes a unique value F associated with a specified query set Q. Thus, for example, $F_1(Q)$ and $F_2(Q)$ might represent two values, such as average salary and age, associated with the same query set, while $F(Q_1)$ and $F(Q_2)$ would represent the same requested value, such as salary, for two different query sets.

Given $F(O)$, $\hat{F}(O)$ represents any estimator to $F(O)$. Any given $F(O)$ may have many estimators, not all of which are unbiased. One estimator is the response to a direct query. Another estimator is the average response $\bar{X}$ to a repeated direct query.

Yet another estimator may be an algebraic combination of responses to a set of distinct queries, possibly no two of which have the same query set. For example suppose $F(O) = F_1(O) - F_2(O)$. The user wishes to estimate $F(O)$ by requesting $F_1(O)$ and $F_2(O)$ separately. The user may take the difference of the responses $\hat{F}_1(O)$ and $\hat{F}_2(O)$ as an estimator of $F(O)$. Since $\hat{F}_1(O) - \hat{F}_2(O)$ is used as an estimator of $F(O)$, it may also be denoted by $\hat{F}(O)$. Unless otherwise specified, however, $\hat{F}(O)$ will denote the point estimator based on a single direct query.

It was previously noted that a database can adhere to two possible policies in perturbing responses. Either the database can return the same value $\hat{F}(O)$ each time $F(O)$ is requested, or a different response can be generated with each request. The former yields an estimate of $F(O)$ based on a sample of one value, since in fact only one observation is generated. The user has no way to improve upon the estimate by repeated queries, and no confidence interval may be constructed based on the sample responses. (One may, however, obtain a confidence interval if one knows the standard deviation σ of the error function $Ferr(O)$.) For example if the distribution is normal, one may be 95% sure that the true value $F(O)$ lies in the interval $[\hat{F}(O)-1.96\sigma, \hat{F}(O)+1.96\sigma]$. Unfortunately, there is no way of improving on this estimate with additional queries. Beck [1] suggests the user estimate F(O) as a function of other queries, e.g. $\hat{F}_1(O_1) - \hat{F}_2(O_2)$. However $\hat{F}(O)$ and $\hat{F}_1(O_1) - \hat{F}_2(O_2)$ are distinct random variables with distinct distributions. One is left with two samples of n=1, not one sample with n=2.

Suppose the database samples from the same error distribution each time the query is posed. Then if the same query is repeated n times, one obtains estimates $\hat{F}_1(O), \hat{F}_2(O), \ldots, \hat{F}_n(O)$. The sample mean is then an unbiased estimator for F(Q). If one

$$x = \frac{\hat{F}_1(Q) + \ldots + \hat{F}_n(Q)}{n}$$

assumes a normally distributed error, one can calculate a confidence interval for F(Q). For example, one may be 95% sure that the true value F(O) lies in the interval

$$x - \frac{t_{\alpha/2}\, s}{\sqrt{n}} < F(Q) < x + \frac{t_{\alpha/2}\, s}{\sqrt{n}}$$

where s is the sample standard deviation of $\{\hat{F}_1(O), \ldots, \hat{F}_n(O)\}$ and $t_{\alpha/2}$ is the value of the t distribution, with n-1 degrees of freedom, leaving an area of 0.025 to the right. (See Figure 1.) Note that as the number of queries increases the confidence interval shrinks. In this sense $\overline{x}$ becomes a better estimate of F(Q) as n increases.

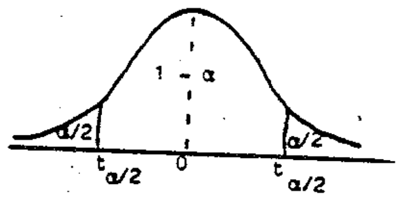


Figure 1. The t distribution

What does it mean to compromise the database in this model?

Beck's [1] definition is expressed solely in terms of the standard deviation σ of the estimator and the distance of the individual value from the mean value across all individuals. In the proposed model the database is said to be compromisable if there is an estimator $\hat{F}(I)$ for which

$$\sigma(\hat{F}(I)) < c\,|F(I)-\bar{F}|$$

where c is a constant for the entire database. According to this definition every database that returns independently generated perturbed responses is compromisable whereas, for a database returning a constant answer to successive queries, c may be selected so that the database is not compromisable.

The following definitions and principles in the generation of error distributions Ferr(Q) for a given query are postulated by the proposed model.

1. All error distributions should be unbiased.
2. The appropriate measure of protection is the standard deviation of the error term. Symbolically we define

$$\text{Protect}(F(Q)) = \sigma(\text{Ferr}(Q)).$$

3. <u>Definition.</u> A value F(Q) is <u>α-compromisable</u> by a given estimator $\hat{F}(Q)$ if

$$P(\,|\hat{F}(Q)-F(Q)| < \text{Protect}(F(Q))\,) > 1-\alpha\ .$$

(See Figure 2 for the meaning of this definition.) This definition reflects the fact that the measure of an estimator's goodness is how likely it is to be close to the true value. The protection level defines what is acceptably close for a given value.

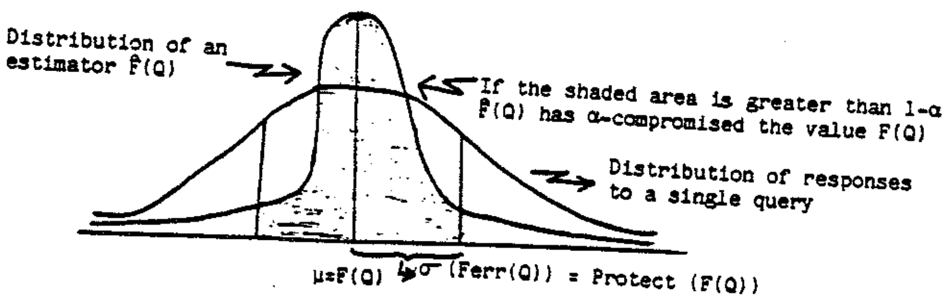

Distribution of an estimator $\hat{F}(Q)$
If the shaded area is greater than 1-α $\hat{F}(Q)$ has α-compromised the value F(Q)
Distribution of responses to a single query
μ=F(Q)
4σ (Ferr(Q)) = Protect (F(Q))

Figure 2. A probabilistic definition of compromise.

In general an unbiased estimator is the mean response $\bar{x}$ to n repeated queries. This becomes a successively better estimator as n becomes large and the standard deviation of $\bar{x}$ shrinks. In fact one can always find an n large enough to α-compromise F(Q).

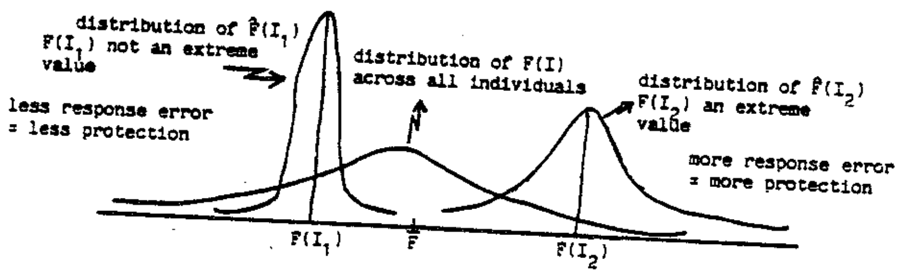
The database is compromisable if, for a predetermined α and integer M, there is an estimator $\hat{F}(I)$ of some individual value F(I) such that F(I) is α-compromisable by $\hat{F}(I)$ and $\hat{F}(I)$ may be expressed as a function of M or fewer queries (with independently generated responses).

Note that definitions in terms of probability apply to databases giving accurate responses as well as to those giving perturbed responses. For those giving accurate responses, a value will be compromisable if and only if its estimator, a function of responses to queries, is exact. Beck's [1] definition on the other hand would make every such database compromisable by any estimator.

4. Protection must be increased for extreme values. Symbolically, if $F_1(O_1)$ is more extreme than $F_2(O_2)$ then $\text{Protect}(F_1(O_1)) > \text{Protect}(F_2(O_2))$. This begs the question as to what constitutes an extreme value. Let $\text{EXT}(F(Q))$ be the <u>measure of extremity</u> for a value $F(Q)$. If the underlying distribution of $F(I)$ is normal, a more traditional measure of deviation from the mean is the normalized random variable

$$Z = \frac{X - \mu}{\sigma}.$$

In this case one could use $Z_I = \frac{F(I) - \overline{F}}{\sigma_F}$ where σ_F is the standard deviation of $F(I)$ across all individuals. (See Figure 3.) σ_F could be approximated by the sample standard deviation s_f for all individual records currently in the database.


distribution of $\hat{F}(I_1)$
$F(I_1)$ not an extreme value
less response error = less protection
distribution of $F(I)$ across all individuals
distribution of $\hat{F}(I_2)$
$F(I_2)$ an extreme value
more response error = more protection
$F(I_1)$
$\overline{F}$
$F(I_2)$

<u>Figure 3.</u> Increased protection for extreme values

$|Z_I|$ would then be the measure of extremity $\text{EXT}(F(I))$. For an arbitrary query set Q, let

$$Z_Q = \frac{F(Q) - \overline{F}}{\sigma_F / \sqrt{n}}$$

when n is the cardinality of the query set.

If the underlying distribution of individual values F(I) is not well approximated by the normal distribution, and if the cardinality of the query set is less than 30, a possible measure of extremity is given by:

$$EXT(F(Q)) = |PCT(F(Q)) - 50| \text{ where}$$

PCT(F(Q)) is the percentile of F(Q) when compared with all values of F over query sets of the same cardinality as Q.

5. Protection should be small for "valid" requests, large for suspicious requests. One can make some a priori judgements about the nature of valid requests as they relate to the size of the query set. In general one would want accurate responses where the query set is the entire database, Q=D. Responses to these would give global averages. Requests where the query set is an individual, Q=I, should be highly protected. The tracker literature has pointed out the possible inferences resulting from set differences. Thus query sets of the form Q = D-I must also be highly protected. A graph of protection versus query set cardinality, not considering other variables, might therefore look like Figure 4. In general all queries will be answered but the smaller the size of the query set the less accurate (hence useful) the response will be.

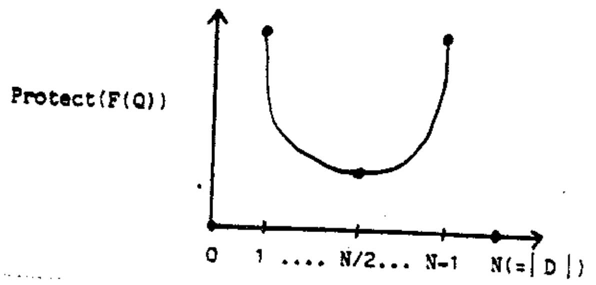


Figure 4. Measure of protection as a function of query set cardinality.

6. In general we know that a potential exits, using trackers, to determine F(I) if the database will accurately respond to F(Q) for most query sets. A global scaling factor $c(t,\alpha)$, changeable over time, could provide flexibility to the DBA if he determines that protection levels are low enough that some F(I) is α-compromisable by an estimator requiring fewer than the number of queries M considered minimally acceptable, or if it is determined that F(I) is compromisable by tracker--obtained estimators (for deductive purposes each response would be considered accurate by the user), or any other deductive method. Thus protection would consist of the product of two kinds of terms:

$$\text{Protect}(F(Q)) = c(t,\alpha)\ H(F(Q)),$$

where H is time independent and $c(t,\alpha)$ is independent of F(Q). At some time t the DBA may decide to increase $c(t,\alpha)$ if he suspects a compromise is possible.

7. A more advanced possibility, discussed by [3], is to develop an "intelligent" system that would be capable of determining inferences that could lead to compromise of an individual's record. The global scaling constant

could then be changed automatically by the threat monitoring system.

CONCLUSIONS AND SUGGESTIONS FOR FUTURE EXPLORATION

Many possible protection formulas fit the above criteria. A generic formula is given by:

$$\text{Protect}(F(Q)) = c(t,\alpha)\ [\text{EXT}(F(Q))]^p\ [T(|Q|)]^q$$

where $p,\alpha \geq 0$, and T is a function of query set cardinality, probably stored as a table because of the discontinuities at the end points. The generic formula points out the relevant elements: protection of extreme values, cardinality of query set as a determinant of a valid query, the need to specify the probabalistic level α at which the protection is provided, and the need to change the level of protection through threat monitoring. It is hoped that the above discussion and the notation developed may be an aid to future investigations, and that various protection formulas fitting the generic formula will be used for experimentation.

Each of the models of user inference discussed in this paper assumes the potential intruder knows the ground rules under which the database operates. Otherwise the intruder could not trust the validity of his conclusions. For example, trackers are only used when query responses can be assumed accurate. On the other hand, the proper use of statistical estimators requires some knowledge of how the system perturbs responses. For example, to estimate values using Beck's [1] sum of random variables one must know the number of random variables in the sum and the frequency with which those random numbers change value. Such knowledge may be in the public domain. It might be interesting, though, to experiment with the degree of inaccuracy one might expect if one attempted to crack a statistical database, whose security mechanisms are not known, using some of the methods proposed in the literature.

Particularly, one might experiment with the relation between the level of protection given a database with perturbed responses, using this paper's model of user inference, and the number of correct inferences of individual values one can obtain using tracker techniques. This might be a relevant question for databases which perturb values but return fixed responses to repeated queries. A user of such a system may assume such a system gives accurate responses and attempt inferences using trackers. In general, though, the distinction between a priori knowledge and deduced knowledge is most appropriate in the discussion of intelligent systems that can determine what the user is capable of inferring [10]. Much interest can be expected in this line of research in the near future.

REFERENCES

1. Beck, L. L. "A Security Mechanism for Statistical Databases." ACM TODS, 5, 3 (September 1980), 316-338.

2. Chin, F. Y. "Security in Statistical Databases for Queries with Small Counts." ACM TODS, 3, 1 (March 1978), 92-104.

3. Chin, F. Y. and Ozsoyoglu, G. "Statistical Database Design." ACM TODS, 6, (March 1981), 113-139.

4. De Jonge, W. "Compromising Statistical Databases Responding to Queries About Means." ACM TODS, 8, 1 (March 1983), 60-80.

5. Denning, D. E., Denning, P. J. and Schwartz, M. D. "The Tracker: A Threat to Statistical Database Security." ACM TODS, 4, 1 (March 1979), 76-96.

6. Denning, D. E. and Denning, P. J. "Data Security." ACM Computing Surveys, 11, 3 (September 1979), 227-249.

7. Dobkin, D., Jones, A. K. and Lipton, R. J. "Secure Databases: Protection Against User Influence." ACM TODS, 4, 1 (March 1979), 97-106.

8. Kam, Y. and Ullman, Y. "A Model for Statistical Databases and their Security." ACM TODS, Vol. 2, No. 1, (March, 1977).

9. Michalewicz, Z. "Compromisability of a Statistical Database." Information Systems, 6, 4 (1981), 301-304.

10. Ozsoyoglu, G. and Chin, F. Y. "Enhancing the Security of Statistical Databases with a Question-Answering System and a Kernel Design." IEEE Transactions on Software Engineering, SE-8, 3 (May 1982), 223-234.