Comparing Inference Control Mechanisms for Statistical Databases with Emphasis on Randomizing

Ernst L. Leiss
University of Houston, Department of Computer Science, Houston, Texas, 77004, U.S.A.
coscel@cs.uh.edu

AND
Jurij Jaklič
University of Ljubljana, Faculty of Economics, Kardeljeva pl. 17, 61000 Ljubljana, Slovenia
jurij.jaklic@uni-lj.si

Keywords: statistical databases, security, comparison, randomizing

Edited by: Jerzy R. Nawrocki
Received: July 27, 1994 **Revised:** February 24, 1995 **Accepted:** April 12, 1995

Statistical databases *are primarily collected for statistical analysis purposes. They usually contain information about persons and organizations which is considered confidential and only aggregate statistics on this confidential attribute are permitted.*
However, deduction of confidential data *(inference) is frequently possible. In order to prevent this possibility several security-control mechanisms have been developed, among them the* randomizing *method.*
We compare randomizing with other methods using several evaluation criteria. Evaluation shows that randomizing has several advantages in comparison with other methods, such as high level of security, robustness, low cost. On the other hand, the problem of bias for small query sets can be considerable for some applications.

1 Introduction

Databases often contain *confidential* information. Various security-control mechanisms deal with different kinds of security problems, such as encryption, identification of users, and access authorization. Here we will study inference, i.e., the deduction of confidential information from legal (i.e. permitted) statistical summary queries.

A *statistical database* (SDB) is a database where certain users are authorized to issue only aggregate (statistical summary) queries, such as sum, maximum, minimum, count, average, median, variance, standard deviation, and k-moment. These users (*researchers*) cannot retrieve information about an individual entry.

Typical examples are Census Bureau databases, salaries of individual persons in a company, and diagnoses of patients in a hospital.

We must enable these users to retrieve aggregate statistics, but prevent them from retrieving values of confidential attributes of individual records. In this way we protect the individual's right to privacy and on the other hand we can process information needed by the society [22].

Problems arise when certain users (*snoopers*) try to derive or infer confidential information from legal aggregate queries. If they are successful, we say that the SDB is *compromised.* There is more than one way how to compromise a SDB, for example the linear system attack [10] or using a tracker [8].

Several methods have been developed in order to protect SDB's against compromises. Most of them are described in [2], where they are also classified and evaluated. Randomizing security-control mechanisms such as the one described in [17] are just mentioned in [2]. Our goal is to evaluate this method and compare it with others.

2 Overview of the Methods

Two models offer frameworks for dealing with the problem of the SDB security [2]: one is called *the conceptual model* [5] and the other *the lattice model* [9]. They support the research in this field but do not offer methods for security control.

There are two approaches to the problem of inference for statistical databases:

- query restriction approach and
- perturbation approach.

Methods belonging to the same group have similar characteristics and are therefore easier to compare. In this section a brief overview of the two approaches is given.

2.1 Query Restriction Approach

The idea is that if we do not permit every aggregate query to be executed, we can achieve better security of the database. The proposed methods differ in the way in which it is decided whether the query is permitted.

The first method (*query set size control*) is to limit the query set size [13]. It has been noticed [10], that if the issued queries have many common entities (records) in their query sets, there is a higher possibility of compromising the database. So, the method *query set overlap control* proposes to restrict the number of records that any pair of issued queries can have in common. One of the methods which can provide a very high level of security is *auditing* [27]. Auditing keeps track of issued queries and checks for each new query whether the database can be compromised. The partitioning method ([4],[25]) and the cell suppression method [6] are other techniques.

2.2 Perturbation Approach

The perturbation approach includes two subgroups of methods. One subgroup replaces original data in the SDB with other data and uses these perturbed data to compute statistics (*data perturbation*) while the other subgroup computes statistics from the original data, but noise is added in one or another way during the computation of the statistics (*output perturbation*).

The *probability distribution methods* [19] replace the original SDB by another sample with the same (assumed) probability distribution. A variant of this method (the *analytical method*) approximates the data distribution by orthogonal polynomials (see [15]). The other proposed approach [23] is to replace the true values of a given attribute with the perturbed values once and forever (*fixed data perturbation*). There are two different methods, one for numerical and one for categorical attributes.

An example of the output perturbation methods is the *random sample queries* method [7]; a statistic of a randomly selected subset of the original query set is given as a response to the issued query.

In the *randomizing method* [17] a response to the given query is computed from a superset of the query set. Records are randomly selected from the database and added to the query set.

Let us assume that a user is interested in a certain statistic of a given query set of size k. Let $i_1, \ldots, i_k$ be the indices of the records in the query set, and let DK be the name of the confidential attribute in whose statistic the user is interested. Instead of computing the true statistic, another $v \geq 0$ entities of the database are selected randomly and added to the query set. Then the statistic of this superset of the original query set with $k+v$ records is computed and returned as the result. Thus for a query of type average we have the perturbed response

$$A = \frac{\sum_{j=1}^{k} DK_{i_j} + \sum_{j=1}^{v} \mathrm{DK}_{s_j}}{k+v} = \\ = \frac{k \cdot a + \sum_{j=1}^{v} \mathrm{DK}_{s_j}}{k+v},$$

where s_j is the index of the j-th randomly selected record and a is the true response $(\sum_{j=1}^{k} DK_{i_j})/k$.

v should be (much) smaller than k, otherwise the precision of the result can be very bad, although the security of this method increases with higher values for v. That yields a *security-accuracy trade-off*. Here we select v to be equal to 1, justified by the observation (see [17]) that even with small introduced noise, the relative error of the inferred values for DK will be considerable for large values of k.

3 Comparison

We use the evaluation criteria proposed in [2] to compare the different methods; they cover the important objectives of a good security control mechanism. Some of the criteria exclude each other, and thus an effort must be directed towards balancing them.

3.1 Security

We consider different kinds of disclosures:

- *Exact disclosure* is possible when a user can obtain the exact value of the confidential attribute.
- *Partial disclosure* is possible when a user can obtain an estimate DK_i' of the value of the confidential attribute DK for the i-th record, such that $Var(DK_i') < c_1^2$ (i.e. variance of the estimate DK_i' is less than c_1^2), where the parameter c_1 is set by the DBA (Data Base Administrator). For the case of categorical attributes a *partial disclosure* is possible when a user can infere that the confidential attribute *has not* a certain value.
- *Statistical disclosure* occurs when the same query is issued several times in order to obtain a small variance of the estimate of the true response - *filtering.*

Let us look first at the exact compromisability of an SDB under different protection methods. There is the possibility of exact disclosure for some methods belonging to the query restriction approach group, namely the query set size control and the query set overlap control.

Exact compromise cannot be done for a SDB protected by the auditing because of the nature of the auditing. This is also true for cell suppression and partitioning (see [6] and [26]).

For methods belonging to the perturbation group, including randomizing, there is no possibility for the exact disclosure, except for some rare cases for the analytical method (see [2]) and for rounding [1]. Conditions under which exact disclosure is possible for these two methods are very severe.

There are more possibilities for partial disclosure. Regardless of which method we use, it is possible to achieve partial disclosure. As for the most of the perturbation approach methods, it is possible to balance the security against the precision also for randomizing. The parameters which influence the security (and precision) and can be set by the DBA are: v, the number of records to be added to the query set and j, the parameter which is used in the method to solve the problem of accuracy (see Section 3.2).

The problem of statistical disclosure has to be considered only in the cases where answers to the same query issued several times differ. For these (perturbation) methods one can achieve a better estimate of the real answer to the query using the method of filtering. The idea is that the user repeats the same query several times. Let A_n denote the perturbed response to the n-th repetition of the query with the true response a. In general $A_i \neq A_j$ for $i \neq j$. Then the user can repeat the query m times and compute the average

$$a' = \frac{A_1 + A_2 + \ldots + A_m}{m}.$$

The result of this expression will converge to a certain value a^* with increasing m. If the perturbed response (after one repetition) is A then we have

$$Var(a') = \frac{Var(A)}{m},$$

if the answers to repeated queries are independent. Thus, we see that the more times one repeats the query, the better an estimate a' of the true response a can be achieved.

Let us see how many queries (m_r) we have to issue if we want to get statistical disclosure of the SDB protected by the basic randomizing method, if the criterion for the statistical disclosure is

$$Var(a') < c_1^2,$$

where c_1 is the parameter set by the DBA. Let us consider a fixed query of type average with the true value a. The perturbed value of that query is

$$A = \frac{a \cdot k + \mathrm{DK}_s}{k+1},$$

where k is the size of the original query set and DK_s is the value of the confidential attribute DK of the randomly selected record. If the random number generator we use is uniform, then we can expect that on the average the value for DK_s is

equal to the average of the values over the database (DK*). Thus the expected value of A is

$$E(A) = \frac{a \cdot k + \mathrm{DK}^*}{k+1}.$$

Now we can compute the variance of A:

$$Var(A) = E\left(\frac{a \cdot k + \mathrm{DK}_s}{k+1} - \frac{a \cdot k + \mathrm{DK}^*}{k+1}\right)^2 = \\ = \frac{E(\mathrm{DK}_s - \mathrm{DK}^*)^2}{(k+1)^2} = \frac{Var(\mathrm{DK})}{(k+1)^2}.$$

Because the responses to the queries are independent of each other, we have

$$Var(a') = \frac{Var(A)}{m} = \frac{Var(\mathrm{DK})}{m \cdot (k+1)^2}.$$

As we expected, the variance of the estimate depends on the variance of the confidential attribute and it is smaller for larger query set size. Thus we have:

$$m_r \geq \frac{Var(\mathrm{DK})}{c_1^2 \cdot (k+1)^2}.$$

Even if the number of queries needed to compromise a SDB can be quite large, it is possible to do it. The reason is that with the increasing m the average introduced noise of the query always converges to the same value DK^*. In order to avoid this one can use the following method.

For each issued query two calls to the random number generator are made, and therefore two indices for the additional record are proposed. The selection of the single record to be added to the query set depends on the values of the confidential attribute of the records which are already in the query set. Let us say that the two proposed indices are x_1 and x_2. Then we choose $\max\{x_1, x_2\}$ if the value of the boolean expression

$$E = [(DK_{i_1} \leq DK_{i_2}) \oplus (DK_{i_2} \leq DK_{i_3}) \oplus \ldots \\ \ldots \oplus (DK_{i_{k-1}} \leq DK_{i_k})]$$

is true, and $\min\{x_1, x_2\}$ otherwise. Here $\oplus$ denotes the XOR operator. Thus the average introduced noise may differ for two different queries and therefore the database cannot be compromised.

3.2 Accuracy of Responses

The problem of the accuracy of responses occurs when we use perturbation methods. Using query restriction control methods, the responses to queries are always equal to the true responses.

We consider two criteria, namely *bias* and *precision*, i.e. variance of the estimator.

As stated in [2] the main disadvantage of the randomizing method in comparison with other output perturbation methods is the problem of bias. A bias occurs when

$$E(a|A = w) \neq w$$

where again a is the true response to a fixed query and A is the perturbed value for that query.

In our case of the randomizing method and for queries of type average, we have

$$a = \frac{A \cdot (k+1) - \mathrm{DK}_s}{k}.$$

¿From this we can compute

$$E(a|A = w) = \frac{(k+1) \cdot w}{k} - \frac{E(\mathrm{DK}_s|A = w)}{k}.$$

Since, the selection of the random record does not depend on the query (for the basic method), we can obtain the final result

$$E(a|A = w) = \frac{(k+1) \cdot w}{k} - \frac{\mathrm{DK}^*}{k} = \\ = w + \frac{w - \mathrm{DK}^*}{k},$$

where DK* is again the average over all database. It follows that in the limit, $k \to \infty$, the bias will be *zero*.

The variance of the perturbed value A for randomizing is $\frac{Var(DK)}{(k+1)^2}$. So, for large values of k (query set size) this method gives us quite good results, which means precise and without bias. The only parameter which influences the precision is the query set size; the variance of the perturbed value is proportional to the variance over all the database.

The problem of accuracy is that the maximal error introduced by the randomizing can be arbitrarily bad [17]. The average error is not so bad, but the maximal error can be very unpleasant, specially for smaller k.

This problem can be solved, if we do not permit that the additional value is very different from the values of the records from the query set. Thus, for a query of type *average* we stipulate that the chosen record satisfy the condition

$$avg - \frac{mx + mn}{2 \cdot j} \leq DK_s \leq avg + \frac{mx + mn}{2 \cdot j},$$

where s is the index of the selected record, avg, mx and mn are the average, minimal and maximal values of the confidential attribute in the specific query set, and j is a parameter set by the DBA. If the first selected record does not satisfy the condition, then we select another record, and so on. Of course we must set a limit on the number of repetitions. Here the selection of j is essential. If j is too small then we do not restrict randomizing; on the other hand if j is too large, possibly no record will satisfy the condition.

It is difficult to say which of the perturbation methods is more precise, because the precision depends to a great deal on the selection of parameters of a given method.

All data perturbation methods, except the fixed data perturbation method for categorical attributes, suffer from the problem of bias. On the other hand, among the output perturbation approaches only randomizing has this problem. This problem might be considerable for small databases with high variance of the confidential attribute.

3.3 Consistency

A security control method is *consistent* if there are no contradictions or paradoxical results, e.g., if we get different responses to the repetition of the same query, or when the response on the average query differs from the quotient of the sum and count queries over the same query set.

All query restriction methods are consistent. The only thing we have to take care about is the possibility that the same query is once restricted and once not (e.g. query set overlap control). Also data perturbation methods do not give contradictory results, but we can obtain some paradoxical results, such as negative salaries.

On the other hand, all probability distribution methods, random sample queries and varying output perturbation methods are inconsistent. Since the randomizing method belongs to the random sample queries methods it is inconsistent too. But we can overcome this problem if we use quasi-randomizing [18] instead of the basic method described in [17].

In order to select a random record to be added to the query set we use a random generator. Each random generator is a function of a parameter *seed*, and for the same seed the same sequence of random numbers is generated. So, when we want to generate a random index of a record, we can use as a seed a function of the query set; thus for the same query set the randomly selected index will be always the same. The requirement for the function which maps a query set into a seed is that it does not change for any permutation of the query set. A simple solution is the sum of the values of the confidential attribute. If

$$QS = \{DK_{i_1}, \ldots, DK_{i_k}\},$$

then

$$\text{seed}(QS) = DK_{i_1} + \cdots + DK_{i_k}$$

or

$$s = g(\text{Rand}(DK_{i_1} + \cdots + DK_{i_k}))$$

where s is the randomly selected index and g is some function which maps random numbers into the set of indices of the records in the database. Another advantage of this method is that statistical disclosure is not possible, because responses to the same query issued several times are always the same. The problem of paradoxical values for randomizing is not as severe as for the fixed data perturbation method.

3.4 Robustness

We say that a security control method is robust if supplementary knowledge does not help a user who wants to compromise the SDB. Supplementary knowledge is considered to be all the information about the database which a user knows from a source other than the system [2]. In general the robustness of the query restriction approach methods is very low, since the responses to queries for these methods are always correct. So with supplementary knowledge about the database one can easily compute other values. Robustness can be controlled for some methods such as partitioning [26]. Very severe is the problem of robustness for auditing because queries with very small query set sizes may be permitted.

The perturbation methods are more robust, clearly because perturbed answers are returned to a user. Their robustness varies from moderate (in most cases) to high for the data swapping method and can be usually controlled by the parameters of a particular method.

The robustness of the randomizing method is high. In fact, if the number of elements known

by the user is small in comparison to the query set size, the SDB is still secure. As stated in [16, pp. 15] also for the number of known elements approximately equal or larger than k (query set size), the number of repetitions of queries one has to issue in order to compromise the database is quite large. This is even more pronounced if quasi-randomizing is used.

3.5 Cost

We consider the cost of the implementation of the security control mechanism, the processing overhead per query and the education of the user necessary to understand the responses.

For randomizing we can say that the initial implementation effort is very low for the basic method and a bit more complicated for the mechanisms which provide higher security (see Section 3.1), higher accuracy (see Section 3.2) and consistency (see Section 3.3).

Some methods have very low processing overhead, for example query set size control, cell suppression, all data perturbation methods and random sample queries. On the other hand there are methods such as query set overlap control and auditing, which are time and space consuming. Even more disturbing than the average complexity of comparisons is the fact that the processing overhead is much larger for the queries issued later than for the queries issued at the beginning.

The randomizing method has low processing overhead, the time overhead required per query is constant and there is no need for additional space. From this point of view all variants of randomizing are basically low in cost.

The randomizing method follows the majority of the methods regarding the cost of the education of the user, which means that it is low in cost - simple to understand.

3.6 Generality

Some of the methods described in [2] are not developed for all types of aggregate statistic queries. For example, the varying output perturbation is developed only for *sum*, *count* and *percentile* statistics.

The randomizing method can be used for almost all types of queries, with the exception of *count*. But there are obviously some differences between the usage of randomizing for statistics such as sum or average and usage of the same method for selector functions as median, min, max.

While the precision for queries of type average is good, i.e. $Var(A) = \frac{Var(DK)}{(k+1)^2}$, we cannot say the same for the selector functions. In the worst case for the max and min queries the difference between the true response and the randomized response can be as large as the difference between the maximal and minimal value of the confidential attribute in the database, $\max_{i=1,\ldots,N}|\mathrm{DK}_i| - \min_{i=1,\ldots,N}|\mathrm{DK}_i|$. On the other hand, a user can get also the true response. If we choose an arbitrary query set of size k, then the probability that the answer given to the user will be exact, is equal to

$$\sum_{m=k}^{N} \frac{\binom{m-1}{k-1}}{\binom{N}{k}} \cdot \frac{m}{N} \approx \frac{k}{N}.$$

In order to solve this problem one can use the restricted randomizing method. However, we know that the value of the perturbed response to a query of type *max* is greater or equal to the true response. Thus, the problem of bias is here more severe.

The situation is the same for queries of type *min*, but not for queries of type *median*: here the answer will change often, but the error is usually small and depends on the variance of the database and the selection of the query set.

Since there are few perturbation methods which can be used for all types of queries, we can consider randomizing as a general method, even though it is not equally good for all types of queries.

3.7 Suitability

It is desirable that a method be applicable for both numeric and categorical attributes. All query restriction approaches are able to deal with both. Two fixed data perturbation methods have been developed, one for numeric attributes [28] and the other one for categorical attributes [2]. The probability distribution and the random sample queries can be used for both. The randomizing method is not suitable for categorical, but only for numeric attributes.

In addition, some methods are suitable for more than one attribute and others for only one attribute. Again all the query restriction methods are basically suitable for more than one attribute.

The only problem is for auditing, where the processing overhead can become very high and this method may have no practical value. From the perturbation methods the following are suitable for more than one attribute: data swapping, analytical method, random sample queries, if the attributes are independent, also the varying output perturbation, as well as randomizing.

The last criteria is suitability to dynamic SDB. For some purposes a SDB can be static, for example a census database. For other purposes it is very important that the database be dynamic and on-line. For such a database the method used must provide security also in cases of changes in the database. Moreover, it must not require too much processing overhead per change in the database. Randomizing is completely suitable for on-line dynamic databases and does not require any additional effort for implementation nor any processing overhead per change in the database. There are some methods which are not suitable to on-line dynamic SDB at all: cell suppression, data swapping, and probability distribution. Note that all the output perturbation methods are suitable for on-line dynamic SDB.

3.8 Information Lost

Information lost is defined as "amount of non-confidential information that is *unnecessarily* eliminated, as well as, in the case of perturbation methods, the statistical quality of the information provided to the users" [2]. This means that in the case of perturbation methods the term information lost corresponds to precision. In other words: the higher the precision is the lower the information lost is. This applies also to the randomizing method.

The situation is different for the query restriction methods. Information lost can be very high for the query set size restriction and for the query overlap restriction approaches.

The auditing method restricts by its nature only queries that can lead to compromise. Following the definition of the information lost (see above) we can say that this method causes no information loss. However, the price that must be paid for this is very high (see Section 3.5).

It is more difficult to give an estimation of information lost for the partitioning and cell suppression methods because it depends on data in the database. Some empirical results (see [26]) show that information loss can be very severe for partitioning. In order to reduce it, *dummy records* have been proposed.

3.9 Conclusions

In summary we can say that the randomizing method has more than one advantage in comparison with other methods. It assures high security, robustness, and precision if the improvements described in this section are used. Another very important advantage is that the method is among the lowest in cost.

The disadvantages are: its bias (however it tends to be small for large sizes of query sets); it is not suitable for categorical data; it is not suitable for some types of queries.

References

[1] Achugbue J.O. and F.Y. Chin. "The effectiveness of output modification by rounding for protection of statistical databases." *INFOR Journal*, Vol. 17, No. 3, August 1979, pp. 209-218.

[2] Adam N.R. and J.C. Wortmann. "Security-Control Methods for Statistical Databases: A Comparative Study." *ACM Computing Surveys*, Vol. 21, No. 4, December 1989, pp. 515-556.

[3] Beck L.L. "A security mechanism for statistical databases." *ACM Trans. Database Syst.*, Vol. 5, No. 3, September 1980, pp. 316-338.

[4] Chin F.Y. and G. Özsoyoğlu. "Security in partitioned dynamic statistical databases." *In Proceedings of the IEEE COMPSAC*, 1979, pp. 594-601.

[5] Chin F.Y. and G. Özsoyoğlu. "Statistical database design." *ACM Trans. Database Syst.*, Vol. 6, No. 1, March 1981, pp. 113-139.

[6] Cox L.H. "Suppression methodology and statistical disclosure control." *Journal of American Statistical Association*, Vol. 75, No. 370, June 1980, pp. 377-385.

[7] Denning D.E. "Secure statistical databases with random sample queries." *ACM Trans. Database Syst.*, Vol. 5, No. 3, September 1980, pp. 291-315.

[8] Denning D.E., P.J. Denning and M.D. Schwartz. "The tracker: A threat to statistical database security." *ACM Trans. Database Syst.*, Vol. 4, No. 1, March 1979, pp. 76-96.

[9] Denning D.E. and J. Schlörer. "Inference control for statistical databases." *Computer*, Vol. 7, No. 16, July 1983, pp. 69-82.

[10] Dobkin D., A.K. Jones and R.J. Lipton. "Secure databases: Protection against user influence." *ACM Trans. Database Syst.*, Vol. 4, No. 1, March 1979, pp. 97-106.

[11] Fellegi I.P. "On the question of statistical confidentiality." *Journal of American Statistical Association*, Vol. 67, No. 337, March 1972, pp. 7-18.

[12] Friedman A.D. and L.J. Hoffman, "Towards a fail-safe approach to secure databases." *In Proceedings of the IEEE Symposium on Security and Privacy*, 1980.

[13] Hoffman L.J. and W.F. Miller. "Getting a personal dossier from a statistical data bank." *Datamation*, Vol. 16, No. 5, May 1970, pp. 74-75.

[14] Jaklič J. "Protecting statistical databases by randomizing and other methods: Comparison and simulation." *Master thesis*, Dept. of Computer Science, University of Houston, December 1992.

[15] Lefons D., A. Silvestri and F. Tangorra. "An analytic approach to statistical databases." *In Proceedings of* 8^{th} *International Conference on Very Large Databases*, 1983, pp. 260-273.

[16] Leiss E.L. "Protecting statistical databases through randomizing." *Technical Report #UH-CS-81-07*, University of Houston, December 1981.

[17] Leiss E.L. "Randomizing, a practical method for protecting statistical databases against compromise." *In Proceedings of* 8^{th} *International Conference on Very Large Databases*, 1982, pp. 189-196.

[18] Leiss E.L. "Principles of data security." *Plenum Press*, 1982.

[19] Liew C.K., W.J. Choi, and C.J. Liew. "A data distortion by probability distribution." *ACM Trans. Database Syst.*, Vol. 10, No. 3, pp. 395-411.

[20] Matloff N.E. "Another look at the use of noise addition for database security." *In Proceedings of the IEEE Symposium on Security and Privacy*, 1986.

[21] Morris J.L. "Computational methods in elementary numerical analysis." *John Wiley & Sons*, 1983.

[22] Palley M.A. and J.S. Simonoff. "The use of regression methodology for compromise of confidential information in statistical databases." *ACM Trans. Database Syst.*, Vol. 12, No. 4, December 1987, pp. 593-608.

[23] Reiss S.P. "Practical data-swapping: The first steps." *ACM Trans. Database Syst.*, Vol. 9, No. 1, March 1984, pp. 20-37.

[24] Reiss S.P. "Practical data-swapping: The first steps." *In Proceedings of the IEEE Symposium on Security and Privacy*, 1980.

[25] Schlörer J. "Information loss in partitioned statistical databases." *Comput. Journal*, Vol. 26, No. 3, 1983, pp. 218-223.

[26] Schlörer J. "Disclosure from statistical databases: Quantitative aspects of trackers." *ACM Trans. Database Syst.* Vol. 5, No. 4, December 1980, pp. 467-492.

[27] Schlörer J. "Confidentality of statistical records: A treat monitoring scheme of on-line dialogue." *Methods Inform. Med.*, Vol. 1, No. 15, 1976, pp. 36-42.

[28] Traub J.F., Y. Yemini and H. Wozniakowski. "The statistical security of a statistical database." *ACM Trans. Database Syst.*, Vol. 9, No. 4, December 1984, pp. 672-679.