T-79.511 Special Course on Cryptology / Privacy-Preserving Data Mining: Revealing Information while Preserving Privacy

Emilia Oikarinen

October 30, 2003

Abstract

This survey examines the tradeoff between privacy and usability in statistical databases as discussed by Irit Dinur and Kobbi Nissim in [4].

1 Introduction

In this survey we explore under which conditions one can achieve statistical information from a database and still preserve privacy for the individual entries. The discussion is based on an approach presented by Dinur and Nissim [4]. Most of the explicit references to [4] are omitted and some parts of [4] are covered only very briefly.

Consider a hospital database consisting of medical history of a population. While the privacy of individuals should be maintained, medical research might advance if statistical information about the database could be used. Thus the hospital needs an access mechanism that allows statistical queries without violating the privacy of any single patient. One solution would involve removing all identifying attributes (e.g. patient's name and social security number) from the database. This, however, does not protect privacy. Usually it is rather easy to identify patients indirectly from the attributes in the database. Identification can be, e.g. based on attributes such as patient's approximate age, sex, marital status and so on. Situation is even worse, if the patient is known to have some rare attribute.

Dinur and Nissim [4] show that if a database of size n is perturbed with random noise of magnitude $\leq \mathcal{E}$, there is a threshold phenomenon where an adversary can reconstruct almost all the database entries if $\mathcal{E} \ll \sqrt{n}$, and if $\mathcal{E} \gg \sqrt{n}$ the adversary can reconstruct none of them.

Organization of this survey is as follows. In Section 2 we discuss the basic concepts of statistical databases and what is meant by database privacy. In Section 3 we present a lower bound on the perturbation needed to maintain any reasonable notion of privacy. In Section 4 we see that if the adversary's complexity is further restricted than it is possible to achieve privacy using smaller perturbation. We end this survey with conclusions in Section 5.

2 Basic Concepts

We start by presenting some notations.

- $\text{neg}(n)$ — a function that is asymptotically smaller than any inverse polynomial, i.e. for all $c > 0$ and for all sufficiently large n, it holds that $\text{neg}(n) < 1/n^c$.
- $\text{dist}(c, d)$ — the *Hamming distance* of two binary strings $c, d \in \{0,1\}^n$, i.e. $\text{dist}(c,d) = |\{i \mid c_i \neq d_i\}|$.
- $\tilde{O}(T(n)) = O(T(n) \log^k(n))$, for some $k > 0$.
- Let $\mathcal{M}$ be a Turing-machine. We denote by $\mathcal{M}^{\mathcal{A}}$ an $\mathcal{A}$-oracle Turing-machine, where $\mathcal{M}$ has an access to algorithm $\mathcal{A}$ and each call to $\mathcal{A}$ costs a unit time.

2.1 Model-Statistical Databases and Statistical Queries

A statistical database is a query-response mechanism that enables users to access its contents via statistical queries. In this survey we consider *binary databases*, where the content is n binary (0/1) entries. A statistical query specifies a subset of the entries in the database and the answer to the query is the number of entries within the specified entries having the value 1.

Definition 1 Let $d = (d_1, \dots, d_n) \in \{0,1\}^n$. A (statistical) query is a subset $q \subseteq \{1, \dots, n\}$. The (exact) answer to a query q is the sum of all database entries in q, i.e. $a_q = \sum_{i \in q} d_i$.

Users issue statistical queries to the database and the response to the queries is computed by a database algorithm accessing the database content. The database algorithm may also keep additional information (in the form of an *internal state*) and update its state whenever invoked.

Definition 2 A (statistical) database $\mathcal{D} = (d, \mathcal{A})$ is a query-response mechanism. The response to a query q is $\mathcal{A}(q, d, \theta)$, where θ is the internal state of the algorithm $\mathcal{A}$. The computation may affect the internal state.

We usually omit d and θ and write $\mathcal{A}(q)$ for $\mathcal{A}(q, d, \theta)$. We are not concerned with the specifics of the database language or its indexing mechanisms. Instead, we assume that the user is able to specify any subset of the entries in the database.

2.2 Privacy Methods for Statistical Databases

The privacy methods for statistical databases can be divided into three main categories: (i) query restriction, (ii) data perturbation and (iii) output perturbation. Let us examine these categories.

In the **query restriction** approach, the queries have to obey a special structure so that the querying adversary is prevented from gaining too much information about the database entries. The usefulness of this approach is limited since it allows only a relatively small number of queries. Another idea related to query restriction is *query auditing* [3]. In query auditing, a log of queries is kept and each new query is checked for a possible compromise on the privacy. The query is allowed or disallowed according to the result of the check. The problem with query auditing is that the auditor's refusals together with the answers to valid queries may leak information about the database.

In the **data perturbation** approach, the database is first perturbed with some noise and the queries are answered according to the perturbed database. The methods for database perturbation include *swapping* [6] and *fixed perturbations* [7]. When swapping is used, portions of the data are replaced by a data taken from the same distribution. In fixed perturbation approach random perturbation is added to each database entry.

In the **output perturbation** approach an exact answer to the query is computed but a noisy version of it is returned. Output perturbation methods include *varying output perturbations* [2] and either *deterministic* [1] or *probabilistic* [5] *rounding.* In varying output perturbations method a random perturbation is added to each query answer with an increasing variance for a repeated query.

In this survey, as in [4], we will consider the perturbation methods. The problem is to find a suitable perturbation level giving some privacy and allowing gaining useful information from the database. The quality of a database algorithm is defined in terms of the magnitude of its perturbation.

Definition 3 An answer $\mathcal{A}(q)$ is within $\mathcal{E}$ perturbation if $a_q - \mathcal{A}(q) \leq \mathcal{E}$. An algorithm $\mathcal{A}$ is within $\mathcal{E}$ perturbation if for all queries $q \subseteq \{1, \dots, n\}$ the answer $\mathcal{A}(q)$ is within $\mathcal{E}$ perturbation[1].

2.3 Database Privacy

To achieve database privacy one has to balance two set of functions: (i) private functions one wishes to hide and (ii) information functions one wishes to reveal. Based on this general view, it is possible to define privacy in many ways. In most cases, single entries of the database are taken as privacy functions, i.e. $\{\pi_i\}_{i \in \{1,\dots,n\}}$, where $\pi_i(d_1, \dots, d_n) = d_i$. This captures the intuition that privacy is compromised if an adversary is able to compute a confidential attribute d_i from its identity i. In statistical database privacy the information functions are usually the sums of subsets of the database entries, i.e. $\{f_q\}_{q \subseteq \{1,\dots,n\}}$, where $f_q(d_1, \dots, d_n) = \sum_{i \in q} d_i$.

[1] Note that it would be sufficient for most of the results in this survey to assume that $\mathcal{A}(q)$ is within $\mathcal{E}$ perturbation for all but a negligible fraction of the queries q.

Dinur and Nissim present a *computational* definition of privacy that asserts that it is *computationally infeasible* to retrieve private information from the database. Other measures of privacy used in previous works include e.g. variance of query answers and the estimator variance. These definitions have some drawbacks. First it is not at all clear that large variance provides privacy as can be seen from the example below. Also, such definitions do not allow us to capitalize the limits of an adversary.

Example 4 Consider $d_i \in \{0, 1\}$ and an estimator $\tilde{d_i} = d_i + Ee$, where $e \in \{-1, 1\}$ is selected at random and E is a large *even* number. Although variance of $\tilde{d_i}$ is very large, d_i can be computed exactly from the last bit of just one sample $\tilde{d_i}$. ■

In **cryptography** privacy is often treated in a complementary manner to the one Dinur and Nissim present. Consider a secure function evaluation. Several parties want to compute a function F of their private inputs $d_1, \ldots, d_n$. Privacy is seen as protecting each party's private input so that other parties cannot deduce any information that is already deducible from the function outcome $F(d_1, \ldots, d_n)$. Thus F dictates which information is to be revealed and the goal is not to leak any information in addition to that. This privacy definition is implicit. Depending on function F a whole lot of information about the private inputs can leak and on the other hand F may be defined in such a manner that no information is leaked.

Dinur and Nissim reverse the order. They define first explicitly which information is not to be revealed and then look for functions that reveal maximum information still possible. Thus instead of implicit privacy, they want privacy definition to be explicit. Instead of giving this explicit definition on privacy first, they consider what privacy is not[2] and define the concept of *non-privacy*, a situation that should not be allowed to occur in any reasonable private database setting. They say that a database in non-private, if a computationally-bounded adversary can expose $1 - \varepsilon$ fraction of the database entries for all constant $\varepsilon > 0$. Thus non-privacy excludes even very weak notions of privacy.

Finally, in Section 4, we consider the definition of privacy given by Dinur and Nissim. Privacy is defined with respect to a bounded adversary with no prior knowledge about the database. The definition tries to capture the fact that the adversary should not be able to predict the ith bit regardless of the content of the rest of the database. The adversary is modeled by a two-phased game. First, the adversary queries adaptively the database and then outputs an index i it intends to guess. In the second phase, the adversary is given the query-response transcript of the first phase plus all but the ith database entries. Privacy is preserved, if with a high probability adversary fails to guess d_i.

3 Impossibility Results

The main result presented in this section is a lower bound on the perturbation needed to maintain any reasonable notion on privacy. Dinur and Nissim show that whenever the perturbation magnitude is smaller than $\sqrt{n}$, an adversary can construct a good approximation of the entire database using only polynomial number of queries.

As discussed in Section 2, we want to formalize non-privacy. Intuitively, if a database is non-private, an adversary can efficiently and very accurately reconstruct the entire database.

Definition 5 (**Non-privacy**) A database $\mathcal{D} = (d, \mathcal{A})$ is $t(n)$-non-private, if for every constant $\varepsilon > 0$ there exists a probabilistic Turing-machine $\mathcal{M}$ with time-complexity $t(n)$ such that

$$\begin{aligned}\Pr[\mathcal{M}^{\mathcal{A}}(1^n) \text{ outputs } c \text{ s.t. } \operatorname{dist}(c, d) < \varepsilon n] \\ \geq 1 - \operatorname{neg}(n),\end{aligned}$$

the probability taken over coin tosses of $\mathcal{A}$ and $\mathcal{M}$.

Note that Definition 5 bounds implicitly the number of queries the adversary can issue by $t(n)$. Also, from now on, we will restrict the adversary by making the queries non-adaptive. Thus, the adversary first specifies its set of queries and is then answered by the database access algorithm.

Also, it is worth noticing that in the results to be shown, the perturbation method can be either data or output perturbation.The adversary's algorithm is oblivious of the perturbation method used.

3.1 Exponential Adversary

We consider first a very powerful adversary that may issue all possible queries (thus the result pre-

[2] There are several definitions for privacy in databases, none of which seems to be good for every possible situation.

sented here pertains only to very small databases). Dinur and Nissim show that such an exponential adversary can violate privacy even if the perturbation magnitude is almost linear.

Theorem 6 *Let $\mathcal{D} = (d, \mathcal{A})$ be a database where $\mathcal{A}$ is within $o(n)$ perturbation. Then $\mathcal{D}$ is* $\mathbf{exp}(n)$*-non-private.*

Proof. Let $\mathcal{A}$ be within $\mathcal{E} = o(n)$ perturbation. Let $\mathcal{M}$ be the following algorithm.

(i) (Query phase)
For all $q \subseteq \{1, \ldots, n\}$, let $\tilde{a}_q = \mathcal{A}(q)$.

(ii) (Weeding phase)
For all $c \in \{0,1\}^n$, if $|\sum_{i \in q} c_i - \tilde{a}_q| \leq \mathcal{E}$ for all $q \subseteq \{1, \ldots, n\}$, then output c and halt.

The algorithm clearly runs in exponential time. Also, $\mathcal{M}$ always halts and outputs a candidate c. We show that c satisfies $\mathrm{dist}(d, c) \leq 4\mathcal{E} = o(n)$. Assume the contrary, i.e. $\mathrm{dist}(d, c) > 4\mathcal{E}$. Let $q_0 = \{i \mid d_i = 1, c_i = 0\}$ and $q_1 = \{i \mid d_i = 0, c_i = 1\}$. It holds $|q_0| + |q_1| = \mathrm{dist}(d, c) > 4\mathcal{E}$ and q_0 and q_1 are disjoint. Therefore at least one of q_0 and q_1 has $2\mathcal{E}+1$ or more elements. Without loss of generality, assume that $|q_1| > 2\mathcal{E}$. We have $\sum_{i \in q_1} d_i = 0$ and thus $\tilde{a}_{q_1} \leq \mathcal{E}$. On the other hand $\sum_{i \in q_1} c_i = |q_1| > 2\mathcal{E}$. Hence we get $|\sum_{i \in q_1} c_i - \tilde{a}_{q_1}| > \mathcal{E}$ which is contradictory to c surviving the weeding phase. □

3.2 Polynomially Bounded Adversary

Next, we consider a more realistic scenario where the adversary is polynomially bounded. Dinur and Nissim show that it is necessary to use a perturbation level of $\Omega(\sqrt{n})$ to achieve even weak privacy. Thus they show that any database algorithm within $o(\sqrt{n})$ perturbation is non-private.

Theorem 7 *Let $\mathcal{D} = (d, \mathcal{A})$ be a database where $\mathcal{A}$ is within $o(\sqrt{n})$ perturbation. Then $\mathcal{D}$ is* $\mathbf{poly}(n)$*-non-private.*

To proof Theorem 7 we need to following lemma.

Lemma 8 (Disqualifying Lemma) *Let $x, d \in [0,1]^n$ and $\mathcal{E} = o(\sqrt{n})$. If $\Pr_i \left[|x_i - d_i| \geq \frac{1}{3}\right] > \varepsilon$ then there exists a constant $\delta > 0$ such that*

$$\Pr_{q \subseteq \{1,\ldots,n\}} [|\sum_{i \in q}(x_i - d_i)| > 2\mathcal{E} + 1] > \delta.$$

For proof, see [4, Appendix A]. Now, let us proof Theorem 7.

Proof. Let $\mathcal{A}$ be within $\mathcal{E} = o(\sqrt{n})$ perturbation. Let $\mathcal{M}$ be the following algorithm.

(i) (Query phase)
Let $t = n \log^2(n)$. For $1 \leq j \leq t$, choose uniformly at random $q_j \subseteq \{1, \ldots, n\}$, and set $\tilde{a}_{q_j} = \mathcal{A}(q_j)$.

(ii) (Weeding phase)
Solve the following linear program (LP) with n unknowns $c_1, \ldots, c_n$.

$$\tilde{a}_{q_j} - \mathcal{E} \leq \sum_{i \in q_j} c_i \leq \tilde{a}_{q_j} + \mathcal{E} \quad \text{for} \quad 1 \leq j \leq t$$
$$0 \leq c_i \leq 1 \quad \text{for} \quad 1 \leq i \leq n$$

(iii) (Rounding phase)
Let $c'_i = 1$ if $c_i > 1/2$ and $c'_i = 0$ otherwise. Output c'.

Clearly, the linear program in the algorithm always has a solution ($c = d$ is a feasible solution), and thus the algorithm always has an output c'. Also, since LP problem can be solved in polynomial time, the algorithm runs in polynomial time.

Thus we need to show that $\mathrm{dist}(c', d) < \varepsilon n$. We use a probabilistic method to show that a random choice of $q_1, \ldots, q_t$ weeds out all possible candidate databases c that are far from the original one.

We fix a precision parameter $k = n$ and define $K = \{0, \frac{1}{k}, \frac{2}{k}, \ldots, \frac{k-1}{k}, 1\}$. For any $x \in [0,1]^n$ we denote by $\overline{x} \in K^n$ the vector obtained from x by rounding each coordinate to the nearest integer multiple of $\frac{1}{k}$.

We apply the triangle inequality with the LP in the weeding phase. We get for $1 \leq j \leq t$,

$$\begin{aligned}
&|\textstyle\sum_{i \in q_j}(\overline{c}_i - d_i)| \\
&= |\textstyle\sum_{i \in q_j}(\overline{c}_i + c_i + c_i - \tilde{a}_{q_j} + \tilde{a}_{q_j} - d_i)| \\
&\leq |\textstyle\sum_{i \in q_j}(\overline{c}_i + c_i)| + |\textstyle\sum_{i \in q_j}(c_i - \tilde{a}_{q_j})| \\
&\quad + |\textstyle\sum_{i \in q_j}(\tilde{a}_{q_j} - d_i)| \\
&\leq \tfrac{|q_j|}{k} + \mathcal{E} + \mathcal{E} \\
&\leq 1 + 2\mathcal{E}.
\end{aligned}$$

We say for any $x \in [0,1]^n$ that a query $q \subseteq \{1, \ldots, n\}$ *disqualifies* $\overline{x}$ if $|\sum_{i \in q}(\overline{x}_i - d_i)| > 2\mathcal{E} + 1$. If q is one of the queries of the algorithm, then x is an invalid solution to the linear program.

We show that if x is far from d on many coordinates, then $\overline{x}$ will be disqualified with a high probability by at least one of the queries $q_1, \ldots, q_t$. We use Lemma 8 and define the set of discrete x's that are far from d as the set

$$X = \{x \in K^n \mid \Pr_i[|x_i - d_i| \geq \frac{1}{3}] > \varepsilon n\}.$$

Let us consider $x \in X$. By Lemma 8 there exists a constant $\delta > 0$ such that

$$\Pr_{q \subseteq \{1,\ldots,n\}}[q \text{ disqualifies } x] \geq \delta.$$

If we choose t independent random queries $q_1, \ldots, q_t \subseteq \{1, \ldots, n\}$, at least one of them disqualifies x with probability $1 - (1-\delta)^t$. Now taking union over X (note, $|X| \leq |K|^n = (k+1)^n = (n+1)^n$), we get

$$\begin{aligned} \Pr_{q_1,\ldots,q_t \subseteq_R \{1,\ldots,n\}}&[\forall x \in X\ \exists i \text{ s.t. } q_i \text{ disqualifies } x] \\ &\geq 1 - (n+1)^n(1-\delta)^t \\ &> 1 - \text{neg}(n). \end{aligned}$$

The last inequality holds if t is large enough. For example $t = n\log^2(n)$ suffices as,

$$\begin{aligned} &(n+1)^n(1-\delta)^t = (n+1)^n(1-\delta)^{n(\log^(n)} \\ &= ((n+1)^{1/\log^2(n)}(1-\delta))^{n\log^2(n)} \underset{n\to\infty}{\longrightarrow} 0 \end{aligned}$$

To complete the proof, note that $\overline{c}$ is *not* disqualified by any of the random subsets $q_1, \ldots, q_t \subseteq \{1, \ldots, n\}$, since $\overline{c}$ is obtained from the solution c of the LP system by rounding it. Thus, if $q_1, \ldots, q_t$ disqualify all $x \in X$, it must be that $\overline{c} \notin X$ and therefore $\text{dist}(c', d) \leq \varepsilon n$. □

The proof for Theorem 7 relies on the fact that a random subset of linear size deviates from the expectation by roughly $\sqrt{n}$. Also, the reconstruction algorithm presented is oblivious of the perturbation method used and the distribution of the database.

3.3 Tightness of the Impossibility Results

Theorem 7 states that for any database distribution it is necessary to have a perturbation of magnitude $\Omega(\sqrt{n})$ to have even a weak notion of privacy. Dinur and Nissim show that this bound is indeed tight. They present a database algorithm that is within $\tilde{O}(\sqrt{n})$ perturbation and private against polynomial adversaries. The algorithm is such that, if the database is queried by a polynomial-time machine, then with a very high probability it does not reveal any information about the data. In the algorithm it is assumed that the database is uniformly distributed over all strings of n bits.

Let $d \in \{0,1\}^n$ at random and set the perturbation magnitude $\mathcal{E} = \sqrt{n}(\log n)^{1+\varepsilon} = \tilde{O}(\sqrt{n})$. Let us consider database $\mathcal{D} = (d, \mathcal{A})$ with algorithm $\mathcal{A}$ defined as follows,

(i) For an input query $q \subseteq \{1, \ldots, n\}$, compute $a_q = \sum_{i \in q} d_i$.

(ii) If $|a_q - \frac{|q|}{2}| < \mathcal{E}$, return $\frac{|q|}{2}$.

(iii) Otherwise, return a_q.

Thus $\mathcal{A}$ uses output perturbation method. Clearly, $\mathcal{A}$ is within $\mathcal{E}$ perturbation. Also, for any probabilistic polynomial-time machine $\mathcal{M}$, the probability (taken over d and coin tosses of $\mathcal{M}$) that $\mathcal{A}$ acts according to rule (iii) is negligible. Although the above algorithm guarantees perturbation level of $\tilde{O}(\sqrt{n})$, it makes the database effectively useless. Users are extremely unlikely to obtain any nontrivial information by querying the database. Hence, must a private database be useless?

This is not necessarily the case as Dinur and Nissim demonstrate by a database algorithm that has some privacy combined with some usability. We relax the requirements in Definition 3 and require that $\mathcal{A}(q)$ is within $\mathcal{E}$ perturbation for *most* q, i.e.

$$\begin{aligned} \Pr_{q \in \{1,\ldots,n\}}&[\mathcal{A}(q) \text{ is within } \mathcal{E} \text{ perturbation}] \\ &= 1 - \text{neg}(n). \end{aligned}$$

Theorem 7 holds for this relaxed definition, too.

Now, let $\mathcal{DB}$ be the uniform distribution over $\{0,1\}^n$ and select $d \in \mathcal{DB}$ at random. The database algorithm $\mathcal{A}$ will use an internal state θ (recall Definition 2) that is initialized upon the first call. The internal state consists of n bits $d' = (d'_1, \ldots, d'_n)$ where $d'_i = d_i$ with probability $1/2 + \delta$ and $d'_i = 1 - d_i$ otherwise. On an input query $q \subseteq \{1, \ldots, n\}$ algorithm $\mathcal{A}$ answers $\tilde{a}_q = \sum_{i \in q} d'_i$. Now, $\mathcal{A}$ is within $\tilde{O}(\sqrt{n})$ perturbation (according to the relaxed definition) and the database has some usability. E.g. it is possible to compute $S \subseteq \{1, \ldots, n\}$ such that significantly more than half of the entries specified by S are set in 1 in the original database. Notice that the algorithm is essentially the same as in *randomized response technique* [8].

The above algorithm essentially creates a private version of the database and the queries are answered according to it. The user may retrieve the whole database d' by querying $q_i = \{i\}$ for $1 \leq i \leq n$, and thereafter answer all other queries by itself. This indicates that it is possible to achieve *some* privacy in a *CD model*, where users get a private version of the database, which they manipulate (their queries are not restricted to be statistical).

4 Feasibility Results

In Section 3 we discussed perturbation level needed when the adversary has an exponential or a polynomial computational power. In this section we briefly investigate whether privacy can be achieved using a smaller perturbation level than $\sqrt{n}$ if adversary's complexity is further restricted. This can also be seen as considering an adversary having a fixed power acting on bigger and bigger databases. Dinur and Nissim present a database access algorithm that preserves privacy with respect to an adversary having running time limited to $\mathcal{T}(n)$ for an arbitrary $\mathcal{T}$. This algorithm uses a perturbation magnitude of about $\sqrt{\mathcal{T}(n)}$.

First, we define what is meant by privacy (note that so far we have only considered non-privacy). In this privacy definition we have to make an assumption regarding the adversary's a-priori knowledge about the database[3]. In the definition below, the prior knowledge is modeled as having the database drawn from an arbitrary distribution $\mathcal{DB}$.

The definition of privacy we will be considering is strong. It requires than even if the adversary learns contents of the entire database except the ith bit, it cannot still predict the ith bit with a good probability. The scenario is modeled by a two-phased adversary: (1) The adversary adaptively queries the database. At the end, the adversary commits a challenge – an index i of the bit it intents to guess. (2) All the database entries except the ith bit are revealed to the adversary. The adversary succeeds if it outputs d_i correctly. In the formal definition below, the two phases are modeled by two Turing machines $\mathcal{M}_1$ and $\mathcal{M}_2$ of which $\mathcal{M}_1$ has an access to the database algorithm.

Definition 9 (Privacy) Let $\mathcal{DB}$ be a distribution over $\{0,1\}^n$ and d is drawn according to $\mathcal{DB}$. A database $\mathcal{D} = (d, \mathcal{A})$ is $(\mathcal{T}(n), \delta)$-private, if for every pair of probabilistic Turing machines $\mathcal{M}_1$ and $\mathcal{M}_2$ having time-complexity $\mathcal{T}(n)$, it holds that

$$\Pr\left[\begin{array}{l}\mathcal{M}_1(1^n) \text{ outputs } (i, view); \\ \mathcal{M}_2(view, d^{-i}) \text{ outputs } d_i\end{array}\right] < \frac{1}{2} + \delta,$$

where $d^{-i} = (d_1, \ldots, d_{i-1}, d_{i+1}, \ldots, d_n)$. The probability is taken over the choice of d from $\mathcal{DB}$ and the coin tosses of all machines involved.

If it is assumed that the adversary has no prior information about the database (modeled by drawing the database from the uniform distribution over n-bit strings), we get the following theorem.

Theorem 10 *Let $\mathcal{T}(n) > \log^k(n)$ and $\delta > 0$. Let $\mathcal{DB}$ be uniform distribution over $\{0,1\}^n$, and select $d \in \mathcal{DB}$ at random. There exists a $\tilde{O}(\sqrt{\mathcal{T}(n)})$-perturbation algorithm $\mathcal{A}$ such that $\mathcal{D} = (d, \mathcal{A})$ is $(\mathcal{T}(n), \delta)$-private.*

For a lengthy and detailed proof, see [4].

5 Conclusions

We have explored the conditions under which a privacy preserving database access mechanism can exist for a statistical database based on discussion by Dinur and Nissim [4]. It is shown that unless the perturbation is as large as $\sqrt{n}$, almost all the database entries can be recovered by a polynomially bounded adversary. Such database is, however, practically almost useless. A bounded adversary model is also briefly discussed. In such model some level of privacy may be achieved.

References

[1] J. Achugbue and F. Chin. *The Effectiveness of Output Modification by Rounding for Protection of Statistical Databases*, INFOR 17(3), pp. 209–218, 1979.

[2] L. Beck, *A Security Mechanism for Statistical Databases*, ACM TODS, 5(3), pp. 316–338. 1980.

[3] Consider a situation in which it is known that d is either 1^n or 0^n in database $\mathcal{D} = (d, \mathcal{A})$. If the perturbation level was less than $n/2$, a single query would reveal the entire database.

[3] F. Chin and G. Ozsoyoglu, *Auditing and Inference Control in Statistical Databases*, IEEE Transactions in Software Engineering, SE-8(6), pp. 113–139. 1982.

[4] I. Dinur and K. Nissim. *Revealing Information while Preserving Privacy.* In Proc. of 22nd ACM SIGMOD-SIGACT-SIGART symposium, pp. 202–210. ACM Press. USA, 2003.

[5] I. Fellegi, *On the Question of Statistical Confidentiality*, Journal of the American Statistical Association, pp. 7–18. 1972.

[6] S. Reiss,*Practical Data Swapping: The First Steps*, ACM TODS, 9(1), pp. 20–37. 1984.

[7] J. Traub, Y. Jemini and H. Wozniakowski, *The Statistical Security of a Statistical Database*, ACM TODS, 9(4), pp. 672–679. 1984.

[8] S. Warner, *Randomized Response: a Survey Technique for Eliminating Evasive Answer Bias.* The American Statistical Association, 60(309), pp. 63-69. 1965.