Query Auditing for Protecting Max/Min Values of Sensitive Attributes in Statistical Databases

Ta Vinh Thong and Levente Buttyán

Laboratory of Cryptography and System Security (CrySyS)
Budapest University of Technology and Economics, Hungary
{thong, buttyan}@crysys.hu
http://www.crysys.hu

Abstract. In this paper, we define a novel setting for query auditing, where instead of detecting or preventing the disclosure of individual sensitive values, we want to detect or prevent the disclosure of aggregate values in the database. More specifically, we study the problem of detecting or preventing the disclosure of the maximum (minimum) value in the database, when the querier is allowed to issue average queries to the database. We propose efficient off-line and on-line query auditors for this problem in the full disclosure model, and an efficient simulatable on-line query auditor in the partial disclosure model.

Keywords: Privacy, query auditing, online auditor, offline auditor, simulatable auditor, probabilistic auditor, statistical database.

1 Introduction

Query Auditing is a problem that has been studied intensively in the context of disclosure control in statistical databases [1]. The goal of a query auditing algorithm is to detect (off-line query auditing) or to prevent (on-line query auditing) the disclosure of sensitive private information from a database that accepts and responds to aggregate queries (e.g., average value of an attribute over a subset of records defined by the query). To the best of our knowledge, in all existing works on query auditing, the private information whose disclosure we want to detect or prevent consists of the sensitive fields of individual records in the database (e.g., the salery of a given employee). The reason may be that statistical databases are mainly used for computing statistics over certain attributes of human users (e.g., the average salary of women employees), and in such applications, each database record corresponds to an individual person. In this paper, we define a novel setting for query auditing, where we want to detect or prevent the disclosure of *aggregate* values in the database (e.g., the maximum salary that occurs in the database).

The motivation behind our work comes from a project[1], called CHIRON, where we use body mounted wireless sensor networks to collect medical data

[1] www.chiron-project.eu

(e.g., ECG signals, blood pressure measurements, temperature samples, etc.) from a patient, and we use a personal device (e.g., a smart phone) to collect those data and provide controlled access to them for external parties (e.g., hospital personnel, personal coach services, and health insurance companies). In this context, the records stored in the database on the personal device all belong to the same patient, and individual values (i.e., sensor readings) may not be sensitive, whereas aggregates computed over those values (e.g., the maximum of the blood pressure in a given time interval) should be protected from unintended disclosure. The reason is that some of those aggregates (extreme values) can be used to infer the health status of the patient, and some of the accessing parties (e.g., health insurance companies) should be prevented to learn that information.

More specifically, in this paper, we study the problem of detecting or preventing the disclosure of the maximum value in the database, when the querier is allowed to issue average queries to the database. We propose efficient off-line and on-line query auditors for this problem in the full disclosure model, and an efficient simulatable on-line query auditor in the partial disclosure model. As for the organization of the paper, we start with an overview of the query auditing problem domain, introduce some terminology, review the state-of-the-art, and then present our model and algorithms together with their detailed analysis. Finally, we note that due to space limitations, we only sketch the proofs, and we focus on the main results of our work. More illustrating examples, explanations and more detailed proofs can be found in our technical report [13].

2 Query Auditing Problems

Query auditing problems can be classified according to the characteristics of the auditor and the attacker model that they resist [1]. In case of offline auditing, the auditor is given a set of t queries $q_1, \ldots, q_t$ and the corresponding answers $a_1, \ldots, a_t$, and its task is to determine offline if a breach of privacy has occurred. In contrast, an online auditor prevents a privacy breach by denying to respond to a new query if doing so would lead to the disclosure of private information. More specifically, given a sequence of $t-1$ queries $q_1, \ldots, q_{t-1}$ that have already been posed and their corresponding answers $a_1, \ldots, a_{t-1}$, when a new query q_t is received, the online auditor denies the answer if it detects that private information would be disclosed by $q_1, q_2, \ldots, q_t$, and $a_1, a_2, \ldots, a_t$, otherwise it gives the true answer a_t.

Let n denote the total number of records in the database. $X = \{x_1, x_2, \ldots, x_n\}$ is the set of the private attribute values in the records. $q = (Q, f)$ is an aggregate query, where Q specifies a subset of records, called the *query set* of q. f is an aggregation function such as MAX, MIN, SUM, AVG, MEDIAN. Finally, let $a = f(Q)$ be the result of applying f to Q, called the answer. In the following, we give an overview of the disclosure models, as well as the notion and concept of simulatable auditor.

In the full disclosure model, the privacy of some data x breaches when x has been uniquely determined.

Definition 1 *Given a set of private values $X = \{x_1, x_2, \ldots, x_n\}$, a set of queries $\mathcal{Q} = \{q_1, q_2, \ldots, q_t\}$, and corresponding answers $\mathcal{A} = \{a_1, a_2, \ldots, a_t\}$, an element x_i is fully disclosed by $(\mathcal{Q}, \mathcal{A})$ if it can be uniquely determined, that is, x_i is the same in all possible data sets X consistent with the answers A to the queries $\mathcal{Q}$.*

One may think that the full disclosure model defines a weak notion of privacy since a private value can be deduced to lie in a tiny interval or even a large interval where the distribution is heavily skewed towards a particular value, yet it is not considered a privacy breach. To deal with this problem, a definition of privacy has been proposed that gives bounds on the ratio of the posteriori probability that an individual value x_i lies in an interval I given the queries and answers to the apriori probability that $x_i \in I$. This is also known as probabilistic (partial) disclosure model [10], which we will introduce next.

Consider an arbitrary data set $X = \{x_1, \ldots, x_n\}$, in which each x_i is chosen independently according to the same distribution $\mathcal{H}$ on $(-\infty, \infty)$. Let $\mathcal{D} = \mathcal{H}^n$ denote the joint distribution. Next we introduce the notion of λ-safe and AllSafe. We say that a sequence of queries and answers is λ-safe for an entry x_i and an interval I if the attacker's confidence that $x_i \in I$ does not change significantly upon seeing the queries and answers.

Definition 2 *The sequence of queries and answers, $q_1, \ldots, q_t$, $a_1, \ldots, a_t$ (denoted by $\wedge_1^t(q_j, a_j)$) is said to be λ-safe with respect to an x_i and an interval $I \subseteq (-\infty, \infty)$ if the next Boolean predicate evaluates to 1:*

$$Safe_{\lambda,i,I}(\wedge_1^t(q_j, a_j)) = \begin{cases} 1 \text{ if } 1/(1+\lambda) \leq \frac{Pr_{\mathcal{D}}(x_i \in I | \wedge_{j=1}^t (f_j(Q_j) = a_j))}{Pr_{\mathcal{D}}(x_i \in I)} \leq (1+\lambda) \\ 0 \text{ otherwise} \end{cases}$$

Definition 3 *Predicate AllSafe evaluates to 1 if and only if $q_1, \ldots, q_t$, $a_1, \ldots, a_t$ is λ-safe for all x_i's and all ω-significant intervals.*

$$AllSafe_{\lambda,\omega}(\wedge_1^t(q_j, a_j)) = \begin{cases} 1 \text{ if } Safe_{\lambda,i,J}(\wedge_1^t(q_j, a_j)) = 1, \forall\ J, \forall\ i \in [n] \\ 0 \text{ otherwise} \end{cases}$$

We say that an interval J is ω-significant if for every $i \in [n]$, $Pr_{\mathcal{D}}(x_i \in J)$ is at least $1/\omega$, and we will only consider the change of probabilities with respect to these intervals. The definition of a randomized auditor for the case of partial disclosure model is as follows.

Definition 4 *A randomized auditor is a randomized function of queries $q_1, \ldots, q_t$, the data set X, and the probability distribution $\mathcal{D}$ that either gives an exact answer to the query q_t or denies the answer.*

Below we introduce the notion of the (λ, ω, T)-privacy game and the (λ, δ, ω, T)-private auditor. The (λ, ω, T)-privacy game is a game between an attacker and an auditor, where each round t (for up to T rounds) is defined as follows:

1. In each round t ($t \leq T$), the attacker poses a query $q_t = (Q_t, f_t)$.

2. The auditor decides whether to respond to q_t or not. The auditor replies with $a_t = f_t(Q_t)$ if q_t is allowed, and denies the response otherwise.
3. The attacker wins if $AllSafe_{\lambda,\omega}(\wedge_1^t(q_j, a_j)) = 0$.

Definition 5 *An auditor is (λ, δ, ω, T)-private if for any attacker A,*

$$Pr\{A \text{ wins the } (\lambda, \omega, T)\text{-privacy game}\} \leq \delta.$$

The probability is taken over the randomness in the distribution $\mathcal{D}$ and the coin tosses of the auditor and the attacker.

Unfortunately, in general an offline auditor cannot directly solve the online auditing problem because even denials can leak information if in choosing to deny, the auditor uses information that is unavailable to the attacker (i.e., the answer to the current query). We refer the reader to the extended report of this paper [13] for an illustrating example. In order to overcome this problem, the concept of *simulatable auditor* has been proposed. Taking into account the crucial observation above, the main idea of simulatable auditing is that the attacker is able to simulate or mimic the auditor's decisions to answer or deny a query. As the attacker can equivalently determine for himself when his queries will be denied, she obtains no additional information from denials. For these reasons denials *provably* leak no information. The definition of simulatable auditor in the full disclosure model is given in Definition 6.

Definition 6 *An online auditor B is simulatable, if there exists another auditor B' that is a function of only $\mathcal{Q} \cup \{q_t\} = \{q_1, q_2, \ldots, q_t\}$ and $\mathcal{A} = \{a_1, a_2, \ldots, a_{t-1}\}$, and whose answer to q_t is always equal to that of B.*

When constructing a simulatable auditor for the probabilistic disclosure model, the auditor should ignore the real answer a_t and instead make guesses about the value of a_t, say a'_t, computed on randomly sampled data sets according to the distribution $\mathcal{D}$ conditioned on the first $t-1$ queries and answers. The definition of simulatable auditor in the probabilistic case is given in Definition 7.

Definition 7 *Let $\mathcal{Q}_t = \{q_1, \ldots, q_t\}$, $\mathcal{A}_{t-1} = \{a_1, \ldots, a_{t-1}\}$. A randomized auditor B is simulatable if there exists another auditor B' that is a probabilistic function of $\langle \mathcal{Q}_t, \mathcal{A}_{t-1}, \mathcal{D} \rangle$, and the outcome of B on $\langle \mathcal{Q}_t, \mathcal{A}_{t-1} \cup \{a_t\}, \mathcal{D} \rangle$ and X is computationally indistinguishable from that of B' on $\langle \mathcal{Q}_t, \mathcal{A}_{t-1}, \mathcal{D} \rangle$.*

A general approach for constructing simulatable auditors works as follows: The input of the auditor is the past $t-1$ queries along with their corresponding answers, and the current query q_t. As mentioned before, the auditor should not consider the true answer a_t when making a decision. Instead, to make it simulatable for the attacker, the auditor repeatedly selects a data set X' consistent with the past $t-1$ queries and answers, and computes the answer a'_t based on q_t and X'. Then, the auditor checks if answering with a'_t leads to a privacy breach. If a privacy breach occurs for any consistent data set (full disclosure model) or for a

large fraction of consistent data sets (partial disclosure model), the response to q_t is denied. Otherwise, it is allowed and the true answer a_t is returned.

While ensuring no information leakage, a simulatable auditor has the main drawback that it can be too strict, and deny too many queries resulting in bad utility.

3 Related Works

We note that *the related works discussed below are concerned with protecting the privacy of individual values, and not aggregated values* that we are addressing in this paper. In case of the full disclosure model, efficient simulatable online auditors have been proposed for SUM [3], MAX, MIN and the combination of MAX and MIN queries [6], [10]. In all these cases the values of private attributes are assumed to be unbounded real numbers. For effectiveness, the MAX and MIN auditors assume that there is no duplication among $x_1,\ldots,\ x_n$ values.

In the full disclosure model, effective offline auditors have been proposed for SUM, MAX, MIN, and the combination of MAX and MIN queries over unbounded real values and under the same conditions as in the online case above [3], [4]. Additionally, SUM auditors have also been proposed for boolean values [7], but the authors proved that the online sum auditing problem over boolean values is coNP-hard. It has been shown that the problem of offline auditing the combination of MAX and SUM (MIN and SUM, MIN and MAX and SUM) queries in the full disclosure model is NP-hard [3].

In [14] an offline SUM auditor has been proposed in which sensitive information about individuals is said to be compromised if an accurate enough interval is obtained into which the value of the sensitive information must fall. In [2] the authors consider the problem of auditing queries where the result is a distance metric between the query input and some secret data.

Similarly, simulatable SUM, MAX, MIN and the combination of MAX and MIN auditors have been proposed for the probabilistic disclosure model [3], [4]. In all cases the private attributes are assumed to take their values randomly accorinding to uniform and log-concave distributions, from an unbounded domain. In [8] the notion of simulatable binding has been proposed that provides better utility than simulatable auditor, but requires more computations.

Targeting the problem of mutable databases, which allow for deleting, modifying, and inserting records, auditors have been proposed in the full disclosure model for MIN, MAX, MIN and MAX, and SUM queries [11].

Next we review a bit more in details the offline SUM auditor proposed in [3] because it is referred to during discussing our method. The main concept of the method is that each query is expressed as a row in a matrix with a 1 wherever there is an index in the query and a 0 otherwise. If the matrix can be reduced to a form where there is a row with one 1 and the rest 0s then some value has been compromised. To make it simulatable, the transformations of the original matrix are performed via elementary row and column operations by ignoring the answers to the queries.

4 Our contributions

We address a new auditing problem by considering an *aggregation* value of a data set to be sensitive and concentrating on protecting the privacy of aggregation values. In contrast to the previous works, we assume that the domain of sensitive values is bounded, which leads to some new problems. We note that in each case below, without loss of generality and for simplicity, we transform each equation $\frac{\sum_1^k x_i}{k} = a$ induced by each AVG query and its answer to the form $\sum_1^k x_i = ak$.

In the rest of the paper, we denote the auditor that receives average queries and protects the privacy of the max (min) value as $\text{Auditor}_{avg}^{max}$ ($\text{Auditor}_{avg}^{min}$), and we denote $\max\{x_1, \ldots, x_n\}$ by *MAX*. We note that in the paper we mainly focus on the privacy of the maximum values, however, auditors can be constructed for minimum values in an equivalent way.

4.1 Offline and Online $\text{Auditor}_{avg}^{max}$ in the full disclosure model

I. The proposed offline auditor: Let us consider t queries $q_1, \ldots, q_t$ over the stored data set $X = \{x_1, \ldots, x_n\}$ and their corresponding answers $a_1, \ldots, a_t$. Each query q_i is of form (Q_i, AVG), where $i \subseteq [n]$, and the value of each x_i is assumed to be a real number that lies in a finite interval $[\alpha, \beta]$, where $\beta > \alpha$. The task of the offline auditor is to detect if the value of MAX is fully disclosed.

Let us refer to the algorithm proposed in [3] as $\mathcal{A}_{sum}$. Using $\mathcal{A}_{sum}$ is not sufficient in our case because it does not consider the bounds of each x_i, as well as the values of the answers. For the purpose of illustration, let us take the following example: let $X = \{x_1, x_2, x_3\}$ and $\forall x_i \in [20, 90]$, let $q_1 = (\{x_1, x_2\}, \text{AVG})$, $q_2 = (\{x_1, x_2, x_3\}, \text{AVG})$ and the corresponding answers $a_1 = 45$, $a_2 = 60$. Finally, let the stored values be $x_1 = 40$, $x_2 = 50$, $x_3 = 90$. According to $\mathcal{A}_{sum}$ the value of *MAX* is not fully disclosed, because the answers and the bounds of x_i's are not considered. We only know that x_3 can be uniquely determined, but nothing about its value. However, in fact *MAX* is fully disclosed because by involving the answers we additionally know that the value of x_3 is 90, which at the same time is the value of *MAX* since 90 is the upperbound of any x_i.

Hence, we have to consider a method that also takes into account the bounds of x_i's and the answers. For this purpose, we propose the application of the well-known linear optimization problem as follows: The t queries are represented by a matrix $\bar{A}$ of t rows and n columns. Each row $r_i = (a_{i,1}, \ldots, a_{i,n})$ of $\bar{A}$ represents the query set Q_i of the query q_i. The value of $a_{i,j}$, $1 \leq i, j \leq n$, is 1 wherever x_j is in the query set Q_i, and is a 0 otherwise. The correponding answers are represented as a column vector $\bar{b} = (b_1, \ldots, b_t)^T$ in which b_i is the answer for q_i.

Since each attribute x_i takes a real value from a bounded interval $[\alpha, \beta]$ we obtain the following special linear equation system, also known as *feasible set*, which includes equations and inequalities:

$$\mathcal{L} = \begin{cases} \bar{A}\bar{x} = \bar{b}, \text{where } \bar{x} \text{ is the vector } (x_1, \ldots, x_n)^T. \\ \alpha \leq x_i \leq \beta, \forall x_i : x_i \in \{x_1, \ldots, x_n\} \end{cases}$$

Then, by appending each objective function $maximize(x_i)$ to $\mathcal{L}$, we get n linear programming problems P_i, for $i \in \{1, \ldots, n\}$. Let $x_i^{max} = maximize(x_i)$, then the maximum value of $x_1, \ldots, x_n$ is the maximum of the n maximized values, $x^{opt} = max\{x_1^{max}, \ldots, x_n^{max}\}$. Let us denote the whole linear programming problem above for determining the maximum value x^{opt} as $\mathcal{P}$. Note that x^{opt} returned by $\mathcal{P}$ is the exact maximum value if (i) $\mathcal{L}$ has a unique solution or (ii) $\mathcal{L}$ does not have a unique solution but there exist some x_i that can be derived to be equal to x^{opt}. To see the meaning of point (ii), let us consider the specific case of $\mathcal{L}$ in which $n = 4$, $\alpha = 0$, $\beta = 5$, and $\bar{A} = \begin{pmatrix} 1\,1\,0\,0 \\ 0\,0\,1\,1 \end{pmatrix}$, $\bar{b} = \begin{pmatrix} 6 \\ 10 \end{pmatrix}$. In this example, $\mathcal{L}$ does not have a unique solution but the exact maximum still can be derived such that $x_3 = x_4 = 5$.

Otherwise, x^{opt} is the best estimation of the exact maximum. We note that in our case $\mathcal{L}$ always has a solution, because one possible solution is actually the values stored in the database.

Based on this linear programming problem, our offline auditor will follow the next steps. Given t queries $q_1, \ldots, q_t$ over $X = \{x_1, \ldots, x_n\}$ and their corresponding answers $a_1, \ldots, a_t$, the value of MAX is fully disclosed in any of the following two cases:

- (F1) In case $\mathcal{L}$ has a unique solution, the value of MAX is equal to x^{opt}.
- (F2) In case $\mathcal{L}$ does not have a unique solution: If by following the solving procedure of $\mathcal{L}$ (e.g., basic row and column operations), there exist some x_i that can be uniquely determined such that $x_i = x^{opt}$, then the value of MAX is x_i. This is because x^{opt} is always at least as large as the value of MAX.

Otherwise, the attacker cannot uniquely deduce the value of MAX. The complexity of the auditor is based on the complexity of $\mathcal{P}$. It is well-known that there are polynomial time methods to solve $\mathcal{P}$, for instance, the path-following algorithm [12], which is one of the most effective method with complexity $O(n^3L)$. Here n is the number of variables while L is the size of the input in bits, and the number of rows is assumed to be $O(n)$. Therefore, our offline auditing method has a polynomial time complexity in the worst case.

II. The proposed online auditor: Let us consider the first $t-1$ queries and answers over the data set similarly defined as in the offline case above. When a new q_t is posed, the task of the online auditor is to make a decision in *real-time* whether to answer or deny the query. More specifically, our goal is to propose an auditor that detects if answering with true a_t causes full disclosure of MAX.

First of all, we discuss the construction of a simulatable auditor for this problem, and we will show the limitation of simulatable auditors in this case. Thereafter, we introduce another method that gets around this limitation. Based on the concept shown in Section 2 and the linear programming problem, the simulatable auditor for this problem is shown in Algorithm 1.

Algorithm 1: Simulatable online auditor $\text{Auditor}_{avg}^{max}$

Inputs: $q_1, \ldots, q_t$, $a_1, \ldots, a_{t-1}$, α, β;
for each consistent data set X' **do** compute the AVG a'_t based on Q_t and X';
  Let $\mathcal{L}_t$ be the feasible set formed by the t queries/answers;
  if $\mathcal{L}_t$ yields an exact maximum **then** output DENY; **endif**
endfor
output a_t;

Algorithm 2: Online auditor $\text{Auditor}_{avg}^{max}$

Inputs: $q_1, \ldots, q_t$, $a_1, \ldots, a_t$, d_{tr}, α, β;
Let $\mathcal{L}_t^*$ be the feasible set formed by the t queries/answers
Let x_t^{opt} be the returned maximum by solving $\mathcal{P}$ with $\mathcal{L}_t^*$
if $|x_t^{opt} - MAX| > d_{tr}$ AND $(MAX - max_t) > d_{tr}$ **then** output a_t; **endif**
else if $|x_t^{opt} - MAX| \leq d_{tr}$ OR $(MAX - max_t) \leq d_{tr}$ **then** output DENY; **endif**

Note that in Algorithm 1, based on the concept of simulatable auditor in Section 2, by ignoring the true answer a_t we examine every data set X', consistent with the past queries and answers, and check if it causes the full disclosure of MAX. This means that the answer a'_t computed based on X' and Q_t, is included in the analysis. The auditor is simulatable because it never looks at the true answer when making a decision. The main drawback, however, of using simulatable auditor in our problem is the bad utility. In order to see this, consider any AVG query q that specifies a subset $\{x_{i1}, x_{i2}, ..., x_{ik}\}$ of X as the query set. There always exist a data set X' for which this query is not safe to respond, namely, the data set where $x_{i1} = x_{i2} = \ldots = x_{ik} = \beta$, as in this case, the true response would be β, and the querier can figure out that all values in the query set must be equal to β. This essentially means that all queries should be denied by a simulatable auditor.

To achieve better utility, hence, we propose a method (Algorithm 2) that is not simulatable but we show that it still ensures, in the full disclosure model, the privacy of the maximum value. Let us denote $|x^{opt} - MAX|$ as the absolute distance between x^{opt} and MAX. Let max_t be the maximum of the first t answers. Let $\mathcal{L}^*$ be the feasible set that is similar to $\mathcal{L}$ but the constraint $\alpha \leq x_i \leq \beta$ is involved only for such x_i's that occurs in the first t queries, and not for all the n variables. Namely, in $\mathcal{L}^*$ the second line of $\mathcal{L}$ is changed to $\alpha \leq x_i \leq \beta$, for all i such that x_i *occurs in in the first* t *queries*. Note that we use $\mathcal{L}^*$ instead of $\mathcal{L}$ in our online auditor because by doing this the auditor leaks less information to the attacker either when answering or denying.

The online auditor works as follows: Recall that $\mathcal{L}^*$ is defined over t queries and answers. Whenever a new query q_t is posed, the auditor computes the true answer a_t, and then it solves the problem $\mathcal{P}$ with $\mathcal{L}^*$, obtaining x^{opt}. If for a given treshold value d_{tr}, $|x^{opt} - MAX| > d_{tr}$ and $(MAX - max_t) > d_{tr}$ then the true answer a_t is provided. Otherwise, if $|x^{opt} - MAX| \leq d_{tr}$ or $(MAX - max_t) \leq d_{tr}$ the auditor denies.

Lemma 1 *The online auditor implemented by the Algorithm 2 provides the privacy of MAX in the full disclosure model.*

Proof. (Sketch) Let $f_{att}(d_{tr},q_1,\ldots,q_t,a_1,\ldots,a_{t-1},\alpha,\beta)$ represent the attacker's based on the input parameters, and returning as output a deny or an answer. We prove that our online auditor does not leak information about MAX, in the full disclosure model by showing that the number of the data sets and the parameter sets for which f_{att} returns deny or answer is always larger than 1. In other words, in every possible scenario, for the attacker the number of possible maximum values will always be greater than 1, hence, the value of MAX cannot be uniquely determined. We apply mathematical induction in each case. □

The utility of the auditor can be measured based on the number of denies. This is controlled by the treshold value d_{tr}. Broadly speaking, if d_{tr} is large then the expected number of denies is greater, while when d_{tr} is small the degree of privacy provided decreases, because the estimated maximum can be very close to the real maximum (MAX). The more specific choice of d_{tr} to achieve a good trade-off between utility and privacy level for the specific application scenarios is an interesting question, for which we will find the answer in our future work.

The worst-case complexity of the online auditor depends on the worst-case complexity of $\mathcal{P}$ and the number of posed queries. We can assume that the number of queries is $O(n)$, where n is the size of the data set. In this case, by applying one of the polynomial time linear program solver methods, the whole complexity remains polynomial.

4.2 Simulatable auditor$^{max}_{avg}$ in the partial disclosure model

We propose a simulatable auditor that prevents the probabilistic disclosure of MAX. By transforming the AVG queries to SUM queries we can adapt one part of the auditor given in [6],[5], but our problem is different from those in [6],[5], because we consider bounded intervals and MAX. Hence, the methods proposed for SUM auditors cannot be used entirely in our case, and although using similar terminology, the proofs are not the same (see [13]).

We assume that each element x_i is independently drawn according to a distribution $\mathcal{G}$ that belongs to the familiy of log-concave distributions over the set $\mathbb{R}$ of real numbers between $[\alpha, \beta]$. Note that we consider the class of log-concave distribution because it covers many important distributions including the guassian distribution. In addition, our online simulatable auditor is based on random sampling, and we want to apply directly the method of Lovasz [9] on effective sampling from log-concave distributions. The main advantage of the sampling method in [9] is that it is polynomial-time and produces only small error.

A distribution over a domain D is said to be log-concave if it has a density function f such that the logarithm of f is concave. Due to the lack of space we only sketch the proofs in this section, but the full proofs can be found in [13].

Lemma 2 *Next we give some relevant points that will make the method in [9] applicable in the construction of our auditor.*

1. *The truncated version of log-concave distribution is also log-concave.*

2. *If $\mathcal{G}$ is a log-concave distribution then the joint distribution $\mathcal{G}^n$ is also log-concave.*
3. *Let the joint distribution $\mathcal{G}^n$ conditioned on $\wedge_{j=1}^{t}(avg(Q_j) = a_j)$, be $\mathcal{G}_t^n$. If $\mathcal{G}$ is a log-concave distribution then $\mathcal{G}_t^n$ is also log-concave.*

Proof. (Sketch)

1. Let the density and the cumulative distribution function of a variable Y be $f(y)$ and $F(y)$, respectively. The truncated version of $f(y)$, $f(y|Y \in I)$, is equal to $\frac{f(y)}{\int_I f(y)dy}$. By assumption, $f(y)$ is log-concave and the denominator is a constant, it follows that $f(y|Y \in I)$ is log-concave. Hence, returning to our problem, each x_i is taken according to a truncated log-concave distribution, which is log-concave.
2. Because the logarithm of the product of log-concave functions is a concave function we get that the product of log-concave distributions is also log-concave. From this the second point of the Lemma follows.
3. Similar to the truncated distribution density function, the density of $\mathcal{G}_t^n$ is as follows: $f_{\mathcal{G}_t^n}(*) = \frac{f_{\mathcal{G}^n}(*)I_{\mathcal{P}}(*)}{Pr(\boldsymbol{x}\in\mathcal{P})}$, where $f_{\mathcal{G}^n}(*)$ is the density of the joint distribution, $I_{\mathcal{P}}(*)$ is an indicator function that returns 1 if $\boldsymbol{x}$ are in the convex constraint $\mathcal{P}$ induced by the t queries and answers, and 0 otherwise. The denominator contains the probability that $\boldsymbol{x}$ being within $\mathcal{P}$, which is a constant value for a given $\mathcal{P}$. According to second point and based on the similar argument as the case in the first point, it follows that $f_{\mathcal{G}_t^n}(*)$ is log-concave.

□

In our case, the predicate λ-Safe and AllSafe is a bit different from the traditional definitions discussed in Section 2, because we are considering the maximum of n values instead of single values. Specifically, in $\text{Safe}_{\lambda,I}(\wedge_1^t(q_j, a_j))$ we require $\frac{P_{\mathcal{G}_{post}^t}(MAX\in I|\wedge_{j=1}^{t}(avg(Q_j)=a_j))}{Pr_{\mathcal{G}_{max}}(MAX\in I)}$ to be within the bound $\left[\frac{1}{1+\lambda}, 1+\lambda\right]$, where $\mathcal{G}_{post}^t$ is the distribution of the posteriori probability, and $\mathcal{G}_{max}$ is the distribution of *MAX*. The definition of AllSafe, $\text{AllSafe}_{\lambda,\omega}(\wedge_1^t(q_j, a_j))$ is then given over all ω-significant intervals J of $[\alpha, \beta]$. Here the notion of ω-significant interval is defined over the maximum value instead of individual values: An interval J is ω-significant if $P_{\mathcal{G}_{max}}(MAX \in J) \geq \frac{1}{\omega}$. The definitions of ($\lambda$, ω, T)-privacy game and (λ, δ, ω, T)-privacy auditor remains unchanged.

In [9] the authors proposed the algorithm $Sample(D, \epsilon)$ for sampling from an arbitrary log-concave distribution D (defined in $\mathbb{R}^n$) with the best running time of $O^*(n^5)$, such that the sampled output follows a distribution D' where the total variation distance between D and D' is at most ϵ. The notation $O^*()$ is taken from [9], and indicates that the polynomial dependence on $\log n$, and the error parameter ϵ are not shown. We make use of this algorithm for constructing our auditor.

The next question is that what kind of, and how many intervals I we need to consider when examining the AllSafe predicate. Of course, in practise, we cannot

examine infinitely many sub-intervals in $[\alpha, \beta]$. Following the approach in [6], we show that it is enough to check only finite number of intervals.

Let us consider the quantiles or quantile function in statistics. Informally, a p-quantile has the value x if the fraction of data smaller than x is p. A quantile function is the inverse of a distribution function. We use the methods for finding quantiles in case of $\mathcal{G}_{max}$ and divide the domain into γ sub-intervals, $I_1,\ldots, I_\gamma$ such that $P_{\mathcal{G}_{max}}(MAX \in I_i) = \frac{1}{\gamma}$, for $1 \leq i \leq \gamma$ (this is related to the inverse distribution function in order statistics). In Lemma 3 we show that if AllSafe evaluates to 1 in case of the γ intervals for a smaller privacy parameter $\widetilde{\lambda}$ (i.e., stricter privacy) then it evaluates to 1 in case of ω-significant intervals as well.

Lemma 3 *Suppose* $Safe_{\widetilde{\lambda},I} = 1$ *for each interval* I *of the* γ *intervals, and* $\widetilde{\lambda} = \frac{\lambda(c-1)-2}{c+1}$*, where* c *is any integer greater than* $1 + 2/\lambda$*. Then,* $Safe_{\lambda,J} = 1$ *for every* ω*-significant interval* J*.*

Proof. (Sketch)

Based on the intuition we use during our proof (see the three cases discussed below) and to achieve that $\widetilde{\lambda}$ is smaller than λ, we set $\widetilde{\lambda}$ such that $\frac{c+1}{c-1}(1 + \widetilde{\lambda})$ $= (1 + \lambda)$. Further, to make $\widetilde{\lambda}$ be positive, based on the setting of $\widetilde{\lambda}$ above we choose the parameter c to be larger than $1+2/\lambda$. In addition, γ is set to be larger than ω, namely, to $\lceil c\omega \rceil$, where the brackets represent ceiling. Finally, let J be a ω-significant interval and denote $P(MAX \in J)$ as P_J^{max}, and let $d = \lceil \gamma P_J^{max} \rceil$. Note that with these settings of γ and d we have $d \geq c$ and $\frac{d+1}{d-1} \leq \frac{c+1}{c-1}$.

Our goal is to prove that the sequence $\wedge_1^t(q_i, a_i)$ is λ-Safe for each ω-significant interval, and to do this, we prove a stronger privacy notion. Specifically, we show that if the sequence $\wedge_1^t(q_i, a_i)$ is $Safe_{\widetilde{\lambda},I} = 1$ for each interval I, then it is $(\frac{d+1}{d-1}(1+\widetilde{\lambda})-1)$-Safe for every interval J. This is a stronger privacy requirement because $\frac{d+1}{d-1}(1 + \widetilde{\lambda}) - 1 \leq \frac{c+1}{c-1}(1 + \widetilde{\lambda}) - 1 = \lambda$. To prove this we examine three possible cases, and we show that this holds in all these cases: (Case 1) J is contained in the union of $d+1$ consecutive intervals, say I_1, I_2, $\ldots$, I_{d+1}, of which J contains the intervals I_2, I_3,$\ldots$, I_d; (Case 2) J is contained in the union of $d+2$ consecutive intervals, say I_1, I_2, $\ldots$, I_{d+2}, of which J contains the intervals I_2, I_3, $\ldots$, I_{d+1}; (Case 3) J is contained in the union of $d+1$ consecutive intervals, say I_1, I_2, $\ldots$, I_{d+1}, of which J contains the d intervals I_1, I_1, $\ldots$, I_d.

□

Now we turn to the construction of the simulatable auditor. According to Definitions 2 and 3, first, we provide the method (Algorithm 3) for checking if the predicate AllSafe is 1 or 0, and then we construct the simulatable auditor (Algorithm 4) based on the concept shown in Section 2 and the definition of (λ, δ, ω, T)-privacy game.

We give the algorithm $\overline{\text{AllSafe}}$, which is an *estimation* of the predicate $\text{AllSafe}_{\lambda,\omega}$. This is because the algorithm makes use of the sampling algorithm $Sample(\mathcal{G}_t^n, \epsilon)$ for estimating the posteriori probability, and instead of examining all the ω-significant intervals, we make an estimation by only taking into account γ intervals: $\overline{\text{AllSafe}}$ takes as inputs (1) the sequence of queries and answers $q_1,\ldots, q_t$,

$a_1,\ldots,\ a_t$; (2) the distribution $\mathcal{G}$; (3) a probability η of error for computing ϵ; (4) the trade-off parameter c such that $\gamma = \lceil c\omega \rceil$, and $\tilde{\lambda} = \frac{\lambda(c-1)-2}{c+1}$, where $\lceil\ \rceil$ represents ceiling; (5) the parameter ω; and (6) the size n of the data set.

The parameter choice is made such that the Lemma 4 holds. In other words, if we modify the privacy parameters in Lemma 4 we have to modify the parameters above as well. Moreover, the intuition behind the parameter choice resides in the proof technique. In our proofs we apply the well-known definitions and theorems related to the Chernoff-bound, Union bound, and some basic statements in statistics and probability theory. Roughly speaking, these parameters have been chosen such that the Chernoff-bound and Union-bound can be applicable. We emphasize that the choice of these specific parameters is only for better illustrating purposes. These specific values of the parameters are one possible choice but not the only one. The general form of parameters is provided in [13].

One drawback of Lemma 3 is that the reverse direction is not necessarily true. Thus, to make claims on the AllSafe = 0 case, we cannot use directly the privacy parameter $\tilde{\lambda}$. Instead, in the algorithm $\overline{\text{AllSafe}}$ we consider an even more stronger privacy notion with a smaller parameter $\lambda' = \tilde{\lambda}/3$. We note that λ' can be any value that is smaller than $\tilde{\lambda}$ (see the proof in [13]), but then we have to modify the privacy parameters in Lemma 4 accordingly. In our case, however, we choose it to be $\tilde{\lambda}/3$ for easier discussion and illustrating purposes. The error ϵ of the algorithm $Sample(\mathcal{G}_t^n, \epsilon)$ is set to be $\frac{\eta}{2N}$. (see [13] for details)

Algorithm 3: $\overline{\text{AllSafe}}$ ($q_1,\ldots,q_t$, $a_1,\ldots,a_t$, $\mathcal{G}$, η, ω, λ, n, c)

Let AllSafe = TRUE;
for each of the γ intervals I in $[\alpha, \beta]$ **do**
  Sample N data sets according to $\mathcal{G}_t^n$, using $Sample(\mathcal{G}_t^n, \epsilon)$;
  Let N_{max}, $N_{max} \subseteq N$, be the number of data sets for which MAX $\in I$;
  if $\left(\frac{\gamma N_{max}}{N} \notin \left[\frac{1}{1+\lambda'}, 1+\lambda'\right]\right)$ **then** Let AllSafe = FALSE; **endif**
endfor
return AllSafe;

Algorithm 4: Simulatable probabilistic auditor

Inputs: $q_1,\ldots,q_{t-1}$, $a_1,\ldots,a_{t-1}$, a new query q_t, $\mathcal{G}$, δ, η, λ, γ, n, T, c;
Let $\epsilon = \delta/10T$;
for $\frac{80T}{9\delta} \ln \frac{T}{\delta}$ times **do**
  Sample a consistent data set X' according to $\mathcal{G}_{t-1}^n$ using Sample($\mathcal{G}_{t-1}^n$, ϵ);
  Let $a_t' = avg_{X'}(Q_t)$; **call** $\overline{\text{AllSafe}}(q_1,\ldots,q_t,a_1,\ldots,a_t',\mathcal{G},\eta,\omega,\lambda,n,c)$;
endfor
if the fraction of data sets X' for which $\overline{\text{AllSafe}}$=FALSE is greater than $\frac{9\delta}{20T}$ **then**
  return DENY; **else return** a_t;
endif;

In Algorithm 3, N denotes the total number of data sets $(x_1,\ldots,x_n)$ sampled according to $Sample(\mathcal{G}_t^n, \epsilon)$, and N_{max}, $N_{max} \subseteq N$, denotes the number of the data sets satisfying $MAX \in I$. Hence, the posteriori probability is estimated by the ratio $\frac{N_{max}}{N}$. In addition, the apriori probability is $\frac{1}{\gamma}$, and according to Definition 2 the probability ratio $\frac{\gamma N_{max}}{N}$ is required to be close to 1.

Intuitively, the steps in Algorithm 3 are as follows: By Lemma 3 instead of checking infinite ω-significant intervals with the privacy parameter λ we check the Safe predicate for each of the γ intervals and the smaller privacy parameter λ'. To estimate the posteriori probability that $MAX \in I$, we sample sufficient number (N) of data sets according to the distribution $\mathcal{G}_t^n$, and compute the fraction (N_{max}) of the data sets for which the maximum value falls in the interval I. Intuitively, by sampling according to $\mathcal{G}_t^n$ we get the data sets that satisfy the condition $\wedge_{j=1}^{t}(avg(Q_j) = a_j)$. If the ratio of the posteriori and apriori probabilities is outside the required bounds then the algorithm returns FALSE, otherwise TRUE is output.

Next we discuss how good estimation Algorithm 3 provides. In the ideal case, we would like that if the predicate $\text{AllSafe}_{\lambda,\omega}$ returns 0 (1) then the algorithm $\overline{\text{AllSafe}}$ returns FALSE (TRUE). However, we cannot make these claims for the next reasons: (i) we do not check all (infinitely many) ω-significant intervals for privacy and instead check only γ intervals; (ii) we estimate the posteriori probability using sampling, which has some error. Hence, instead of achieving the ideal case we provide the following claims:

Lemma 4 *1. If* $AllSafe_{\lambda,\omega}(q_1, \ldots, q_t, a_1, \ldots, a_t) = 0$ *then Algorithm* $\overline{AllSafe}$ *returns FALSE with probability at least* $1 - \eta$.

2. If $AllSafe_{\tilde{\lambda}/9,\gamma}(q_1, \ldots, q_t, a_1, \ldots, a_t) = 1$ *then Algorithm* $\overline{AllSafe}$ *returns TRUE with probability at least* $1 - 2\gamma\eta$.

Proof. (Sketch) The proof and the parameter setting for this Lemma is based on the application of the well-known Chernoff-bound and Union-bound. Let X_1, $\ldots$, X_n be independent Bernoulli trials (or Poisson trials), with $P(X_i = 1) = p$ (or $P(X_i = 1) = p_i$ in case of Poisson trials). Let X be $\sum_1^n X_i$ with μ be $E[X]$, and $\theta \in (0, 1]$. The Chernoff-bound says: $P(X \leq \mu(1-\theta)) \leq e^{-\mu\theta^2/2} \leq e^{-\mu\theta^2/4}$, and $P(X \geq \mu(1+\theta)) \leq e^{-\mu\theta^2/4}$. The Union-bound says that if we have the events e_1,$\ldots$, e_n then by applying the Chernoff-bound we can give a bound for the union of these events, that is, $P[e_1 \cup \cdots \cup e_n] \leq \sum_1^n P[e_i] \leq \sum_1^n bound_i$. □

Intuitively, with probability close to 1, whenever $\text{AllSafe}_{\lambda,\omega} = 0$ the algorithm $\overline{\text{AllSafe}}$ also returns FALSE, and for a smaller privacy parameter $\tilde{\lambda}/9$ whenever $AllSafe_{\tilde{\lambda}/9,\gamma} = 1$ then $\overline{\text{AllSafe}}$ returns TRUE. For the region in between, no guarantees can be made. We note that in the general case, by choosing properly the input parameters, in the second point of the Lemma, we can choose any privacy parameter smaller than $\tilde{\lambda}$. The question is that, with these chosen parameters, how large should N be? We show that, based on the Chernoff-bound (see [13]), setting $N = \frac{9\gamma^2 ln(2/\eta)}{\tilde{\lambda}^2} * (1+\lambda')^2 * max((1+\tilde{\lambda})^2, (3+\lambda')^2)$ is suitable for fullfiling the claims in the Lemma.

Now that we have an algorithm that evaluates the predicate $\text{AllSafe}_{\lambda,\omega}$, we turn to discuss the construction of the simulatable auditor itself. During the auditor construction, besides making use of the algorithm $\overline{\text{AllSafe}}$ we also take into account the notion of the T-round privacy game discussed in Section 2.

In Algorithm 4, beyond the parameters used in $\overline{\text{AllSafe}}$, additional parameters δ and T are concerning the (λ, ω, T)-privacy game and the (λ, δ, ω, T)-privacy auditor, and ϵ is the sampling error. Intuitively, the auditor repeatedly samples, according to the distribution $\mathcal{G}^n_{t-1}$, a data set X' that is consistent with the previous $t-1$ queries and answers. Then the corresponding answer a'_t is computed based on X' and the query set Q_t of the query q_t. Thereafter, we call the algorithm $\overline{\text{AllSafe}}$ with the previous queries and answers, along with q_t and a'_t. If the *fraction* of data sets for which $\overline{\text{AllSafe}}$ returns FALSE is larger than $9\delta/20T$ then the auditor denies, otherwise it returns the true answer a_t. The reason of choosing $9\delta/20T$ is that we want to fullfil the definition of (λ, δ, ω, T)-privacy auditor. The proof that Algorithm 4 implements a (λ, δ, ω, T)-privacy auditor is based on the well-known theorems of the Chernoff bound and Union bound over T rounds of the privacy game.

Theorem 1 *Algorithm 4 implements a $(\lambda, \delta, \omega, T)$-private simulatable auditor, and its running time is $N\gamma\frac{80T}{9\delta}\ln\frac{T}{\delta}T_{samp}(\mathcal{D}_c, \epsilon)$, where $T_{samp}(\mathcal{D}_c, \epsilon)$ is the running time of the algorithm Sample$(\mathcal{D}_c, \epsilon)$, and $\mathcal{D}_c$ represents either $\mathcal{G}^n_{t-1}$ or $\mathcal{G}^n_t$. Finally, the running time of the simulatable auditor after t queries is $t\gamma N\frac{80T}{9\delta}$ $log\frac{T}{\delta}T_{samp}(\mathcal{D}_{cond}, \epsilon)$.*

Proof. (Sketch) Again, the proof of the first point is based on the Chernoff-bound and Union-bound. The running time results from the fact that we check γ intervals and sample N data sets in each of the $\frac{80T}{9\delta}\ln\frac{T}{\delta}$ round, using the algorithm Sample. Finally, this process is executed totally t times after t queries. □

Since the running time of the algorithm Sample is polynomial [9], the running time of the Algorithm 4 is polynomial. We assume that our simulatable auditor does not include the quantile computation procedure, however, note that there is a large class of $\mathcal{G}$ for which the quantile computation is polynomial-time.

5 Conclusion

We defined a novel setting for query auditing, where instead of detecting or preventing the disclosure of individual sensitive values, we want to detect or prevent the disclosure of aggregate values in the database. As a specific instance of this setting, in this paper, we studied the problem of detecting or preventing the disclosure of the maximum value in the database, when the querier is allowed to issue average queries to the database. We proposed efficient off-line and on-line query auditors for this problem in the full disclosure model, and an efficient simulatable on-line query auditor in the partial disclosure model. Our future work is concerned with looking at other instances (e.g., other types of aggregates in the queires) and prototypical implementation of our algorithms for experimentation in the context of the CHIRON project.

Acknowledgments. The work presented in this paper has been carried out in the context of the CHIRON Project (www.chiron-project.eu), which receives funding from the European Community in the context of the ARTEMIS Programme (grant agreement no. 225186). The authors are also partially supported by the grant TAMOP - 4.2.2.B-10/12010-0009 at the Budapest University of Technology and Economics.

References

1. C. C. Aggarwal and P. S. Yu, editors: Privacy-Preserving Data Mining - Models and Algorithms, vol. 34 of Advances in Database Systems, Springer, (2008).
2. Y. Chen and D. Evans: Auditing information leakage for distance metrics. In: 3rd IEEE International Conference on Privacy, Security, Risk and Trust, pp. 1131–1140. IEEE, 2011.
3. F. Chin: Security problems on inference control for sum, max, and min queries. J. ACM, 33:451–464, May 1986.
4. F. Chin and G. Ozsoyoglu: Auditing for secure statistical databases. In Proceedings of the ACM '81 conference, pp. 53–59, New York, USA, 1981
5. K. Kenthapadi: Models and algorithms for data privacy. Ph.D. Thesis, Computer Science Department, Stanford University, 2006.
6. K. Kenthapadi, N. Mishra, and K. Nissim: Simulatable auditing. In 25th Symposium on Principles of Database Systems(PODS), pp. 118–127, 2005.
7. J. Kleinberg, C. Papadimitriou, and P. Raghavan: Auditing boolean attributes. In Journal of Computer and System Sciences, pages 86–91, 2000.
8. Z. Lei, J. Sushil, and B. Alexander. Simulatable binding: Beyond simulatable auditing. In Proceedings of the 5th VLDB workshop on Secure Data Management, pp. 16–31. Springer-Verlag, 2008.
9. L. Lovász and S. Vempala: The geometry of logconcave functions and sampling algorithms. In Journal Random Struct. Algorithms, 30:307–358, May 2007.
10. S. U. Nabar, B. Marthi, K. Kenthapadi, N. Mishra, and R. Motwani: Towards robustness in query auditing. In Proceedings of the 5th VLDB workshop on Secure Data Management, pp. 151–162, 2006.
11. S. U. Nabar, B. Marthi, K. Kenthapadi, N. Mishra, and R. Motwani: Towards robustness in query auditing. Technical Report, Stanford University, 2006.
12. J. Renegar: A polynomial-time algorithm, based on Newton's method, for linear programming. Mathematical Sciences Research Institute (Berkeley, Calif.), 1st edition, 1986.
13. T. V. Thong and L. Buttyán: Query auditing for protecting max/min values of sensitive attributes in statistical databases. `http://www.crysys.hu/members/tvthong/QA/ThB12QATech.pdf`, 2012.
14. L. Yingjiu, W. Lingyu, W. X. Sean, and J. Sushil: Auditing interval-based inference. In Proceedings of the 14th International Conference on Advanced Information Systems Engineering, CAiSE '02, pp. 553–567. Springer-Verlag, 2002.