M-score: A Misuseability Weight Measure

Amir Harel, Asaf Shabtai, Lior Rokach and Yuval Elovici

Abstract—Detecting and preventing data leakage and data misuse poses a serious challenge for organizations, especially when dealing with insiders with legitimate permissions to access the organization's systems and its critical data. In this paper, we present a new concept, Misuseability Weight, for estimating the risk emanating from data exposed to insiders. This concept focuses on assigning a score that represents the sensitivity level of the data exposed to the user and by that predicts the ability of the user to maliciously exploit this data. Then, we propose a new measure, the *M*-score, which assigns a misuseability weight to tabular data, discuss some of its properties, and demonstrate its usefulness in several leakage scenarios. One of the main challenges in applying the *M*-score measure is in acquiring the required knowledge from a domain expert. Therefore, we present and evaluate two approaches toward eliciting misuseability conceptions from the domain expert.

Index Terms—Data Leakage; Data Misuse; Security Measures; Misuseability Weight.

1 Introduction

SENSITIVE information such as customer or patient data and business secrets constitute the main assets of an organization. Such information is essential for the organization's employees, sub-contractors, or partners to perform their tasks. Conversely, limiting access to the information in the interests of preserving secrecy might damage their ability to implement the actions that can best serve the organization. Thus, data leakage and data misuse detection mechanisms are essential in identifying malicious insiders.

The task of detecting malicious insiders is very challenging as the methods of deception become more and more sophisticated. According to the 2010 Cyber Security Watch Survey [1] 26% of the cyber-security events, recorded in a 12-month period, were caused by insiders. These insiders were the most damaging with 43% of the respondents reporting that their organization suffered data loss. Of the attacks, 16% were caused by theft of sensitive data and 15% by exposure of confidential data.

The focus of this paper is on mitigating leakage or misuse incidents of data stored in databases (i.e., tabular data) by an insider having legitimate privileges to access the data. There have been numerous attempts to deal with the malicious insider scenario. The methods that have been devised are generally based on user behavioral profiles that define normal user behavior and issue an alert whenever a user's behavior significantly deviates from the normal profile. The most common approach for representing user behavioral profiles is by analyzing the SQL statement submitted by an application server to the database (as a result of user requests), and extracting various features from these SQL statements [2]. Another approach focuses on analyzing the actual data exposed to the user, i.e., the result-sets [3]. However, none of the proposed methods consider the different sensitivity levels of the data to which an insider is exposed. This factor has a great impact in estimating the damage that can be caused to an organization when data is leaked or misused. Security-related data measures including *k*-Anonymity [4], *l*-Diversity [5] and (*a*,*k*)-Anonymity [6] are mainly used for privacy-preserving and are not relevant when the user has free access to the data. Therefore, we present a new concept, *Misuseability Weight*, which assigns a sensitivity score to datasets, thereby estimating the level of harm that might be inflicted upon the organization when the data is leaked. Four optional usages of the misuseability weight are proposed: (1) applying anomaly detection by learning the normal behavior of an insider in terms of the sensitivity level of the data she is usually exposed to; (2) improving the process of handling leakage incidents identified by other misuse detection systems by enabling the security officer to focus on incidents involving more sensitive data; (3) implementing a dynamic misuseability-based access control, designed to regulate user access to sensitive data stored in relational databases; and (4) reducing the misuseability of the data.

The rest of this paper is organized as follows. In section 2 we review related works in the domain of data misuse detection and data privacy measures. Section 3 introduces the misuseability weight concept and in sections 4 and 5 we present and illustrate the *M*-score, a misuseability weight measure for tabular data. Section 6 presents extensions to the basic *M*-score definition. In section 7 we present several applications of the *M*-score measure. In section 8 we describe an experiment that was conducted in order to show that the *M*-score fulfills its goal of weighting misuseability, as well as to determine the best approach to acquire the sensitivity function from domain experts. Finally, section 9 concludes the paper.

2 Related Work

2.1 Misuse Detection in Databases

In recent years, several methods have been proposed for

A. Harel, A. Shabtai, L. Rokach, and Y. Elovici are with the Information Systems Engineering Department and Deutsche Telekom Laboratories at BGU, Ben-Gurion University of the Negev, Beer-Sheva, Israel. E-mail: harelam@bgu.ac.il; shabtaia@bgu.ac.il; rokach@bgu.ac.il; elovici@bgu.ac.il.

Manuscript received 7 November, 2011.

mitigating data leakage and data misuse in database systems. These methods can generally be classified as syntax-centric or data-centric. The syntax-centric approach relies on the SQL-expression representation of queries to construct user profiles. For example, a query can be represented by a vector of features extracted from the SQL statement, such as the query type (e.g., SELECT or INSERT), and the tables or attributes requested by the query [2]. Celikel et al. [7] present a model for risk management in distributed database systems. The model is used to measure the risk poses by a user in order to prevent her from misusing or abusing her role privileges. In the model, a *Risk Priority Number* (RPN) is calculated for each user, which is the product of the *Occurrence Rating* (*OR*) that reflects the number of times the same query was issued with respect to the other users in the same role; the *Severity Rating* (*SR*) that measures the risk by referring to the quality of the data the user might get from the queries she issued; and the *Detection Rating* (*DR*) indicates how close the behavior of the user is to the behavior of users in other roles. Another syntax-centric method is the framework to enforce access control over data streams [8] that define a set of secure actions (e.g., secure join) that replaces any unsecure action (e.g., join) the user makes. When a user issues an unsecure action, the appropriate secure action is used instead, and by addressing the user permissions, retrieves only data that this user is eligible to see.

The data-centric approach focuses on what the user is trying to access instead of how she expresses it. With this approach, an action is modeled by extracting features from the obtained result-set. Since we are dealing with data leakage, we assume that analyzing what a user sees (i.e., the result-sets) can provide a more direct indication of a possible data misuse. An interesting work [3] presents a data-centric approach and considers a query's expression syntax as irrelevant for discerning user intent; only the resulting data matters. For every access to a database, a *statistical vector* (S-Vector) is created, holding various statistical details on the result-set data, such as minimum, maximum and average for numeric attributes, or counts of the different values for text attributes. Evaluation results showed that the S-Vector significantly outperforms the syntax centric approach presented in [2]. Yaseen et al. [9] also proposed a data-centric method that uses dependency graphs based on domain expert knowledge. These graphs are used in order to predict the ability of a user to infer sensitive information that might harm the organization using information she already obtained. Then, utilizing dependency graphs, the system prevents unauthorized users from gaining information that enables them to infer or calculate restricted data they are not eligible to have.

Closely related to this line of works are the preventive approaches. The insider prediction tool [10] uses a taxonomy of insider threats to calculate the *Evaluated Potential Threat* (EPT) measure. This measure tries to estimate whether a user's action is correlated with a part of the taxonomy that is labeled as malicious. The EPT is calculated by considering features describing the user, the context of the action and the action itself. In addition, the tool uses a set of malicious actions that were previously discovered. To prevent insiders from misusing their privileges, Bishop and Gates [11] suggested the *group-based access control* (GBAC) mechanism, which is a generalization of RBAC mechanism. This mechanism uses, in addition to the user's basic job description (role), the user characteristics and behavioral attributes such as the time she normally comes to work or the customers with whom she usually interacts.

As already mentioned, none of the proposed methods consider the sensitivity level of the data to which the user may be exposed. This factor can greatly impact the outcome when trying to estimate the potential damage to the organization if the data is leaked or misused. Consequently, we adopted the data-centric approach - the data retrieved by a user action is examined and its sensitivity level computed.

2.2 Privacy-Preserving Data Publishing

During the past decade, several measures in the field of *privacy-preserving data publishing* (PPDP) were introduced [12]. Examples of such measures are *k*-Anonymity [4], *l*-Diversity [5] and (*a*,*k*)-Anonymity [6]. These measures attempt to estimate how easy it is to compromise an individual's privacy in a given publication, where publication refers to a table of data containing *quasi-identifier* attributes, *sensitive* attributes and *additional* (other) attributes. The main goal of these measures is to estimate the ability of an attacker to infer who are the individuals (also called victims) behind the quasi-identifier, and thus reveal sensitive attribute values (e.g., disease).

PPDP algorithms are useful when there is a need for exporting data (e.g., for research) while retaining the privacy of individuals in the published dataset. It can also be used in a limited way for estimating the level of misuseability of data. The harder it is to identify who is the entity in a record, the lower the potential risk of a perpetrator maliciously exploiting that information. This approach, however, is not effective in other scenarios that assume a user has full access to the data.

Sweeney [4] proposed the *k*-anonymity measure that indicates how hard it is to fully identify who the owner is of each record in a published table *T*, given a publicly available database (e.g., Yellow Pages). The measure determines that *T* satisfies *k*-anonymity if and only if each value of the quasi-identifier in *T* appears at least *k* times.

A known disadvantage of *k*-anonymity is that it does not consider the diversity of the sensitive attribute value (also known as the common sensitive attribute problem). In an effort to deal with this issue, the *l*-Diversity measure [5] employs a different approach that considers the diversity of the sensitive values denoted by *T*. (*α*,*k*)-Anonymity [6] is a hybrid approach that adds to *k*-anonymity a requirement that for every different value of quasi-identifier, every different value of the sensitive attributes appears with a frequency of no less than $\alpha \in [0,1]$.

A closely related research topic is *differential privacy*. The goal of differential privacy is to ensure that statistical (or aggregation) queries can be executed on a database with high accuracy while preserving the privacy of the

entities in the database [13],[14]. This approach is relevant only when exposing statistical information rather than individual records (e.g., for analytics or data mining tasks). However, in most cases, performing different tasks require exposing the individual records. The *M*-score measure is mainly used for deriving the misuseability level of the individual records exposed to the user.

Next, we discuss the shortcomings of the PPDP measures in the context of measuring the misuseability level and why a new measure should be introduced.

3. Misuseability Weight Concept

Data stored in an organization's computers is extremely important and embodies the core of the organization's power. An organization undoubtedly wants to preserve and retain this power. On the other hand, this data is necessary for daily work processes. Users within the organization's perimeter (e.g., employees, sub-contractors, or partners) perform various actions on this data (e.g., query, report, and search) and may be exposed to sensitive information embodied within the data they access.

In an effort to determine the extent of damage to an organization that a user can cause using the information she has obtained, we introduce the concept of *Misuseability Weight*. By assigning a score that represents the sensitivity level of the data that a user is exposed to, the misuseability weight can determine the extent of damage to the organization if the data is misused. Using this information, the organization can then take appropriate steps to prevent or minimize the damage

3.1 Dimensions of Misuseability

Assigning a misuseability weight to a given dataset is strongly related to the way the data is presented (e.g., tabular data, structured or free text) and is domain specific. Therefore, one measure of misuseability weight cannot fit all types of data in every domain. In this section, we describe four general dimensions of misuseability. These dimensions, which may have different levels of importance for various domains, can serve as guidelines when defining a misuseability weight measure. While the first two dimensions are related to entities (e.g., customers, patients, or projects) that appear in the data, the last two dimensions are related to the information (or properties) that are exposed about these entities. The four dimensions are:

Number of entities – This is the data size with respect to the different entities that appear in the data. Having data about more entities obviously increase the potential damage as a result of a misuse of this data.

Anonymity level – While the number of different entities in the data can increase the misuseability weight, the anonymity level of the data can decrease it. The anonymity level is regarded as the effort that is required in order to fully identify a specific entity in the data.

Number of properties – Data can include a variety of details, or properties, on each entity (e.g., employee salary or patient disease). Since each additional property can increase the damage as a result of a misuse the number of different properties (i.e., amount of information on each entity) should affect the misuseability weight.

Values of properties – The property value of an entity can greatly affect the misuseability level of the data. For example, a patient record with disease property equals to HIV should probably be more sensitive than a record concerning patient with a simple flu.

In the context of these four dimensions, we claim that PPDP measures are only effective in a limited way through their capability of measuring the anonymity level dimension of the data. These measures, however, lack any reference to the other important dimensions that are necessary for weighting misuseability. For example, consider a table that shows employee names and salaries. Even if we double all the salaries that appear in the table, there may not be any change in neither of these measures' scores, and therefore no reference to the values of properties dimension. As a result of this lack, as well as others, we conclude that PPDP measures are not sufficiently expressive to serve as a misuseability weight measure and that a new measure is needed. In the following section we introduce our proposal for addressing this need.

4 The M-Score Measure

To measure the misuseability weight, we propose a new algorithm - the *M-score*. This algorithm considers and measures different aspects related to the misuseability of the data in order to indicate the true level of damage that can result if an organization's data falls into wrong hands. The *M*-score measure is tailored for tabular datasets (e.g., result sets of relational database queries) and cannot be applied to non-tabular data such as intellectual property, business plans, etc. It is a domain independent measure that assigns a score, which represents the misuseability weight of each table exposed to the user, by using a sensitivity score function acquired from the domain expert.

4.1 Formal Definition

In this section we provide the formal definitions for the *M*-score. Without loss of generality, we assume that only a single database exists. Nevertheless, the measure can be easily extended to cope with multiple databases. The first definition discusses the building blocks of our measure – table and attributes.

Definition 1. Table and Attribute. A table $T(A_1, \ldots, A_n)$ is a set of r records. Each record is a tuple of n values. The value i of a record, is a value from a closed set of values defined by A_i, the i's *Attribute* of T. Therefore, we can define A_i either as the name of the column i of T, or as a domain of values.

We define three, non-intersecting types of attributes: *quasi-identifier* attributes [15]; *sensitive* attributes; and *other* attributes, which are of no importance to our discussion. To exemplify the computation of the *M*-score, we use throughout this paper the database structure of a cellular company as represented in Fig. 1.

Definition 2. Quasi-Identifier attributes. *Quasi-identifier attributes* $Q = \{q_{i1},\ldots,q_{ik}\} \subseteq \{A_i,\ldots,A_n\}$ are attrib-

utes that can be linked, possibly using an external data source, to reveal a specific entity that the specific information is about. In addition, any subset of the quasi-identifiers (consisting of one or more attributes of Q) is a quasi-identifier itself.

In Fig. 1, seven quasi-identifier attributes are presented: q_1 = *First Name*; q_2 = *Last Name*; q_3 = *Job*; q_4 = *City*; q_5 = *Sex*; q_6 = *Area Code*; and q_7 = *Phone Number*.

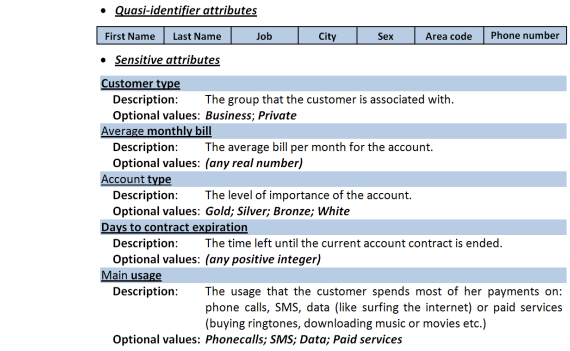

Fig. 1. An example of quasi-identifier and sensitive attributes.

DEFINITION 3. Sensitive attributes. *Sensitive attributes* $S_j = \{s_{j1},\ldots,s_{jk}\} \subseteq \{A_i,\ldots,A_n\}$ are attributes that are used to evaluate the risk derived from exposing the data.

The sensitive attributes are mutually excluded from the quasi-identifier attributes (i.e., $\forall j\ S_j \cap Q = \emptyset$).
In our example, we have five different sensitive attributes – from s_1 = *Customer Group* to s_5 = *Main Usage*.

The next definition introduces the function we use in order to determine the sensitivity level of a record in the table. Previous studies have shown that privacy (and therefore misuseability) of data, are fundamentally context-driven (e.g., [16]). Barth et al. [17] reject the claim that the definition of private versus public does not include a given context. In light of this works, context is also a parameter in our sensitivity score function. The context in which the table was exposed, denoted by C, is a vector of m contextual attributes $<c_1, \ldots, c_m>$. Contextual attributes can be, for example, the time when the action was performed (e.g., daytime, working hours, weekend); the location in which it happened (e.g., the hospital in which the patient is hospitalized or a clinic in another part of the country); or the user's role. The specific context is defined by the combination of the values of the contextual attributes. The degree of sensitivity of individual records (and therefore the sensitivity of a table) is context dependant; i.e., the same table may have a different sensitivity rank within different contexts.

DEFINITION 4. Sensitivity score function. The sensitivity score function $f: C \times S_j \rightarrow [0,1]$ assigns a sensitivity score to each possible value x of S_j, according to the specific context $c \in C$ in which the table was exposed.
For each record r, we denote the value x_r of S_j as $S_j[x_r]$.

The sensitivity score function should be defined by the data owner (e.g., the organization) and it reflects the data owner's perception of the data's importance in different contexts. When defining this function, the data owner might take into consideration factors such as privacy and legislation, and assign a higher score to information that eventually can harm others (for example, customer data that can be used for identity theft and might result in compensatory costs). In addition, the data owner should define the exact context attributes. For simplicity reasons, throughout the paper and experiments, we assumed that there is only one context. However, we are aware of the implications of acquiring a context-based, sensitivity score function and leave this for future work.

In Fig. 2, an example of full definition of sensitivity score function f is presented. In this example we assume that there is only one context. As shown, f can be defined for both discrete attributes (e.g., Account type) and continuous ones (e.g., Average monthly bill).

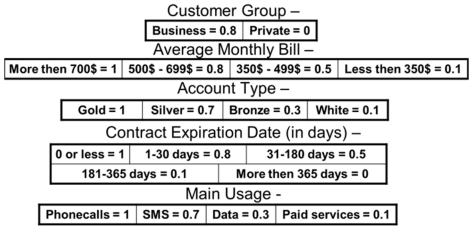

Fig. 2. An example of sensitivity score function

4.2 Calculating the *M*-Score

The *M*-score incorporates three main factors-

1. *Quality of data* - the importance of the information.
2. *Quantity data* - how much information is exposed.
3. *The distinguishing factor* - given the quasi-identifiers, the amount of efforts required in order to discover the specific entities that the table refers to.

In order to demonstrate the process of calculating the *M*-score, we use the example presented in Table 1. Table 1a represents our source table (i.e., our "database") while Table 1b is a published table that was selected from the source table and for which we calculate the *M*-score.
In the following sections, we explain each step in the proposed measure calculation.

TABLE 1
SOURCE AND PUBLISHED TABLES

(A) THE SOURCE TABLE

Job	City	Sex	Account Type	Average Monthly Bill
Lawyer	NY	Female	Gold	$350
Gardener	LA	Male	White	$160
Gardener	LA	Female	Silver	$200
Lawyer	NY	Female	Bronze	$600
Teacher	DC	Female	Silver	$300
Gardener	LA	Male	Bronze	$200
Teacher	DC	Female	Gold	$875
Programmer	DC	Male	White	$20
Teacher	DC	Female	White	$160

(B) THE PUBLISHED TABLE

Job	City	Sex	Account Type	Average Monthly Bill
Lawyer	NY	Female	Gold	$350
Lawyer	NY	Female	Bronze	$600
Teacher	DC	Female	Silver	$300
Gardener	LA	Male	Bronze	$200
Programmer	DC	Male	White	$20
Teacher	DC	Female	White	$160

4.2.1 Calculating Raw Record Score

The calculation of the raw record score of record i (or RRS_i), is based on the sensitive attributes of the table, their value in this record, and the table context. This score determines the quality factor of the final *M*-score, using the sensitivity score function f, defined in definition 4.

DEFINITION 5. Raw Record Score.

$$RRS_i = min\left(1, \sum_{S_j \in T} f(c, S_j[x_i])\right)$$

For a record i, RRS_i will be the sum of all the sensitive values score in that record, with a maximum of 1.

When comparing two tables with different number of attributes, the table with the larger number of sensitive attributes will tend to have a higher sensitivity value for each individual record. In order to be able to compare the sensitivity of tables having different number of attributes, we need to eliminate this factor. Therefore, we have set an upper bound on the RRS_i by taking the minimum between 1 and the sum of sensitivity scores of the sensitive attributes. For example, in Table 1b there are two sensitive attributes: account type and average monthly bill. Therefore, $RRS_1 = min(1,1+0.5)=1$ since, according to Fig. 2, f(Account Type[Gold])=1 and f(Average Monthly Bill[$350])=0.5. Similarly, $RRS_3 = min(1,0.7+0.1)=0.8$, since f(Account Type[Silver])=0.7 and f(Average Monthly Bill[$300])=0.1).

4.2.2 Calculating Record Distinguishing Factor

Using the distinguishing factor (*DF*), the *M*-score incorporates the uniqueness of the quasi-identifier's value in the table when weighting its misuseability. The *DF* measures to what extent a quasi-identifier reveals the specific entity it represents (e.g., a customer). It assigns a score in the range of [0,1], when the lower the score is, the harder it is to distinguish one entity from another, given this quasi-identifier. In other words, the DF of record i indicates the effort a user will have to invest in order to find the exact entity she is looking for.

Formally, the distinguishing factor function DF:{*quasi-identifiers*}→[0,1], maps a given quasi-identifier value to the true frequency of the quasi-identifier in the population of the relevant entities. For example, given a quasi-identifier "Job = Teacher" under the assumption that the population is "all US citizens", *DF* should return: (# US citizens that are also teachers) / (# US citizens).

Usually, the *DF* is not easily acquired, and therefore we use the record distinguishing factor (D_i) as an approximation. The record distinguishing factor (D_i) is a *k*-anonymity-like measure, with a different reference table from which to calculate *k*. While *k*-anonymity calculates, for each quasi-identifier, how many identical values are in the published table, the distinguishing factor's reference is "Yellow Pages". This means that an unknown data source, denoted by R_0, contains the same quasi-identifier attributes that exist in the organization's source table, denoted by R_1 (for example, Table 1a). In addition, the quasi-identifier values of R_1 are a sub-set of the quasi-identifier values in R_0, or more formally- $\Pi_{\text{quasi-identifier}}R_1 \subseteq \Pi_{\text{quasi-identifier}}R_0$. We assume that the user might hold R_0 and that $D_{R0}(x)=DF(x)^{-1}$. However, since R_0 is unknown, and since $\Pi_{\text{quasi-identifier}}R_1 \subseteq \Pi_{\text{quasi-identifier}}R_0 \Rightarrow D_{R1}(x) \leq D_{R0}(x)$, then $D_{R1}(x) \approx DF(x)^{-1}$. Therefore, we use R_1 as an approximation for calculating the distinguishing factor.

In the example presented in Table 1b the distinguishing factor of the first record is equal to two (i.e., $D_1 = 2$) since the tuple {*Lawyer*, *NY*, *Female*} appears twice in Table 1a. Similarly, $D_3 = 3$, ({*Teacher*, *DC*, *Female*} appears three times in Table 1a); $D_4 = 2$; and $D_5 = 1$.

If there are no quasi-identifier attributes in the published table, we define that for each record i, D_i equals to the published table size.

As previously mentioned, the *k*-anonymity may suffer from the common sensitive attribute problem in which an adversary may not be able to match a record with its true entity, but she can still know the sensitive values. We opt to use the variation of the *k*-anonymity measure since it is well-known and widely-used in various tasks and implementations. However, other PPDP measures such as *l*-Diversity and (*a*,*k*)-Anonymity can be used as well.

4.2.3 Calculating the Final Record Score

The Final Record Score (*RS*) uses the records' RSS_i and D_i, in order to assign a final score to all records in the table.

DEFINITION 6. Final Record Score. Given a table with r records, *RS* is calculated as follows:

$$RS = \max_{0\leq i\leq r}(RS_i) = \max_{0\leq i\leq r}\left(\frac{RRS_i}{D_i}\right)$$

For each record i, *RS* calculate the weighted sensitivity score RS_i by dividing the record's sensitivity score (RRS_i) by its distinguishing factor (D_i). This ensures that as the record's distinguishing factor increases (i.e., it is harder to identify the record in the reference table) the weighted sensitivity score decreases. The *RS* of the table is the maximal weighted sensitivity score.

For example, the *RS* score of Table 1b is calculated as follows:

$$RS(1b) = max\left(\frac{1}{2},\frac{1}{2},\frac{0.8}{3},\frac{0.4}{2},\frac{0.2}{1},\frac{0.2}{3}\right) = \frac{1}{2}$$

4.2.4 Calculating the M-Score

Finally, the *M*-score measure of a table combines the sensitivity level of the records defined by *RS* and the quantity factor (the number of records in the published table, denoted by r). In the final step of calculating the *M*-score, we use a settable parameter x ($x \geq 1$). This parameter sets the importance of the quantity factor within the table's final *M*-score. The higher we set x, the lower the effect of the quantity factor on the final *M*-score.

DEFINITION 7. *M*-Score. Given a table with r records, the table's *M*-score is calculated as follows:

$$\boldsymbol{MScore} = r^{1/x} \times RS = r^{1/x} \times \max_{0\leq i\leq r}\left(\frac{RRS_i}{D_i}\right)$$

where r is the number of records in the table, x is a given parameter and *RS* is the final Record Score presented in Definition 6.

For example, for $x = 2 \Rightarrow 1/x = 1/2$, the *M*-score of Table 1b is, *M*-score(1b) = $\sqrt{6} \times 0.5 = 1.224$.

The derived *M*-score value is not bounded. Thus, it is difficult to understand the meaning of the derived value and in particular the level of threat that is reflected by the *M*-score value. Therefore, we propose the following procedure for normalizing the *M*-score to the range [0,1]. Assume that T is the published table which is derived by applying the selection operator on the source table S, given a set of conditions, and then the projection operator: $T=\Pi_{a1,a2,\ldots,an}(\sigma_{\text{condition}}(S))$. Let T^* be the projection on a_1, a_2, … , a_n on the source table: $T^*= \Pi_{a1,a2,\ldots,an}(S)$. The *M*-score of table T can be normalized by dividing the *M*-score of T by the *M*-score of T^*: Norm*M*-Score(T)=*M*-Score(T)/*M*-Score(T^*).

4.3 The *M*-Score Properties

In this section we present two interesting properties of the *M*-score measure.

4.3.1 Monotonic Increasing

When calculating the *M*-score of two tables, where one is a sub-set of the other, the *M*-score of the super-set table is equal to or greater than the one of the sub-set tables.

Claim 1. *Let T_1, T_2 tables. If $T_1 \subseteq T_2$, then M-score(T_1) ≤ M-score(T_2).*

Proof. Let, r_i be the number of records in T_i (i.e., $r_i = |T_i|$); $\max_i$ the final record score of T_i (i.e., $\max_i = RS(T_i)$); and m_i the first record in T_i where $\max_i = RS_{m_i}$.

Since $T_1 \subseteq T_2$, we know that $r_1 \leq r_2$.

If $m_2 \in T_1$, then $\max_2 = \max_1$ and therefore,

$M\text{-score}(T_2) = r_2^{1/x} \times \max_1 \geq r_1^{1/x} \times \max_1 = M\text{-score}(T_1)$

Else, if $m_2 \notin T_1$, then $\max_2 \geq \max_1$ (Otherwise, $\max_2 < \max_1$ which is a contradiction to the definition of $\max_2$ - the maximum value of all records in T_2).

Therefore,

$M\text{-score}(T_2) = r_2^{1/x} \times \max_2 \geq r1^{1/x} \times \max_1 = M\text{-score}(T_1)$ □

4.3.2 M-score of union of tables

When calculating *M*-score with x=1, then the *M*-score of the union table (*Bag Algebra* union) is equal to or greater than the sum of *M*-scores of two tables with the same attributes.

Claim 2. *Let $T_1(A_1,\ldots,A_n)$, $T_2(B_1,\ldots,B_n)$ tables, where $\forall i$, $A_i=B_i$. If x=1, then M-score($T_1 \cup T_2$)≥M-score(T_1)+M-score(T_2).*

Proof. Let, r_i be the number of records in T_i (i.e., $r_i = |T_i|$); $\max_i$ the final record score of T_i (i.e., $\max_i = RS(T_i)$); and max' the maximal between $\max_1$ and $\max_2$. Then,

$M\text{-score}(T_1) + M\text{-score}(T_2) = r_1 \times \max_1 + r_2 \times \max_2 \leq$

$r_1 \times \max' + r_2 \times \max' = (r_1 + r_2) \times \max' = M\text{-score}(T_1 \cup T_2)$ □

This property suggests that in order to avoid detection while obtaining a large amount of sensitive data, the user has to work harder and must obtain the data piece by piece, a small portion at a time. Otherwise, *M*-score would rank the actions with a high misuseability weight.

4.4 Complexity Analysis

In this section we analyze the complexity of the *M*-score computation. For this purpose, we denote r to be the number of records in the published table and n the number of records in the source table.

Claim 3. *The computational complexity of the M-score calculation of a given table is $O(r \times n)$.*

Proof. The computational complexity of the *M*-score calculation is mainly affected by three factors: the *raw record score* of each record (RRS_i); the *distinguishing factor* of each record (D_i); and the *final record score* (RS).

To calculate RRS_i, the *sensitivity score function* needs to be calculated for each sensitive attribute's value. Given a sensitivity score function that maps each triplet of (context × sensitive attribute × value) to a score (as presented in definition 4), and under the assumption that the number of contexts, attributes and attributes sets of possible values in the source table are constant, the calculation of RRS_i is $O(1)$ (summing up the sensitivity scores of each sensitive attribute of record i).

To calculate D_i for a record i, each of the quasi-identifier values needs to be counted in the source table. Since we assume that the number of quasi-identifier attributes is constant the calculation of D_i is $O(n)$ since we need to compare the record's quasi-identifier with each of the records in the source table.

Therefore, the calculation of RS_i, the record weighted sensitivity score, for all the records in the published table is $O(r \times n)$.

RS is calculated by finding the record with the maximal RS_i, and therefore is $O(r)$.

Consequently, the computational complexity of the *M*-score calculation is $O(r \times n)$ □

Nonetheless, the calculation of the *M*-score can actually be done in $O(r)$, if the quasi-identifier values in the source table are preprocessed and counted in advanced, so that extracting the distinguishing factor of a quasi-identifier can be done in $O(1)$.

5 Illustration

In this section we test the *M*-score as a misuseability weight measure, and illustrate, using different scenarios, how the *M*-score addresses each misuseability dimension that we defined.

Scenario 1 – Publishing data on more entities

The *number of entities* dimension can highly affect the misuseability of a data table. Therefore, the *M*-score is incorporating the quantity factor in its calculation (denoted by r). However, as stated, in different domains there can be varying definitions about what constitutes a massive leak. In some cases, even a few records of data containing information about a highly important entity are regarded as a big risk. For others, the information itself is secondary to the amount of data that was lost.

In light of these considerations, *M*-score uses the settable parameter x for adjusting the effect of the table size on the final score. There are three possible settings for x: (1) If the organization wants to detect users who are exposed to a vast amount of data and regards the sensitivity of the data as less important, x can be set to 1; (2) If there is little interest in the quantity and only users who were exposed to highly sensitive data are being sought, then $x \to \infty$; (3) In all other cases, x can be set to represent the tradeoff between these two factors.

The illustration in Table 2 presents the *M*-score values for x=1, 2 and 100 (as an approximation of infinity) of two identical queries that differ only in the amount of returned customer records. The table shows that when increasing the value of x, the difference between the *M*-scores of the two queries becomes less significant.

TABLE 2
M-SCORE RESULTS FOR LARGE DATA WITH RESPECT TO X

Query	x = 1	x = 2	x = 100
Select top 5,000 "Business" customers	1166.500	16.497	0.254
Select top 500 "Business" customers	116.650	5.217	0.248

Scenario 2 – Reveal the specific entities

The *anonymity level* dimension is also addressed by the *M*-score measure, by taking into consideration the distinguishing factor, since the calculation of the *M*-score gives less sensitivity weight to records that are harder to identify. For example, a table that shows only the customer's city will be ranked with a lower *M*-score than the same table if we were to add the user's name to it. In other words, since knowing only costumer's city is significantly less useful in order to fully identify her, the distinguishing factor will reflect this status.

Scenario 3 - Exposing more properties

The sensitivity factor incorporated in the *M*-score is the way it addresses both the *number of properties* and the *values of properties* dimensions. Usually, exposing more details means that more harm can be inflicted on the organization. If the details also reveal that an entity in the data is a valuable one, the risk is even higher. Definition 5 showed us that the *M*-score considers all the different sensitive attributes. To illustrate this, we consider Tables 3a and 3b that show data about the same customer. However, while the latter shows only the customer's average monthly bill, the former also adds his account type. Calculating their score results in *M*-score(3a) = min(1, 0.3+0.5) = 0.8, and *M*-score(3b) = min(1, 0.3) = 0.3. As expected, *M*-score(3b), which expose less details, is lower.

The calculation of the *M*-score also considers the specific value of each sensitive attribute. If the average monthly bill on Table 3b was 'white', which is less sensitive then 'bronze', than *M*-score(3b) = 0.1.

TABLE 3
EXAMPLE OF COLLECTING MORE DETAILS ON A CUSTOMER

(A) TWO SENSITIVE ATTRIBUTES

First Name	Last Name	Account Type	Average Monthly Bill
Anton	Richter	Bronze	$450

(B) ONE SENSITIVE ATTRIBUTES

First Name	Last Name	Average Monthly Bill
Anton	Richter	Bronze

6 EXTENDING THE *M*-SCORE

Until now, we were describing how the *M*-score can measure the misuseability weight of a single publication, without considering the information the user already has; i.e., "prior knowledge". Prior knowledge can be: (1) previous publications (previous data tables the user was already exposed to); and (2) knowledge on the definition of the publication (e.g., the user can see the WHERE clause of the SQL query). In this section we extend the *M*-score basic definition and address these issues.

6.1 Multiple publications

A malicious insider can gain valuable information from accumulated publications by executing a series of requests. The result of each request possibly revealing information about new entities, or enriching the details of entities already known to her. Here, we focus on the case where the user can uniquely identify each entity (e.g., customer) in the result-set, i.e., the distinguishing factor is equal to 1 (D_i=1). We leave the case of publications with D_i>1 to future work. Fig. 3 depicts nine optional cases resulting from two fully identifiable sequential publications. Each case is determined by the relation (equal, overlapping or distinct) between the two publications with respect to the publications' sensitive attributes (marked in shades of green) and the exposed entities which are the distinct identifier values (marked in red). For example, in case 1 on Fig. 3, the publications share the same schema (i.e., include the same attributes in all tuples), but have no common entities; case 6 presents two publications that share some of the entities, but each publication holds different attributes on them.

Based on these nine possible cases we introduce the *Construct Publication Ensemble* procedure (Fig. 4) that constructs an ensemble set E on which the *M*-score should be calculated, where $<T_1, \dots, T_{n-1}>$ are the previous publications; T_n is the current (new) publication; and F is the time frame in which we still consider previous publication. By calculating the *M*-score of the ensemble set E, we actually consider the relevant prior knowledge the user has so far.

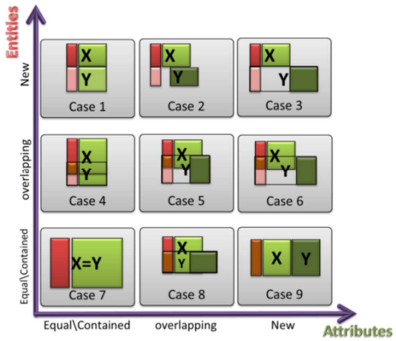

Fig. 3. Nine cases resulting from two fully identifiable publications

The *Construct Publication Ensemble* procedure is recursive. For each new publication, the procedure first creates an ensemble set X of all the previous publications that are within the time frame F (lines 5 to 7). Then, the procedure checks which case in Fig. 3 fits the current publications and acts accordingly (lines 8 to 16). Finally, on line 17 the resulting ensemble set is returned.

Construct Publication Ensemble **(<$T_1, \dots, T_{n-1}$>, T_n, F)**

1. **START**
2. **IF** $n = 1$ [1]
3. **THEN** return T_n
4. **ELSE**
5. ***RemoveOldPublications***(<$T_1, \dots, T_{n-1}$>,F) [2]
6. X ← ***ConstructPublicationEnsemble***(<$T_1, \dots, T_{n-2}$>, T_{n-1},F)
7. Y ← T_n
8. **IF** X∩Y = Y **(case 7 in Fig. 2)**
9. **THEN** E ← Y
10. **ELSE**
11. **IF** Entities(X) ∩ Entities(Y) = ∅ [3] **(cases 1-3 in Fig. 2)**
12. **THEN** E ← X ∪ Y
13. **ELSE**
14. **IF** SensAttr(X) = SensAttr(Y) [4] **(case 4 in Fig. 2)**
15. **THEN** E ← X ∪ Y-X
16. **ELSE** E ← X JOIN Y **(cases 5,6,8 and 9 in Fig. 2)**
17. return E
18. **END**

(1) No previous publications exist.
(2) Removes previous publications that are out of the time frame and shouldn't be considered
(3) Entities(X) are all the *values* of the identifiers attributes in set X.
(4) SensAttr (X) are all the *sensitive attributes* of set X.

Fig. 4. The Construct Publication Ensemble procedure

6.2 Multi-relational schema

In this subsection we address the scenario of multi-relational schema in which more than one table is re-

leased. In particular following Nergiz et al. [18] we assume that we are given a multi-relational schema that consists of a set of tables $T_1,\ldots T_n$, and one main table PT, where each tuple corresponds to a single entity (for example in Table 1 the main entity is the customer). The joined table JT is defined as $JT=PT \bowtie T_1 \bowtie \ldots \bowtie T_n$. Note that the quasi-identifier set can span across various tables, namely the "quasi-identifier set for a schema is the set of attributes in JT that can be used to externally link or identify a given tuple in PT" [18].

The various ingredients of the M-score can be calculated on an individually basis by using JT. For each entity i in PT we calculate the RRS_i by summing the scores of all sensitive values that appear in all her records in JT after eliminating duplicate values (for example if there are two records in JT that correspond to the same customer and each one of these records redundantly indicate that the customer is living in NY, then the sensitive score for the city NY will be counted only once). The D_i for an entity should be calculated by first calculating the D_i for each record in JT as described in section 4.2.2 (Note that Nergiz et al. [18] explain how k-anonymity is calculated in case of multiple-relations). Then, the entity's D_i is set to the minimum among all her records' D_i in JT. Finally the M-score is calculated using Definition 7.

6.3 Knowledge on request definition

A user may have additional knowledge on the data she receives emanating from knowing the structure of the request created this data, such as the request's constraints. In such cases, the basic M-score does not consider such knowledge. For example, a user might submit the following request: "select 'Name' of customers with 'Account type'='gold'" In this case, the user knows that all customers are 'gold' customers. However, since the result-set of this request will only include the names, the M-score cannot correctly compute its misuseability weight. In order to extend the M-score to consider this type of prior knowledge, RES(R) and COND(R) operators are defined.

DEFINITION 8. RES(R) = $\{A_1,\ldots,A_n\}$ is the set of attributes in the table that was retrieved following the request R.

DEFINITION 9. COND(R) = $\{S_1, \ldots,S_m\}$ is the set of sensitive attributes that are included in request R constraints, such that RES(R)∩COND(R)=Ø. The calculated sensitivity value of attribute S_j is denoted by COND(R)$_j$

For example, in the request R="*select 'Job' and 'Average monthly bill' of customers with 'Account Type' = BRONZE, 'Average monthly bill' > 100 AND 'City' = NY*", RES(R) = {'Job', 'Average Monthly Bill'}. The constraints' attributes of R is the set {'Account type', 'Average monthly bill', City}. However, since 'Average monthly bill'∈RES(R) and 'City' is not a sensitive attribute, COND(R) = {Account Type}.

COND(R)$_j$ can be calculated according to a different strategies depending on the attribute type (i.e., discrete or continuous). Some of the possible strategies are presented using the following example:

$\boldsymbol{R_{exp1}}$ = select the first and last name of all customers with Account type = Bronze or Silver

$\boldsymbol{R_{exp2}}$ = select the first and last name of all customers with Average monthly bill between 100 to 300

Sensitivity maxima: In case of discrete attributes, the COND(R_{exp1})$_j$ can be set to be the maximal value returning from the sensitivity score function, from all possible values specified in R_{exp} condition. In our example, the possible values of 'Account type' are Bronze or Silver. If we use the sensitivity score function presented in Fig. 2, then f(Account type[Silver]) = 0.7 > f(Account type[Bronze]) = 0.3. So, COND(R_{exp1})$_j$ should be set to be 0.7

Weighted value: According to this strategy, the value COND(R_{exp1})$_j$ of a discrete attribute is set according to the relative frequency in the source table of each possible value in R_{exp} constraint, with respect to the other possible values. For example, if in the source table the Account Type of α records are Bronze and of β records Silver, then COND(R_{exp1})$_j$ = f(AccountType[Bronze])×($\alpha/\alpha+\beta$)+f(AccountType[Silver])×($\beta/\alpha+\beta$) = 0.3×($\alpha/\alpha+\beta$) + 0.7×($\beta/\alpha+\beta$) .

Strategies for continuous attributes: To set the value of COND(R_{exp2})$_j$, the strategy might be to take either the average value in the condition value range (e.g., 200 in R_{exp2} case), or the maximal sensitivity score function given from the lower or the higher bound of the given range.

In the basic definition of the M-score, M-score(R) is given by ranking the values of RES(R) in the result table. In order to extend the measure to also consider prior knowledge, a change in the definition of the *Raw Record Score* should be introduced so it will also consider the values of COND(R) attributes.

DEFINITION 10. Extended Raw Record Score.

$$RRS_i = min\left(1, \sum_{S_j \in \mathrm{RES}(R)} f(c, S_j[x_i]) + \sum_{S_j \in \mathrm{COND}(R)} \mathrm{COND}(R)_j\right)$$

For a record i the extended RRS_i is the sum of all sensitive values scores in that record plus the sum of all the values given by COND(R)$_j$ with the maximum score of 1.

7 APPLICATIONS OF THE *M*-SCORE

In this section, four interesting applications of the M-score are presented: using the M-score as an *access control* mechanism, using it to *improve existing detection* methods, using it as the base of an *anomaly detection method* or, using it to implement a proactive misuseability reduction mechanism.

7.1 Dynamic Misuseability-Based Access Control

We propose using the M-score as the basis for a new mandatory access control mechanism for relational databases. The MAC mechanism regulates user access to data according to predefined classifications of both the user (the *subject*) and the data (the *object*) [19]. The classification is based on partially ordered access classes (e.g., top secret, secret, confidential, unclassified). Both the objects and the subjects are labeled with one of these classes, and the permission is granted by comparing the subject access class to that of the object in question. Basic MAC implementations for relational databases partition the database records to sub-sets, with each sub-set holding all records

with the same access class. According to the proposed method, the *M*-score is used for dynamically assigning an "access class" to a given set of records (i.e., a table).

The new proposed access control mechanism, which we call *Dynamic Misuseability-Based Access Control* (*DMBAC*), can be used to regulate user access to sensitive data stored in relational databases; it is an extension of the basic MAC mechanism.

The DMBAC is enforced as follows. First, each user is assigned with a "misuseability clearance", i.e., the maximal *M*-score that this subject is eligible to access. Then, for each query that a user submits, the *M*-score of the returned result-set is calculated. The derived *M*-score, which represents the *dynamic access class* of that result-set, is compared with the misuseability clearance of the subject in order to decide whether she is entitled to access the data she is requesting. Note that similar to the basic MAC, the DMBAC can be enforced in addition to existing access control layers such as role-based or discretionary access control.

The DMBAC approach presents several advantages over the basic MAC mechanism. First, as opposed to the finite number of access classes in MAC, in DMBAC there can be an infinite number of dynamic access classes, allowing more flexibility and fine-grained access control enforcement. Second, while manual labeling of tuples is required in MAC, in DMBAC, once the sensitivity score function is acquired, every result-set can be labeled automatically. Third, the dynamic approach enables the access control mechanism to derive a context-based access label, for example, the amount of tuples that were exposed or the data that the subject already possesses (using the extensions presented in Section 6). Last, while in the basic MAC subjects are only permitted to write to objects with access class higher or equal to their own (to prevent exposure of data to unauthorized subjects), in DMBAC the access class is assigned dynamically and therefore subjects are not limited in their writing.

The proposed DMBAC mechanism can operate in the following two modes: *binary* and *subset disclosure*. In the *binary mode*, if the misuseability clearance of the subject is lower than the *M*-score of the result-set, no data will be presented at all. In the *subset disclosure mode*, a subset of the result-set might be presented to the user. The subset of records can be selected, for example, by iteratively removing the most sensitive record from the result-set and exploiting the fact that the *M*-score is greatly affected by its score. Doing so will eventually create a subset whose *M*-score is lower than or equal to the subject's misuseability clearance.

Note, however, that assigning a clearance level for each user or role is a challenging task. It is challenging in the "classic" MAC model where users are assigned with a discrete clearance level (e.g., "top secret", "secret") and it is even more challenging in our case where the clearance level is a numeric value. We think that an iterative process, which assigns an initial clearance and refines this value a long time, as well as machine learning and statistical methods for assigning a clearance level, can all be suggested and explored in future work.

7.2 *M*-score-based Anomaly Detection

A different usage scenario arises in implementing *M*-score-based anomaly detection. During the learning phase, the normal behavior of each user or role is extracted. The normal behavior represents the sensitivity level of the data to which users are exposed, during their regular activity within different contexts (e.g., time of day, location). During the detection phase, the *M*-score of each action is computed and validated against the behavioral model that was derived in the learning phase. A significant deviation from the normal behavior (i.e., access to data with a sensitivity level significantly higher than the data that the user normally accesses) will trigger an alert.

7.3 Dynamic Threshold Mechanism

The *M*-score measure can be used for improving the detection performance of existing detection mechanisms. Detection mechanisms are usually set with a predefined threshold such that the IT manager is notified about incidents with an alert level that exceeds this threshold. Normally, the threshold is set only according to a static set of user's features (e.g., her role). The *M*-score can be used for implementing a dynamic, context-based, threshold that is higher when only low-sensitive data is involved and lower when the data is highly sensitive. This enables the IT manager to focus on potential misuse incidents that involve more sensitive data.

7.4 Proactive Misuseability Reduction

In some cases, data misuse can be prevented or contained by altering the data so that its sensitivity\misuseability level is reduced. However, at the same time, the modified data should still be useful for performing the desired tasks. Therefore, we propose a proactive mechanism that consists of the following three elements: (1) measuring the potential risk from exposure of the data; (2) reducing that potential risk, possibly by altering the data; and (3) measuring the utility of the altered data and its usefulness for performing the task.

As a misuseability weight measure, the *M*-score can be used to estimate the expected risk of exposing the data.

In order to reduce the misuseability score, we can reduce the *M*-score quality factor (e.g., by exposing fewer sensitive attributes); reduce the quantity factor (e.g., by removing some of the records), or, increase its distinguishing factor. Increasing the distinguishing factor means increasing the anonymity of the entities presented in the table; for example, by using the *personalized privacy preservation* method proposed by Xiao and Tao [20]. The personalized privacy preservation method uses *generalization* [21] in order to increase the anonymity of a given table. This can be done, for example, by replacing quasi-identifiers' values (e.g., replacing age=21 with age$\in$[20,30]) and/or blurring the sensitive attribute using a taxonomy of the sensitive values (e.g., replacing disease='dyspepsia' with disease='stomach disease'). While the basic definition of generalization suggests equally generalizing the table's records, Xiao and Tao [20] suggested consulting the data owner and specifying for each record the required "degree of privacy", thereby avoiding over-generalizing rec-

ords (e.g., when a patient has a simple flu) and under-generalizing (e.g., generalizing 'dyspepsia' to 'stomach disease' is not sufficient and should be generalized into 'some disease'). In our case, the organization can use the "personalized" mechanism to define how to generalize its data. For example, according to the sensitive attributes in Fig. 1, the organization can decide to generalize only records involving 'gold' type costumers.

An organization would like to reduce its risk by generalizing the dataset to a sufficient level while at the same time allowing its employees to use the data. Therefore, we propose using a *data utility metric* in order to prevent "over generalizing" of the data [22]. Several data utility metrics have been proposed; some are very simple and intuitive, such as generalization counting [4], while others are more complex and address the intended use of the data (e.g., classification or regression) [23]. The specific metric should be chosen according to the organization's needs, and a proper threshold should be defined.

8 Eliciting Misuseability Conceptions

In this section, we present an experiment we conducted. The main target of this experiment was to check if the *M*-score fulfills its target of measuring misuseability weight. In addition, one of the main challenges in applying the *M*-score is acquiring knowledge required for deriving the sensitivity score function. Acquiring such a function is a challenging task, especially in domains with large number of attributes, each with many possible values. Then, the function must be able to score many possible combinations. Consequently, we propose and evaluate two approaches for acquiring the domain expert knowledge necessary for deriving the sensitivity score function.

8.1 Eliciting the Score Function

In each of the two approaches presented here, we asked the domain expert to describe her expertise by filling out a relatively short questionnaire. The goal was twofold: to "capture" simply and quickly the relevant knowledge of the domain expert and to collect enough data to extract the expert's intentions. Using this captured knowledge, we then derive the scoring model (the sensitivity score function) by using different methods. This section presents the different approaches for acquiring the knowledge and the methods that can be used in order to extract the function from the collected data.

8.1.1 Records Ranking

In this approach, the domain expert is requested to assign a sensitivity score to *individual* records. Thus, the domain expert expresses the sensitivity level of different combinations of sensitive values. Fig. 5 depicts an example of assigning a sensitivity score (in the range 0 to 100) to 4 records, each with two sensitive attributes.

Account type	Customer group	Score
Gold	Business	100
Silver	Business	90
Bronze	Private	45
White	Private	0

Fig. 5. Scored records example

Once the expert has finished ranking the records, a model generalizing the scored record-set is derived. This model should be able to assign a sensitivity score to any given record, even if the combination of values in it did not exist in the record-set ranked by the user beforehand.

There are two challenges when applying the records ranking method: (1) choosing a record-set that will make it possible to derive a general model that will be as small and compact as possible (since it is not possible to rank all records in the database); and (2) choosing an algorithm for deriving the scoring model.

The first challenge can be addressed in several ways, such as choosing the most frequent records that appear in the database. In our experiment, we used the *Orthogonal Arrays* method [24] that is usually utilized for reducing the number of cases necessary for regression testing while maximizing the coverage of all sets of *n* combinations.

Tackling the second challenge is a bit more complicated because many different methods, each with its pros and cons, can be chosen for building a knowledge-model from a list of ranked records. One of the most prominent differences between methods is the functional dependencies among the attributes, and therefore, to derive the function, we examined two different, complementary methods: *linear regression model* and *CART model*.

Linear regression model. Linear regression is a well-known statistical method that fits a linear model describing the relationship between a set of attributes and the dependent variables. The regression model is trained on labeled records that include different combinations of the sensitive attribute values (including "blank" as a legal value indicating that the value is unknown). Considering as many different combinations as possible for attribute values in the learning process allows the model to better reflect the real sensitivity function. We can regard the problem of finding the *M*-score sensitivity score function like that of fitting a linear function, and use the sensitivity score given by the domain expert, as shown in Fig. 5. Fig. 6 illustrates a simple regression model trained of record similar to Fig. 5.

$$\begin{aligned}\text{Score} = {} & 61.88 \\ & + 42.5 \times \text{Account.type[gold]} \\ & + 30 \times \text{Account.type[silver]} \\ & - 35 \times \text{Account.type[white]} \\ & - 18.75 \times \text{Customer.group[Private]}\end{aligned}$$

Fig. 6. Linear regression model

CART model. CART (Classification and Regression Tree) [25] is a learning method that uses a tree-like structure in which each split of the tree is a logical if-else condition that considers the value of a specific attribute. In the leaves, CART uses a regression to predict the dependent variable. The tree structure is used because no assumption is made that the relationships between the attributes and the dependent variable are linear. This is the main difference between CART and the linear regression method. For the evaluation of our experiment, we use R rPart [26] implementation of ANOVA-trees, which is a CART-like model. Fig. 7 illustrates an rPart ANOVA-tree created with a dataset similar to Fig. 5. For each split in the tree, the right branch means the condition is true. The left

branch indicates that the condition is false.

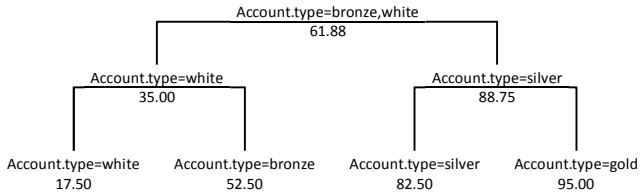

Fig. 7. rPart output of CART-like regression tree.

In both methods, the prediction model is built according to the data collected with the record ranking approach. Then, the sensitivity score function is deduced using the prediction for a given new record. We refer to these methods as Records Ranking[LR] (linear regression based on record ranking) and Records Ranking[CART] (CART based on record ranking).

8.1.2 Pairwise Comparison

People usually best express their opinion about a subject by comparisons with other subjects, rather than expressing it solely on the given subject, with nothing to compare to [27]. Therefore, pairwise comparison might help the domain expert to better describe the sensitivity level of an attribute value by allowing him to compare it to different values. In the pairwise comparison approach, the domain expert is required to *compare pairs* of sensitive attributes and pairs of possible values of a specific attribute. Fig. 8 presents the comparison of two sensitive attributes, and then the comparison of the optional values of the customer group attribute.

In order to derive the scoring model, we chose the *analytic hierarchy process* (AHP) [28] that is used for deducing preferences based on sets of pairwise comparisons. AHP is a decision support tool for handling multi-level problems that can be presented as a tree of chosen values. Using the pairwise comparisons data it weights the importance of each of the values with respect to the other possible values on the same level. Then, the importance of a path in the tree can be extracted by multiplying the weights of the different weights in it.

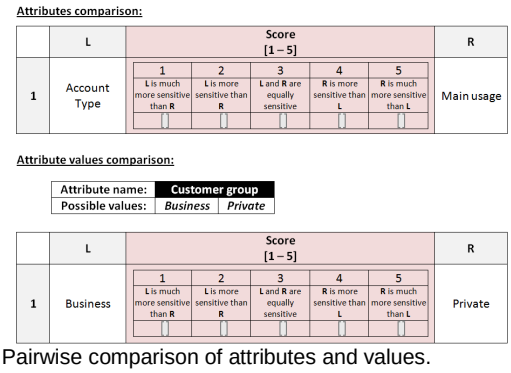

Fig. 8. Pairwise comparison of attributes and values.

In our case, we defined the problem of finding the sensitivity score function as a 3-level AHP problem (see Fig. 9). The top level defines the problem of finding the weight of a given sensitive attribute value. Having only one option, this level has a single node with a weight of 1. The next level includes the sensitive attributes (e.g., Account type, Customer group). The AHP tree leaves define the possible values of the sensitive attribute (e.g., gold, business). We suggest using pairwise comparisons in which the expert is asked to first compare each possible pair of attributes and then the possible pairs of values of the same attribute. This makes it possible to learn the weight of each node. Then, in order to extract the sensitivity score function, we simply look at the weight of the path to each value as its sensitivity. For example, using the AHP-tree in Fig. 9, if we want to infer the sensitivity of Account type silver, we simply need to calculate:
weight(Finding sensitive value weight) × weight(Account type) × weight(silver) = 1 × 0.75 × 0.3 = 0.225

Extracting the sensitivity score function with this method results in relative scores, – the sensitivity scores of all leaves of the tree (i.e., the actual values) sum up to 1. Thus, this method, unlike the other methods presented previously, enables the expert to directly define which values are more important than others. We refer to this method as Pairwise Comparison[AHP].

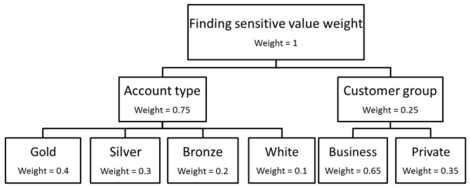

Fig. 9. Example of AHP-tree with 3 levels.

8.2 Experiment Description

In the experiment, we attempted to answer the following research questions:

1. Does the *M*-score fulfill its goal of weighting the misuseability weight of tables of data?
2. Which method (Records Ranking[LR], Records Ranking[CART] or Pairwise Comparison[AHP]) creates the knowledge-model that calculates the sensitivity score function which best fits the scores given by the domain expert?
3. Which approach (record ranking or pairwise comparisons) allows the expert to give sufficient information for creating good models within the shortest period of time?
4. Which approach do the domain experts prefer?
5. Is it feasible to derive a model that can rank the sensitivity of data records using a domain expert's knowledge?

In this section we first explain the course of the experiment and present the questionnaire that was used. Then, we elaborate upon the way different parts of the questionnaire were used in the experiment. Finally, we show the results of using each of the methods described above. For simplicity, we conduct the experiment as if a single context exists. We believe that the methods that we present can be easily extended to deal with multiple contexts (i.e., by acquiring the data from the experts with respect to the context and creating a model-per-context). This issue, however, is left for future work.

8.2.1 Experiment questionnaire

In order to conduct the experiment, we designed a four-part questionnaire. The first two parts of the questionnaire (A and B) were utilized to acquire knowledge from

the experts using the two approaches presented in section 8.1 (record ranking and pairwise comparison). The last two parts of the questionnaire (C and D) were used for evaluating the quality of the knowledge-model created.

Since domain experts from Deutsche Telekom were to answer the questionnaire, we used the customer records from the cellular phone service domain, as presented in Fig. 1. The attributes "days to contract expiration" and "monthly average bill" were discretized by the experts according to predefined ranges.

In part A of the questionnaire, records containing one of the different possible values of each sensitive attribute are presented. In each record, there were also "blanks" in some of the attributes, indicating unknown values. The records in this part were selected by using the *Orthogonal Arrays* method and covering all 3-way possibilities (all combinations of 3 different values of all the attributes). The participant was asked to rank the sensitivity level of each of the given records on a scale of 0 to 100 (similar to Fig. 5). Using the ranked records, knowledge models were derived using both the Records Ranking[LR] and Records Ranking[CART] methods.

In part B, pairs <L , R> of sensitive attributes or their values were presented to the participant who was asked to decide which of the two possibilities is more sensitive on a scale of 1 (L is much more sensitive than R) to 5 (R is much more sensitive than L) as shown in Fig. 8. This scale was chosen according to psychological studies, which have shown that it is best to use discrete scales of 7±2, depending on the granularity level needed in the decision [29]. With the data acquired from this part, we extracted the Pairwise Comparison[AHP] sensitivity score function.

In both parts A and B, the time required for completing the questions was measured. In addition, the participant was asked to rank which part was more difficult to complete on a scale of 1 (A was much more difficult) to 5 (B was much more difficult).

Part C of the questionnaire included a list of tables containing both customer identifiers and sensitive data. Each table contained a different subset of attributes from the set of sensitive and identifying attributes on Fig. 1. The participant was asked to assign a sensitivity rank between 0 and 100 to each of the tables (see an example on Fig. 10).

First name	Last name	Customer group
Georg	Beckenbauer	Business
Mareike	Noelter	Business
Lars	Schmidt	Business

Rank:

Fig. 10. Example of table ranking

In the last part of the questionnaire (part D) pairs of tables, such as the table shown in Fig. 10, were presented to the participant who was asked to decide which of the two tables is more sensitive.

8.2.2 Measurements

To address research question 2 we analyzed 10 questionnaires that Deutsche Telekom security experts completed. For the analysis, we used parts C and D in each questionnaire to evaluate the different sensitivity score functions created using the data collected in parts A and B.

First, the tables from part C were ranked with the different *M*-scores extracted from the three sensitivity score functions. We will refer to them as *M*-score-LR (for the *M*-score that is calculated using the Records Ranking[LR] model); *M*-score-CART (using Records Ranking[CART]); and *M*-score-AHP. Then, using these ranks and the ranks that were given by the expert to each table, we constructed four vectors (M-score-LR$_i$, M-score-CART$_i$, M-score-AHP$_i$ and Expert-score$_i$, respectively, where i represents the specific expert). The vectors were sorted according to the sensitivity of the tables, from the least sensitive table to the most sensitive one. Finally, using the *Kendall Tau* measure [30] we compared each of the *M*-score vectors to the Expert-score$_i$ vector. The *Kendall Tau* measure is a known statistic test for ranking the similarity of ordering of vector coefficients. It assigns ranks in the range [-1,1], when -1 indicates that one vector is the reverse order of the other and 1 indicates the vectors are identical. Consequently, in our case we would like to have ranks as close to 1 as possible.

In order to measure the accuracy of each of the methods, we used the comparisons from part D. First, as in part C, we calculated each table's vectors. Then, using these calculated *M*-scores, each comparison was "classified" to one of three classes: L (i.e., left table is more sensitive); R (right table is more sensitive); or E (the tables are equally sensitive). Finally, using the class given by the expert in the questionnaire, the classification accuracy of each *M*-score was measured.

8.2.3 Experimental Results

In this part, we elaborate on the empirical results of our experiment, as presented in Tables 4 and 5.

Derived score function accuracy

Table 4a depicts the Kendall Tau measure results of the correlation between pairs of the Expert-score$_i$ vector and each of the other *M*-score vectors. On all tests, the p-value indicates that the correlation was statistically significant. From the table it can be seen that the Pairwise Comparison[AHP] model gave the best results (τ=0.512, which means the vectors are 75.64% correlated), followed by the Records Ranking[LR] (0.488, 74.43%).

Table 4b depicts the results of the classification accuracy. While analyzing the results, we encountered many situations where the expert classified a pair as class E (both tables are equally sensitive). However, the calculated tables *M*-scores were very close, but were not exactly equal. Thus, many pairs that could actually be considered as class E were classified as R or L. Therefore, in addition to the regular accuracy calculation that appears in Table 4b, we added an "extended accuracy" that also considers comparisons when the difference between the calculated *M*-scores of both tables is insignificant. These pairs were counted with a lower weight (0.5) in the extended accuracy calculation. (That is, if a pair was not classified as E, but the calculated *M*-scores of the tables in this pair were very close, we added 0.5 to the accuracy numerator). From the table we can see that the Records Ranking[LR] model was the most accurate (average accuracy=0.69, avg. extended accuracy=0.8), followed by the Pairwise Comparison[AHP] (0.66 and 0.77, respectively).

From the results, we cannot give a clear answer about which method created the best sensitivity score function. It can be noted, though, that both Records Ranking[LR] and Pairwise Comparison[AHP] significantly outperformed the Records Ranking[CART].

TABLE 4
EXPERIMENT RESULTS ON EXPERT VECTORS
(A) KENDALL TAU RESULTS ON EXPERT VECTORS

Vectors	Average τ (% correlation)	p-value
Expert-score$_i$: M-score-LR$_i$	0.488 (74.43%)	< 0.01
Expert-score$_i$: M-score-CART$_i$	0.477 (73.86%)	< 0.01
Expert-score$_i$: M-score-AHP$_i$	0.512 (75.64%)	< 0.01

(B) ACCURACY OF THE METHODS

Method	Average Accuracy	Average Extended Accuracy
Regression	0.69	0.8
CART	0.57	0.67
AHP	0.66	0.77

Completion time and preferred approach

As was stated above, we measured the time each participant took to complete parts A (record ranking) and B (pairwise comparisons). The results showed that the time to complete part A (25 minutes on average) was considerably longer than the time to complete part B (7 minutes on average). *Paired two sample for means T-test* showed that these results are statistically significant ($P(T \le t) < 0.01$).

After participants had finished both parts A and B, we asked each to express an opinion about which part was harder to complete on a range of 1 (part B was much harder) to 5 (part A was much harder). Of the 10 participants, 8 responded with a 5; and the average of the responses was 4.7. We assume that this fact is strongly connected to the fact that completing part A took more than 3 times longer than part B.

Correlation between experts

In addition to the Kendall Tau that we calculated for each expert, as shown in Table 4a, we also computed the Kendall Tau on pairs of vectors across experts. For each type of vector - *M*-score-LR$_i$, *M*-score-CART$_i$, *M*-score-AHP$_i$ and Expert-score$_i$ - we matched pairs of the same vector type across all experts. Table 5 presents the average results for that experiment, and the average for all the methods vectors (i.e., excluding the Expert-score$_i$). Interestingly, the correlation between the different expert ranks (τ=0.391) is significantly lower than the models correlation (0.797 in average).

TABLE 5
AVERAGE KENDALL TAU BETWEEN ALL PAIRS *I*,*K* OF EXPERTS

Vectors	Average τ (% correlation)	p-value
Expert-score$_i$: Expert-score$_k$	0.391 (69.55%)	< 0.05
M-score-LR$_i$: M-score-LR$_k$	0.812 (90.65%)	< 0.05
M-score-CART$_i$: M-score-CART$_k$	0.839 (91.95%)	< 0.05
M-score-AHP$_i$: M-score-AHP$_k$	0.74 (87.02%)	< 0.05
Average of models' vectors (M-score-LR$_i$, M-score-CART$_i$, M-score-AHP$_i$)	0.797 (89.87%)	< 0.05

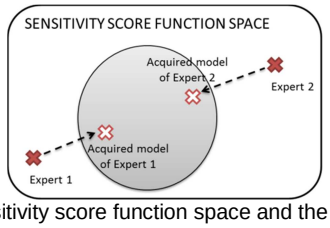

Fig. 11. The sensitivity score function space and the model subspace

This fact might be explained by Fig. 11. The figure illustrates the space of all possible sensitivity score functions, when the marks of **Expert 1** and **Expert 2** show the specific functions of these experts in the space. Since the methods we propose for deriving expert functions are rather simple, they are limited to creating only a subset of all possible score functions, which are only a sub-space (as illustrated by the circle in the figure). Thus, the sensitivity score functions of different experts, as reflected by the models (shown in the figure as **Acquired model of Expert 1** and **Acquired model of Expert 2**), are closer to each other and therefore have a stronger correlation. This fact is very important for our purposes because it allows us to assume that using knowledge acquired from one expert is sufficient to calculate the *M*-score for the entire domain.

8.3 Experiment Conclusions

The main goal of this experiment was to find whether the *M*-score fulfills its goal of measuring misuseability weight (research question 1). To test this, we examined the top three most sensitive tables for every Expert-score$_i$, *M*-score-LR$_i$, *M*-score-CART$_i$ and *M*-score-AHP$_i$ vectors, for each expert *i*. on 95.55% of the cases, the same three tables were the most sensitive for all the vectors of the same expert. Another important observation is that when we examined the instances of disagreement between the calculated *M*-score and the expert's score in regard to analyzing the cases that lowered either the Kendall Tau or the accuracy, we discovered that in many cases the intention of the expert was better expressed in the calculation and not by her own score. For example, in one case one expert clearly expressed on the pairwise comparison (part B) that the 'data' value of the Main usage attribute was much more sensitive than all the other values. However, in part D, the expert indicated that customer records with different Main usage values are more sensitive. After showing the expert this case, he indeed admitted being wrong in his answer on part D. We believe that directly ranking tables as a method of creating misuseability measure, although possible, is much more cognitively complicated and would be much less accurate.

Another goal was to find the best approach for acquiring the sensitivity score function, in terms of accuracy, time and experts' preferences (research questions 2, 3 and 4). The results show that the accuracy and correlation of the Records Ranking[CART] were significantly lower than the other two methods. On the other hand, both Records Ranking[LR] and Pairwise Comparison[AHP] presented good results that were fairly equal. The main drawback of the record ranking approach was the time required for acquiring the necessary data. In addition, participants strongly rejected this approach (possibly, as a result of the time factor). Although the Pairwise Comparison[AHP] model results do not suffer from these drawbacks, it lacks the ability to handle non-discrete attributes. In our experiment, for example, we had to discretize two attributes, and if we had used only the records ranking methods, we could have left these attributes non-discrete and given the expert concrete values to rank rather than fixed ranges. To conclude, if the domain contains only

discrete attributes, or if a set of fixed ranges to the non-discrete attributes can be defined, the Pairwise Comparison[AHP] is the preferred approach to eliciting expert knowledge and deriving the sensitivity score. If discrete attributes cannot be discretized, then the use of Records Ranking[LR] to create the knowledge-model is preferred. Our last goal (research question 5) was to understand whether it is feasible to derive a knowledge-model that can be used as sensitivity score function by collecting a reasonable amount of expert data. Our experiment shows that deriving such model is indeed feasible and that the expert only needs to make a relatively small effort to supply enough data. To emphasize the feasibility of learning the expert's knowledge, we can compare the presented results to a base line of randomly ranked records (in part A), and randomly compared attributes\values (in part B). Random behavior brings the Kendall Tau values to approximately 0 (~50% correlation), as opposed to a ~75% correlation using expert knowledge. The extended accuracy of the random approach stands at approximately 0.55, while with expert knowledge it can reach 0.8.

9 Conclusions and Future Work

We introduced a new concept of misuseability weight and discussed the importance of measuring the sensitivity level of the data that an insider is exposed to. We defined four dimensions that a misuseability weight measure must consider. To the best of our knowledge and based on the literature survey we conducted, there is no previously proposed method for estimating the potential harm that might be caused by leaked or misused data while considering important dimensions of the nature of the exposed data. Consequently, a new misuseability measure, the *M*-score, was proposed. We extended the *M*-score basic definition to consider prior knowledge the user might have and presented four applications using the extended definition. Finally, we explored different approaches for efficiently acquiring the knowledge required for computing the *M*-score, and showed that the *M*-score is both feasible and can fulfill its main goals.

Two important issues, which relate to the knowledge elicitation and representation, should be further investigated: the temporal aspect of the *M*-score and the validity of the knowledge, acquired from the experts, over time; and the knowledge acquisition that might be subjective and not consistent among different experts which, in turn, may lead to an inaccurate sensitivity function.

In regards to the time factor, we assumed that the sensitivity level of an attribute's value will change in rare cases and especially the order of the values with respect to their sensitivity level. For example, the value of a gold customer will not change and will remain more sensitive than a silver customer. A customer's type may change from gold to silver and this will be reflected when computing the *M*-score of the customer's record. However, we are aware of the need to validate and re-acquire the knowledge from time-to-time, and although we showed in the experiments that the knowledge can be acquired accurately with relatively minimal effort (in terms of experts time) using the pairwise comparison approach, we plan to explore methods for incremental learning, or post-learning fine tuning of the elicited sensitivity score function in future work.

With respect to the subjectivity of the elicited scoring function, our experiments indicate that the methods used ensure that the acquired knowledge is not biased. In fact we showed that using knowledge acquired from one expert is sufficient in order to calculate sound *M*-scores for the entire domain. We plan to further investigate this important issue and check the effect of combining knowledge from several experts (e.g., ensemble of knowledge models) on the quality of the acquired knowledge and the accuracy of the *M*-score. In addition, in some cases the value of customers can be calculated by using known knowledge on the customer (e.g., how much she spends) and by predicting future revenue from the customer. In such cases, the sensitivity level of sensitive attributes can be objectively obtained by using machine learning techniques; in particular by fitting the sensitive parameter values to the customer value [31].

We also plan to extend the *M*-score to support multiple publications with D_i>1, and the sensitivity of combinations of sensitive values; evaluate the measure using data from different domains, such as patient medical records, and using multiple contexts; and investigating other misuseability weight measures that are designed to handle data in formats other than tabular data.

References

[1] 2010 CyberSecurity Watch Survey, http://www.cert.org/archive/pdf/ecrimesummary10.pdf

[2] A. Kamra, E. Terzi, and E. Bertino, "Detecting Anomalous Access Patterns in Relational Databases," *International Journal on Very Large Databases*, 17(5):1063-1077. 2008.

[3] S. Mathew, M. Petropoulos, H. Q. Ngo, and S. Upadhyaya, "Data-Centric Approach to Insider Attack Detection in Database Systems," *Recent Advances in Intrusion Detection*, 2010.

[4] L. Sweeney, "k-Anonymity: a model for protecting privacy," *International Journal on Uncertainty, Fuzziness and Knowledge Based Systems*, 10(5):571-588, 2002.

[5] A. Machanavajjhala, et al., "l-diversity: Privacy beyond k-anonymity," *ACM Trans. on Knowledge Discovery from Data*, 1(1), 2007.

[6] R. C. Wong, L. Jiuyong, A. W. Fu and W. Ke, "(α,k)-Anonymity: An Enhanced k-Anonymity Model for Privacy-Preserving Data Publishing," *Knowledge Discovery and Data Mining*, 2006.

[7] E. Celikel, et al., "A risk management approach to RBAC," *Risk and Decision Analysis*, 1(2):21–33, 2009.

[8] B. Carminati, E. Ferrari, J. Cao, and K. Lee Tan, "A framework to enforce access control over data streams," *ACM Trans. on Information Systems Security*, 13(3), 2010.

[9] Q. Yaseen, and B. Panda, "Knowledge Acquisition and Insider Threat Prediction in Relational Database Systems," *Computational Science and Engineering*, pp. 450-455, 2009.

[10] G. B. Magklaras and S. M. Furnell, "Insider Threat Prediction Tool: Evaluating the probability of IT misuse," *Computers & Security*, 21(1):62-73, 2002

[11] M. Bishop and C. Gates. "Defining the insider threat," *Cyber Security and Information Intelligence Research*, 1-3, 2008

[12] C. M. Fung, K. Wang, R. Chen, and P. S. Yu, "Privacy-preserving data publishing: A survey on recent developments," *ACM Computing Surveys*, 42(4), 2010.

[13] A. Friedman, and A. Schuster, "Data Mining with Differential

Privacy," *Knowledge Discovery and Data Mining*, 493-502, 2010.
[14] C. Dwork, "Differential Privacy: A Survey of Results," *Theory and Applications of Models of Computation*, 1-19, 2008.
[15] T. Dalenius, "Finding a Needle in a Haystack or Identifying Anonymous Census Records," *Journal of Official Statistics*, 2(3):329-336, 1986.
[16] B. Berendt, O. Günther, and S. Spiekermann, "Privacy in e-commerce: stated preferences vs. actual behavior," *Comm. of the ACM*, 48(4):101-106, 2005.
[17] A. Barth, A. Datta, J. C. Mitchell and H. Nissenbaum, "Privacy and Contextual Integrity: Framework and Applications," *IEEE Symposium on Security and Privacy*, 184-198, 2006.
[18] M.E. Nergiz, et al. "Multirelational k-Anonymity," *IEEE Trans. on Knowledge and Data Engineering*, 21(8):1104-1117, 2009.
[19] E. Bertino, and R. Sandhu. "Database Security-Concepts, Approaches, and Challenges," *IEEE Trans. on Dependable and Secure Computing*, 2(1):2-19, 2005
[20] X. Xiao, and Y. Tao, "Personalized Privacy Preservation," *ACM Conference on Management of Data*, 229-240, 2006.
[21] L. Sweeney, "Achieving *k*-Anonymity Privacy Protection Using Generalization and Suppression," *Inter. Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):571-588, 2002.
[22] Y. Yuan, et al. "Evolution of Privacy-Preserving Data Publishing," *Anti Counterfeiting Security and Identification*, 34-37, 2011.
[23] K. LeFevre, D. DeWitt, and R. Ramakrishnan, "Workload-Aware Anonymity," *International Conference on Knowledge Discovery and Data Mining*, 277-286, 2006.
[24] A. S. Hedayat, N. J. A. Sloane, and J. Stufken, *Orthogonal Arrays - Theory and Applications*. New York: Springer-Verlag, 1999.
[25] L. Breiman, et al., *Classification and Regression Trees*. Monterey, Calif.: Wadsworth and Brooks, 1984.
[26] R Development Core Team. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. http://www.R-project.org. 2010.
[27] T. Satty, "A scaling method for priorities in hierarchical structures," *Journal of mathematical psychology*, 125:234-281, 1977.
[28] T. Satty, *Multicriteria Decision Making*, McGraw-Hill, 1980.
[29] G. A. Miller, "The magical number seven, plus or minus two: Some limits on our capacity for processing information," *The Psychological Review*, 63(2):81-97, 1956.
[30] M. Lapata, "Automatic evaluation of information ordering: Kendall's tau," *Comput. Linguist.*, 32(4):471-484, 2006.
[31] S. Rosset, E. Neumann, U. Eick, N. Vatnik, and Y. Idan, "Customer lifetime value modeling and its use for customer retention planning," *Knowledge Discovery and Data Mining*, 2002.