Constraints, Inference Channels and Secure Databases

Alexander Brodsky, Csilla Farkas, Duminda Wijesekera, and X. Sean Wang

Department of Information and Software Engineering,
George Mason University, Fairfax, VA 22030-4444.
{brodsky,cfarkas,duminda,xywang}@gmu.edu

Abstract. This paper investigates the problem of confidentiality violations via illegal data inferences that occur when arithmetic constraints are combined with non-confidential numeric data to infer confidential information. The database is represented as a point in an $(n+k)$-dimensional constraint space, where n is the number of numerical data items stored in the database (extensional database) and k is the number of derivable attributes (intensional database). Database constraints over both extensional and intensional databases form an $(n+k)$-dimensional constraint object. A query answer over a data item x is an interval I of values along the x axis of the database such that I is correct (i.e., the actual data value is within I) and safe (i.e., users cannot infer which point within I is the actual data value). The security requirements are expressed by the accuracy with which users are allowed to disclose data items. More specifically, we develop two classification methods: (1) volume-based classification, where the entire volume of the disclosed constraint object that contains the data item is considered and (2) interval based classification, where the length of the interval that contains the data item is considered. We develop correct and safe inference algorithms for both cases.

Contact Author: Csilla Farkas, Mail Stop 4A4, George Mason University, Fairfax, VA 22030-4444. Telephone: 703–993–1629, Fax: 703–993-1638, Internet: cfarkas@gmu.edu

1 Introduction

Databases that contain confidential information require that users can access - directly or indirectly - only data for which they have the proper authorization. Direct data accesses are usually controlled by some form of lattice-based (mandatory) access control mechanisms [BL75,Den76]. Each data request is evaluated by comparing the security level of the requested data with the security clearance of the user. If the security requirements are satisfied (i.e., the user's security clearance dominates the security classification of all data) the data access is permitted. It is rejected otherwise. However, lattice-based access control is insufficient to prevent secrecy violations via inferences when non-confidential data is combined with meta-data to derive confidential information. Moreover, the strict security requirements of mandatory access control may unnecessarily limit data availability. In a number of applications it is desirable to provide a range of valid answers that would still allow the users to perform their jobs with limited authority while preserving confidentiality. For example, military support services should be able to perform scheduled maintenance work without knowing the mission (e.g., destination) of the serviced equipment. This paper provides a framework to define *flexible security requirements* on data and presents algorithms that compute *correct* and *safe* answers to queries. Intuitively correctness means that the actual data item is contained in the returned interval, and safety means that the user is not able to infer anything beyond what is released.

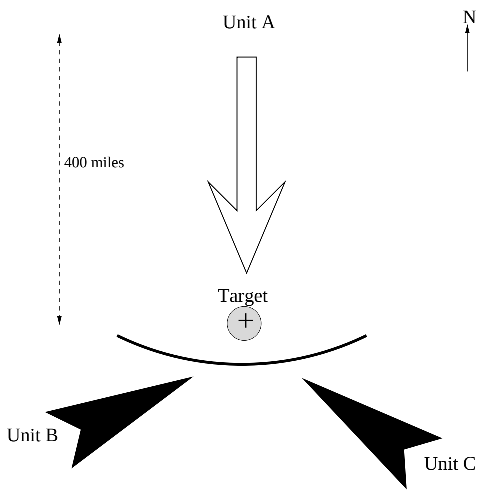

Fig. 1. Military Attack Plan

Most indirect data accesses result from inferences combine non-confidential data with constraints. We illustrate an illegal inference using a military database that contains data about different military units, such as types and specifications of their vehicles, destination distance, fuel and food supply (see Table 1). Assume that the attack strategy is that units **B** and **C** from the south engage in a long-term attack with the main force of the enemy; while, unit **A**, that is equipped with light weapons and thus able to move fast, is summoned from the north to launch a surprise attack on the unprotected northern front of the enemy. Fig. 1 shows the geographical locations of the units and the planned attacks. It is necessary that only authorized officers of the command center and of the units **A**, **B** and **C** know about the surprise attack. However, keeping the military plan secret does not guarantee the secrecy of this information. Consider the situation when an enemy spy is able to gain information about the amount of fuel requested by the different units. Since the number of the vehicles of the units and the mileages of these vehicles are usually known, the amount of available fuel defines the maximal travel distance and, therefore the potential destination of units. In our example, the knowledge of the destination of unit **A** would reveal the attack plan, thus reducing the strategic advantage of surprise.

How can we prevent such inferences? One possible solution is to keep fuel volume secret. However, this would severely limit data availability and require that all users accessing this information must be authorized to access information about the military plan as well. In this paper we present a different approach where users are allowed to access a range of possible data values but not the precise value. This flexible solution allows us to preserve confidentiality and provides improved data availability compared to the previous approaches.

The inference problem in databases first was considered in statistical databases where the security requirement is that aggregate information about groups of individuals can be released but no specific information about an individual should be accessed (see Denning [Den82] for survey). Illegal inferences revealing private information are based on the size of the database used to compute the statistics (e.g., number of individuals is a group) and/or the overlap among the queries requesting related information. However, researchers do not consider inferences raised by combining meta-data with non-confidential data to disclose confidential information. Since the beginning of 1980s, researchers, focusing on multi-level secure relational databases, identified the problem of indirect access to confidential data via combining meta-data with non-confidential data [GM84,Mor88,SO87,Hin88,Smi90,Buc90,Den85,Thu87,MSS88,RJHS95,ST90]. However, these techniques often result in over-classification of data and, therefore reduce data availability. Moreover, most authors, with the exception of [Den85,Hin88,SO91,DdVS99a,DdVS99b], do not consider the problem of actual inference for specific families of constraints; rather they develop a framework assuming that disclosure inference algorithms are readily available. It is our view, however, that the main technical difficulty of solving the inference channel problem lies in developing inference algorithms that guarantee data confidentiality.

Finally, the existing inference prevention techniques, with the exception of [RJHS95], are based on withholding answers from the users. We propose a new approach where confidentiality in numeric databases is achieved by controlling the accuracy of the released data. Our model guarantees that no illegal data access is possible even in the presence of database dependencies while supporting maximal data availability. The work of Rath et. al [RJHS95] is the closest to ours. They consider functional dependencies (FDs) to raise illegal imprecise inferences in databases containing numeric values. While their approach is secure, it is limited in the sense of considering FDs only and does not define security requirements in the context of range values.

In this paper we consider a data model consisting of a set of variables. A database is an instantiation of values to the variables that satisfies the database (arithmetic) constraints. The database is represented as a single point in an $(n+k)$-dimensional space, where n is the number of data items stored in the database (extensional database) and k is the number of values that can be derived from the stored data (intensional database).[1] The arithmetic constraints over extensional and intensional data form an $(n+k)$-dimensional constraint object. When a user requests a data value a range of possible values, represented as two new constraints, are generated and returned to the user.

The security requirements on the data items are expressed by the accuracy with which users are allowed to know the data values. To the best of our knowledge, this is the first paper that introduces a security model in which the classification of the data items is based on the precision of numeric answers and thus continuous. We propose two methods to assign security classification to the data items: first, the entire *volume* of the constraint object that contains the data item is considered; and second, the length of the *interval* that contains the data item is considered. While volume-based classification is easier to implement than the interval-based one, the second technique is more flexible and it allows higher data availability. For both methods we develop algorithms that generate query answers which are *correct* (i.e., the data item is within the returned interval) and *safe* (i.e., the user is unable to infer which point within the answer is the actual data item).

The organization of the paper is as follows. In Section 2 we give a detailed description of the considered problem and provide intuitive explanations of the concepts involved. In Section 3 we formally define the considered model and the security requirements. Sections 4 and 5 contain the algorithms to generate correct and safe query answers for volume and interval based classification methods, respectively. Finally, in Section 6 we conclude and recommend new directions to extend our research.

2 Problem Formulation

This paper formalizes a security model applicable to numerical databases when a database instance is a vector of numerical values. Users and data items have security classifications. Users (subjects) can access data (objects) through queries.

[1] emNote, that the real underlying database may adhere to any existing model, e.g. relational or object-oriented.

Queries over numeric data return an interval of valid answers, such that the actual data is contained in the interval. The size (length) of the interval (i.e., *precision* of the released data) depends on the security classification of the requested data and the security clearance of the user. Precision of released information can be measured either by the length of the returned data interval or by the volume of the valid constraint space defined by this interval. The system presented in this paper ensures that the interval-based query answers do not allow the users to infer any unauthorized information. The research problem is formulated in terms of the following components: (1) *data model* of which the (numeric) instances consist of objects of discourse, (2) *arithmetic constraints* imposed upon the database, (3) *security classifications* of users and data items, and (4) *queries* requesting numeric values.

The next section provides an example that will be used throughout this paper for illustrative purposes.

2.1 The Running Example

Consider the military database containing information about units **A, B** and **C** mentioned in the Introduction. It contains information about the *number of vehicles, amount of fuel, destination distance, food supply, number of soldiers* etc, as shown in Table 1. Some of these attributes are such as *amount of fuel* are stored, while others such as the *travel range* is derivable. In addition, the database is expected to satisfy the following constraints.

$$\begin{aligned}
\text{number of soldiers} &\leq 10,000 \\
\text{amount of fuel} &\leq 20,000 (gallon) \\
\text{number of vehicles} &\geq 45 \\
3 \times \text{amount of fuel} &\geq \text{travel range (miles)}
\end{aligned}$$

Users of the database are generals, military planners, field commanders, field soldiers, supply supervisors, supply personnel, press and general public. Based on their rank and tasks, users have different access rights to data items. For example, generals are allowed to access all the data in the database while supply personnels are allowed to access information related to supply but can not access information about the field information of the units.

The queries in our model request numerical values, such as *amount of fuel* used by the military unit **A**. Based on the data confidentiality (security classification) the answer is a range of values. In calculating the returned range, we ensure that the user cannot infer other attribute ranges such as the *travel range* of the military unit **A** more precisely than allowed by the security model. Note that we assume that all constraints are known to all users revealing additional information to the users about valid database values.

3 Data Model, Queries and Security Requirements

3.1 Data Model

The data model consists of two sets of attributes of numerical type, and a set of constraints over them.

Database Each entity of our model consists of a set of n attributes that are directly stored in the data base. Furthermore, there is a set of k derivable attributes, not stored in the database. Consequently, every entity is viewed as a $(n+k)$-dimensional vector $(a_1, \ldots, a_n, a_{n+1} \ldots a_{n+k})$ of numerical type. We say that $(n+k)$ is the dimension of the data model. The set $x_1, \ldots, x_n, x_{n+1} \ldots x_{n+k}$ of variables represents attributes, and $x_1 \mid a_1, \ldots, x_n \mid a_n, x_{n+1} \mid a_{n+1}, \ldots, x_{n+k} \mid a_{n+k}$ represent instantiations of values $a_1, \ldots, a_{n+k}$ to the variables. In the running example the attributes (variables) are unit name, soldiers, food supply, fuel, number of vehicles, destination, fuel/vehicle, and travel range. The database is given in Table 1. As stated, the derivable attributes are not stored explicitly in the database, but can be computed from publically available statistics, and the values stored in the database. We use both derivable and explicitly stored attributes without distinction.

Stored Attributes						Derived Attributes	
Unit Name	Soldiers	Food Suppl. (days)	Fuel (gals)	Number Vehicl.	Destination (miles)	Fuel/Vehicl. (gals)	Travel Rng. (miles)
A	1000	10	15,000	75	400	200	600
B	500	18	7,000	140	85	50	150
C	100	25	3,000	45	67	60	180

Table 1. A Database of Military Operations

Arithmetic Constraints We consider conjunctions of arithmetic constraints of the following forms:

$$f(x_1, x_2, \ldots, x_n, x_{n+1}, \ldots, x_{n+k}) \geq b$$

$$f(x_1, x_2, \ldots, x_n, x_{n+1}, \ldots, x_{n+k}) > b$$

where f is a function, $x_1, \ldots, x_{n+k}$ are variables, and b is a real number.[2]

Let C be the conjunction of constraints of the above forms over variables $x_1, \ldots, x_n$, $x_{n+1}, \ldots, x_{n+k}$. C defines an $(n+k)$-dimensional *constraint object*, that is all possible instances of database entities that satisfy C.

[2] Note that the conjunction of a finite number of arithmetic constraints of the above form can express equalities as well.

Consider our military example again. Fig. 2 shows the valid constraint space (possible instances) of the database over attributes amount of fuel (per vehicle) and the destination. The point $A(400, 200)$ correspond to the instance of unit **A**, where the available fuel is 200 gallons for each vehicle and the destination distance is 400 miles. Clearly, $A(400, 200)$ satisfies the constraints.

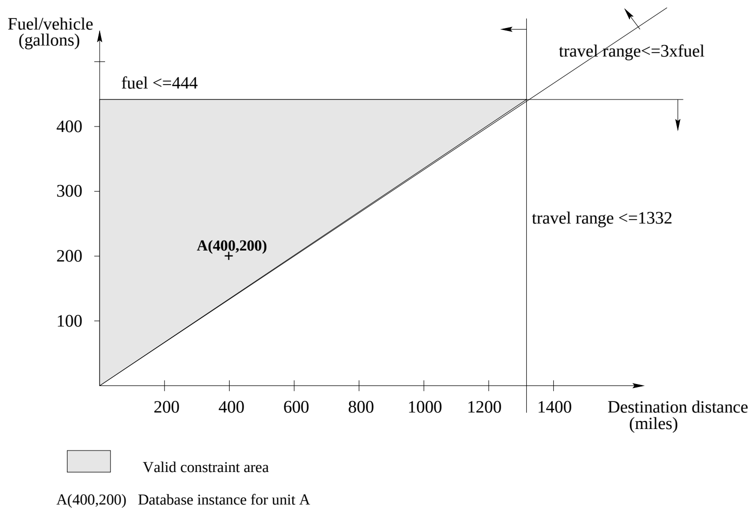

Fig. 2. Fuel and travel range data of the military database

Queries and Answers We consider *queries* where a user requests a particular data value of x_i, i.e., the coordinate (a_i) of the database along the x_i axis. The *answer* to the query is an interval $I = [a_{i_1}, a_{i_2}]$ of data values, such that $a_{i_1} \leq a_i \leq a_{i_2}$. The query answer can be viewed as two new constraints of the form $a_{i_1} \leq x_i$ and $x_i \leq a_{i_2}$. The *data revealed* by the answer of the query is the new constraint object C' defined by $C \wedge (a_{i_1} \leq x_i) \wedge (x_i \leq a_{i_2})$. C' is a restriction of the original constraint object C.

3.2 Security Model

Multi-level secure relational database systems contain data classified at difference security levels. Users of the database have security clearances assigned. Security classifications and clearances are expressed by security labels that contain two components:(1) a hierarchical component (e.g., top-secret < secret < unclassified) and (2) a non-hierarchical categories (e.g., { supply, press, field-info }). Security labels form a lattice structure with the *dominance* ($\leq$) among the labels. A data access is permitted if the security clearance label of the user dominates the security classification label of the requested data items. For example, in our model, the attribute *destination distance* of unit **A** could be classified as *(top-secret, field-info)* and only officers with security clearances that dominates *(top-secret, field-info)* are allowed to access this data, e.g., generals, with security clearance *(top-secret,{press,supply,field-info})*. However, supply officers with security clearance *(top-secret, supply)* are not allowed to access this data.

For simplicity, we assume that the subjects of our security model are the users.[3] To define the security objects of our model, it is necessary to understand what information a user gains if a query is answered. Let us give an intuitive explanations before formally defining the objects and their security classifications of our model.

Originally a user knows:
- The (n+k) -dimensional constraint object C.
- The data instances satisfy C.
- If C is bounded for a data value of x_i, then the value of x_i must be in the interval $[min(a_i), max(a_i)]$, where $min(a_i)$ and $max(a_i)$ are the smallest and largest possible values of x_i defined by C.

After Answering a Query that is an interval $[a_{i_1}, a_{i_2}]$ containing a_i, the value of x_i the user knows:
- C', the restriction of C by the two new constraints $a_{i_1} \leq x_i$ and $x_i \leq a_{i_2}$.
- The database is a point within C'.
- If C' is bounded for a data item x', than the value of x' must be in the interval $[a', a'']$, where a' and a'' are the smallest and largest possible values of x' defined by C'.

Intuitively, the size of the constraint object (or its projection) known by the user to contain the database represents how accurately the user knows the actual data value. The smaller the volume (projection), the more accurate the user's knowledge about the actual data values. We assume that all possible instances of an entity within a constraint space have the same probability to be the actual database, i.e., the database is *uniformly distributed* within the constraint object. We use the properties of the constraint object, such as volume, projection, to define the security requirements on the data items.

Definition 1. *(Volume and Projection of a constraint Object)*
Let C be an $(n+k)$-dimensional constraint object. The volume *of C is defined as the volume of the $(n+k)$-dimensional geometrical shape. The* projection *of C on axis x (data item x) is an interval $[a_1, a_2]$, where a_1 and a_2 are the minimal and maximal values along the x axis defined by C.*

In this papers we present two approaches to define the *security* of objects in our database. First, we use the *volume* of C as the security object. While this method is simple and can be easily implemented, it does not allow to assign different sensitivity levels to the different data items. [4] Second, we propose a

[3] Usually in security, a distinction is made between the users and the principals (subjects) acting on behalf of them.
[4] Same as database level of security granularity in relational databases.

security classification based on *projections* of C to represent the security restrictions on the individual data items. This method is enhanced by a probability measurement that incorporates the shape of the constraint object, which is not reflected in the projection. The second approach allows not only the assignment of different security classifications to the individual data items but also, to represent partial disclosure. The following sections contain the detailed descriptions of the two methods and algorithms that generate safe query answers according to the security restrictions.

4 Security by Volume

In this section we present a security model in which the absolute volumes of the disclosed (known by the user to contain the database) constraint objects are used to represent the sensitivity of the database.

Definition 2. *(Volume Objects)*
Let $l_1, \ldots, l_k$ be a set of security labels and C a constraint object with volume V. For each security label l_i, $i = 1, \ldots, k$ we create a volume object*, denoted by V_{l_i}, as follows:*

1. *$V_{l_i} \leq V$, i.e., the volume object must be smaller than the volume of the constraint object.*
2. *If $l_i \leq l_j$ then $V_{l_i} \geq V_{l_j}$, i.e., volume objects at lower security levels can not be smaller than volume objects at higher security level.*

The secrecy of the database is violated if a user with security clearance l_i is able to disclose a constraint object with volume less than V_{l_i}. We propose a security mechanism based on controlling the *volume* of the constraint object disclosed by the user. For this approach it is necessary that the original constraint object C is bounded from every direction.

To visualize the security classifications, consider Fig. 2. The shaded area defines the valid data values. Since knowing that the database must satisfy the constraints, the user knows that the available fuel per vehicle cannot exceed 444 gallons (maximum 20,000 gallons fuel available for each unit divided by the minimum number of vehicles 45 for each unit) and, therefore, the maximal travel distance (3 times the available gallons of fuel per vehicle) is less than or equal to 1332 miles. The actual value of available fuel for each vehicle of unit **A** is 200 gallons, and the destination distance is 400 miles. The total volume of the valid constraint space is 295,704.

Assume, that in addition to the constraints of the database, the user knows that the amount of fuel available per vehicle is between 150 and 300 gallons. Clearly, the user's knowledge about the fuel is more accurate now than before. The darkly shaded area of Fig. 3 shows the new constraint object C' after the addition of $150 \leq fuel/vehicle \leq 300$. The volume of C' is 101,250. The increase of the user's knowledge about the data value is reflected by the decrease of the size (volume) of constraint object. Observation of the reduced constraint

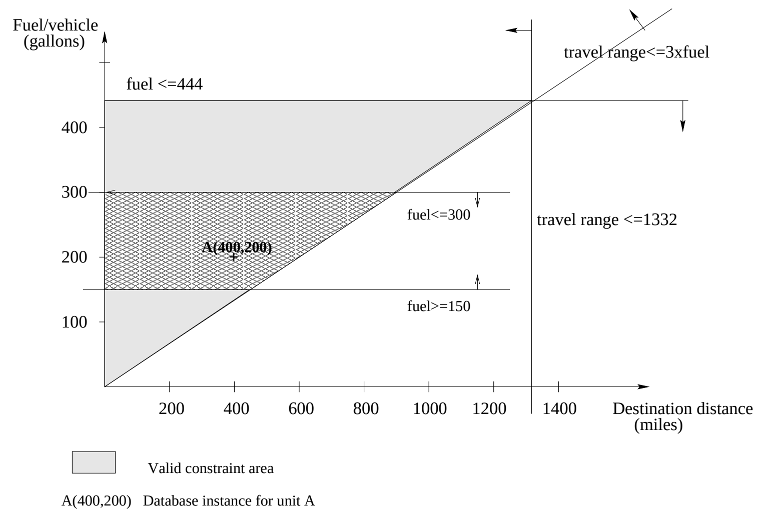

Fig. 3. Restricted constraint object

object indicates that the user also gained information about travel distance. The maximal travel distance cannot exceed 900 miles.

Definition 3. *(Correct, Safe and Efficient Query Answer by Volume)*
Let DB be the database, x the data item in DB requested by the user, l the user's security clearance, C the constraint object known by the user to contain the database, and V_l the volume object at security level l. A query answer $[a_1, a_2]$ is

1. Correct *if $a \in [a_1, a_2]$, i.e., the actual data value a of x is within the returned interval.*
2. Safe *if the query answer does not violate the security requirements, i.e., if C' is the new constraint object $C \wedge (a_1 \leq x) \wedge (x \leq a_2)$ with volume V' then:*
   (a) $V' \geq V_l$ and
   (b) DB is uniformly distributed in C', i.e., the user knows only that the data item satisfies C'.
3. Efficient *if given $l_i < l$ and V_{l_i} the volume object at security level l_i $V' \leq V_{l_i}$, i.e., released information is more accurate at higher security level.*

The algorithm given in Fig. 4 generates a correct, safe and efficient query answer. It uses a random number generator based on the uniform distribution. To ensure that the algorithm terminates, we use a threshold value 100,000 to limit the number of trials to generate correct, safe, and efficient query answer. If no such answer is found, the algorithm returns a default answer, that is the minimal and maximal valid values defined by the constraint object.

Theorem 1. *Algorithm 1 (1)* terminates *and returns a (2)* correct*, (3)* safe *and (4)* efficient *answer according to Definition 3.*

Proof. 1. *Termination:* straightforward by Step 3.
2. *Correctness:* By the construction of points a_1 and a_2 in Steps 3.b and 3.c, respectively, Algorithm 1 ensures that the data value a is within the generated interval.
3. *Safety:* (a) By comparing the volume of the restricted constraint object V' with the volume object V_l in Step 6, Algorithm 1 ensures that V' is not smaller than V_l. (b) By using a random number generator based on uniform distribution, any point within the restricted constraint structure C' has the same probability to be the database, i.e., the database is uniformly distributed within C'. Also, since any new information about a data item is fully contained in the previous answers thus no possibly hazardous overlap can occur.
4. *Efficiency:* Follows from step 7.II.a.

Returning back to our running example, assume that the answer given as a result of a query made by a user with clearance *(secret,press)* for the amount of fuel used by unit **A** is $[150, 300]$, as given by Fig. 3. Furthermore assume that the volume objet associated with this security level is $V_{(secret,press)} = 150,000$. Since the volume of the restricted constraint object as given is 101,250, the answer is correct, safe, and efficient (but not optimal).

5 Security by Interval

Similarly to the previous section, the information disclosed by a user is reflected in the *constraint object* by this users. However, in this section we explore a flexible security classification that is based on the projection of the constraint object on the data items. The proposed method allows the security officer to individually classify each data item.

First, let us give an intuitive example. Consider the military database we presented earlier and assume that the travel range is more sensitive than the amount of fuel, e.g., the enemy (public) is not allowed to deduce the travel range with accuracy of 200 miles, while fuel information is allowed for release within 50 gallons range, e.g., each vehicle has fuel between 180 and 230 gallons. However, by releasing this fuel information, the travel range is reduced to $690 - 540 = 150$ miles. While the security requirement is satisfied for fuel, it is clearly violated for travel range.

Moreover, it is possible, that the security officer would like to consider disclosures, where the data item is not fully disclosed within a given interval size but has high probability. Note, that we assumed that originally the database is uniformly distributed within the constraint object C. Because of this uniform distribution, the probability that the database is within a sub-object C' of C is proportion to the volumes of C' and C. For example, if the the volume of C is V and the volume of C' is V', the probability that the database is contained

Algorithm 1: Query Answer by Volume	
INPUT	• User's query Q requesting data item x • User's previous query answers $q_1, \ldots, q_m$ • User's security clearance l • Constraints object C known by the user to contain the DB • Database point DB • Volume objects $V_{l_1}, \ldots, V_{l_h}$
OUTPUT	• Answer to Q • Restricted constraint object C'

METHOD	1. **If** $q_1, \ldots, q_m$ contains an answer $[a'_1, a'_2]$ for data item x, **then**
	  (a) Return $[a'_1, a'_2]$ as the answer.
	  (b) *Generate* C' as $C' = C$.
	2. Initialization:
	  (a) Let a' be the smallest and a'' the largest possible value of x defined by C.
	  (b) $r = 0$
	3. **If** $r > 100,000$ **then** Return $[a', a'']$ as default answer.
	  Else *Generate* two points a_1 and a_2 as follows:
	  (a) Let a be the actual value of data item x
	  (b) *Randomly* pick a_1 from the interval $[a', a]$
	  (c) *Randomly* pick a_2 from the interval $[a, a'']$
	4. *Generate* two linear constraints:
	  (a) $a_1 \leq x$
	  (b) $x \leq a_2$
	5. *Generate* the new constraint object $C' = C \wedge (a_1 \leq x) \wedge (x \leq a_2)$
	6. *Calculate* volume V' of C'
	7. *Verify* security requirements:
	  I. **If** V' smaller than V_l (security requirement is violated) **then**
	  (a) $r = r + 1$ and return to 3
	  II. **Else** (security is not violated) **then**
	  (a) **If** $V_{l'} \leq V'$, where $V_{l'}$ is the volume object at security level l' such that $l' \leq l$ **then** (more accurate answer is possible)
	    $r = r + 1$ and return to 3
	  (b) **Else**
	    i. *Return* $[a_1, a_2]$ as the answer to the query
	    ii. *Store* C' as the new constraint object known by the user.

Fig. 4. Query Answer by Volume

in C' is V'/V. This observation allows the security officer to protect data from partial disclosures. For example, we can assign that users with unclassified security clearances are not allowed to know the travel distance within 100 miles with probability 85%.

Definition 4. *(Interval objects)*
Let $x_1, \ldots, x_{n+k}$ be a database, $l_1, \ldots, l_h$ a set of security labels and C a constraint object with volume V. For each data item x_i $i = 1, \ldots, n+k$ and for each security label l_j, $j = 1, \ldots, h$ we create an interval object, *denoted by (I_{i_j}, p_{i_j}), as follows:*

1. *I_{i_j} defines the smallest interval length (accuracy) with which users with security clearances l_j are allowed to know the value of x_i with probability p_{i_j}.*
2. *If $l_s \leq l_r$ and $p_{i_s} = p_{i_r}$ then $I_{i_s} \geq I_{i_r}$, i.e., interval size of an object at a low security level can not be smaller than the interval size of the same object at high security level.*
3. *If $l_s \leq l_r$ and $I_{i_s} = I_{i_r}$ then $p_{i_s} \leq p_{i_r}$, i.e., if two interval sizes of a data item are the same at different security levels, then the high security level can not have lower probability assigned to than the low security level.*

Intuitively, given a data item x and its interval object (I, p) at security level l, the secrecy of x is violated if a user with security clearance l is able to disclose a constraint object C such that C contains a sub-object C', with projection size on x smaller than or equal to I, that contains the database with probability higher than p.

Definition 5. *(Correct and Safe Query Answer - Interval)*
Let $x_1, \ldots, x_{n+k}$ be the database, x_i the data item in DB requested by the user, l the user's security clearance, C the constraint object known by the user to contain the database, and $(I_{1_l}, p_{1_l}), \ldots, (I_{n+k_l}, p_{n+k_l})$ the interval objects of data items $x_1, \ldots, x_i, \ldots, x_{n+k}$ at security level l. A query answer $[a_1, a_2]$ is correct and safe if

1. Correct*: $a \in [a_1, a_2]$, where a is the actual data value of x_i.*
2. Safe*: the query answer does not violate the security requirements, i.e., if C' is the new constraint object $C \wedge (a_1 \leq x_i) \wedge (x_i \leq a_2)$ with volume V' then*
   (a) *C' does not contain a sub-object C'_j with projection size I'_{j_l} on data item x_j such that $I'_{j_l} \leq I_{j_l}$ $j = 1, \ldots, n+k$ and volume V'_j, such that V'_j/V' greater than p_{j_l} and*
   (b) *DB is uniformly distributed in C', i.e., the user knows only that the database is any point within C'.*

We propose a security mechanism to control the *length of the answer interval* returned to the user ensure that the security requirements are satisfied. This approach does not require that the original constraint object is bounded.

Theorem 2. *Algorithm 2 (1)* terminates *and returns a (2)* correct *and (3)* safe *query answer according to Definition 5.*

Algorithm 2: Query Answer by Interval

INPUT

1. User's query requesting data item x
2. User's previous query answers $q_1, \ldots, q_m$
3. User's security clearance l
4. Constraints object C known by the user
5. Database point DB
6. Set S of Interval objects of the form (I_l, p_l)

OUTPUT

1. Answer to Q
2. Restricted constraint object C'

METHOD

1. **If** $q_1, \ldots, q_m$ contains an answer $[a'_1, a'_2]$ for x **then**
   (a) Return $[a'_1, a'_2]$ as the answer
   (b) Generate C' as $C' = C$
2. Initialization:
   (a) Let a' be the smallest and a'' the largest possible value of x defined by C.
   (b) $r = 0$
3. **If** $r > 100,000$ **then** Return $[a', a'']$ as default answer.
   Else *Generate* a random length l
4. Randomly *pick a point* a_1 in the interval $[a - l, a]$ (starting point of the answer), where a is the actual value of x in the database.
5. **If** $a_1 < a'$ **then** (security is violated) **then**
   $r = r + 1$ and return to Point 3.
6. **If** $r > 100,000$ **then** Return $[a', a'']$ as default answer.
   Else *Generate* a random length l'
7. **If** $l' \leq l$ (incorrect answer) or $l' < I_l$ (security is violated) **then**
   $r = r + 1$ and return to Point 6
   Else generate $a_2 = a_1 + l'$ (end point of the answer)
8. **If** $a_2 > a''$ **then** (security is violated) **then**
   $r = r + 1$ and return to Point 6.
9. *Generate* two linear constraints:
   (a) $a_1 \leq x$
   (b) $x \leq a_2$
10. *Generate* the new constraint object $C' = C \wedge (a_1 \leq x) \wedge (x \leq a_2)$
11. *Calculate* volume V' of C'
12. *Verify* security requirements for every data item y in the DB
    If there exist a sub-object C'_i of C' such that
    (a) $\mid C'_i \mid_y \leq I_{y_l}$, where $\mid C'_i \mid_y$ is the projection of C'_i on y and I_{y_l} is from $(I_{y_l}, p_{y_l},$ and
    (b) $V'_i / V' > p_{y_l}$
    then (security is violated) $r = r + 1$ and return to Point 3
13. **Else** (security is not violated)
    (a) *Return* $[a_1, a_2]$ as the answer to the query
    (b) *Store* C' as the new constraint object known by the user.

Fig. 5. Query Answer by Interval

Proof. 1. *Termination:* straightforward by Step 3 and 6.

2. *Correctness:* By the construction of lengths l (Step 3), l' (Step 6) and the points a_1 (Steps 4,5) and a_2 (Steps 7,8), Algorithm 2 ensures that the data value a is within the generated interval.
3. *Safety:* (a) By verifying for every data item that no disallowed *interval object* is disclosed (Step 12), Algorithm 2 ensures that the security requirements are not violated. Further, (b) by using a random number generator based on the uniform distribution, any point within the restricted constraint structure C' has the same probability to be the database, i.e., the database is uniformly distributed within C'. Similarly to the security classifications via volume objects, note that any new information about a data item is fully contained in the previous answers thus no possibly hazardous overlap can occur.

Note, that since we do not enforce any restriction on the correlation of interval size and probability we can't define efficient answer similarly to the volume based model. However, if one of these measurements are fixed, e.g., all interval objects given with 0.8 probability, we can define effective answer similarly to the volume based method.

6 Conclusions

We have presented a model to specify the accuracy with which users are allowed to access numeric values. As shown, restrictions can be enforced either on the volume of the disclosed constraint object that surrounds the protected data items, or on the interval of a specific data value. While volume based accuracy is easier to enforce than the interval based one, the later provides more flexibility, therefore improved data availability. We developed algorithms to compute correct and safe answers to queries for both measures of accuracy.

Finally, we conclude by recommending further research directions. Currently our algorithms provide an effectively accurate answer that is not optimal, i.e., the most accurate value allowed to the user. While we believe it is impossible to provide a general algorithm that generates safe, correct, and optimal query answer such a method may exist for restricted families of arithmetic constraints. Also, in this paper we considered queries that request the value of a single data item. Our work could be extended to incorporate queries requesting several data items simultaneously and incorporating selection conditions. Finally, the interval based accuracy can handle sets of sensitive intervals and probabilities for each data item. For example, a data item x may be accessed by a user only if none of the classifications $(I_{x_1}, p_{x_1}), \ldots, (I_{x_k}, p_{x_k})$ is violated.

References

[BL75] D.E. Bell and L.J. LaPadula. Secure computer systems: Mathematical foundation and model. Technical report, Mitre Corp. Report No. M74-244, Bedford, Mass., 1975.

[Buc90] L.J. Buczkowski. Database inference controller. In D.L. Spooner and C. Landwehr, editors, *Database Security III: Status and Prospects*, pages 311–322. North-Holland, Amsterdam, 1990.

[DdVS99a] S. Dawson, S. De Capitani di Vimercati, and P. Samarati. Minimal data upgrating to prevent inference and association attacks. In *Proc. of the 18th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 114–125, 1999.

[DdVS99b] S. Dawson, S. De Capitani di Vimercati, and P. Samarati. Specification and enforcement of classification and inference constraints. In *Proc. IEEE Symp. on Security and Privacy*, 1999.

[Den76] D.E. Denning. A lattice model of secure information flow. *Comm. ACM*, 19(5):236–243, May 1976.

[Den82] D.E. Denning. *Cryptography and Data Security.* Addison-Wesley, Mass., 1982.

[Den85] D.E. Denning. Commutative filters for reducing inference threats in multilevel database systems. In *Proc. IEEE Symp. on Security and Privacy*, pages 134–146, 1985.

[GM84] J.A. Goguen and J. Meseguer. Unwinding and inference control. In *Proc. IEEE Symp. on Security and Privacy*, pages 75–86, 1984.

[Hin88] T.H. Hinke. Inference aggregation detection in database management systems. In *Proc. IEEE Symp. on Security and Privacy*, pages 96–106, 1988.

[Mor88] M. Morgenstern. Controlling logical inference in multilevel database systems. In *Proc. IEEE Symp. on Security and Privacy*, pages 245–255, 1988.

[MSS88] S. Mazumdar, D. Stemple, and T. Sheard. Resolving the tension between integrity and security using a theorem prover. In *Proc. ACM Int'l Conf. Management of Data*, pages 233–242, 1988.

[RJHS95] S. Rath, D. Jones, J. Hale, and S. Shenoi. A tool for inference detection and knowledge discovery in databases. In *Proc. of the 9th IFIP WG11.3 Workshop on Database Security*, pages 317–332, 1995.

[Smi90] G.W. Smith. Modeling security-relevant data semantics. In *Proc. IEEE Symp. Research in Security and Privacy*, pages 384–391, 1990.

[SO87] T. Su and G. Ozsoyoglu. Data dependencies and inference control in multilevel relational database systems. In *Proc. IEEE Symp. Security and Privacy*, pages 202–211, 1987.

[SO91] T. Su and G. Ozsoyoglu. Inference in MLS database systems. *IEEE Trans. Knowledge and Data Eng.*, 3(4):474–485, December 1991.

[ST90] P.D. Stachour and B. Thuraisingham. Design of LDV: A multilevel secure relational database management system. *IEEE Trans. Knowledge and Data Eng.*, 2(2):190–209, June 1990.

[Thu87] B.M. Thuraisingham. Security checking in relational database management systems augmented with inference engines. *Computers and Security*, 6:479–492, 1987.