Neighborhood Data and Database Security

Kioumars Yazdanian, Frédéric Cuppens
e-mail: yaz@tls-cs.cert.fr - cuppens@tls-cs.cert.fr

CERT / ONERA, Dept. of Computer Science
2 avenue E. Belin, B.P. 4025, 31055 Toulouse Cédex (FRANCE)

Abstract

Data protection in a database involves inference controls which are sometimes of a semantic nature. The notion of neighborhood in a given context between two data is considered and a representation for context and neighborhood is proposed as well as a measurable definition for the neighborhood based on first order mathematical logic.

Introduction

A database is a representation of a subset of real-world knowledge.

Classical databases manage elementary data which are known facts, e.g:

- Paul is rich
- Paul has four brand new cars
- Paul has a big house
- Jane is fond of flowers
- Jane is fond of roses
- Jane has roses

are examples of elementary data.

Other kinds of database management systems are suited to represent and manage fuzzy data (data with unknown or probabilistic truth value) such as:

- Paul is probably rich
- Paul has a few brand new cars but I don't know how many.

In distributed databases, several local databases are connected through a network in order to share data and to provide answers to queries with respect to all data in all databases. The security problem of data disclosure is, therefore, more complex. Each local database has its particular schema and protection rules. Global protection rules can be defined on the distributed database conceptual schema restricting access to some data in one database depending on data accessible in other databases.

Merging several local databases schema into one global distributed database schema is technically feasible but as far as security is concerned, the problem of semantic relationships between data in different local databases remains.

This paper will present the basis for a database management system able to express the semantic similarity of two data. The "Neighborhood" terminology will be used to express such a similarity.

For instance, "My tailor has four brand new cars" can be considered to be similar to "My tailor is rich" and therefore, the first datum is considered to be a neighbor of the second.

This concept of neighborhood introduces a new paradigm in security policy domain. Up till now data protection has been enforced on given and well-defined data or a set of data called the data grain. The concept of granularity could be extended to define larger "grains" of data depending on new rules involving more or less similar data.

In the security domain it is interesting to express the semantic proximity or neighborhoo d of data by which the knowledge of one data implies more or less the knowledge of the other. This is considered a semantic covert channel or a data neighborhood n-deduction as will be defined in the following sections.

In both cases however neither the covert channel nor the n-deduction provide the precise data expected by the user but some indication. This will be the basic difference between classical deduction and neighborhood deduction.

This paper is organized in six sections.

With respect to the relational model, two kinds of neighborhood can be defined; they are presented in the first section.

An important application of the neighborhood concept in the security domain is the semantic covert channel which is introduced in the second section.

In some cases, a datum could be considered as a neighbor of another and in some other cases not. Therefore the definition of "context" is given in the third section.

The fourth and fifth sections propose a formalism for context and neighborhood representation using the relational model for data representation. Database formalization using First order Logic, either in the proof theoretic approach or in the model approach are considered as known ([4], [6]).

The last section shows how a measure of neighborhood can be defined.

©1993 ACM 0-89791-635-2 $1.50

Upper-case letters used as an argument of an n-ary relation or predicate will denote tuples of degree less than or equal to n, and lower-case letters will denote one argument. Generally, letters at the beginning of the alphabet will denote constant (fully instantiated tuples) and letters at the end of the alphabet will denote variables if not otherwise stated.

I - The two kinds of neighborhood

In relational databases, information is represented by tuples in a relation. A tuple is a combination of elements belonging to "domains" which are sets of elements. Usually each element is considered as an attribute value. For instance, 30 is an element of the domain of ages, 2000 is an element of the domain of salaries, Clerk is an element of the domain of jobs.

One kind of neighborhood of data occurs when two data are similar because they have the same structure (for instance two tuples of the same relation) and their only difference is a difference of domain values which could be considered as neighbors (for instance two tuples having a "salary" attribute values which are respectively 2000 and 2010 could therefore be considered as similar). In this case a neighborhood of values implies a neighborhood of data and the concept of neighborhood of value has to be defined in a domain.

The other kind of neighborhood involves two different data with different representations, but similar semantics. For instance "Paul has four brand new cars" represented by a set of tuples in the relation

PERSON-CAR(name,car#,model,date-of-purchase,price)

can be considered as a neighbor of "Paul is rich" which is a tuple of the relation

STATUS(name,status)

In classical Database, an answer to a query is a set of data represented by tuples which satisfy the constraints expressed in the query expression. These tuples can be considered as the exact solution of the problem the query expresses. In some cases, it can be useful to also provide information close to the constraints of the query but not exactly matching them. This was the initial idea in defining neighborhoods of data.

For instance, someone asking for a group of young people interested in football would ask the Database for persons under 15, but if there is not enough people to form his group, he is interested also in having persons under 16 or 17. This means that the constraint on the age in his query is not a very strict constraint to be enforced; age values can be considered to be similar depending on a given policy which can be either a difference less than 3 or a difference less than a given percentage, or user defined by subsets of domain (for instance for market analysis, we have subsets of ages such as {<9} {9-12} {13-15} {17-20}{21-30} {31-45} {46-55} {56-70} {>71}).

Other people asking for an airplane time-table from town A to town B, will also be interested in knowing possibilities to flight to town C close to B and take a train from C to B.

This problem is different from the problem of incomplete data (where a value of data is unknown) and from the problem of fuzzy information (where there is ponderation on truth value).

The examples above illustrate the two different kinds of neighborhood notions, the first one based on neighbor values, the second one representing general data neighborhood.

Let us notice that in security application, neighborhood of data implies similarity of their protection level, or at least a hierarchy between them.

II - Semantic Covert Channel

In the security domain, covert channels are a difficult problem very often related since what one is not allowed to know is coveted. Dealing with database and information management, a specific Covert Channel is bound to the problem of inference. Inference Control has been addressed in [5] where inference channel is defined as: "some information P can be used to derive partial or complete information about some other information Q where Q is classified higher than P".

Several inference channels have been considered, such as:

. Deductive channel (using logical deduction)

. Abductive channel (using Abduction to infer data)

. Probabilistic channel (using probability to find out data)

All these inference channels can be represented through standard formalisms (see [5]).

The deductive channel is based on the Modus Ponens (MP) derivation rule. Deductive relationship follows this rule strictly: when we have both P --> Q and P then we have Q:

$$\frac{P,\ P \dashrightarrow Q}{Q}$$

The abductive channel is also based on the Modus Ponens (MP) derivation rule. It intends to point out missing hypothesis (premisse): when we have P --> Q and Q then P is guessed:

$$\frac{Q,\ P \dashrightarrow Q}{P}$$

The probabilistic channel uses probabilistic properties.

Other inferences are possible based on semantic relationships existing between data. This kind of relationship is only described through the meaning of the data: it cannot be expressed by a syntactic expression without a new formalism.

In many cases, transcription of the semantic into a well formed expression is possible and thus allows its automated management. This is the case of the three kinds of semantic relationship described above.

The problem addressed by the neighborhood notion is also a problem of semantic relationship between two data; this relationship is not expressible using a classical formula, but the knowledge of one of the data discloses knowledge on another.

Geographical vicinity could be a good example to illustrate neighborhood of data. Let us consider two towns A and B, B being close to A and the destination relation DEST.

Let us assume that John is going to A, i.e. DEST("John","A") is true. In some context, it can be considered that if John goes to A, it can be considered that John goes to B, and therefore DEST("John","B"), even not strictly true, can be considered as an answer to the query where is John going ? This kind of relationship between DEST("John","A") and DEST("John","B") is not a deductive, neither is it an abductive nor probabilistic.

III - Neighborhood and Context

Examples above give a first idea of neighborhood data. In a Relational Database, data is represented by tuples of a relation extension, i.e. by elements of a cartesian products of domains. The first form of neighborhood notion is related to neighborhood of elements in a domain. For instance, in the domain of ages, 16 is a neighbor of 15; a neighborhood of data can be defined correspondingly: two data are neighbor data if they are different with respect to the values of one of their arguments and if the two values are neighbor elements in the corresponding domain. For instance within the relation:

LIVE (name,age,town)

the data "Live (John,17,Paris)" can be considered a neighbor of the data "LIVE (Jane,19,Rome)" since 19 is a neighbor of 17 in the age domain. This preliminary definition raises some problems:

1 - The data "John is 17 years old and lives in Paris" and "Jane is 19 years old and lives in Rome" can be considered as neighbors with respect to the age but not with respect to the town.

In the same way, "Live (John,17,Orsay)" can be considered a neighbor of the data "LIVE (Jane,40,Paris)" since Orsay is a neighbor of Paris with respect to the location but not with respect to age.

Therefore the notion of neighborhood is related to what will be called the CONTEXT.

2 - It is the same for the values 15 and 19 which can be considered as neighbor elements of the domain of ages in some contexts only (for instance, they are neighbors when dealing with young people but they are not when looking for people allowed to have a driving license).

3 - The neighborhood relation is not necessary symmetric. For instance, living in Orsay (a suburb of Paris) is neighbor of living in Paris but not the converse, since looking for somebody's address, it may happen that we look for his home in Paris since he told us "I live in Paris" instead of "I live in Orsay" (Paris is more well-known than Orsay and Orsay is in fact very close to Paris); in this way living in Orsay is a neighbor data of living in Paris but the converse is not true as long as nobody will say "I am living in Orsay" instead of "I live in Paris".

Similarly,

OWNER("Jane","Rolls") is a neighbor of
RICH("Jane")

but the converse is not true. To know if somebody is rich, it is possible to ask if he is the owner of a "Rolls" but to know if somebody owns a "Rolls", it is not meaningful to know that he is rich.

This illustrates the notion of neighborhood with respect to query answering. Let A1 and A2 be respectively the answers to queries Q1 and Q2. If answer to Q1 is interesting when Q2 is asked, then A1 is considered as a neighbor of A2 but the converse is not true since the answer to Q2 is not interesting when Q1 is asked. It can be said that A2 is the precise answer to Q2, and A1 is a related answer to Q2.

To solve the first two problems, let us introduce the notion of context and emphasize the asymmetry of the neighborhood relation for the third problem.

IV - Notion of Context

Context is defined as conditions or circumstances in the environment including not only the database content but also the query expression and the user characteristics. Context can be defined in a meta-level using more complex mechanisms which will not be discussed here. Such a choice have been made in [3] for cooperative answering in a database.

The use of neighborhood notion can be considered as a kind of cooperative answering, but the method proposed here is quite different, since it stays within the basic representation of data in relational database i.e. first order logic and does not use another model to modify query. The application of neighborhood to data security is also quite different from cooperative answering, since neighborhood and context notions are used to avoid disclosure of information via semantic covert channel.

In the sequel, it will be shown how the notion context can be integrated in the database and how neighborhood notion is managed.

Representation of a context can be done in two ways and is closely related to the way of using context in information representation.

A context can be considered as an element of a specific domain, the domain of contexts, and if a data P(A) is true in a given context c, it can be represented by using a corresponding predicate P'(A,c). The other representation of context is by using a unary predicate C(x) which is a classical way to represent a set of elements in the relational model. Using this representation, the fact that a data P(A) is true in a given context c can be expressed by the formula: C(c) --> P(A)

Using the first representation have the drawback to have to introduce new specific relation (or predicate) P' for each existing P. But both proof theoretic or model approach can be used for database representation. In the second approach, all data which are context dependent have an implicit representation in the database, and deductive mechanism is needed to find context dependent data.

In both cases, context has to be defined in terms of other data or information, for instance, a given context c1 may apply when the user is interested in a person's age. This can be expressed either by a meta level expression: "if the user is interested in ages then C(c1)" or by the introduction of predicates "USER(x)" and "INTEREST(x,y)" and the following expression:

USER("Dick") & INTEREST ("Dick","ages") -->
C(c1)

or more generally

USER(x) & INTEREST (x,"ages") --> C(c1)

Context change is another important matter when dealing with neighborhood notion. It can be done naturally by modifying conditions that define context, e.g. changing the user in the above example in a way that the new user is not interested in "ages".

Of course two contexts are not necessarily disjoints as long as their definitions do not lead to inconsistency.

V - Notion of Neighborhood

What is needed to express the notion of neighborhood between two data is to say that in some circumstances one of the data can be used instead of the other. It has also been noticed that the converse is not always true, so the neighborhood notion must be expressed by a non symmetrical expression.

Here again the logical implication enables the statement that, in a given context, a given datum implies (is neighbor of) another datum. And implication is not a symmetrical operator.

Thus, let us consider the notion of "context" as defined above. In a context c1, the neighborhood of a data P(A) with Q(B) (where P and Q are predicate symbols of degree np and nq and A and B are fully instantiated tuples) will be expressed by the following formulae with respect to the two notations used for context expression:

C(c1) & P(A) --> Q(B)

which is equivalent to

C(c1) --> (P(A) --> Q(B))

versus

P'(A,c1) --> Q'(B,c1)

P' and Q' are predicate symbols of degree np+1 and nq+1.

Since the logical implication is used to represent both neighborhood of data and deduced (implicit) data, it is necessary to define a different syntax for the neighborhood connector for instance:

++>

The interesting point is that ++> and --> have exactly the same behavior in the formal representation of a database, since they corresponds to the logical implication symbol, and depending on the application, a deductive process can be used either to define and manage neighbor data or implicit (deduced) data.

It is necessary to have two different representation symbols to distinguish what is logically deduced from a set of elementary data and general rules (by the way the conclusion of a general rule is true when its premise is true) from what is considered as a neighbor data of another (the conclusion of the general rule is semantically close to its premise).

The two different interpretations given to the logical implication operator can be managed independently using different models associated with a first order theory. The problem is that in the first order theory, specific well-formed formulae stated as axioms for the general rules of the represented real-world are not the same for implicit data representation and for neighborhood representation.

For instance

USER("Dick") & INTEREST ("Dick","ages") -->
C(c1)

is a classical deduction rule to infer C(c1) from the premise.

but

C(Ages) & LIVE(Jane,19,Rome) ++>
LIVE(John,17,Paris)

is a neighborhood representation rule.

Instead of using two different symbols (++> and -->) it is possible to use the same implication symbol but to distinguish two different and exclusive subsets of the general rules.

The next step will introduce some differencies between the two implication symbols.

VI - A Measure of Neighborhood

In [7], the concepts of n-deduction and n-consistency were defined in order to solve the non-decidability of first order theories applied to databases.

N-deduction introduces a deduction path length within which if a data is not proven, then it is considered as not provable. In this way, first order theories are proven to be decidable with respect to n-deduction.

An application of n-deduction to the neighborhood notion is straightforward. The interesting point is that while classical deduction is transitive, to say if P --> Q and Q --> R then P --> R, the neighborhood notion is not necessary:

if the n-deduction path length is limited to one
it is possible to have P ++> Q and Q ++> R
but not P ++> R

Until which level is a neighbor of a neighbor a neighbor? In the age neighborhood example, it can be said that two ages are neighbors if their difference in age is less than 2 years. But since 29 is a neighbor of 27, and 27 is a neighbor of 25, and so on, it does not follow that 29 can be considered a neighbor of 1. So it is useful to define a measure of neighborhood (or proximity) in order to say that over a given "distance" data are no longer considered neighbors.

N-deduction could be a means for defining such a distance measure. Given a set of rules and an inference process, we can consider all data that can be derived from the original set of data by one application of the inference process. This yields an extended set of data (distance ≤1). A second application of the process rule on this extended set will yield a third set of data (distance ≤2). and so on...

In the context of security, the problem remains the same. The general definition of neighborhood implies that if P(A) is a neighbor of Q(B) then the level of Q(B) must be lower than or equal to the level of P(A). But using general deduction concept leads to the statement that P(A) is a neighbor of R(C) even if there are many neighborhood definition rules between them.

Therefore, use of n-deduction can be suggested, associated with the neighborhood symbol ++> and the neighborhood rules to manage the neighborhood concept and to be able to express a measure of the neighborhood or proximity of two data.

Conclusion

The concept of neighborhood of data has been defined as well as its representation in the relational database model. It can be used to give some protection rules on data in order to avoid disclosure of information through semantic covert channel: some data becomes evident to a user because he is able to obtain another closely related data.

The concept of n-deduction can be used as a tool to express a measure of neighborhood between two data and for a consistent management of protection rules over data in a relational database.

Acknowledgments

Acknowledgments to H. Hosmer and J. Manning for their help in english improvement.

Bibliography

[1]K. Bowen, R. Kowalski
"Amalgamating language and metalanguage", in "Logic Programming", Clark Tarnlund Ed. 1980

[2] C.L. Chang, C.T. Lee
"Symbolic Logic and Mechanical Theorem Proving", Academic Press

[3] F. Cuppens, R. Demolombe
"Cooperative Answering: A methodology to provide intelligent access to Databases", 2nd International Conference on Expert Database Systems, Virginia, 1988

[4] H. Gallaire, J. Minker, J.-M. Nicolas
"Logic and Databases: a deductive approach", ACM Surveys, vol 16, n° 2, 1984

[5] T. Garvey, T. F. Lunt
"Cover Stories for Database Security", 5th IFIP WG 11.3 Working Conf. on Database Security, Shepherdstown, 1991

[6] R. Reither
"Toward a Logical Reconstruction of Relational Database Theory", in "Conceptual Modeling" Springer Verlag Ed. 1984

[7] K. Yazdanian
"Coherence des bases de donnees deductives", Thèse 3eme cycle, Université Paul Sabatier , Toulouse 1977