International Journal of Advanced Computer Technology (IJACT)

ISSN:2319-7900

Enhanced Privacy-Preserving Updates to Anonymous and Confidential Databases

Mr. Patil Pravin, Prof. Shirgave S K.

Abstract

In order to update *k*-anonymous and confidential database, the suppression based and generalization based updating protocol technique has been proposed. These protocols check whether the database inserted with the new tuple is still *k* - anonymous without knowing the content of the table and database respectively. But these methods will not work if initial database is empty. Also, if the incoming tuple that fails the test of these updating protocols, there is no solution for which action to be taken. So, in this paper we propose two solutions based on pending tuple set (i.e. a collection of all tuples that fails anonymous property of database)namely the private extraction of *k*-anonymous part of pending tuple set or *k*-anonymozation of pending tuple set by privately suppressing entries.

Index Terms—Anonymity, Data management, Privacy, Secure computation.

1. Introduction

Today it is well understood that databases represent an important asset for many applications and thus protection of privacy has become a very important concern. In particular, databases that contain sensitive information (e.g., health information) have often been available to public access, frequently with identifiers stripped off in order to protect privacy. Such a requirement has motivated a large variety of approaches aiming at better protecting data confidentiality and data ownership. To address such problem, Samarati and Sweeney [2] have been developed technique called *k-anonymization*,thus making it more difficult to link sensitive information to specific individuals. Such technique protects privacy by modifying the data so that the probability of linking a given data value, for example a given disease, to a specific individual is very less. Once database maintains with anonymous technique problem arises when data stored in a confidential, anonymity preserving database need to be updated. The operation of updating such a database, e.g., by inserting a tuple containing information about a given individual, introduces problems can the database owner decide if the updated database still preserves the privacy of individuals without directly knowing the new data to be inserted? All protocols have been developed rely on the fact that the anonymity of DB is not affected by inserting t if the information contained in *t*, properly anonymized, is already contained in the DB.

This is achieved by privately checking whether there is a match between (a properly anonymized version of) t and (at least) one tuple contained in the DB. The first protocol is based on suppression-based anonymous technique, and it allows the owner of DB to properly anonymous the tuple *t*, without gaining any useful knowledge on its contents and without having to send to its owner newly generated data. To achieve such goal, the parties secure their messages by use a commutative and homomorphic encryption scheme.

2. Problem Statement

As mentioned before, since DB contains privacy sensitive data, one main concern is to protect the privacy of each individual. Such task is guaranteed through the use of anonymization.But protocols used for modifying work well on the database that are previously anonymized i.e (containing some dummy anonymized data in the database). That addresses some issues : (i) the definition of a mechanism for actually performing the update, once k-anonymity has been verified; (ii) the specification of the actions to take in case Protocols yield a negative answer; (iii) how to initially populate an empty table. In this paper, we sketch the solutions developed in order to address these questions and which comprise our overall methodology for the private database update. As a general approach, we separate the process of database k-anonymity checking and the actual update into two different

ISSN:2319-7900

phases, managed by two different sub-systems: the Private Checker and the Private Updater.

3. Proposed Solutions

All method we proposed here are relying on the concept of the pending tuple set i.e.Suppose that a tuple fails the tests of anonymous database updating protocols then, the system does not insert the tuple to the k-anonymous database, and waits until *k*-1 other tuples fail the insertion. At this point, the system checks whether such set of tuples, referred to as pending tuple set. In Figure 1 such modules are represented along with labeled arrows denoting what information is exchanged among them.

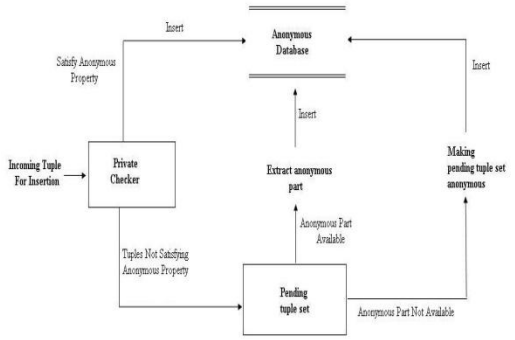

Fig. 1. Prototype Architecture overview.

Specially, we consider a setting in which there is a set of customers, each of whom has a row of a table, and a miner, who wants to mine the entire table. Our objective is to design protocols that allow the miner to obtain a *k*-anonymous table representing the customer data in such a way that does not reveal any extra information that can be used to link sensitive attributes to corresponding identifiers. We give two different methods of this problem. In the first method, given a table, the protocol needs to extract the *k*-anonymous part (i.e., the maximum sub-set of rows that is already *k*-anonymous) from it. The privacy requirement is that the sensitive attributes out-side the *k*-anonymous part should be hidden from any individual participant including the miner. This method is suitable if the original table is already close to *k*-anonymity. In the second method, given a table, the protocol needs to suppress some entries of the quasi-identifier attributes, so that the entire table is *k*-anonymized. The privacy requirement is that the suppressed entries should be hidden from any individual participant. This method is suitable even if the original table is not close to *k*-anonymity and also used for initial populating of empty tables .

3.1 Problem Formulations

Consider a table with *m* quasi- identifier attributes, (s*1,….,sm*), and *n* sensitive attributes, (*a*1,…, *an*). Without loss of generality, we assume that there are no other attributes except these *m*+*n*. Suppose that there are *N* +1 involved parties: *N* customers and one miner.For convenience, the miner assigns indices 1 through *N* to the customers in the sequel, by "customer *i*" we mean the "customer with indices *i*". Note that the indices are *not* identifiers because they are arbitrarily assigned by the miner, who does not know the identifiers of the customers. Each customer *i* have a row of the table,which is denoted by *Ri* = (*s*(*i*)1 ,……,*s*(*i*)*m* ; *a*(*i*)1,…….., *a*(*i*)*n*). We assume there are private unidentified channels between each customer and the miner. That is, the channels are unstoppable and the miner has no information about which customer is using which channel, but each channel is used by exactly one customer. The overall objective is to enable the miner to obtain a *k*-anonymized table in a private manner (so that he can mine the table).This can be achieved in two ways, described in detail in Sections 3.2 and 3.3: either enable the miner to extract the *k*-anonymous part of the pending tuple set, or enable him to obtain a *k*-anonymized pending tuple set in which some entries of the quasi-identifier attributes are suppressed.

3.2 Formulation 1: Private Extraction of *k*-Anonymous Part

In the first problem formulation, the miner extracts the *k*-anonymous part of the table (i.e., the maximum subset of rows that is *k*-anonymous), but does not learn extra information about the sensitive attributes of the rows outside the *k*-anonymous part. Consequently, the miner cannot link the sensitive attributes of any row to the corresponding identifiers.

Intuitively, our privacy requirement states that, for each party(miner or customer), the view of the protocol seen by that party can be simulated by an algorithm that has no knowledge of the sensitive attributes outside the *k*-anonymous part.This captures the requirement that any individual party cannot learn any extra information about these sensitive attributes

ISSN:2319-7900

by virtue of engaging in the protocol.To formalize this requirement, first define the *view* of each party during an execution of the protocol.

3.3 Formulation 2: *k*-Anonymization by Privately Suppressing Entries

It has been studied how to *k*-anonymize a table by suppressing entries ideally suppressing as few as possible [6]. Our second problem formulation supports suppression in our distributed setting. Let Anonymized(*T*) denote the output (which is a *k*-anonymized table) of a protocol that *k*-anonyms pending tuple set *T* by suppressing entries.

4. Solution For Formulation

In this section, we solve the first formulation of the problem.That is, we design a protocol that privately extracts the *k*-anonymous part of a table. The basic idea of our design is that each customer encrypts her sensitive attributes using a m encryption key that can be derived if and only if there are at least *k*- rows whose quasi-identifiers are equal. Specifically,the key to encrypt the sensitive attributes ($a(i)1$,……,$a(i)n$) is a function of the corresponding quasi-identifier ($s(i)1$,….,$s(i)m$) and it is shared among the customers with threshold *k*. Each customer submits to the miner one share of the key(s) corresponding to her quasi-identifier. As a result, if and only if there are at least *k* customers whose quasi-identifiers are equal, the miner is able to recover a key.The remaining technical question is how each customer selects the key.

We resolve this problem by assuming a *(2N,K) Shamir secret sharing* [7] of a "seed" key *x*, where each customer *i* have two shares $x2i$-1 and $x2i$ of the seed key. (Note the meaning of the two parameters of Shamir secret sharing: 2*N* is the overall number of shares and *k* is the threshold number of shares needed to recover *x*.) A very useful property of Shamir secret sharing is that with *k* or more shares one can easily derive all other shares using Lagrange interpolation, while with less than *k*-shares one has no information about any other shares at all.

5. Our Solution For Formulation Second

In this section, we solve the second formulation of the problem. Specifically, we provide a protocol that privately *k*-anonymous a table by suppressing entries. Our protocol is based on Meyerson and Williams's algorithm (which refer to as MW) for *k*-anonymizing a database [23]. Our protocol provides quantifiable, though not ideal, privacy. Namely, it keeps all information about the suppressed entries private from each individual party, except revealing the distance between each pair of rows.Our protocol consists of three phases. In the first phase, the protocol allows the miner to compute the distance between each pair of rows. In the second phase, the miner uses the MW algorithm to compute a *k-partition* of the table. (A *k*- partition is a collection of disjoint subsets of rows in which each subset contains at least *k* -rows and the union of these subsets is the entire table.) In the third phase, the protocol allows the miner to compute the *k*-anonymized table. The second phase is a direct computation of part of MW (which relies only on the inter-row distances already known to the miner).We now overview the more complex first and third phases;we describe all three phases in complete detail in Section 5.1.

5.1Protocols

This technique makes pending tuple set *k*-anonymized by suppressing entries ideally suppressing as few as possible. Let Anonymized (*T*) denote the output (which is a *k-an*onymized table) of a protocol that *k-an*onyms the table *T* by suppressing entries. This technique is based on Meyersons and Williams's algorithm (which is known as MW) for *k*-anonymizing a database. Namely, it keeps all information about the suppressed entries private from each individual party, except revealing the distance between each pair of rows.

Algorithm:

Phase 1: Compute the distance between every two rows.

The distance between two rows is the how many numbers of quasi-id attributes in which they have dissimilar entries.

Phase 2: Make a *k*-partition of the table. (Data organizer can perform it locally)

A *k*-partition is a collection of disjoint subsets of rows in which each subset contains at least *k* rows and the union of these subsets is the original table.

Phase 3: determine the *k*-anonymous table.

ISSN:2319-7900

5.2. Phase 1: Protocol for Computing Distances between Rows

For any two rows and on the *i'* th quasi-id attribute, define

$$\begin{cases} 1 & \text{If two rows have the same Value} \\ \text{Uniform\_random} & \text{Otherwise} \end{cases}$$

The distance between the any two rows = the number of *1*s in Computing $i \in [1, m]\}$

i) If all rows in this subset are agreeing on this quasi-id, no operation is needed;

ii) Otherwise, mask the values of this quasi-id with *.

$$distance = \frac{i \text{ th quasi\_id of one row}}{i \text{ th quasi\_id of second row}}$$

Only need to compute the quotient of quasi-ids.

5.2. Phase 2: A *k*-partition is a combination of subsets of rows in which each subset contains at least *k* rows and the union of these subsets is the entire table

5.2. Phase 3: Compute a *k*-partition of the table

The output of Phase 2 is a divide the table into subsets of rows.In each of these subsets, for each quasi-id attribute,

References

[1] N. R. Adam, J. C. Wortmann. "Security-control methods for statistical databases: a comparative study". *ACM Computing Surveys 1989*

[2] L. Sweeney. "k-anonymity: a model for protecting privacy". *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, 2002*

[3] A. Trombetta, E. Bertino. "Private updates to anonymous databases". *In Proc. Int'l Conf. On Data Engineering (ICDE), 2006.*

[4] O. Goldreich. "Foundations of Cryptography. Basic Applications", *2004*

[5] J. Li, B.C. Ooi, W. Wang. "Anonymzing streaming data for privacy protection". In *Proc. of IEEE Int'l Conf. On Database Engineering (ICDE),2008.*

[6] S. Zhong, Z. Yang, R. N. Wright. "Privacy-enhancing *k*-anonymization of customer data". *In Proc. ACM Symposium on Principles of Database Systems (PODS), 2005*

[7] M. K. Reiter, A. Rubin. Crowds: anonymity with Web transactions.ACM Transactions on Information and System Security (TISSEC), 1998.

[8] Alberto Trombetta, Wei Jiang, Elisa Bertino and Lorenzo Bossi. "Privacy –Preserving Update To Anonymous and Confidential Database".*IEEE Transactions on Knowledge and Data Engineering in Nov ,2011.*

Biographies

Mr.Patil Pravin R received B.E degree in Information Technology from Shivaji University,Kolhapur Maharastra,India in 2009,the M.E degree in Computer Science and Engineering pursues from Shivaji University.

Prof.Shirgave S K received B.E degree in Computer Science and Engineering from Shivaji University,Kolhapur ,Maharastra,India ,the M.E degree in Computer Science and Engineering from Shivaji University Kolhapur,Maharastra,India. Currently, He is an associate professor of Computer Science Engineering at Shivaji University .His teaching and research areas include web mining, database security and computer algorithm.