International Journal of Computer Applications Technology and Research
Volume 4– Issue 4, 274 - 279, 2015, ISSN:- 2319–8656

Anonymizing and Confidential Databases for Privacy Protection Using Suppression and Generalization Based Protocols

K.Sathyamoorthy
Department of CSE
Panimalar Institute of Technology
Chennai,India

S.Venkata Lakshmi
Department of CSE
Panimalar Institute of Technology
Chennai,India

Tina Belinda Miranda
Department of CSE
Panimalar Institute of Technology
Chennai,India

Abstract—The technique of k-anonymization has been proposed in the literature as an alternative way to release public information, while ensuring both data privacy and data confidentiality. "X" owns a k-anonymous database and needs to determine whether "X" database, when inserted with a tuple owned by "Y", is still k-anonymous. Clearly, allowing "X" to directly read the contents of the tuple breaks the privacy of "Y". In this place,"Y" not get the privacy of own information because the information of "Y" can be accessed by "X" without the prior knowledge of "Y". On the other hand, the confidentiality of the database managed by "X" is violated once "Y" has access to the contents of database. Thus, the problem is to check whether the database inserted with the tuple is still k-anonymous, without letting "X" and "Y" knows the contents of the tuple and database respectively. In this paper, we propose two protocols solving this problem that is suppression-Based & Generalization-Based k-anonymous and confidential databases using through prototype architecture. And also those two protocols maintain privacy and confidential information in k-anonymous database.

Keywords — Confidentiality, Anonymity, Privacy, Secure Computation.

1. INTRODUCTION

TODAY'S globally networked society places great demand on the collection and sharing of person-specific data for many new uses [1]. This happens at a time when more and more historically public information is also electronically available. When these data are linked together, they provide an electronic image of a person that is as identifying and personal as a fingerprint even when the information contains no explicit identifiers, such as name and phone number. Other distinctive data, such as birth date and postal code, often combine uniquely[2] and can be linked to publicly available information to re-identify individuals. Data confidentiality is particularly relevant because of the value, often not only monetary, that data have. For example, medical data collected by following the history of patients over several years may represent an invaluable asset that needs to be adequately protected. Such a requirement has motivated a large variety of approaches aiming at better protecting data confidentiality and data ownership. Relevant approaches include query processing techniques for encrypted data and data watermarking techniques.

Data confidentiality is not, however, the only requirement that needs to be addressed. Today there is an increased concern for privacy. The availability of huge numbers of databases recording a large variety of information about individuals makes it possible to discover information about specific individuals by simply correlating all the available databases. Although confidentiality and privacy are often used as synonyms, they are different concepts: data confidentiality is about the difficulty by an unauthorized user to learn anything about data stored in the database. Usually, confidentiality is achieved by enforcing an access policy, or possibly by using some cryptographic tools. Privacy relates to what data can be safely disclosed without leaking sensitive information regarding the legitimate owner [5].

To better understand the difference between confidentiality and anonymity, consider the case of a medical facility connected with a research institution. Suppose that all patients treated at the facility are asked before leaving the facility to donate their personal health care records and medical histories to the research institution, which collects the records in a research database. To guarantee the maximum privacy to each patient, the medical facility only sends to the research database an anonymized version of the patient record. Once this anonymized record is stored in the research database, the non anonymized version of the record is removed from the system of the medical facility.

Thus, the research database used by the researchers is anonymous. Suppose that certain data concerning patients are related to the use of a drug over a period of four years and certain side effects have been observed and recorded by the researchers in the research database. It is clear that these data (even if anonymized) need to be kept confidential and accessible only to the few researchers of the institution working on this project, until further evidence is found about the drug. If these anonymous data were to be disclosed, privacy of the patients would not be at risk; however the company manufacturing the drug may be adversely affected. Recently, techniques addressing the problem of privacy via data anonymization have been developed, thus making it more difficult to link sensitive information to specific individuals.

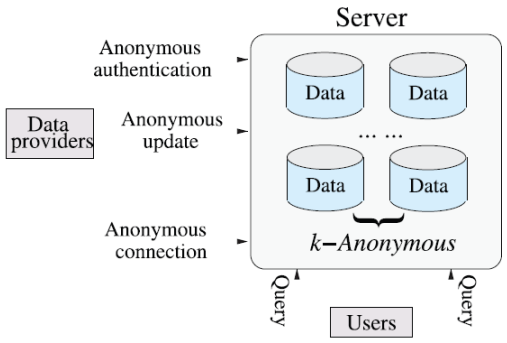

Figure 1: Anonymous Database

One well-known technique is k-anonymization. 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 small. So far, the problems of data confidentiality and anonymization have been considered separately. However, a relevant 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 two problems concerning both the anonymity and confidentiality of the data stored in the database and the privacy of the individual to whom the data to be inserted are related: 1) Is the updated database still privacy preserving? and 2) Does the database owner need to know the data to be inserted? Clearly, the two problems are related in the sense that they can be combined into the following problem: can the database owner decide if the updated database still preserves privacy of individuals without directly knowing the new data to be inserted? The answer we give in this work is affirmative. It is important to note that assuring that a database maintains the privacy of individuals to whom data are referred is often of interest not only to these individuals, but also to the organization owning the database. Because of current regulations, like HIPA organizations collecting data about individuals are under the obligation of assuring individual privacy. It is thus, in their interest to check the data that are entered in their databases do not violate privacy, and to perform such verification without seeing any sensitive data of an individual.

1.1 Problem Statement

Figure.1 captures the main participating parties in our application domain. We assume that the information concerning a single patient (or data provider) is stored in a single tuple, and DB is kept confidentially at the server. The users in Figure.1 can be treated as medical researchers who have the access to DB. Since DB is anonymous, the data provider's privacy is protected from these researchers. As mentioned before, since DB contains privacy-sensitive data, one main concern is to protect the privacy of patients. Such task is guaranteed through the use of anonymization. Intuitively, if the database DB is anonymous, it is not possible to infer the patients' identities from the information contained in DB. This is achieved by blending information about patients. Suppose now that a new patient has to be treated. Obviously, this means that the database has to be updated in order to store the tuple t containing the medical data of this patient.

The modification of the anonymous database DB can be naively performed as follows: the party who is managing the database or the server simply checks whether the updated database DB is still anonymous. Under this approach, the entire tuple t has to be revealed to the party managing the database server, thus violating the privacy of the patient. Another possibility would be to make available the entire database to the patient so that the patient can verify by himself/herself if the insertion of his/her data violates his/her own privacy. This approach however, requires making available the entire database to the patient thus violating data confidentiality. In order to devise a suitable solution, several problems need to be addressed: Problem 1: without revealing the contents of t and DB, how to preserve data integrity by establishing the anonymity of DB. Problem 2: once such anonymity is established, how to perform this update? Problem3: what can be done if database anonymity is not preserved? Finally, problem 4: what is the initial content of the database, when no data about users has been inserted yet? In this paper, we propose two protocols solving Problem 1, which is the central problem addressed by our paper. However, because the other problems are crucial from a more practical point of view. An approach that can be used is based on techniques for user anonymous authentication and credential verification [20]. The above discussion illustrates that the problem of anonymous updates to confidential databases is complex and requires the combination of several techniques, some of which are proposed for the first time in this paper. Figure.1 summarizes the various phases of a comprehensive approach to the problem of anonymous updates to confidential databases.

1.2 Proposed Solutions

All protocols we propose to solve Problem 1 rely on the fact that the anonymity of DB is not affected by inserting t if the information contained in t, properly anonymized, and is already contained in DB. Then, Problem 1 is equivalent to privately checking whether there is a match between (a properly anonymized version of) t and (at least) one tuple contained in DB. The first protocol is aimed at suppression-based anonymous databases, and it allows the owner of DB to properly anonymize the tuple t, without gaining any useful knowledge on its contents and without having to send to t's owner newly generated data. To achieve such goal, the parties secure their messages by encrypting them. In order to perform the privacy-preserving verification of the database anonymity upon the insertion, the parties use a commutative and homomorphic encryption scheme. The second protocol is aimed at generalization-based anonymous databases, and it relies on a secure set intersection protocol, such as the one found in [3], to support privacy-preserving updates on a generalization-based k-anonymous DB.

2. RELATED WORK

A preliminary approach to this problem was investigated in [33]. However, these protocols have some serious limitations, in that they do not support generalization-based updates, which is the main strategy adopted for data anonymization. Therefore, if the database is not anonymous with respect to a tuple to be inserted, the insertion cannot be performed. In addition one of the protocols is extremely inefficient. In the current paper, we present two efficient protocols, one of which also supports the private update of a generalization

based anonymous database. We also provide security proofs and experimental results for both protocols. So far no experimental results had been reported concerning such type of protocols; our results show that both protocols perform very efficiently.

The first research direction deals with algorithms for database anonymization. The idea of protecting databases through data suppression has been extensively investigated in the area of statistical databases [1]. The problem of protecting the privacy of time varying data have recently spurred an intense research activity which can be roughly divided into two broad groups depending on whether data are continuously released in a stream and anonymized in an online fashion, or data are produced in different releases and subsequently anonymized in order to prevent correlations among different releases. The second research direction is related to Secure Multiparty Computation (SMC) techniques. SMC represents an important class of techniques widely investigated in the area of cryptography. The third research direction is related to the area of private information retrieval, which can be seen as an application of the secure multiparty computation techniques to the area of data management. Here, the focus is to devise efficient techniques for posing expressive queries over a database without letting the database know the actual queries [10]. Thus, the goal is to protect data confidentiality from the external entities managing the data; however, data are fully available to the clients, which is not the case under our approach.

3. BASIC DEFINITIONS AND PRIMITIVES

3.1 Anonymity Definition: When using a suppression-based anonymization method, we mask with the special value ∂, the value deployed by Alice for the anonymization. When using a generalization-based anonymization method, original values are replaced by more general ones, according to a priori established value generalization hierarchies The information age has witnessed a huge growth in the amount of personal data that can be collected and analyzed. This has led to an increasing use of data mining tools with the basic goal of inferring trends in order to predict the future. However, this goal conflicts with the desire for privacy of personal data. In many scenarios, access to large amounts of personal data is essential in order for accurate inferences to be drawn. We adopt the following notations thereafter:

AREA	POSITION	SALARY
Data Mining	Associate Professor	$90,000
Intrusion Detection	Assistant Professor	$78,000
Handheld Systems	Research Assistant	$17,000
Handheld Systems	Research Assistant	$15,500
Query Processing	Associate Professor	$100,000
Digital Forensics	Assistant Professor	$78,000

TABLE.1. Original Data Set

AREA	POSITION	SALARY
*	Associate Professor	*
*	Assistant Professor	*
Handheld Systems	Research Assistant	*
Handheld Systems	Research Assistant	*
*	Associate Professor	*
*	Assistant Professor	*

TABLE 2.Suppressed with K=2

. Quasi-Identifier (QI): A set of attributes that can be used with certain external information to identify a specific individual.

. T½QI_: T½QI_ is the projection of T to the set of attributes contained in QI.

Definition 3.1. T½QI_ satisfies k-anonymity if and only if each record in it appears at least k times [32].

3.2.Cryptographic Primitives

A commutative, product-homomorphic encryption scheme ensures that the order in which encryptions are performed is irrelevant (commutativity) and it allows to consistently performing arithmetic operations over encrypted data (homomorphic property). Further, for the security proofs we require that the encryption scheme E satisfies the in distinguishability property. We extend the definition of commutative, indistinguishable encryption scheme presented in [3], in order to obtain an encryption scheme which also product-homomorphic. Given a finite set K of keys and a finite domain D, a commutative, product homomorphic encryption scheme E is a polynomial time computable function E : K X D→D satisfying the following properties:

1.Commutativity.

For all key pairs K_1,K_2€K andvalue d € D, the following equality holds:

$$E_{K1}(E_{K2}(d)) = E_{K2}(E_{K1}(d)) \quad (1)$$

2. Product-homomorphism.

For every K € K and everyvalue pairs d1,d2 € D, the following equality holds:

$$E_k(d_1) \cdot E_K(d_2) = E_K(d_1 \cdot d_2) \quad (2)$$

3. Indistinguishability

It is infeasible to distinguish an encryption from a randomly chosen value in the same domain and having the same length. In otherwords, it is infeasible for an adversary, with finite computational capability, to extract information about a plain text from the cipher text.

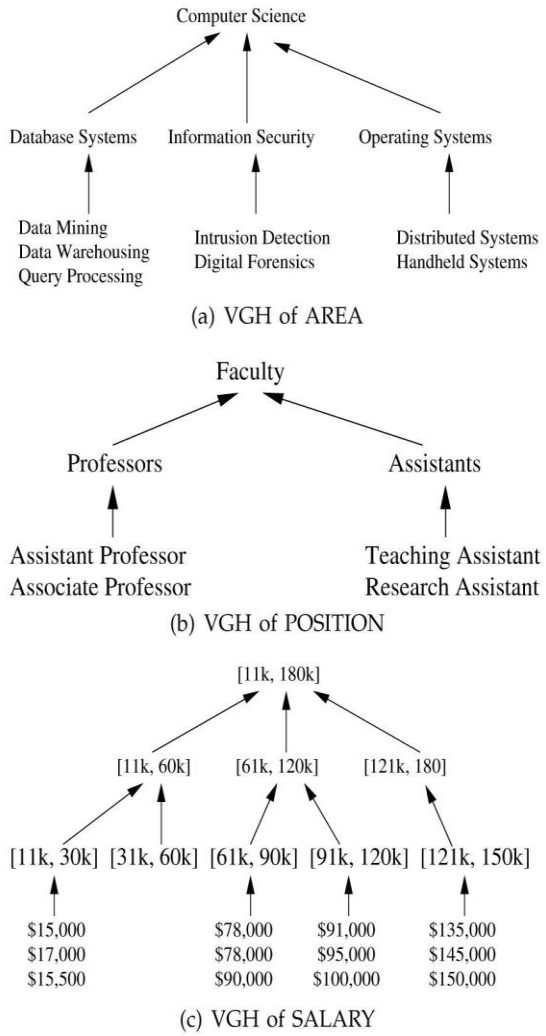

Figure. 2. Value Generalization Hierarchies.

AREA	POSITION	SALARY
Database Systems	Associate Professor	[61k, 120k]
Information Security	Assistant Professor	[61k, 120k]
Operating Systems	Research Assistant	[11k, 30k]
Operation Systems	Research Assistant	[11k, 30k]
Database Systems	Associate Professor	[61k, 120k]
Information Security	Assistant Professor	[61k, 120k]

TABLE 3: Generalized with K=2

4. PRIVATE UPDATE BASED ON SUPPRESSION BASED PROTOCOLS

The idea of suppressing an attribute is a simple concept. A value is replaced by a less specific, more general value that is faithful to the original. In a classical relational database system, domains are used to describe the set of values that attributes assume. For example, there might be a ZIP domain, a *number* domain and a *string* domain. In the original database, where every value is as specific as possible, every attribute is considered to be in a ground domain. In this section, we assume that the database is anonymized using a suppression-based method. Note that our protocols are not required to further improve the privacy of users other than that provided by the fact that the updated database is still k-anonymous.It allows the owner of DB to properly anonymized the tuple t, without gaining any useful knowledge on its contents and without having to send to t's owner newly generated data. To achieve such goal, the parties secure their messages by encrypting them. In order to perform the privacy-preserving verification of the database anonymity upon the insertion, the parties use a commutative and homomorphic encryption scheme.

Suppose that X owns a k-anonymous table T over the QI attributes. X has to decide whether T U t, where t is a tuple owned by Y is still k-anonymous, without directly knowing the values in t(assuming t and T have the same schema). This problem amounts to decide whether t matches any tuple in T on thenon-suppressed QI attributes. If this is the case, then t,properly anonymized, can be inserted into T. Otherwise, the insertion of t into T is rejected. A solution that addresses such drawback is based on the following protocol. Assume, X and Y agree on acommutative and product-homomorphic encryption scheme. Unless otherwise stated, the term *data* refers to person-specific information that is conceptually organized as a table of rows (or records) and columns (orfields). Each row is termed a *tuple*. Tuples within a table are not necessarily unique. Each column is called an *attribute* and denotes a semantic category of information that is a set of possible values; therefore, an attribute is also a domain. Attributes within a table are unique. So by observing a table, each row is an ordered *n*-tuple of values <*d1*, *d2*, …, *dn*> such that each value *dj*is in the domain of the *j*-th column, for *j*=1, 2, …, *n* where *n* is the number of columns.

Protocol 4.1

1. X codes his tuple ∂_i into c([v'$_1$…….v'$_{t]}$, denoted asC(∂_i). Then, X encrypts c(∂_i)with his private key and sends $E_A(c(\partial_i))$to Y.

2. Y individually codes each attribute value in t to get the tuple of coded values [c(v_1) . . . c(v_u)] encrypt search coding and $E_A(c(\partial_i))$ with his key B and sends (i)

[$E_B(c(v_1))$ . . . $E_B(c(v_u))$], and (ii) $E_B(E_A(c(\partial_i)))$ to X.

3. Since E is a commutative encryption scheme,

$E_B(E_A(c(\partial_i))) = E_A(E_B(c(\partial i)))$,

X decrypts $E_A(E_B(c(\partial_{i)}))$ to obtain $E_B(c(\partial_i))$

4. Since the encrypted values sent by Y are ordered according to the ordering of the attributes in T(assume this is a public information known to both X and Y), X knows which are, among the encrypted values sent by Y, the one corresponding to the suppressed and non suppressed QI attributes. Thus, X computes

$E_B(c(v_1))$ X . . . X $E_B(c(v_s))$ (5)

where v_1 . . . v_s are the values of nonsuppressed attributes contained in tuple t. As already mentioned, E is a product-homomorphic encryption scheme. Based also on the definition of function c(∂), this implies that Expression 5 is equal to

$E_B(c([v_1 \ldots v_s]))$ (6)

5. X checks whether

$E_B(c([v_1 \ldots v_s])) = E_B(c([v_1 \ldots v_0]))$

If true, t (properly anonymized) can be inserted to table T. Otherwise, when inserted to T, t breaks k-anonymity.

5. PRIVATE UPDATE USING GENERALIZATION BASED

CONFIDENTIAL AND ANONYMOUS DATA

The idea is that there are some numbers of records in *T* that can be considered outliers. For this reason, up to a certain number of records (the *maximum suppression threshold*) may be completely excluded from *V*. Under this combined scheme, *V* is obtained through full-domain generalization, with selected outlier tuples removed entirely. For any anonymization mechanism, it is desirable to some notion of minimality. Intuitively, a k-anonymization should not generalize, suppress, or distort the data more than is necessary to achieve k-anonymity. Indeed, there are a number of ways to minimality. One notion of minimal full-domain generalization using the distance vector of the domain generalization. Informally, this definition says that a full-domain generalization *V* is minimal if *V* is k-anonymous, and the height of the resulting generalization is less than or equal to that of any other k-anonymous full-domain generalization.

In this section, we assume that the table T is anonymized using a generalization-based method; let $T_1. . . . .T_u$ be udisjoint VGHs corresponding to $A_1…..A_u€ A_t^{anon}$ known to X. Let ∂ € T, and let GetSpec(∂[$A_1 . . .A_u$],$T_1 . . . T_u$) (GetSpec(∂) for short) denote a function which returns a set $_\gamma$ of specific values (values at the bottom of a VGH)related to each attribute $A_i€A_t^{anon}$ such that every value in $_\gamma$ can be generalized to ∂[Ai] for some i according to T_i. For example, let T refer to Table 4 and A_t^{anon}={AREA,POSITION, SALARY}. If T= [Operating Systems, Research Assistant, [11k, 30k]], then based on the VGHs (presented in Figure. 2) GetSpec(T) = {Distributed Systems, Handheld Systems, Research Assistant, $15,000, $17,000, $15,500}.

Protocol 5.1

1. X randomly chooses a ∂ € T_w.
2. Y compute $^\gamma$ = GetSpec(∂).
3. X and Y collaboratively compute s = SSI(γ, T).
4. If s = u then t's generalized form can be safely insertedto T.
5. Otherwise, X computes T_w←T_w– {∂} and repeatthe above procedures until either s = u or $T_{u=¥}$

5.1 Security Analysis

The security of Protocol 5.1 depends on that of the SSI protocol, and detailed security analyses of SSI can be found in [3], [13]. The SSI protocol presented in [3] is easy to implement and efficient to perform. Although the protocol leaks the intersection size between $^\gamma$ and T to the participating parties, it does provide sufficient privacy protection in our application. In case this linkage of intersection sizes is not acceptable, we can adopt one variation of the SSI protocol presented in [13]. We can make the protocol only return whether or not acceptable without disclosing the intersection size. Under the context of Secure Multiparty Computation, this variation of SSI does not leak any information that cannot be inferred from the final result and the private input data. Thus, using SSI proposed in [13], Protocol 5.1 can achieve very high security.

6. ARCHITECTURAL DESIGN

Our prototype of a Private Checker (that is, X) is composed by the following modules: a crypto module that is in charge of encrypting all the tuples exchanged between an user (that is, Y) and the Private Updater, using the techniques and a checker module that performs all the controls, as prescribed by Protocols 4.1 and 5.1; a loader module that reads chunks of anonymized tuples from the k-anonymous DB. The chunk size is fixed in order to minimize the network overload. In Figure. 3 such modules are represented along with labelled arrows denoting what information are exchanged among them. Note that the functionality provided by the Private Checker prototype regards the check on whether the tuple insertion into the k-anonymous DB is possible. We do not address the issue of actually inserting a properly anonymized version of the tuple.

The information flow across the above mentioned modules is as follows: after an initial setup phase in which the user and the Private Checker prototype exchange public values for correctly performing the subsequent cryptographic operations, the user sends the encryption E(c(∂i)) of her/his tuple to the Private Checker; the loader module reads from the k-anonymous DB the first chunk of tuples to be checked with E(c(∂i)). Such tuples are then encrypted by the crypto module.

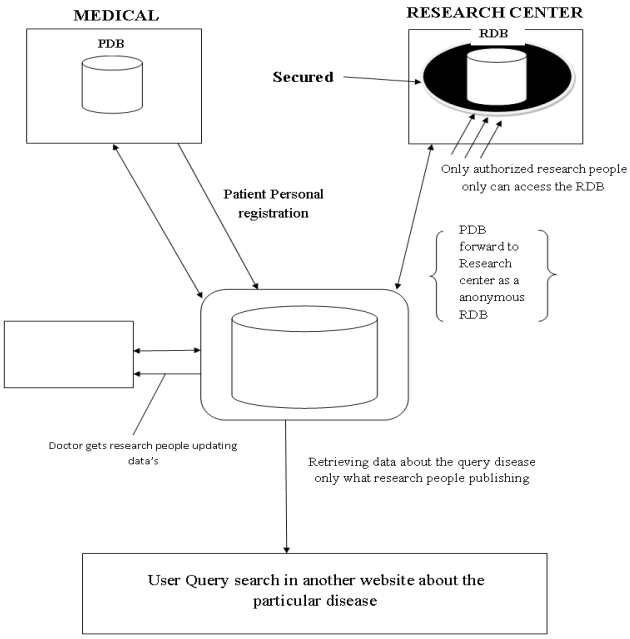

Figure. 3. Prototype architecture overview.

7. CONCLUSION / FUTURE WORK

In this paper, we have presented two secure protocols for privately checking whether k-anonymous database retains its anonymity once a new tuple is being inserted to it. Since the proposed protocols ensure the updated database remains k-anonymous, the results returned from a user's (or a medical researcher's) query are also k-anonymous. Thus, the patient or the data provider's privacy cannot be violated from any query. As long as the database is updated properly using the proposed protocols, the user queries under our application domain are always privacy-preserving. In order for a database system to effectively perform privacy preserving updates to a k-anonymous table, Protocols 4.1 and 5.1 are necessary but clearly not sufficient. As already mentioned in Section 1, other important issues are to be addressed:

1. The definition of a mechanism for actually performing the update, once k-anonymity has been verified.

2. The specification of the actions to take in case Protocols 4.1 or 5.1 yield a negative answer.

3. How to initially populate an empty table.

4. The integration with a privacy-preserving query system.

In addition to the problem of falling insertion, there are other interesting and relevant issues that remain to be addressed:

- Devising private update techniques to database systems that supports notions of anonymity different than k-anonymity.
- Dealing with the case of malicious parties by the introduction of an untrusted, non colluding third party .
- Implementing a real-world anonymous database system.
- Improving the efficiency of protocols, in terms of number of messages exchanged and in terms of their sizes, as well.

We believe that all these issues are very important and worthwhile to be pursued in the future.

8. REFERENCES

[1] N.R. Adam and J.C. Wortmann, "Security-Control Methods for Statistical Databases: A Comparative study," ACM Computing Surveys, vol. 21, no. 4, pp. 515-556, 1989.

[2] G. Aggarwal, T. Feder, K. Kenthapadi, R. Motwani, R. Panigrahy,D. Thomas, and A. Zhu, "Anonymizing Tables," Proc. Int'l Conf .Database Theory (ICDT), 2005.

[3] R. Agrawal, A. Evfimievski, and R. Srikant, "Information Sharingacross Private Databases," Proc. ACM SIGMOD Int'l Conf .Management of Data, 2003.

[4] C. Blake and C. Merz, "UCI Repository of Machine MLRepository.html, 1998.

[5] E. Bertino and R. Sandhu, "Database Security—Concepts, Approaches and Challenges," IEEE Trans. Dependable and SecureComputing, vol. 2, no. 1, pp. 2-19, Jan.-Mar. 2005.

[6] D. Boneh, "The Decision Diffie-Hellman Problem," Proc. Int'lAlgorithmic Number Theory Symp., pp. 48-63, 1998.

[7] D. Boneh, G. di Crescenzo, R. Ostrowsky, and G. Persiano, "PublicKey Encryption with Keyword Search," Proc. Eurocrypt Conf., 2004.

[8] S. Brands, "Untraceable Offline Cash in Wallets with Observers,"Proc. CRYPTO Int'l Conf., pp. 302-318, 1994.

[9] J.W. Byun, T. Li, E. Bertino, N. Li, and Y. Sohn, "Privacy-Preserving Incremental Data Dissemination," J. Computer Security,vol. 17, no. 1, pp. 43-68, 2009.

[10] R. Canetti, Y. Ishai, R. Kumar, M.K. Reiter, R. Rubinfeld, and R.N.Wright, "Selective Private Function Evaluation with Application to Private Statistics," Proc. ACM Symp.Principles of DistributedComputing (PODC), 2001.

[11] S. Chawla, C. Dwork, F. McSherry, A. Smith, and H. Wee,"Towards Privacy in Public Databases," Proc. Theory of CryptographyConf. (TCC), 2005.

[12] U. Feige, J. Kilian, and M. Naor, "A Minimal Model for SecureComputation," Proc. ACM Symp. Theory of Computing (STOC),1994.

[13] M.J. Freedman, M. Naor, and B. Pinkas, "Efficient PrivateMatching and Set Intersection," Proc. Eurocrypt Conf., 2004.

Mr.K.Sathyamoorthy is currently working as a Assistant Professor, Computer Science and Engineering, at Panimalar Institute of Technology, Poonamalle, Chennai 600123. He has completed his graduate in Anna University and Post graduate in Sathyabama University. His research interest includes image processing and soft computing.

Ms.S.Venkata Lakshmi is currently working as a Assistant Professor Grade 1, Computer Science and Engineering, at Panimalar Institute of Technology, Poonamalle, Chennai 600123.She has completed her graduate in M.S.University and Post graduate in DR.MGR university where she is currently working toward the Ph.D degree. She has published research papers in International, National conference proceedings. Her research interest includes image processing and medical imaging.

Ms.Tina Belinda Miranda is currently working as an Lecturer, Computer Science and Engineering at Panimalar Institute of Technology, Poonamalle, Chennai 600123.She has completed her graduate in Anna university and Post graduate in Anna university. Her research interest includes networks and Image processing.