PRIVACY PRESERVING MINING OF ASSOCIATION RULES IN HORIZONTALLY DISTRIBUTED DATABASES

S.Jayalakshmi[*]

JebaMoses.T[**]

Abstract—

This paper, provide a secure mining in horizontally distributed databases by using association rules. We used Fast Distributed Mining (FDM) algorithm which is unsecured distributed version of the Apriori algorithm. Horizontally Distributed Databases divides the relation tuples, so that first subset of rows is present within the site 1 and another subset is present within the site 2. Original relation is obtained by taking union of all the sets. Association rules are if/then statements that help uncover relationships between unrelated data relational database and other information repository.

Index Terms—Privacy Preserving, Data Mining, Distributed Computation, Frequent Item sets, Association Rules.

[*] B.Tech(IT)-Final year, IFET College of Engineering, villupuram.

[**] Assistant Professor, Department of Information Technology, IFET College of Engineering, villupuram.

1. Introduction

To study here the problem of secure mining of association rules in horizontally partitioned databases. In that setting, there are several sites (or players) that hold homogeneous databases, i.e., databases that share the same schema but hold information on different entities. The goal is to find all association rules with support at least *s* and confidence at least *c*, for some given minimal support size *s* and confidence level *c*, that hold in the unified database, while minimizing the information disclosed about the private databases held by those players.

Herein we propose an alternative protocol for the secure computation of the union of private subsets. The proposed protocol improves upon that in [1] in terms of simplicity and efficiency as well as privacy. In particular, our protocol does not depend on commutative encryption and oblivious transfer (what simplifies it significantly and contributes towards much reduced communication and computational costs). While our solution is still not perfectly secure, it leaks excess information only to a small number (three) of possible coalitions, unlike the protocol of [1] that discloses information also to some single players. In addition, we claim that the excess information that our protocol may leak is less sensitive than the excess information leaked by the protocol of [1].

2. The Fast Distributed Mining algorithm

The Fast Distributed Mining (FDM) algorithm of Cheung et al. [2], which is an unsecured distributed version of the Apriori algorithm. Its main idea is that any *s*-frequent itemset must be also locally *s*-frequent in at least one of the sites. Hence, in order to find all globally *s*-frequent itemsets, each player reveals his locally *s*-frequent itemsets and then the players check each of them to see if they are *s*-frequent also globally. The FDM algorithm proceeds as follows:

(1) Initialization

(2) Candidate Sets Generation

(3) Local Pruning

(4) Unifying the candidate itemsets

(5) Computing local supports

(6) Broadcast Mining Results

2.1. Overview of the paper

The FDM algorithm violates privacy in two stages: In Step 4, where the players broadcast the itemsets that are locally frequent in their private databases, and in Step 6, where they broadcast the sizes of the local supports of candidate itemsets. Kantarcioglu and Clifton [1] proposed secure implementations of those two steps. Our improvement is with regard to the secure implementation of Step 4, which is the more costly stage of the protocol, and the one in which the protocol of [1] leaks excess information. Kantarcioglu and Clifton's secure implementation of Step 4. Then describe our alternative implementation and proceed to analyze the two implementations in terms of privacy and efficiency and compare them. Our protocol offers better privacy and that it is simpler and is significantly more efficient in terms of communication rounds, communication cost and computational cost.

3. Kantarcioglu and Clifton

Protocol 1 is the protocol that was suggested by Kantarcioglu and Clifton [1] for computing the unified list of all locally frequent itemsets, without disclosing the sizes of the subsets nor their contents. The protocol is applied the set of all itemsets that are globally *s*-frequent. Refer to hereinafter as Protocol UNIFI-KC (Unifying lists of locally Frequent Itemsets — Kantarcioglu and Clifton).

Protocol UNIFI-KC works as follows: First, each player adds to his private subset fake itemsets, in order to hide its size. Then, the players jointly compute the encryption of their private subsets by applying on those subsets a commutative encryption, where each player adds, in his turn, his own layer of encryption using his private secret key. At the end of that stage, every itemset in each subset is encrypted by all of the players; the usage of a commutative encryption scheme ensures that all itemsets are, eventually, encrypted in the same manner. Then, they compute the union of those subsets in their encrypted form. Finally, they decrypt the union set and remove from it itemsets which are identified as fake. We now proceed to describe the protocol in detail.

3.1. Secure multiparty protocol

A protocol for computing that function which is much simpler to understand and program and much more efficient than those generic solutions. It is also much simpler than Protocol UNIFI-KC and employs less cryptographic primitives. Our protocol (Protocol 2) computes a wider range of functions, which we call threshold functions.

Two comments are in order:

(1) If the index i had not been part of the input to the hash function (Steps 2-3), then two equal components in P_1's input vector, say $s(i) = s(j)$, would have been mapped to two equal signatures, $s^{¢}(i) = s^{¢}(j)$. Hence, in that case player P_2 would have learnt that in P_1's input vector the ith and jth components are equal. To prevent such leakage of information, we include the index i in the input to the hash function.

(2) An event in which $s^{¢}(i)$ Î Q$^{¢}(i)$ while $s(i)$ Ï Q(i) indicates a collusion; Hash functions are designed so that the prob-ability of such collusions is negligible, whence the risk of a collusion can be ignored. However, it is possible for player P_M to check upfront the selected random key.

Protocol THRESHOLD operates correctly if the in-equality verifications in Step 7 are carried out correctly, since $(s(i) + s_M(i))$ mod $(M + 1)$ equals the ith component $a(i)$ in the sum vector. The inequality verification is correct if Protocol SETINC is correct.

The susceptibility of Protocol THRESHOLD-C to coalitions is not very significant because of two reasons:

- The entries of the sum vector **a** do not reveal information about specific input vectors. Namely, knowing that $a(i) = p$ only indicates that p out of the M bits $b_m(i)$, 1 £ m £ M, equal 1, but it reveals no information regarding which of the M bits are those.
- There are only three players that can collude in order to learn information beyond the intention of the protocol. Such a situation is far less severe than a situation in which any player may participate in a coalition, since if it is revealed that collusion took place, there is a small set of suspects.

4. System architecture

Admin is used to view user details. Admin to view the item set based on the user processing details using association role with Apriori algorithm.

Association rules are if/then statements that help uncover relationships between seemingly unrelated data in a relational database or other information repository.

Association rules are created by analyzing data for frequent if/then patterns and using the criteria support and confidence to identify the most important relationships. Support is an indication of how frequently the items appear in the database. Confidence indicates the number of times the if/then statements have been found to be true.

Apriori is designed to operate on database containing transactions. The purpose of the Apriori Algorithm is to find associations between different sets of data. It is sometimes referred to as "Market Basket Analysis". Each set of data has a number of items and is called a transaction. The output of Apriori is sets of rules that tell us how often items are contained in sets of data.

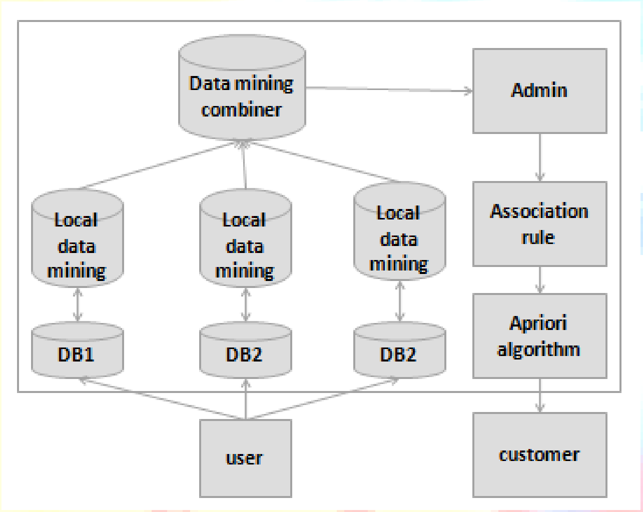


Fig.1 System architecture.

5. Privacy

We begin by analyzing the privacy offered by Protocol UNIFI-KC. That protocol does not respect perfect privacy since it reveals to the players information that is not implied by their own input and the final output. In Step 11 of Phase 1 of the protocol, each player augments the set X_m by fake itemsets. To avoid unnecessary hash and encryption computations, those fake itemsets are random strings in the cipher text domain of the chosen commutative cipher. Hence, every encrypted itemset that appears in two different lists indicates with high probability a true itemset that is locally *s*-frequent in both of the corresponding sites. Therefore, Protocol UNIFI-KC reveals the following excess information:

(1) P_1 may deduce for any subset of the odd players, the number of itemsets that are locally supported by all of them.

(2) P_2 may deduce for any subset of the even players, the number of itemsets that are locally supported by all of them.

(3) P_1 may deduce the number of itemsets that are supported by at least one odd player and at least one even player.

(4) If P_1 and P_2 collude, they reveal for any subset of the players the number of itemsets that are locally supported by all of them.

5.1 Communication cost

By analyze the communication and computational costs of Protocols UNIFI-KC and UNIFI.

In evaluating the communication cost, we consider three parameters: Total number of communication rounds, total number of messages sent, and the overall size of the messages sent.

The communication costs based on two protocols are,

(1) **Communication cost of Protocol UNIFI-KC:** As Protocol UNIFI-KC hashes the itemsets and then encrypts them, *t* should be at least the recommended cipher text length in commutative ciphers.

(2) **Communication cost of Protocol UNIFI:** Protocol UNIFI consists of four communication rounds (in each of the iterations): One for Step 2 of Protocol THRESHOLD that it invokes; one for Step 4 of that protocol; one for Steps 4-5 in Protocol SETINC which is used for the inequality verifications in Protocol THRESHOLD; and one for Step 7 in Protocol SETINC.

5.2 Computational cost

In Protocol UNIFI-KC each of the players needs to perform hash evaluations as well as encryptions and decryptions. As the cost of hash evaluations is significantly smaller than the cost of commutative encryption, we focus on the cost of the latter operations. Since commutative encryption is typically based on modular exponentiation, the overall computational cost of the protocol is $Q(Mt^3n)$ bit operations per player.

In Protocol THRESHOLD, which Protocol UNIFI calls, each player needs to generate $(M-1)n$ (pseudo)random $(\log_2 M)$-bit numbers (Step 1). Then, each player performs $(M-1)n$ additions of such numbers in Step 1 as well as in Step 3. Player P_1 has to perform also $(M-2)n$ additions in Step 5. Therefore, the computational cost for each player is $Q(Mn\log_2 M)$ bit operations. In addition, Players 1 and M need to perform n hash evaluations. Compared to a computational cost of $Q(Mt^3n)$ bit operations per player, we see that Protocol UNIFI offers a significantly improvement with respect to Protocol UNIFI-KC also in terms of computational cost.

6. Experimental setup

To compared the performance of two secure implementations of the FDM algorithm. In the first implementation (denoted FDM-KC), we executed the unification step (Step 4 in FDM) using Protocol UNIFI-KC, where the commutative cipher was 1024-bit RSA [8]; in the second implementation (denoted FDM) we used our Protocol UNIFI, where the keyed-hash function was HMAC [4]. In both implementations, we implemented Step 5 of the FDM algorithm in the secure manner that was described in Section 3. We tested the two implementations with respect to three measures:

1) Total computation time of the complete protocols (FDM- KC and FDM) over all players. That measure includes the Apriori computation time, and the time to identify the globally *s*-frequent itemsets, as described in Section 3. (The latter two procedures are implemented in the same way in both Protocols FDM-KC and FDM.)
2) 2) Total computation time of the unification protocols only (UNIFI-KC and UNIFI) over all players.
3) Total message size.

We ran three experiment set, where each set tested the dependence of the above measures on a different parameter:

-N- the number of transactions in the unified database,

-M- the number of players, and

- S- the threshold support size.

6.1 **Related work**

Previous work in privacy preserving data mining has considered two related settings. One, in which the data owner and the data miner are two different entities, and another, in which the data is distributed among several parties who aim to jointly perform data mining on the unified corpus of data that they hold.

In the first setting, the goal is to protect the data records from the data miner. Hence, the data owner aims at anonymizing the data prior to its release. The main approach in this context is to apply data perturbation [3], [4]. The idea is that the perturbed data can be used to infer general trends in the data, without revealing original record information.

In the second setting, the goal is to perform data mining while protecting the data records of each of the data owners from the other data owners. This is a problem of secure multi- party computation. The usual approach here is cryptographic rather than probabilistic. Lindell and Pinkas [7] showed how to securely build an ID3 decision tree when the training set is distributed horizontally. Lin et al. [6] discussed secure clustering using the EM algorithm over horizontally distributed data. The problem of distributed association rule mining was studied in [5], [9], [10] in the vertical setting, where each party holds a different set of attributes, and in [1] in the horizontal setting. Also the work of [8] considered this problem in the horizontal setting, but they considered large-scale systems in which, on top of the parties that hold the data records (resources) there are also managers which are computers that assist the resources to decrypt messages.

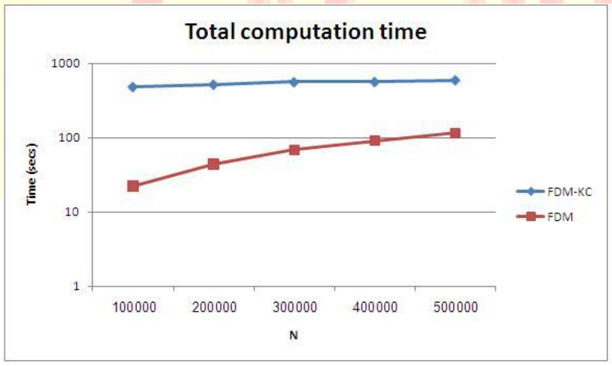


Fig.2 Computation cost

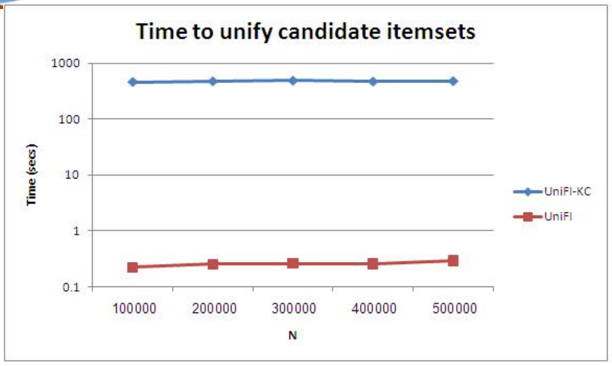


Fig.3 Communication cost

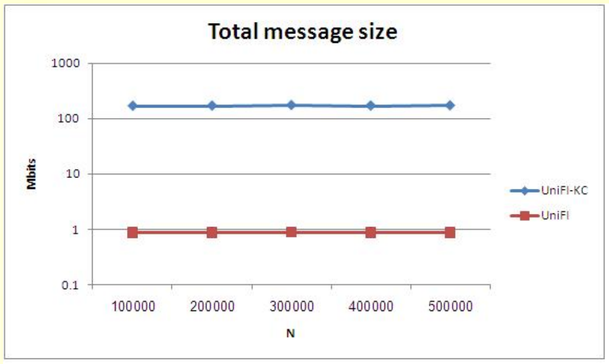


Fig.4 Number of transaction N

8. Result and Discussion

The overall result of Privacy Preserving Mining of Association Rule in Horizontally Distributed Database is discussed with help of screen shots.

The home page contains three different parties. They are client, data owner and server. Each party contains own login page to view their content.

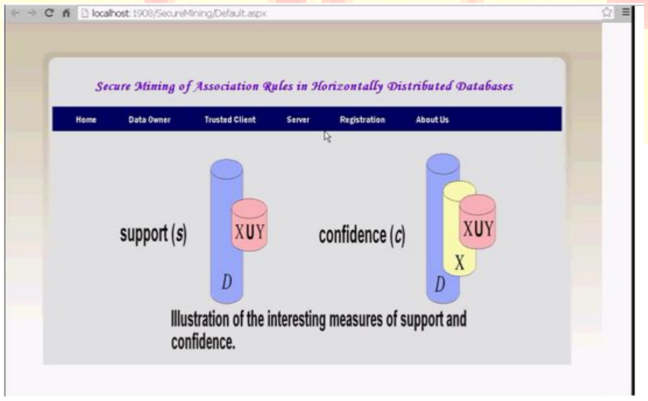
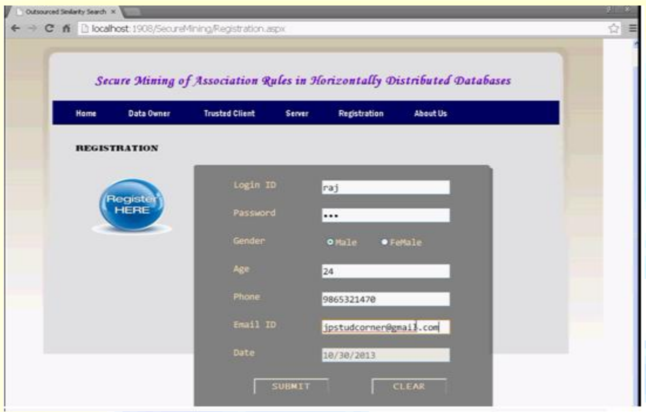

localhost:1908/SecureMining/Default.aspx
Secure Mining of Association Rules in Horizontally Distributed Databases
Home
Data Owner
Trusted Client
Server
Registration
About Us
support (s)
XUY
D
confidence (c)
XUY
X
D
Illustration of the interesting measures of support and confidence.

Fig.5 Home Page

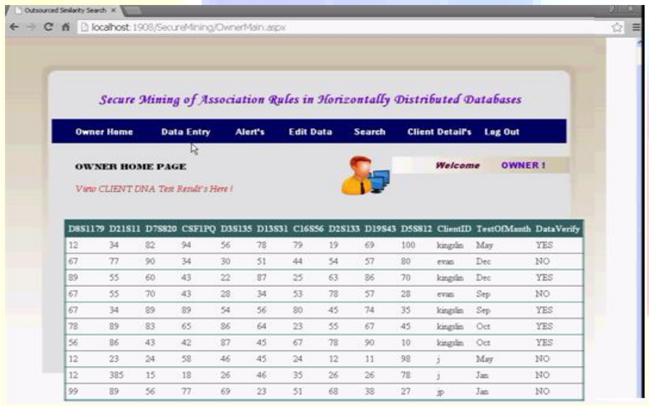
The new client wants to register to get login id and password. The data owner enters the data for particular then only record id is created. The data owner can see a client details and their content. They have right to edit the content but other cannot have rights to change the content (client, server). If server is try to change the content while verification the alert message will send to both the data owner and the client. Through the alert message they understand server trying to change the content.


Outsourced Similarity Search
localhost:1908/SecureMining/Registration.aspx
Secure Mining of Association Rules in Horizontally Distributed Databases
Home
Data Owner
Trusted Client
Server
Registration
About Us
Register HERE
SUBMIT
CLEAR

REGISTRATION

Login ID	raj
Password	•••
Gender	Male FeMale
Age	24
Phone	9865321470
Email ID	jpstudcorner@gmail.com
Date	10/30/2013

Fig.6 Registration page


Outsourced Similarity Search
localhost:1908/SecureMining/OwnerMain.aspx
Secure Mining of Association Rules in Horizontally Distributed Databases
Owner Home
Data Entry
Alert's
Edit Data
Search
Client Detail's
Log Out
Welcome OWNER !

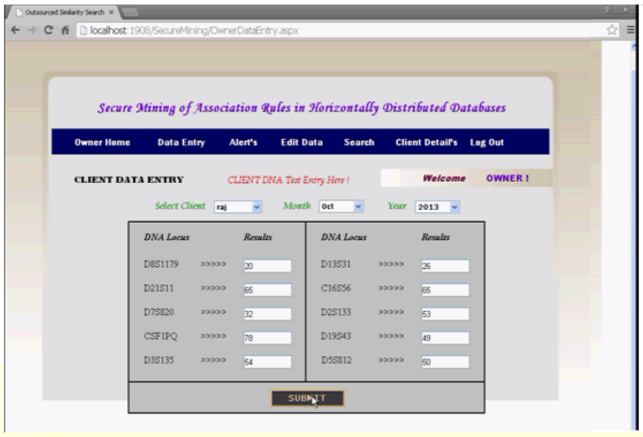
OWNER HOME PAGE

View CLIENT DNA Test Result's Here !

D8S1179	D21S11	D7S820	CSF1PQ	D3S135	D13S31	C16S56	D2S133	D19S43	D5S812	ClientID	TestOfMonth	DataVerify
12	34	82	94	56	78	79	19	69	100	kingslin	May	YES
67	77	90	34	30	51	44	54	57	80	evan	Dec	NO
89	55	60	43	22	87	25	63	86	70	kingslin	Dec	YES
67	55	70	43	28	34	53	78	57	28	evan	Sep	NO
67	34	89	89	54	56	80	45	74	35	kingslin	Sep	YES
78	89	83	65	86	64	23	55	67	45	kingslin	Oct	YES
56	86	43	42	87	45	67	78	90	10	kingslin	Oct	YES
12	23	24	58	46	45	24	12	11	98	j	May	NO
12	385	15	18	26	46	35	26	26	78	j	Jan	NO
99	89	56	77	69	23	51	68	38	27	jp	Jan	NO

Fig.7 Data owner view the client details.

The data owner has only the rights to create client id, record id and key to decrypt the data which is entered by data owner. When the server tries to edit the content but the content will not be change only the alert message will send.


Outsourced Similarity Search
localhost:1908/SecureMining/OwnerDataEntry.aspx
Owner Home
Data Entry
Alert's
Edit Data
Search
Client Detail's
Log Out
SUBMIT

Secure Mining of Association Rules in Horizontally Distributed Databases

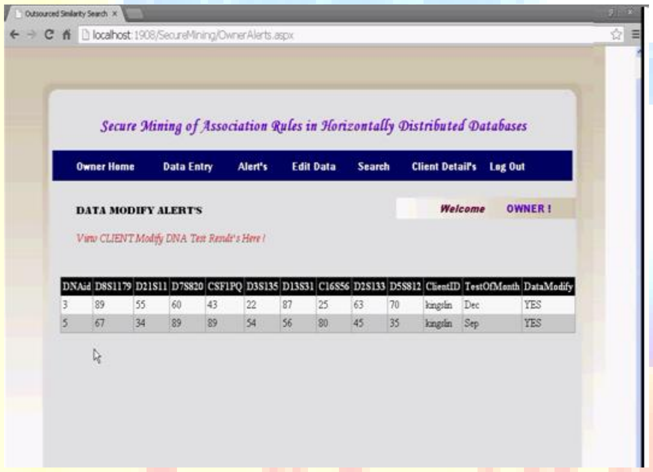
CLIENT DATA ENTRY CLIENT DNA Test Entry Here ! Welcome OWNER !

Select Client raj Month Oct Year 2013

DNA Locus		Results	DNA Locus		Results
D8S1179	>>>>>	20	D13S31	>>>>>	26
D21S11	>>>>>	65	C16S56	>>>>>	65
D7S820	>>>>>	32	D2S133	>>>>>	53
CSF1PQ	>>>>>	78	D19S43	>>>>>	49
D3S135	>>>>>	54	D5S812	>>>>>	50

Fig.8 Client data entry.

The data owner has only the rights to create client id, record id and key to decrypt the data which is entered by data owner. When the server tries to edit the content but the content will not be change only the alert message will send.


Outsourced Similarity Search
localhost:1908/SecureMining/OwnerAlerts.aspx
Owner Home
Data Entry
Alert's
Edit Data
Search
Client Detail's
Log Out

Secure Mining of Association Rules in Horizontally Distributed Databases

DATA MODIFY ALERT'S Welcome OWNER !

View CLIENT Modify DNA Test Result's Here !

DNAid	D8S1179	D21S11	D7S820	CSF1PQ	D3S135	D13S31	C16S56	D2S133	D5S812	ClientID	TestOfMonth	DataModify
3	89	55	60	43	22	87	25	63	70	kingslin	Dec	YES
5	67	34	89	89	54	56	80	45	35	kingslin	Sep	YES

Fig.9 Alert message.

The data owner can view the data in three different ways. They are encrypted hierarchical index search, metric preserving transformation and flexible distance based hashing.

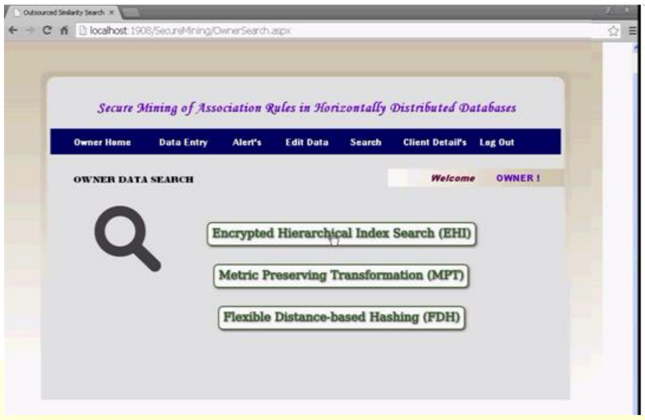
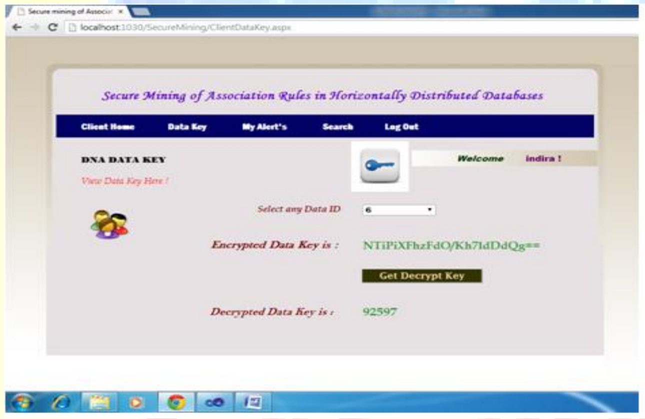

localhost:1908/SecureMining/OwnerSearch.aspx
Secure Mining of Association Rules in Horizontally Distributed Databases
Owner Home
Data Entry
Alert's
Edit Data
Search
Client Detail's
Log Out
OWNER DATA SEARCH
Welcome OWNER !
Encrypted Hierarchical Index Search (EHI)
Metric Preserving Transformation (MPT)
Flexible Distance-based Hashing (FDH)

Fig.10 Owner data search.

To get key for decrypt the data (to view in original format). The key is uniquely generated for each record. The key can view by three of them called data owner, client and server. The key also changed only by data owner. The client and the server are view and use the key but not edit.


localhost:1030/SecureMining/ClientDataKey.aspx
Secure Mining of Association Rules in Horizontally Distributed Databases
Client Home
Data Key
My Alert's
Search
Log Out
DNA DATA KEY
View Data Key Here !
Welcome indira !
Get Decrypt Key

Select any Data ID 6

Encrypted Data Key is : NTiPiXFhzFdO/Kh7ldDdQg==

Decrypted Data Key is : 92597

Fig.11 To get key.

8. Conclusion

I have proposed a protocol for secure mining of association rules in horizontally distributed databases that improves significantly upon the current leading protocol in terms of privacy and efficiency. One of the main ingredients in this proposed protocol is a novel secure multi-party protocol for computing the union (or intersection) of private subsets that each of the interacting players holds. Another ingredient is a protocol that tests the inclusion of an element held by one player in a subset held by another. Those protocols exploit the fact that the underlying problem is of interest only when the number of players is greater than two. An efficient protocol for inequality verifications that uses the existence of a semi honest third party. Such a protocol might enable to further improve upon the communication and computational costs of the second and third stages of the protocol. While the solution is still not perfectly secure, it leaks excess information only to a small number (three) of possible coalitions, unlike the protocol of Fast algorithms for mining association rules in large databases that discloses information also to some single players.

Acknowledgement: I thank for my HOD and Guide.

References:

[1] M. Kantarcioglu and C. Clifton. Privacy preserving distributed mining of association rules on horizontally partitioned data. *IEEE Transactions on Knowledge and Data Engineering*, 16:1026–1037, 2004.

[2] D.W.L. Cheung, J. Han, V.T.Y. Ng, A.W.C. Fu, and Y. Fu. A fast distributed algorithm for mining association rules. In *PDIS*, pages 31– 42, 1996.

[3] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *SIGMOD Conference*, pages 439–450, 2000.

[4] A.V. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke. Privacy preserving mining of association rules. In *KDD*, pages 217–228, 2002.

[5] M. Kantarcioglu, R. Nix, and J. Vaidya. An efficient approximate protocol for privacy-preserving association rule mining. In *PAKDD*, pages 515–524, 2009.

[6] X. Lin, C. Clifton, and M.Y. Zhu. Privacy-preserving clustering with distributed EM mixture modeling. *Knowl. Inf. Syst.*, 8:68–81, 2005.

[7] Y. Lindell and B. Pinkas. Privacy preserving data mining. In *Crypto*, pages 36–54, 2000.

[8] A. Schuster, R. Wolff, and B. Gilburd. Privacy-preserving association rule mining in large-scale distributed systems. In *CCGRID*, pages 411– 418, 2004.

[9] J. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. In *KDD*, pages 639–644, 2002.

[10] J. Zhan, S. Matwin, and L. Chang. Privacy preserving collaborative association rule mining. In *Data and Applications Security*, pages 153– 165, 2005.