A Novel Technique for Secure Mining of Horizontally Distributed Databases: Model and Mechanism

G. Kalyani Rajeswari [#1], P. Srinivas [*2], K .V. N. Rajesh [*3]

[#1] M.TECH, Department of Information Technology, Vignan's Institute of Information Technology, JNTU-KAKINADA,
Andhra Pradesh, India
[#1]rajeswari.k242@gmail.com

[*2] Assistant Professor, Department of Information Technology, Vignan's Institute of Information Technology,
Visakhapatnam, Andhra Pradesh, India
[*2]srinivasp3@gmail.com

[*3]Assistant Professor, Department of Information Technology, Vignan's Institute of Information Technology,
Visakhapatnam, Andhra Pradesh, India
[*3]kvnrajesh@vignanvizag.com

Abstract

We propose a novel protocol for the authentication of the players participating in secure mining of association rules in horizontally distributed databases. The novel protocol which is proposed in this paper is based on the Fast Distributed Mining (FDM) algorithm. Our proposed algorithm mainly computes the union of private subsets that each of the interacting players holds and tests whether the element is present in the subset held by another player imposing authentication to the data held by players for finding the associations. In our proposed application we took an example of university website and the players constitute faculty, student, admin, principal and so on. Our proposed algorithm uses the symmetric key encryption for the inequality verification of the player's data. The symmetric key encryption algorithm we use here is tiny encryption algorithm. Tiny encryption algorithm proves to be efficient in terms of security and response than the other symmetric key encryption algorithms. By conducting several experiments on some university web site data we finally prove that TEA algorithm is efficient in mining horizontal data bases in a secure manner.

Keywords

Secure Mining, Fast Distributed Mining, Tiny Encryption Algorithm, Horizontal Database, Associations.

1. Introduction

Generally, data mining (sometimes called data or) is the process of analyzing data from different perspectives knowledge discovery and summarizing it into useful information - information that can be used to increase revenue, cuts costs, or both. Data mining software is one of a number of analytical tools for analyzing data. It allows users to analyze data from many different dimensions or angles, categorize it, and summarize the relationships identified. Technically, data mining is the process of finding correlations or patterns among dozens of fields in large relational databases [1].

1.1 Major Elements in Data Mining

Data Mining consists of Five Major Elements:

1) Extract, transform, and load transaction data onto the data warehouse system.
2) Store and manage the data in a multidimensional database system.
3) Provide data access to business analysts and information technology professionals.
4) Analyze the data by application software.
5) Present the data in a useful format, such as a graph or table.

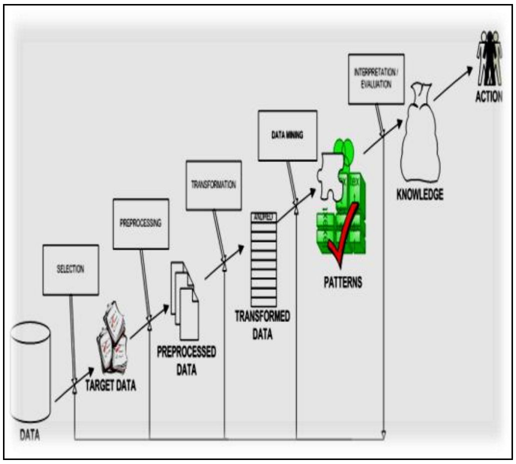


Figure 1.Structure of Data Mining

Data mining is the discovery of the unknown patterns from both heterogeneous and homogeneous database. Secure Data Mining helps to discover association rules which are being shared by homogeneous databases (same schema but the data is present on different entities).The algorithm not only finds the union and intersection of association rules with support and confidence which hold in the total database, while ensuring the data held by players to be authenticated.

Let us assume homogeneous databases having the same schema but the information holds on different entities. The target we need from these databases is that the players data are authenticated using a secured third party. The algorithm after authentication proceeds further to find union and intersection of association rules. The TEA does the job which is symmetric key encryption algorithm for the third party authentication. Its simplicity and performance proves to be part with resource intensive algorithms such as DES (Data Encryption Standard). It can be written into any program on any computer as it is short and uses less set up time. The algorithm can be extended in such a way as a-prior to the distributed case using the following lemma:

If a rule has support > k% globally, it must have support > k% on at least one of the individual sites. A distributed algorithm for this would work as follows:

Request that each site send all rules with support at least k. For each rule returned, request that all sites send the count of their transactions that support the rule, and the total count of all transactions at the site. From this, we can compute the global support of each rule and (from the lemma) be certain that all rules with support at least k have been found. The above approach protects individual data privacy, but it does require that each site disclose what rules it supports, and how much it supports each potential global rule.

1.2 Preliminaries

In this section we will be mention the preliminaries that are used in solving this approach.

1.2.1 The Fast Distributed Mining Algorithm

The protocol of [1], as well as our current algorithm, are based on the Fast Distributed Mining (FDM) algorithm of author Cheung et al. [2], which is an unsecured distributed version of the Apriori, be also locally s-frequent in at least one of the sites. Hence, in order to find all globally s-frequent item sets, each player reveals his locally s-frequent item sets and then the players check each of them to see if they are s-freq check each of them to see if they are s-frequent also globally.

The FDM algorithm proceeds as follows:

(1) Initialization: It is assumed that the players have already jointly calculated F_s^{k-1} . The goal is to proceed and calculate F_s^{k}.

(2) Candidate Sets Generation: Each player P computes the set of all (k − 1)-item sets that are locally frequent in his site and also globally frequent; namely, P_m computes the set $F_s^{k-1, m}$. He then applies on that set the Apriori algorithm in order to generate the set B of candidate k-item sets.

(3) Local Pruning: For each X € $B_s^{k,m}$, P_m computes $supp_m$ (X). He then retains only those item sets that are locally s-frequent. We denote this collection of item sets by $C_s^{k,m}$.

1.2.2 A Practical Example

Let D be a database of N =18 itemsets over a set of L =5 items, A = {1, 2, 3, 4, 5}. It is partitioned between M =3 players, and the corresponding partial databases are:

D = {12, 12345, 124, 1245, 14, 145, 235, 24, 24}
D = {1234, 134, 23, 234, 2345}
D = {1234, 124, 134, 23}.

Setting s =1 /3, an item set is s-frequent in D if it is supp(X ∪ Y)/supp(X).) Supported by at least 6=sN of its transactions. In this case,

F_s^{1} = {1, 2, 3, 4}

F_s2 = {12, 14, 23, 24, 34}

$F_s 3$ = {124}

F_s^{4} = F = ∅

2. Literature Survey

In this section we will describe the assumptions that are used in the proposed paper. This section mainly surveys on the literature of our proposed project.

2.1 Frequent Itemset Example

To illustrate the problem of data mining of frequent occurring patterns, consider a sample database of transactions shown in Figure 2. The set of items for a given transaction could be the buying habits of users, such as, books written by Jane Austen, Agatha Chris tie, Sir Arthur Conan Doyle, etc.

Transaction	Items
1	ACTW
2	CDW
3	ACTWHG
4	ACDWHF
5	ACDTWHGK
6	CDTHFK
7	HFKQR
8	HGKQR
9	QRS
10	QRS

This example has been adapted from Lin 2003.
Figure 2 Sample Database

The database consists of seven transactions with twelve different items. Let θ denote the set of items in the database. A set N ⊆ θ consisting of items from the database is called an Item set.

For example, N = {A, C, D} is an itemset. For notational convenience, we will write ACD to denote the itemset N consisting of items A, C, and D. Suppose that one is interested in identifying the item sets that occur in at least 2 transactions (i.e., the set of authors whose books are commonly bought). Given the sample database, the item sets are A, C, D, H, F, K, Q, and R. A common y used terminology in the data mining literature to denote the number of transactions in which an itemset occurs as a subset is support. The problem of finding patterns in the database can be restated as: identify the itemsets that have at least the user-specified level of support. The user-specified level of support is known as minimum support (or minsup for short) and itemsets that satisfy minsup are known as frequent itemsets [3],[4].

2.2 Data Representation

The transactions database can be viewed as a two-dimensional matrix: the rows represent individual transactions and the columns represent items. For designing data mining algorithms, the data can be represented in either a horizontal view or a vertical view [Lin2003]:

Horizontal view

It consists of representing each row with a unique transaction identifier and a bitmap to represent the items involved in the transaction. For example, if there could be 10 items involved in a transaction, then the bit-string 1000100010 means those items 1, 5, and 9 were involved.

Vertical view

It consists of assigning a unique identifier to each column (i.e., item) and a bitmap that represents the transactions in which that particular item is involved. For example, if there are1610 transactions that involve a particular item, then the bit string 1000100010 means that transactions 1, 5, and 9 are involved.

In their paper, Lin et. al [Lin2003] suggests that a vertical representation is a natural choice for mining frequent itemsets. This is because the vertical representation allows operating on only those item sets that are frequent. Furthermore, for itemsets that are not frequent, their associated bitmap representation can be discarded, thereby leading to a reduced memory footprint. Consequently, Lin et. al uses a vertical representation in their algorithm.

3. Identifying the Globally S-Frequent Itemsets

Protocols UNIFI-KC and UNIFI yield the set C that consists of all item sets that are locally s-frequent in at least one site. Those are the k-item sets that have potential to be also globally s-frequent. In order to reveal which of those item sets is globally s-frequent there is a need to securely compute the support of each of those item sets. That computation must not reveal the local support in any of the sites [6],[7]. Let x be one of the candidate item sets in C. Then x is globally s-frequent if and only if

$$\Delta(x) := supp(x) - sN = \sum_{m=1}^{M} (supp_m(x) - sN_m) \geq 0\,.$$

We describe here the solution that was proposed by Kantar- cioglu and Clifton. They considered two possible settings. If the required output includes all globally s-frequent item sets, as well as the sizes of their supports, then the values of $\Delta(x)$ can be revealed for all x. In such a case, those values may be computed using a secure summation protocol (e.g. [5]), where the private addend of P is $supp_m$ (x) – sN_m.

4. Methodology of Proposed Paper

In this section we mainly concentrate on the methodology or algorithm that we are using in the proposed application. In this paper we are mainly using ATEA algorithm for mining horizontally distributed data bases.Now let us discuss about the proposed algorithm in detail.

Advanced Tiny Encryption Algorithm (ATEA)

In cryptography, the **Advanced Tiny Encryption (ATE) Algorithm** is a new block cipher notable for its simplicity of description and implementation, typically a few lines of code. It was designed by David Wheeler and Roger Needham of the Cambridge Computer Laboratory; it was first presented at the Fast Software Encryption workshop in Leuven in 1994, and first published in the proceedings of that workshop which is clearly shown in figure 3.

XTE Algorithm

In cryptography, **XTE** (**eXtended TE Algorithm**) is a block cipher designed to correct weaknesses in ATE Algorithm. The cipher's designers were David Wheeler and Roger Needham of the Cambridge Computer Laboratory, and the algorithm was presented in an unpublished technical report in 1997 (Needham and Wheeler, 1997). It is not subject to any patents.Like TE , XTE is a 64-bit block Feistel cipher with a 128-bit key and a suggested 64 rounds. Several differences from TE are apparent, including a somewhat more complex key-schedule and a rearrangement of the shifts, XORs, and additions.

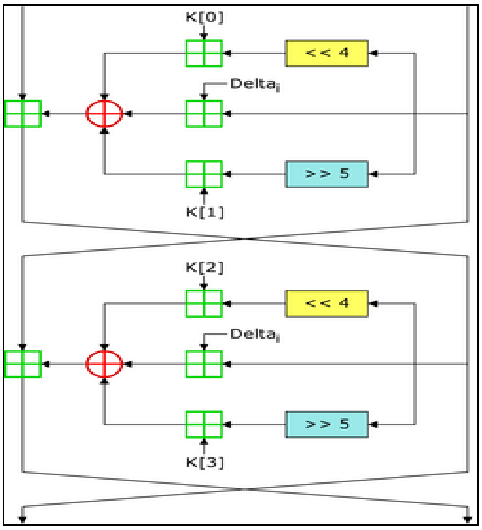
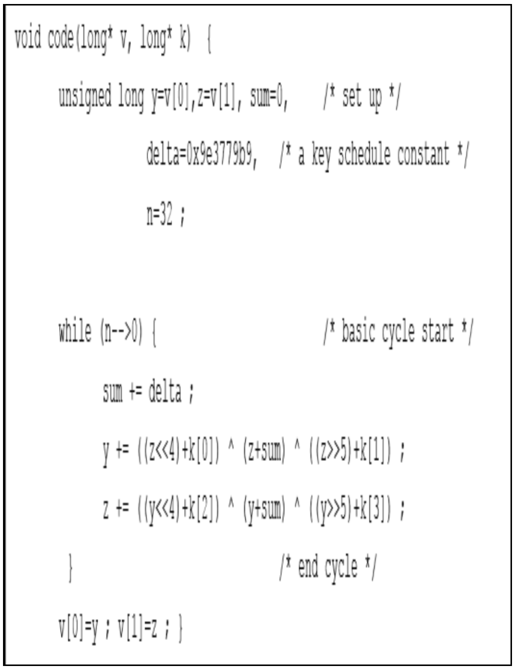


Figure 3.Two Feistel rounds (one cycle) of TE Algorithm

Block TE Algorithm:

Which uses the XTE round function, but Block TE applies it cyclically across an entire message for several iterations. Because it operates on the entire message, Block TE has the property that it does not need a mode of operation. An attack on the full Block TE was described in (Saarinen, 1998), which also details a weakness in Block TE's successor, XXTE.

Pseudo-code for ATE Algorithm



```
void code(long* v, long* k)  {
    unsigned long y=v[0],z=v[1], sum=0,     /* set up */
              delta=0x9e3779b9,   /* a key schedule constant */
              n=32 ;

    while (n-->0) {                       /* basic cycle start */
        sum += delta ;
        y += ((z<<4)+k[0]) ^ (z+sum) ^ ((z>>5)+k[1]) ;
        z += ((y<<4)+k[2]) ^ (y+sum) ^ ((y>>5)+k[3]) ;
    }                              /* end cycle */
    v[0]=y ; v[1]=z ; }
```

5. Implementation Modules

Implementation is the state where the theoretical design is originally converted into practical design. This is divided into following four modules. They are as follows:

5.1 Main Modules

1. Privacy Preserving Data Mining
2. Distributed Computation
3. Frequent Itemsets
4. Association Rules

1. Privacy Preserving Data Mining:

This is one of the main module, 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 anonym zing the data prior to its release. The main approach in this context is to apply data perturbation. The idea is that. Computation and communication costs versus the number of transactions *N* 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 multiparty computation. The usual approach here is cryptographic rather than probabilistic.

2. Distributed Computation:

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

1) Total computation time of the complete protocols (FDMKC and FDM) over all players. That measure includes the Apriori computation time, and the time to identify the globally *s*-frequent item sets, as described in later.

2) Total computation time of the unification protocols only (UNIFI-KC and UNIFI) over all players.
3) Total message size. We ran three experiment sets, where each set tested the dependence of the above measures on a different parameter:

• *N* — the number of transactions in the unified database,

3. Frequent Itemsets:

We describe here the solution that was proposed by Kantarcioglu and Clifton. They considered two possible settings. If the required output includes all globally *s*-frequent item sets, as well as the sizes of their supports, then the values of $\Delta(x)$ can be revealed for all. In such a case, those values may be computed using a secure summation protocol, where the private addend of *Pm* is *suppm*(*x*) − *sNm*. The more interesting setting, however, is the one where the support sizes are not part of the required output. We proceed to discuss it.

4. Association Rules:

Once the set *Fs* of all *s*-frequent itemsets is found, we may proceed to look for all (*s*, *c*)-association rules (rules with support at least *sN* and confidence at least *c*). In order to derive from *Fs* all (*s*, *c*)-association rules in an efficient manner we rely upon the straightforward lemma.

5.2 Main Source Code for Current Application

Here we will discuss the main source code for the proposed application implementation. Here I will show the data base logic through which the current application is connecting its frond end user interface with the storage data. For this application we are using JSP as front end and My Sql as back end data base.

Sample Code for Data Base Connection

```
package databaseconnection;

import java.sql.*;
public class databasecon
{
        static Connection con;
        public static Connection getconnection()
        {try{
        Class.forName("com.mysql.jdbc.Driver");
        con=
DriverManager.getConnection("jdbc:mysql://localho
st:3306/secure_mining_horizontally_distributed","ro
ot","admin");
                }
                catch(Exception e)
                {
        System.out.println("class error");
                }
                return con;
        }
}
```

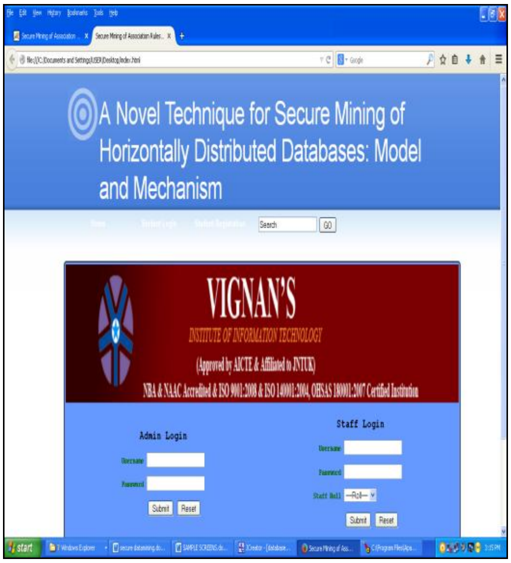
Sample Code for Graph Function

```
<%@ page import="java.sql.*" %>

<%@ page import="java.io.*" %>

<%@ page import="org.jfree.chart.ChartFactory" %>

<%@ page import="org.jfree.chart.ChartUtilities" %>

<%@ page import="org.jfree.chart.JFreeChart" %>

%@page import="org.jfree.chart.plot.PlotOrientation"%

<%@ page import="org.jfree.data.*" %>

<%@pageimport="org.jfree.data.jdbc.JDBCCategoryDatas
et"%>

<% String a2 = request.getQueryString();

session.setAttribute("gp",a2);

String query=
"SELECT sno,total_student,pass_student,fail_student
from ece_perform where semester='"+a2+"'";

JDBCCategoryDataset dataset=new
JDBCCategoryDataset("jdbc:mysql://localhost:3306/dept_
ece_secure_mining","com.mysql.jdbc.Driver","root","admi
n");

dataset.executeQuery(query);

JFreeChart chart = ChartFactory
.createBarChart3D("StudentPerformance","Details","Num
bers",dataset,PlotOrientation.VERTICAL,true, true, false);

try

{
ChartUtilities.saveChartAsJPEG(new File("F:/apache-
tomcat6.0.18/webapps/Secure_Mining_of_Horizontal_Dat
abase/Graph/graph_ece.jpg"), chart, 600, 400);

response.sendRedirect("graph_ece_all_semester.jsp");

}catch (IOException e)
{

System.out.println("Problem in creating chart.");
}
%>
```

6. Experimental Results

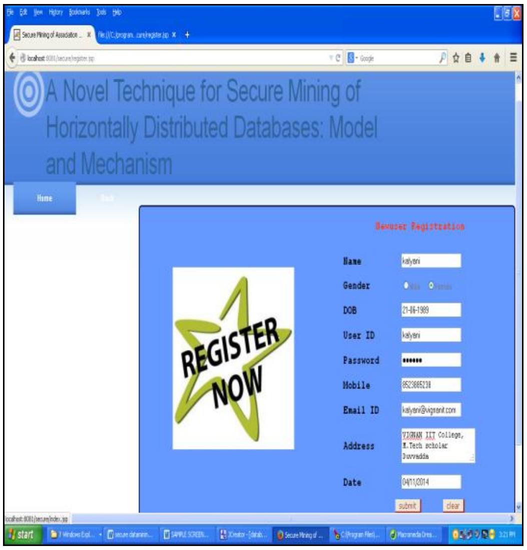
In this paper, we have proposed a new secure algorithm called as ATE Algorithm for mining horizontally distributed data bases.For example the students personal information which is stored in an college or university, it should be accessed by any other person who is of same college and department or of other college or department, who work under an chain of educational wing. For this reason only we have examined the proposed paper on behalf of university campus environment by providing privacy for data publishing. We have shown the application in a web interface with JEE 6.0 edition .In this JEE we are using front end as Java Server pages (JSP) and HTML pages. As we are deploying the application in web interface, we are using tomcat server for deploying the application .Hence we use tomcat 7.0 as the deployment web server. We are using My Sql data base for storing the data temporarily on to our system and then retrieve the same data whenever needed. This MY-SQL data base is chosen on highest priority as it has GUI support and it is always Auto Commit by its nature.

Sample Screens


A Novel Technique for Secure Mining of Horizontally Distributed Databases: Model and Mechanism

VIGNAN'S
INSTITUTE OF INFORMATION TECHNOLOGY
(Approved by AICTE & Affiliated to JNTUK)
NBA & NAAC Accredited & ISO 9001:2008 & ISO 14001:2004, OHSAS 18001:2007 Certified Institution

Welcome Page of the current application


A Novel Technique for Secure Mining of Horizontally Distributed Databases: Model and Mechanism

Newuser Registration

Name	kalyani
Gender	Male Female
DOB	21-06-1989
User ID	kalyani
Password	••••••
Mobile	8523885238
Email ID	kalyani@vignanit.com
Address	VIGNAN IIT College, M.Tech scholar Duvvadda
Date	04/11/2014

Sample screen for registration of Students

7. Conclusion

Mining of association rules in horizontally distributed databases introduces some security issues. To overcome the security problems we devised ATEA to act as the firewall undergoing for the mining activities. Here we used Fast Distributed Mining to find the union and intersection of association rules for the distributed databases. ATEA is symmetric key algorithm for the third party authentication. Its performance, strength, simplicity proves to far better than some symmetric key encryption algorithm s. We can replace ATEA with asymmetric key encryption algorithms to increase its performance but the response time would be more than ATEA.

8. Future Enhancement

In the future we can extend the research to mine generalized association rules, multiple level association rules, and quantitative rules. Other research problems that this study suggests is the implementation of the techniques presented here to the problem of distributed association rule mining in the vertical setting especially in all other domains include medical, shopping, banking and so on.

9. 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] M. Bellare, R. Canetti, and H. Krawczyk. Keying hash functions for message authentication. In Crypto, pages 1–15, 1996.

[4] A. Ben-David, N. Nisan, and B. Pinkas. FairplayMP - A system for secure multi-party computation. In CCS, pages 257–266, 2008.

[5] J.C. Benaloh. Secret sharing homomorphisms: Keeping shares of a secret secret. In Crypto, pages 251–260, 1986.

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

[7] R. Fagin, M. Naor, and P. Winkler. Comparing Information Without Leaking It. Communications of the ACM, 39:77–85, 1996.