Abstract - Privacy and security issues in data mining become an important property in any data mining system. A considerable research has focused on developing new data mining algorithms that incorporate privacy constraints. In this paper, we focus on privately mining association rules in vertically partitioned data where the problem has been reduced to privately computing Boolean scalar products. We propose a modification of steganography-based multiparty protocols for this problem. The proposed modification fine tune the performance to be faster in case of very large database, with acceptable level of reduction in privacy.

I. INTRODUCTION

Privacy preserving data mining is an important property that any mining system must satisfy. There are many methods for privacy preserving distributed association rule mining across private databases. So these methods try to compute the answer to the mining without revealing any additional information about user privacy. An application that needs privacy preserving distributed association rule mining across private databases, like medical research. There are some existing techniques that can be used for building this application, but they are inadequate related to some disadvantages. One from these techniques is trusted third party. The main parties give the data to a "trusted" third party and have the third party do the computation [1]. However, the third party has to be completely trusted, both with respect to intent and competence against security breaches. The level of trust required is too high for this solution to be acceptable. Also data perturbation technique has different idea, the idea is that the distorted data does not reveal private information, and thus is "safe" to use for mining. The key result is that the distorted data, and information on the distribution of the random data used to distort the data, can be used to generate an approximation to the original data distribution, without revealing the original data values. The distribution is used to improve mining results over mining the distorted data directly, primarily through selection of split points to "bin" continuous data. Later refinement of this approach tightened the bounds on what private information is disclosed, by showing that the ability to reconstruct the distribution can be used to tighten estimates of original values based on the distorted data [2].

Another approach is secure multi-party computation. In this approach given two parties with inputs x and y respectively, the goal of secure multi-party computation is to compute a function f(x,y) such that the two parties learn only f(x,y), and nothing else [3]. In [4] an efficient protocol for Yao’s millionaires’ problem showed that any multi-party computation can be solved by building a combinatorial circuit, and simulating that circuit. A variant of Yao’s protocol is presented in [5] where the oblivious transfers is used to make secure decision tree learning using ID3 with efficient cryptographic protocol. There are many methods for privacy preserving distributed association rule mining across private databases when databases partitioned vertically. The problem is reduced to compute scalar product between these databases [6]. Any one can use set-intersection protocols for online recommendation services, online dating services, medical databases, and many other applications and also for mining over vertical data bases. There are some existing techniques that one might use for solving the problem of private scalar product but they have some problem like increasing the running time of computing the scalar product [7]. In any method of the above, the main concentration is to make better privacy preserving with high performance. When the data base goes large the overhead of adding the privacy will degrades the performance, so some algorithms are proposed to solve this problem of mining very large databases as in [8],[9]. To gain high performance with acceptable level of privacy in case of large database, we need fast technique for computing scalar product. So we try to reduce the computation time of computing scalar product, by using smaller matrix to hide the vectors used in computing the scalar product. In this work we propose a modification of steganography-based multiparty protocol for computing scalar product. Our modification gives suitable solution for tradeoff between the performance and privacy.

The organization of this paper is as follows. An overview about the problem and related work in the area of privacy preserving data mining for association rule mining on distributed heterogeneous (vertically partitioned) databases given in section 2. Section 3 shows the details of modified algorithm of computing the scalar product, illustrative examples, and security analysis and evaluation metrics are presented. Section 4 describes the implementation and results of modified an algorithm versus the old algorithm. Finally, some conclusions are put forward in Section 5.

II. DISTRIBUTED ASSOCIATION RULE MINING AND PROBLEM DEFINITION

Association rule mining is one of the most important data mining techniques used in many real life applications. It is used to reveal unexpected relationships in the data. By assuming heterogeneous databases: each site has different schema. The goal is to produce association rules that hold globally, while limiting the information shared about each site to preserve the privacy of data in each site.

Let there be k parties S1, S2, . . . , Sk. We consider a heterogeneous database scenario, a vertical partitioning of the database between the k parties. The association rule mining
problem can be formally stated as follows: Let \( I = \{i_1, i_2, \ldots, i_n\} \) be a set of literals, called items. Let \( D \) be a set of transactions, where each transaction \( T \) is a set of items such that \( T \subseteq I \). Associated with each transaction is a unique identifier, called its TID. We say that a transaction \( T \) contains \( X \), a set of some items in \( I \), if \( X \subseteq T \). An association rule is an implication of the form \( X \Rightarrow Y \), \( X \subseteq I \), and \( Y \subseteq I \), and \( X \cap Y = \emptyset \). The rule \( X \Rightarrow Y \) holds in the transaction set \( D \) with confidence \( c \) if \( c \% \) of transactions in \( D \) that contain \( X \) also contain \( Y \). The rule \( X \Rightarrow Y \) has support \( s \) in \( D \) if \( s \% \) of the transactions in \( D \) contain \( X \cup Y \).

The association rule mining problem can be decomposed into two distinct sub problems first generate all combinations of items that have support at least minimum support then for every frequent found in the first step, generate all rules from it with minimum confidence. Most of the work done so far has focused on the first problem since generating the corresponding association rules from the frequent items is a not difficult task. The most known algorithm for mining frequent itemsets is the Apriori algorithm [10]. With this algorithm, the set of transactions is viewed as a database \( D \) with \( n \) rows and \( m \) columns, every row corresponding to a transaction and every column corresponding to an item. Each entry in the database is 0 or 1, specifying the absence or presence of items in the set of transactions. In other words, if the \( i \)th row in the database corresponds to transaction \( t_i \) and the \( j \)th column corresponds to item \( I_j \), then the \( j \)th entry in row \( i \) (denoted by \( t_{i,j} \)) indicate whether or not \( t_i \) contains \( I_j \).

There are two main partitioning methods for distributed data base are horizontal and vertical partitioning. For studying privacy-preserving data mining it is useful to consider how data may be partitioned among the involved parties. In some cases, organizations may collect the same kind of data about different entities (for example people, traffic, etc.). From a database perspective, we may then say that the data is partitioned horizontally; that is, the same schema is used to store the data at each site. In other cases, organizations may organize data using different schemas, meaning that they collect different kinds of data, perhaps on the same entities. We then say that the data is partitioned vertically.

Consider the database is partitioned vertically into two sets of columns, the first set (denoted by \( DB1 \)) consisting of the first \( a \) columns (more precisely, the columns corresponding to items \( I_1, I_2, \ldots, I_a \)), and the second one (denoted by \( DB2 \)) consisting of the remaining \( m-a \) columns (i.e., the columns corresponding to \( I_{a+1}, I_{a+2}, \ldots, I_m \)). Also, consider that we have two parties \( A \) and \( B \), such that \( DB1 \) belongs to party \( A \) and \( DB2 \) belongs to party \( B \). The two parties want to collaboratively find the frequent itemsets in \( DB = DB1 \cup DB2 \) without any party revealing its own vector.

The scalar product between \( \vec{x} \) and \( \vec{y} \) is defined as in equation 3.

\[
\vec{x} \cdot \vec{y} = \sum_{i=1}^{n} x_i y_i
\]

where \( x_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,n}\} \) is the \( i \)th transaction in database \( DB1 \) and \( y_i = \{y_{i,1}, y_{i,2}, \ldots, y_{i,n}\} \) is the \( i \)th transaction in database \( DB2 \).

**Figure 1.** Apriori Algorithm for Vertically Partitioned Data

Determining if the itemset \( \{I_{a1}, I_{a2}, \ldots, I_{ap}, I_{b1}, I_{b2}, \ldots, I_{bq}\} \) is frequent reduces to testing if \( \vec{x} \cdot \vec{y} \geq \text{minimum support} \). Thus, the two parties want to compute the scalar product \( \vec{x} \cdot \vec{y} \) without any party revealing its own vector. This idea can be easily incorporated into the Apriori algorithm. Also this idea can be easily extended to more than two parties.

**A. Related Work**

Within the context of privacy-preserving data mining [11], several private shared scalar product protocols have been proposed. The goal is that one of the participants obtains the scalar product of the private vectors of all parties. Additionally, it is often required that no information about the private vectors, except what can be deduced from the scalar product, will be revealed during the protocol. Moreover, since data mining applications work with a huge amount of data [8], it is desirable that the scalar product protocol is also very efficient. A secure scalar product protocol has various applications in privacy preserving data mining, starting with privacy-preserving frequent pattern mining on vertically distributed database and ending with privacy-preserving cooperative statistical analysis.

**B. Fast Steganography-based Multi-Party Protocols for Privacy-Preserving Association Rule Mining in Vertically Partitioned Data**

Steganography [17] is the science of hiding secret messages in other messages. So this prevents an observer from learning anything unusual is taking place. Some
methods use this concept in the context of computing scalar products under privacy constraints.

In [12] proposed a method for computing the scalar protocol. Main idea is to hide the data of all sites in the protocol in a large matrix with the use of some random variables. These variables used to mark the needed values to obtain the final results. The method working as following:

Let $P_1, P_2, \ldots, P_k$ be $k$ parties, and let $\mathcal{X}_i$ be the Boolean column vector corresponding to party $P_i$ (all vectors have the same size $n$). Also, consider that $P_1$ is the initiator of the protocol. The parties want to collaboratively compute the scalar product $X^* \cdot X$, where $X^*$ is the $t$th column in $M_k$, and the rest of the columns in $M_k$ are randomly generated. The number of columns in $M_k$ (denoted by $q$) is fixed by party $P_1$ (or can be considered as an input parameter for the protocol).

The important thing here is that $t$ is randomly generated by $P_k$ and not revealed to the other parties. Then, for each $i = k, k-1, \ldots, 2$ party $P_i$ sends $M_i$ to party $P_{i-1}$. When $P_{i-1}$ receives $M_i$, it forms a new matrix $M_{i-1}$, such that $\mathcal{X}_i$ is the $t$th column in $M_k$, and the rest of the columns in $M_k$ are randomly generated. The number of columns in $M_k$ (denoted by $q$) is fixed by party $P_1$ (or can be considered as an input parameter for the protocol). The protocol makes use of a randomly generated parameter, denoted by $r$, which is available only to the initiator of the protocol (in our case, $P_1$), and is used to hide the information in it, to enhance the performance.

A. **Proposed new algorithm**

To improve the performance of the protocol which described above in [12], we need to reduce the size of matrix used to hide the information in it, to enhance the performance. Our main idea is to reduce the number of columns. Divide the vector of any site to $x$ sub-vectors and each sub-vector has size $(n/x)$ elements. The matrix that will used to hide the vectors during the protocol has size $n \times 2^x$ where $n$ is the number of elements in any one of vectors used in computing the scalar product and $x$ is number of sub-vectors. Then apply the algorithm as in figure 2.

![Figure 2](image-url)

**Figure 2** proposed new algorithm

**B. Illustrative Examples**

We describe the new protocol by considering a running example involving three parties. The extension to more than three parties follows the same idea. Let $P_1, P_2, P_3$ be three parties that want to collaboratively compute the scalar product $LV_1 \cdot LV_2 \cdot LV_3$, where

\[
LV_{1,s_1} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad LV_{2,s_1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad LV_{3,s_1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}
\]

are the vectors associated with $P_1$, $P_2$, $P_3$, respectively. Each of the vectors has size $n = 5$. Let $P_1$ be the initiator of the protocol. The protocol makes use of a randomly generated parameter, denoted by $r$, which is available only to the initiator of the protocol (in our case, $P_1$), and is used to hide the input vectors within the matrices constructed during the protocol. In our running example, we choose the number of sub vectors is $x = 3$ then we need three random number $r_1, r_2, r_3$ and the size of the sub vector is $n/x$.

**Step 1.** First, $P_1$ forms an $(n \times 2^x)$ matrix where $r_1$th column of $M$ is first sub-vector and $r_2$th column of $M$ is second sub-vector and $r_3$th column of $M$ is third sub-vector and the rest of the entries are randomly generated. Then, $P_1$ sends $M$ to $P_2$.

\[
M_{s_1,s_2} = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}
\]

**Step 2.** Upon receiving $M$ from $P_1$, $P_2$ forms an $(n \times 2^x)$ matrix where $M[l][c] = X_2[l] \cdot M[l][c]$ for all $l, c$. Then, $P_2$ sends $M$ to $P_3$. And when received at $P_3$ make the same thing as in $P_2$. 

...
At site 2:
\[
M_{1,i} = \begin{bmatrix}
1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 
\end{bmatrix}
\]

At site 3:
\[
M_{2,i} = \begin{bmatrix}
1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 
\end{bmatrix}
\]

Finally send \(M\) to \(P_1\).

**Step 3.** Construct vector \(P\) that \(P[(i-1)x+j] = M[j][ri]\) where \(i = 1,2,\ldots,x\) and \(j = 1,2,3,\ldots,n/x\). Then publish the product
\[
P = \sum_{i=1}^{n} P[i]
\]

We can extend the above idea by using larger vectors. Suppose that we have five vectors with number of elements equal to 256. We choose size of sub-vectors to be 4 then we have 64 sub-vectors. The matrix used to hide the vectors will be 256 x \(2^6\). As above we will insert the sub-vectors of first site in the new matrix and apply the above algorithm as in figure 2.

C. **Security Analysis and evaluation metrics**

To ensure that our protocol preserve the privacy we need to proof that, in addition to privacy we also measure performance to evaluate our protocol using the computation cost of running the algorithm.

- **Security Analysis**

**Theorem 1:** Our algorithm privately computes the scalar product of any number of vectors present in the database without revealing any private information about users of data base.

**Proof.** To show that our algorithm preserves the privacy can be done by using the idea of simulating every thing during the protocol running to know what data every site see in running the protocol [3]. The proof as following. During the protocol all values in \(M\) is random based in choosing \(X\) and \(Z\) so we will compare our method with old one using the probability that \(k-1\) can know the data of site \(k\). For our protocol the probability is \((1/2^x) \times n/x\) instead of \(1/2^x\). We can say that our new protocol is faster and also with acceptable level of privacy in large data bases.

- **Evaluation metrics**

To measure the performance of our method we use computation cost as performance metrics. Cost estimation for association rule mining using the method we have presented can be computed as following: The number of sites is \(k\). Let the total number entries in local vector \(LV\) is \(n\). The factor that used to make sub-vectors is \(n/x\). Then total computation cost for our protocol is \(O(k^n * 2^x)\) and the algorithm in [12] need \(O(k^n * 2^x)\) and \(2^n\) is number of columns in message of the protocol. So if \(2^n\) and \(2^x\) are close then the two algorithms have the same running time. Our algorithm gives a suitable solution for trade off between the performance and privacy. From this we can say that our protocol is flexible in the size of matrix used to hide the data during the protocol. In very large scale data bases we can use appropriate matrix size to gain better performance.

IV. IMPLEMENTATION AND RESULTS OF PROPOSED METHOD

We implement our new method and algorithm in [12] using java. Our test in data that represent based on 0/1 matrix. By running the new algorithm 500 tests done, and 600 tests for old algorithm on heterogamous data bases with different size from 1000 bytes to 60000 bytes with number of attributes 5 and 10 by 10 tests for every data base. The 10 values for every test are very closed to each other. The values listed in figure 3 are the average values. Testing done by using P4 (2.8 GHZ) with Java (SDK 1.6). In testing we choose our new algorithm with sub-vector size \((n/x)\) equal to 20,10,5,3, and 2.

<table>
<thead>
<tr>
<th>K</th>
<th>N</th>
<th>Algorithm in [12]</th>
<th>New Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1000</td>
<td>0.015476995</td>
<td>0.006159643</td>
</tr>
<tr>
<td></td>
<td>5000</td>
<td>0.091357644</td>
<td>0.046743371</td>
</tr>
<tr>
<td></td>
<td>10000</td>
<td>0.156267739</td>
<td>0.093134072</td>
</tr>
<tr>
<td></td>
<td>20000</td>
<td>0.356476356</td>
<td>0.197512594</td>
</tr>
<tr>
<td></td>
<td>50000</td>
<td>1.0527975</td>
<td>0.539032216</td>
</tr>
<tr>
<td></td>
<td>60000</td>
<td>1.138129</td>
<td>0.6238082</td>
</tr>
<tr>
<td>10</td>
<td>1000</td>
<td>0.018204381</td>
<td>0.010225604</td>
</tr>
<tr>
<td></td>
<td>5000</td>
<td>0.115394613</td>
<td>0.065117794</td>
</tr>
<tr>
<td></td>
<td>10000</td>
<td>0.236252681</td>
<td>0.142883324</td>
</tr>
<tr>
<td></td>
<td>20000</td>
<td>0.467338339</td>
<td>0.280670675</td>
</tr>
<tr>
<td></td>
<td>50000</td>
<td>1.3382765</td>
<td>0.749534475</td>
</tr>
<tr>
<td></td>
<td>60000</td>
<td>1.517329</td>
<td>0.8557666</td>
</tr>
</tbody>
</table>

(a) \(n/x = 2\)

<table>
<thead>
<tr>
<th>K</th>
<th>N</th>
<th>Algorithm in [12]</th>
<th>New Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1000</td>
<td>0.0171710148</td>
<td>0.004915597</td>
</tr>
<tr>
<td></td>
<td>5000</td>
<td>0.07957408</td>
<td>0.029405301</td>
</tr>
<tr>
<td></td>
<td>10000</td>
<td>0.171128647</td>
<td>0.059467752</td>
</tr>
<tr>
<td></td>
<td>20000</td>
<td>0.356956273</td>
<td>0.120856515</td>
</tr>
<tr>
<td></td>
<td>50000</td>
<td>0.932710185</td>
<td>0.302567231</td>
</tr>
<tr>
<td></td>
<td>60000</td>
<td>1.1737853</td>
<td>0.419425222</td>
</tr>
<tr>
<td>10</td>
<td>1000</td>
<td>0.018929445</td>
<td>0.007384402</td>
</tr>
<tr>
<td></td>
<td>5000</td>
<td>0.124361949</td>
<td>0.039841181</td>
</tr>
<tr>
<td></td>
<td>10000</td>
<td>0.225873788</td>
<td>0.085167362</td>
</tr>
<tr>
<td></td>
<td>20000</td>
<td>0.460194522</td>
<td>0.171453284</td>
</tr>
<tr>
<td></td>
<td>50000</td>
<td>1.3394662</td>
<td>0.485033493</td>
</tr>
<tr>
<td></td>
<td>60000</td>
<td>1.63570226</td>
<td>0.594194932</td>
</tr>
</tbody>
</table>

(b) \(n/x = 3\)

<table>
<thead>
<tr>
<th>K</th>
<th>N</th>
<th>Algorithm in [12]</th>
<th>New Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1000</td>
<td>0.019353</td>
<td>0.0002424</td>
</tr>
<tr>
<td></td>
<td>5000</td>
<td>0.08589</td>
<td>0.0138333</td>
</tr>
<tr>
<td></td>
<td>10000</td>
<td>0.174361</td>
<td>0.0237979</td>
</tr>
<tr>
<td></td>
<td>20000</td>
<td>0.342355</td>
<td>0.0511433</td>
</tr>
<tr>
<td></td>
<td>50000</td>
<td>1.010554</td>
<td>0.1369872</td>
</tr>
<tr>
<td></td>
<td>60000</td>
<td>1.106744</td>
<td>0.1772911</td>
</tr>
<tr>
<td>10</td>
<td>1000</td>
<td>0.032366</td>
<td>0.0039643</td>
</tr>
<tr>
<td></td>
<td>5000</td>
<td>0.181384</td>
<td>0.024516</td>
</tr>
<tr>
<td></td>
<td>10000</td>
<td>0.345825</td>
<td>0.0519070</td>
</tr>
<tr>
<td></td>
<td>20000</td>
<td>0.719146</td>
<td>0.1087745</td>
</tr>
<tr>
<td></td>
<td>50000</td>
<td>1.086134</td>
<td>0.1428010</td>
</tr>
<tr>
<td></td>
<td>60000</td>
<td>1.609831</td>
<td>0.2711111</td>
</tr>
</tbody>
</table>

(c) \(n/x = 5\)
In this paper we presented privacy preserving association rule mining algorithms of have been recently introduced with the aim of preventing the discovery of sensible information. The new algorithm is stenography based. We modify an algorithm of privacy preserving association rule mining on distributed heterogeneous data by optimize the computation required for all sites. The results showed that our algorithm solving the trade off between performance and privacy in large data base. Also an implementation for modified algorithm is presented. From the results obtained we can say that our algorithm is good privacy preserving algorithm with better performance. In our work we test the new algorithm with data up to 60000 row, for future we can make more tests in large database up to 1 million row.

REFERENCES