

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

Dragoş Trincă and Sanguthevar Rajasekaran
Department of Computer Science and Engineering
University of Connecticut
Storrs, CT 06269, USA
{dtrinca,rajasek}@enr.uconn.edu

Abstract

Recently, with the emergence of privacy issues in data mining, considerable research has focused on developing new data mining algorithms that incorporate privacy constraints, and, in the same time, are as efficient as possible in terms of accuracy of the results. In this paper, we focus on privately mining association rules in vertically partitioned data, and propose two steganography-based multi-party protocols for this problem. The proposed protocols are shown to be not only at least as secure as other protocols for the same problem, but also much faster. We also present some applications of the proposed steganographic protocols in medicine.

1. Introduction

With the recent concerns about privacy of personal information, considerable work has been done in the relatively new area of privacy-preserving data mining [2, 7]. The goal is not only to develop new data mining algorithms that incorporate privacy constraints, but also to augment the existing ones with privacy-preserving capabilities. Such algorithms should also be fast and as efficient as possible in terms of accuracy of the results. Most of these algorithms are increasingly important in homeland security and counterterrorism-related applications, where the data is usually sensitive.

In this paper, we focus on the problem of privately mining association rules in vertically partitioned data. The association rule mining problem was first proposed by Agrawal et. al in [3], and since then, has become one of the most important problems in data mining. Under privacy constraints, the association rule mining problem was first stud-

ied in [4], where the authors gave a mathematical treatment for a class of randomization algorithms and derived formulae for support and variance prediction, and showed how to incorporate these formulae into mining algorithms. They also presented experimental results that validate their algorithm by applying it on real datasets. In [6], the authors have addressed the privacy-preserving association rule mining problem over horizontally partitioned data. Their methods incorporate cryptographic techniques to minimize the information shared, while adding little overhead to the mining task.

The problem of privately mining association rules in vertically partitioned data was recently addressed in [11], [10], [12], and [5]. In [11] and [5], the authors have proposed two distinct algebraic protocols for the special case of the problem when there are only two parties sharing the database. Both of these algebraic techniques are interesting, although the protocol given in [5] requires at least twice the bitwise communication cost of the protocol presented in [11]. Another protocol for the special case when there are only two parties sharing the database has been proposed in [12], but their protocol is cryptographic in nature, since it is based on homomorphic encryption. The *only* work that has proposed solutions to the general case when multiple parties share the database is [10], where the author has proposed two distinct cryptographic protocols. In the first one, the author has reduced the problem to secure set intersection and then solved the latter using some cryptographic tools. However, this protocol is shown to produce leakage of information for all parties. The second cryptographic protocol produces no leakage of information, but it is not fast at all (according to the results reported in [10] and this paper). Moreover, as it was also pointed out in [10, 11, 5], cryptographic protocols are not so interesting from a practical point of view, since they usually introduce a large overhead in the computational part of the problem.

In this paper, we propose two steganography-based pro-

*This research has been supported in part by the NSF Grant ITR-0326155.

protocols for this problem. Our protocols are shown to be not only at least as secure as other protocols for the same problem, but also much faster. The paper is organized as follows. In Section 2, we provide the reader with a concise introduction to the problem of privately mining association rules. In Section 3, we propose two steganography-based protocols for evaluating itemsets that preserve the privacy of the individual parties. The security offered by these two protocols is analyzed in Section 4. Results of our implementations are reported in Section 5. In Section 6 we present some applications of the proposed steganographic protocols in medicine. Finally, in the last section, we provide some conclusions and future work directions.

2. Problem definition

The problem of mining association rules, as it was first posed in [3], can be formally stated as follows. Let $\mathcal{I} = \{I_1, I_2, \dots, I_m\}$ be a set of attributes, usually called items, and let \mathcal{D} be a set of transactions. Each transaction in \mathcal{D} is a set $T \subseteq \mathcal{I}$ of items. An *association rule* is an implication of the form $X \implies Y$, where $X \subset \mathcal{I}$, $Y \subset \mathcal{I}$, and $X \cap Y = \emptyset$. The rule $X \implies Y$ has *support* s in \mathcal{D} if $s\%$ of the transactions in \mathcal{D} contain $X \cup Y$. On the other hand, we say that the rule $X \implies Y$ holds with *confidence* c in \mathcal{D} if $c\%$ of the transactions in \mathcal{D} that contain X also contain Y . The problem is to find all association rules with support and confidence above certain thresholds (usually referred to as *minsup* and *minconf*).

The association rule mining problem can be decomposed into two distinct subproblems:

1. Generate all combinations of items that have support at least *minsup*.
2. For every frequent itemset $Y = \{I_{i_1}, I_{i_2}, \dots, I_{i_k}\}$ found in the first step, consider all rules of the form $X \implies I_{i_j}$ ($1 \leq j \leq k$), where $X = Y - \{I_{i_j}\}$. Clearly, every such rule has support at least *minsup*. However, not all such rules may have confidence at least *minconf*. In order to see whether the rule $X \implies I_{i_j}$ is satisfied or not, just divide the support of Y by the support of X . If the ratio is at least *minconf*, then the rule is satisfied; otherwise, it is not.

Most of the work done so far has focused on the first subproblem, since generating the corresponding association rules from the frequent itemsets is a trivial task. The most known algorithm for mining frequent itemsets is the Apriori algorithm [1].

Within this framework, 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, spec-

ifying 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 $t_i[j]$) indicates whether or not t_i contains I_j .

Consider now that the database is partitioned vertically into two sets of columns, the first set (denoted by D_1) consisting of the first a columns (more precisely, the columns corresponding to items I_1, I_2, \dots, I_a), and the second one (denoted by D_2) consisting of the remaining $m - a$ columns (i.e., the columns corresponding to $I_{a+1}, I_{a+2}, \dots, I_m$). Also, consider that we have two parties \mathcal{A} and \mathcal{B} , such that D_1 belongs to party \mathcal{A} and D_2 belongs to party \mathcal{B} . The two parties want to collaboratively find the frequent itemsets in $D = D_1 \cup D_2$ without any party revealing its own database.

```

Input:  $D_1$  and  $D_2$ 
Output: the frequent itemsets in  $D = D_1 \cup D_2$ 
-----
1:  $L_1 \leftarrow \{\text{large 1-itemsets}\}$ 
2:  $k \leftarrow 2$ 
3: WHILE  $L_{k-1} \neq \emptyset$  DO
4:    $C_k \leftarrow \text{Apriori-gen}(L_{k-1})$ 
5:   FOR all candidates  $c \in C_k$  DO
6:     IF all the items in  $c$  are entirely at  $\mathcal{A}$  or  $\mathcal{B}$  THEN
7:       that party independently computes  $c.\text{count}$ 
8:     ELSE
9:       Let  $\mathcal{A}$  have items  $I_{a_1}, I_{a_2}, \dots, I_{a_p}$  and  $\mathcal{B}$ 
10:      have the remaining items  $I_{b_1}, I_{b_2}, \dots, I_{b_q}$ .
11:       $\vec{X}[i] \leftarrow \prod_{j=1}^p t_{\mathcal{A},i}[a_j]$  for all  $i = 1, \dots, n$ 
12:       $\vec{Y}[i] \leftarrow \prod_{j=1}^q t_{\mathcal{B},i}[b_j]$  for all  $i = 1, \dots, n$ 
13:       $c.\text{count} \leftarrow \vec{X} \cdot \vec{Y}$ 
14:     ENDFOR
15:   ENDFOR
16:    $L_k \leftarrow L_k \cup \{c \mid c.\text{count} \geq \text{minsup}\}$ 
17:    $k \leftarrow k + 1$ 
18: ENDWHILE
19: return  $L_1 \cup \dots \cup L_{k-2}$ 

```

Figure 1. The Apriori Algorithm for Vertically Partitioned Data

Assume that the two parties want to find out if the itemset $\{I_{a_1}, I_{a_2}, \dots, I_{a_p}, I_{b_1}, I_{b_2}, \dots, I_{b_q}\}$ is frequent, where $\{I_{a_1}, I_{a_2}, \dots, I_{a_p}\} \subseteq \{I_1, I_2, \dots, I_a\}$ and $\{I_{b_1}, I_{b_2}, \dots, I_{b_q}\} \subseteq \{I_{a+1}, I_{a+2}, \dots, I_m\}$. First, party \mathcal{A} forms an n -dimensional column vector \vec{X} , whereas party \mathcal{B} forms an n -dimensional column vector \vec{Y} , such that the i -th component of \vec{X} (denoted by $\vec{X}[i]$) is $\prod_{j=1}^p t_{\mathcal{A},i}[a_j]$ and the i -th component of \vec{Y} (denoted by $\vec{Y}[i]$) is $\prod_{j=1}^q t_{\mathcal{B},i}[b_j]$, where $t_{\mathcal{A},i}$ is the i -th transaction in database D_1 and $t_{\mathcal{B},i}$ is the i -th transaction in database D_2 . The scalar product between \vec{X} and \vec{Y} is defined as $\vec{X} \cdot \vec{Y} = \sum_{i=1}^n \vec{X}[i] \vec{Y}[i]$. Determining if the itemset

$\{I_{a_1}, I_{a_2}, \dots, I_{a_p}, I_{b_1}, I_{b_2}, \dots, I_{b_q}\}$ is frequent reduces to testing if $\vec{X} \cdot \vec{Y} \geq \text{minsup}$. 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, as shown in Figure 1. This algorithm was proposed in [11], and can be easily extended to more than two parties.

As pointed out in [11], the only part of the algorithm that shares database entries between the two parties is the computation of the scalar product $\vec{X} \cdot \vec{Y}$ in line 13. In [11], the authors have proposed an algebraic protocol for computing the scalar product $\vec{X} \cdot \vec{Y}$ without any party revealing its own vector. However, their protocol works only for two parties. The only multi-party protocols for computing scalar products under these privacy constraints are the ones proposed in [10]. In this paper, we propose two steganography-based protocols that are much faster than the protocols given in [10].

3. Fast steganography-based multi-party protocols for computing scalar products under privacy constraints

Steganography [9] is the science of embedding secret messages in other messages – in a way that prevents an observer from learning that anything unusual is taking place. Here, we use this concept in the context of computing scalar products under privacy constraints. The basic idea of our multi-party protocol is the following. Let $\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_k$ be k parties, and let \vec{X}_i be the boolean column vector corresponding to party \mathcal{P}_i (all vectors have the same size n). Also, consider that \mathcal{P}_k is the initiator of the protocol. The parties want to collaboratively compute the scalar product $\vec{X}_1 \cdot \vec{X}_2 \cdot \dots \cdot \vec{X}_k$ without any party revealing its own vector to the other parties. First, \mathcal{P}_k forms a boolean matrix M_k such that \vec{X}_k 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 apriori by party \mathcal{P}_k (or can be considered as an input parameter for the protocol). The important thing here is that t is randomly generated by \mathcal{P}_k and *not* revealed to the other parties. Then, for each $i = k, k-1, \dots, 2$ party \mathcal{P}_i sends M_i to party \mathcal{P}_{i-1} . When \mathcal{P}_{i-1} receives M_i , it forms a new matrix M_{i-1} , such that $M_{i-1}[l, c] = M_i[l, c] \cdot \vec{X}_i[l]$. Finally, party \mathcal{P}_1 sends to party \mathcal{P}_k a tuple (p_1, p_2, \dots, p_q) , where p_j is the number of 1's in the j -th column of M_1 . At this point, party \mathcal{P}_k knows that $p_t = \vec{X}_1 \cdot \vec{X}_2 \cdot \dots \cdot \vec{X}_k$.

This protocol, denoted by Protocol-I, is given in Figure 2. As one can remark, t is randomly generated from the interval $[1, q]$, so that parties $\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_{k-1}$ do not know which column of the matrix that each of them forms leads to the final scalar product.

Input : $\vec{X}_1, \vec{X}_2, \dots, \vec{X}_k$, and q
Output : the scalar product $\vec{X}_1 \cdot \vec{X}_2 \cdot \dots \cdot \vec{X}_k$

- 1 : \mathcal{P}_k randomly generates a natural number $t \in [1, q]$.
- 2 : \mathcal{P}_k forms an $n \times q$ boolean matrix M_k , where the t -th column of M_k is \vec{X}_k and the rest of the entries in M_k are randomly generated.
- 4 : generated.
- 5 : FOR $i = k$ downto 2 DO
- 6 : Party \mathcal{P}_i sends M_i to party \mathcal{P}_{i-1} . When \mathcal{P}_{i-1} receives
- 7 : M_i , it forms an $n \times q$ boolean matrix M_{i-1} , where
- 8 : $M_{i-1}[l, c] = M_i[l, c] \cdot \vec{X}_i[l]$.
- 9 : ENDFOR
- 10 : \mathcal{P}_1 sends to \mathcal{P}_k the tuple (p_1, p_2, \dots, p_q) , where p_j is the
- 11 : number of 1's in the j -th column of M_1 .

Figure 2. Protocol-I

Example 1 Let $\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3$ be three parties that want to collaboratively compute the scalar product $\vec{X}_1 \cdot \vec{X}_2 \cdot \vec{X}_3$ using Protocol-I, where

$$\vec{X}_1 = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \vec{X}_2 = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \vec{X}_3 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

are vectors of size $n = 6$ each. Also, assume that $q = 4$, $t = 2$, and

$$M_3 = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}.$$

It can be seen that the second column of M_3 is actually \vec{X}_3 . The rest of the entries in M_3 are randomly generated. Also, it can be easily checked that

$$M_2 = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}, M_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}.$$

In the last step of the protocol, \mathcal{P}_1 sends to \mathcal{P}_3 the tuple $(p_1 = 3, p_2 = 3, p_3 = 3, p_4 = 2)$. At this point, \mathcal{P}_3 knows that the scalar product $\vec{X}_1 \cdot \vec{X}_2 \cdot \vec{X}_3$ is $p_t = 3$. Then, \mathcal{P}_3 can send the result (i.e. the scalar product) to \mathcal{P}_1 and \mathcal{P}_2 . Since t is available only to \mathcal{P}_3 , the three parties compute the scalar product without any party revealing its own vector to the other parties. Indeed, when \mathcal{P}_2 receives M_3 from \mathcal{P}_3 , it does not know which of the four columns of M_3 corresponds

to \vec{X}_3 . Also, when \mathcal{P}_1 receives the matrix M_2 from \mathcal{P}_2 , it does not know which of the four columns of M_2 corresponds to the boolean AND between \vec{X}_2 and \vec{X}_3 .

After a careful analysis, one can see that the time complexity of the protocol is $\mathcal{O}(knq)$, since each of the k parties performs $\mathcal{O}(nq)$ basic operations. Except for \mathcal{P}_1 , party \mathcal{P}_i sends to party \mathcal{P}_{i-1} an $n \times q$ boolean matrix. So, the communication complexity of the protocol is also $\mathcal{O}(knq)$.

The security level offered by Protocol-I heavily depends on the parameter q . In fact, the only attack that could lead to information loss works as follows. Suppose that party \mathcal{P}_{k-1} wants to find out which of the q columns of M_k corresponds to \vec{X}_k . In order to do so, \mathcal{P}_{k-1} randomly generates a number $r \in [1, q]$ and concludes that the r -th column of M_k corresponds to \vec{X}_k . The probability that the r -th column of M_k is indeed \vec{X}_k is $\frac{1}{q}$. However, even if $r = t$ (i.e. the r -th column of M_k is indeed \vec{X}_k), party \mathcal{P}_{k-1} simply has no guarantee that the column chosen is indeed the one sought. In other words, there is no attack that would allow \mathcal{P}_{k-1} to find (with probability 1) the components of \vec{X}_k . The same is true for the other parties.

One method to increase the security offered by Protocol-I works as follows. First, party \mathcal{P}_k randomly generates n indices c_1, c_2, \dots, c_n , where $1 \leq c_i \leq q$ for all i . Then, \mathcal{P}_k forms an $n \times q$ boolean matrix M_k , such that $M_k[i, c_i] = \vec{X}_k[i]$ for all i , and the rest of the entries in M_k are randomly generated. For each $i = k, k-1, \dots, 2$, party \mathcal{P}_i sends M_i to party \mathcal{P}_{i-1} . Upon receiving M_i , \mathcal{P}_{i-1} forms a new matrix M_{i-1} such that $M_{i-1}[l, c] = M_i[l, c] \vec{X}_{i-1}[l]$. Finally, \mathcal{P}_1 sends the matrix M_1 to \mathcal{P}_k , who finds that $\vec{X}_1 \cdot \dots \cdot \vec{X}_k = M_1[1, c_1] + \dots + M_1[n, c_n]$. Since only \mathcal{P}_k knows the indices c_1, c_2, \dots, c_n , only \mathcal{P}_k finds the scalar product $\vec{X}_1 \cdot \dots \cdot \vec{X}_k$. Definitely, \mathcal{P}_k can share this scalar product with the other parties. This variant of Protocol-I, denoted by Protocol-II, is given in Figure 3.

We note that, as in Protocol-I, the time complexity of Protocol-II is $\mathcal{O}(knq)$. The communication complexity is $\mathcal{O}(knq)$ as well.

Regarding the security offered by Protocol-II, consider the same probabilistic attack discussed earlier. More precisely, \mathcal{P}_{k-1} wants to find out which of the q entries in $M_k[i, -]$ corresponds to $\vec{X}_k[i]$, for all $i = 1, 2, \dots, n$. In order to do so, \mathcal{P}_{k-1} randomly generates $r_1 \in [1, q], \dots, r_n \in [1, q]$, and concludes that the r_i -th entry in $M_k[i, -]$ is $\vec{X}_k[i]$, for all $i = 1, 2, \dots, n/2$. The probability that $r_i = c_i$ for all $i = 1, 2, \dots, n$ is $\frac{1}{q^n}$. When n is large, this means that even a value of 2 for q might suffice. In the case of \mathcal{P}_{k-1} , if $r_i = c_i$ for at least some i , then \mathcal{P}_{k-1} will have the chance to find partial information about \vec{X}_k , but, as stated earlier, \mathcal{P}_{k-1} simply has no guarantee that $r_i = c_i$ for at least some i . So, in conclusion, if we care about partial information loss, then Protocol-I provides a higher security

level. If we do not care about partial information loss, then Protocol-II is more secure.

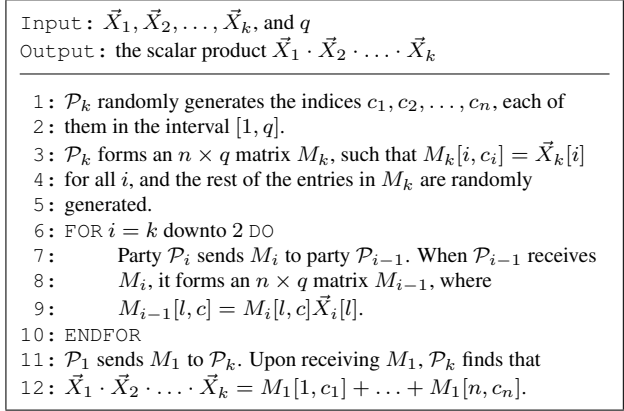


Figure 3. Protocol-II

Example 2 Let $\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3$ be three parties that want to collaboratively compute the scalar product $\vec{X}_1 \cdot \vec{X}_2 \cdot \vec{X}_3$ using Protocol-II, where $\vec{X}_1, \vec{X}_2, \vec{X}_3$ are the vectors from Example 1. Also, assume that $q = 4$, $c_1 = 3$, $c_2 = 1$, $c_3 = 4$, $c_4 = 2$, $c_5 = 2$, $c_6 = 4$, and

$$M_3 = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}.$$

It can be easily seen that $M_3[i, c_i] = \vec{X}_3[i]$ for all i . The rest of the entries in M_3 are randomly generated. Also, it can be easily checked that

$$M_2 = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}, M_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}.$$

In the last step of the protocol, \mathcal{P}_1 sends M_1 to \mathcal{P}_3 . At this point, \mathcal{P}_3 finds that $\vec{X}_1 \cdot \vec{X}_2 \cdot \vec{X}_3 = \sum_{i=1}^6 M_1[i, c_i] = 3$. Since the c_i 's are available only to \mathcal{P}_3 , the parties collaboratively compute the scalar product without any party revealing its own vector to the other parties. Also, it can be easily checked that the amount of work involved is almost the same as in Example 1. However, as discussed earlier, Protocol-II is more secure as long as we do not care about partial information loss.

Since the randomly generated parameter t (or, in the case of Protocol-II, the parameters c_1, \dots, c_n) is available only to \mathcal{P}_k , parties $\mathcal{P}_1, \dots, \mathcal{P}_{k-1}$ have no method to find out which column of the matrix they receive during the protocol leads to the final scalar product. Therefore, we can say that these steganography-based protocols are less vulnerable to key-recovery attacks than the cryptographic protocols proposed in [10].

4. Security

Denote the probabilistic attack described in Section 3 by Attack-I. Consider also the following probabilistic attack, in which one of the parties (say \mathcal{P}_{k-1}) wants to find the vector \vec{X}_k corresponding to \mathcal{P}_k : party \mathcal{P}_{k-1} randomly generates a boolean column vector of size n and concludes that the generated vector is \vec{X}_k . The probability that the randomly generated vector is indeed \vec{X}_k is $\frac{1}{2^n}$, since there are 2^n possible boolean vectors of size n . Denote this probabilistic attack by Attack-II. Attack-II is the *least* efficient attack that can be applied to any protocol for this problem.

However, in the case of Protocol-I, if $q \geq 2^n$ and every possible boolean vector of size n is present within the columns of M_k , then Attack-I is as inefficient as Attack-II. Also, in the case of Protocol-II, if $q \geq 2$ and both values (0 and 1) are present in each row of M_k , then Attack-I is as inefficient as Attack-II. Therefore, in these cases, Protocol-I and Protocol-II are at least as secure as *any* other possible protocol for this problem, since Attack-II can be applied to *any* protocol and Attack-II is also the *least* efficient attack that can be applied to any protocol for this problem. This is indeed true, since Attack-I and Attack-II are the only attacks that can be applied to Protocol-I and Protocol-II.

5. Implementation

We have implemented Protocol-I, Protocol-II, and the second cryptographic protocol proposed in [10]. As in [10], we have implemented their second cryptographic protocol using the Okamoto-Uchiyama public-key cryptosystem [8]. All protocols were tested on a GenuineIntel server with an Intel Pentium 4 CPU at 2.40GHz.

The results are reported in Table 1. As one can see, Protocol-I and Protocol-II are much faster than the second cryptographic protocol proposed in [10], mostly due to the fact that there are no modular exponentiations or other cryptographic functions involved in Protocol-I or Protocol-II. Also, since the time complexities of Protocol-I and Protocol-II are the same, their runtimes are almost the same.

Table 1. Comparisons between our steganography-based protocols (Protocol-I and Protocol-II) and the second cryptographic protocol given in [10] (denoted by OU)

k	n	Protocol-I	Protocol-II	OU
5	1,000	0.02sec	0.03sec	9.27sec
	5,000	0.07sec	0.07sec	25.10sec
	10,000	0.14sec	0.14sec	52.65sec
	50,000	0.66sec	0.68sec	6.67min
	100,000	1.31sec	1.34sec	17.41min
10	1,000	0.03sec	0.03sec	17.70sec
	5,000	0.13sec	0.14sec	49.40sec
	10,000	0.27sec	0.27sec	1.75min
	50,000	1.32sec	1.33sec	13.32min
	100,000	2.63sec	2.70sec	34.86min

6. Applications in Medicine

We can use the above multi-party protocols in the context of comparing health care records in a privacy protecting manner as follows. Consider the case of two parties where each party has a vector of records. Let the records with party one be $R_1^1, R_2^1, \dots, R_n^1$. Treat this sequence as a vector \vec{R}_1 . Let the corresponding records with party two be $R_1^2, R_2^2, \dots, R_n^2$. Let this sequence constitute vector \vec{R}_2 . Assume that we are interested in finding the difference $|R_i^1 - R_i^2|$, for $1 \leq i \leq n$. Assume also that the two parties do not want to share their records with each other and there is a trusted party who is in-charge of computing the differences without knowing the individual records themselves. Under this setting, party one generates an $n \times q$ matrix M . It stores R_i^1 in a random column c_i of matrix M , for $1 \leq i \leq n$. The other elements of M are chosen randomly. Party one then sends this M to party two. Party two modifies M as follows: $M[i, j] = |M[i, j] - R_i^2|$, for $1 \leq i \leq n$ and $1 \leq j \leq q$. The modified M is sent to the trusted party. The trusted party also receives the column numbers c_i (for $1 \leq i \leq n$) from party one. Thus the trusted party will be able to infer the differences of interest from the M it receives from party two. Note that neither party one nor the trusted party gets to see \vec{R}_2 . Also, the trusted party does not get to see \vec{R}_1 . The probability of party two guessing \vec{R}_1 is $\frac{1}{q^n}$. The above technique can be generalized to a multi-party scheme easily. One possibility is for the trusted party to compute pair-wise differences of records (using the above two-party algorithm) and then compute averages.

7. Conclusions and future work

In this paper, we have focused on the problem of privately mining association rules in vertically distributed boolean databases. We proposed two steganography-based multi-party protocols for evaluating itemsets that preserve the privacy of the individual parties. The two protocols are shown to be not only at least as secure as other protocols for the same problem, but also much faster. As future work, it would be interesting to design and test parallel variants of the proposed multi-party protocols.

References

- [1] R. Agrawal, R. Srikant. Fast Algorithms for Mining Association Rules. In: *Proceedings of the 20th International Conference on Very Large Databases*, 1994, pages 487–499.
- [2] R. Agrawal, R. Srikant. Privacy-Preserving Data Mining. In: *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2000, pages 439–450.
- [3] R. Agrawal, T. Imielinski, A. Swami. Mining Association Rules between Sets of Items in Large Databases. In: *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216.
- [4] A.V. Evfimievski, R. Srikant, R. Agrawal, J. Gehrke. Privacy Preserving Mining of Association Rules. In: *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002, pages 217–228.
- [5] I. Ioannidis, A. Grama, M.J. Atallah. A Secure Protocol for Computing Dot-Products in Clustered and Distributed Environments. In: *Proceedings of the 31st International Conference on Parallel Processing*, 2002, pages 379–384.
- [6] M. Kantarcioglu, C. Clifton. Privacy-Preserving Distributed Mining of Association Rules on Horizontally Partitioned Data. In: *Proceedings of the 2002 ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*.
- [7] Y. Lindell, B. Pinkas. Privacy Preserving Data Mining. *Journal of Cryptology*, 15(3):177–206, 2002.
- [8] T. Okamoto, S. Uchiyama. A New Public-Key Cryptosystem as Secure as Factoring. In: *Proceedings of EUROCRYPT'98*, pages 308–318.
- [9] Steganography Links and Whitepapers: available at <http://www.data-hiding.com>.
- [10] J. Vaidya. Privacy Preserving Data Mining over Vertically Partitioned Data. Ph.D. Thesis, Purdue University, August 2004.
- [11] J. Vaidya, C. Clifton. Privacy Preserving Association Rule Mining in Vertically Partitioned Data. In: *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002, pages 639–644.
- [12] J.Z. Zhan, S. Matwin, L. Chang. Private Mining of Association Rules. In: *Proceedings of the 2005 IEEE International Conference on Intelligence and Security Informatics*, pages 72–80.