Efficient Privacy-Preserving Data Mining in Malicious Model

Keita Emura[1], Atsuko Miyaji[2], and Mohammad Shahriar Rahman[2]

[1] Center for Highly Dependable Embedded Systems Technology
[2] School of Information Science
Japan Advanced Institute of Science and Technology
1-1 Asahidai, Nomi, Ishikawa, Japan 923-1292
{k-emura,miyaji,mohammad}@jaist.ac.jp

Abstract. In many distributed data mining settings, disclosure of the original data sets is not acceptable due to privacy concerns. To address such concerns, privacy-preserving data mining has been an active research area in recent years. While confidentiality is a key issue, scalability is also an important aspect to assess the performance of a privacy-preserving data mining algorithms for practical applications. With this in mind, Kantarcioglu et al. proposed secure dot product and secure set-intersection protocols for privacy-preserving data mining in malicious adversarial model using zero knowledge proofs, since the assumption of semi-honest adversary is unrealistic in some settings. Both the computation and communication complexities are linear with the number of data items in the protocols proposed by Kantarcioglu et al. In this paper, we build efficient and secure dot product and set-intersection protocols in malicious model. In our work, the complexity of computation and communication for proof of knowledge is always constant (independent of the number of data items), while the complexity of computation and communication for the encrypted messages remains the same as in Kantarcioglu et al.'s work (linear with the number of data items). Furthermore, we provide the security model in Universal Composability framework.

KeyWords: Privacy-preserving Data Mining, Malicious Model, Threshold Two-party Computation, Efficiency

1 Introduction

1.1 Background

The information age has enabled many organizations to gather large volumes of data through data mining. However, the usefulness of this data is negligible if 'meaningful information' or 'knowledge' cannot be extracted from it. Confidentiality is a key issue that arises in any huge collection of data. The need for privacy is sometimes due to law (e.g., for medical databases) or can be motivated by business interests. However, a key utility of large databases today

is scientific or economic research. Despite the potential gain, this is often not possible due to the confidentiality issues which arise, leading to concerns over privacy infringement while performing the data mining operations.To address this problem, several privacy-preserving distributed data mining protocols using cryptographic techniques have been suggested. Depending on the adversarial behavior assumptions, those protocols use different models. Classically, two main categories of adversaries have been considered:

Malicious adversaries: These adversaries may behave arbitrarily and are not bound in any way to follow a specified protocol. Protocols that are secure in the malicious model provide a very strong security guarantee as honest parties are 'protected' irrespective of an adversarial behavior of corrupted parties.

Semi-honest adversaries: These adversaries correctly follow the protocol specification, yet may attempt to learn additional information by analyzing the transcript of messages received during the execution.

Scalability is another important aspect to assess the performance of a privacy-preserving data mining algorithm. In particular, scalability describes the efficiency trends when data sizes increase. Such parameter concerns the increase of both performance and storage requirements as well as the costs of the communications required by a data mining algorithm when it is used to compute some joint data. For this reason, a privacy-preserving data mining algorithm has to be designed and implemented with the capability of handling huge datasets that may still keep growing. Therefore, the scalability measure is very important in determining practical privacy-preserving data mining techniques.

Kantarcioglu et al. [10] showed several protocols on equality, dot product and set-intersection operations that are useful to design privacy-preserving data mining algorithms in malicious model for the first time. To the best of our knowledge, this is the only work on designing dot product and set-intersection protocols in privacy-preserving data mining area that involves malicious model. They utilize homomorphic encryption with Non-Interactive Zero Knowledge (NIZK) proof to realize their protocols. For a dot product, they provide an efficient protocol designed against malicious adversaries assuming that at least one party is semi-honest. They also prove that the given protocols are secure against malicious adversaries. The computation and communication overhead for the NIZK proofs of their secure dot product protocol can be expressed as $O(n)$ where n is the size of the input dataset, and the complexity of their secure set-intersection is $O(D)$ where D is the domain of data items (Given that party P_0 has n items and party P_1 has m items, where both m and n are bigger than $O(\sqrt{D})$, or where n or m equal to total item domain size D). Therefore, the efficiency of the protocol is highly dependent on the size of the input dataset. In their protocols, for n data items n NIZK proofs are computed and sent over the communication channel to other party.

We estimate the size of n in the real world application. Let us consider a practical scenario, where two parties having huge amount of data want to mine their data in a privacy-preserving way through some joint function using Kantarcioglu et al.'s approach. For an example, according to [17],

- The US Library of Congress, among the largest databases of the world, holds over 125 million items.

Now, if organizations having such very large databases want to compute some data mining algorithm (for an example, two such libraries want to perform data mining operations for research on efficient library-management) in a privacy-preserving way using Kantarcioglu et al's model, the computation and communication complexity for the n NIZK proofs of n data items (n= 125 million) will pose a huge burden on both the computation and communication resources. Each NIZK proof of plaintext used in Kantarcioglu et al.'s work requires 4 exponentiations in $\mathbb{Z}_N$. That means, 500 million exponentiations will be required to be computed!!! In other words, the number of NIZK proof grows linearly with the number of data items n in the protocols proposed by Kantarcioglu et al.; such a performance bottleneck must be addressed to design efficient privacy-preserving data mining algorithms for practical implementation.

Applications of Dot Product and Set-Intersection: K-means clustering is a simple and very commonly used clustering algorithm in data mining. It starts with an unclustered dataset with n elements and one attributes and outputs cluster assignments of each data element in the set. It requires prior knowledge of the number of clusters k [8, 16, 2]. K-means clustering uses dot product and equality protocols as building blocks. Some recent studies [18, 19] provide privacy-preserving association rule mining algorithms using vertically partitioned data. These algorithms involve secure dot product computation with inputs of length n, where n can be arbitrarily large. As for the secure set-intersection, to determine which customers appear on a 'do-not-receive-advertisements' list, a store must perform a set-intersection operation between its private customer list and the producer's list.

1.2 Related Work

Cryptographic techniques have been used to design many different distributed privacy-preserving data mining algorithms. In general, there are two types of assumptions on data distribution: vertical and horizontal partitioning. In the case of horizontally partitioned data, different sites collect the same set of information about different entities. For example, different credit card companies may collect credit card transactions of different individuals. Secure distributed protocols have been developed for horizontally partitioned data for mining decision trees [13], k-means clustering [12], k-nn classifiers [9]. In the case of vertically partitioned data, it is assumed that different sites collect information about the same set of entities but they collect different feature sets. For example, both a university and a hospital may collect information about a student. Again, secure protocols for the vertically partitioned case have been developed for mining association rules [18], and k-means clusters [8, 16]. All of those previous protocols claimed to be secure only in the semi-honest model (we do not consider the proposals which have not used standard cryptographic notions). In [10], authors present two-party secure protocols in the malicious model for data mining. They follow the generic malicious model definitions from the cryptographic literature,

and also focus on the security issues in the malicious model, and provide the malicious versions of the subprotocols commonly used in previous privacy-preserving data mining algorithms. There has been some other works related to secure two-party computation [1, 14]. In [1], the protocol has been shown secure assuming that at least one-party behaves in semi-honest model. However, the protocol requires both parties to engage in a 'proof of decryption' ability (where a sender sends a set of ciphertexts to the receiver and checks whether the receiver can decrypt all the ciphertexts or not), which increases the communication overhead. On the other hand, [14] proposed a two-party protocol to securely evaluate a 2DNF (Disjunctive Normal Form) formula using homomorphic encryption from vector decomposition. But this protocol has been shown secure only in the semi-honest adversarial model. Recently, [7] proposed efficient set operations against the malicious adversaries. It is based on oblivious pseudorandom function evaluation in the standard model. They assume no trusted set up or trusted third party for the computation, thus increasing the communication overhead. But for data mining applications, assuming the existence of a trusted set up is not unrealistic in practice (e.g., a government organization acting as a trusted party may want to perform data mining operations on the data of some hospitals who do not have any trust among themselves).

1.3 Our Contribution

Considering the problems mentioned above, we provide sophisticated modifications that lead to bigger increases in efficiency of the privacy-preserving data mining algorithms. In this paper, we build efficient and secure dot product and set-intersection protocols in the malicious model. Our approach is as follows:

- Each party generates the ciphertexts of its private data using Paillier's encryption.
- Product of all the ciphertexts is computed.
- The NIZK proof is computed using the result of the product, not for each of the ciphertexts.
- All the ciphertexts and the proof of knowledge are sent to other party.
- The receiver first computes the product of the received ciphertexts, and then verifies the proof of knowledge.
- Both the parties use Cramer's threshold two-party computation to jointly compute their data.

We apply the homomorphic property of Paillier cryptosystem to reduce the number of NIZK proofs in our protocols. Although the constructions of our protocols do not deviate a lot from that of [10], we say that the effect of our construction for efficient implementation is huge. Our approach reduces the computational and communication complexity of the proof of knowledge drastically. In our work, computation and communication for the NIZK proofs are always constant, i.e., they are independent of number of data items. In other words, to perform data mining operation on n data items, we need only one NIZK proof. However, the computation and communication for the ciphertexts are linear with the number of data items ($O(n)$, similar to that of [10]). We also provide the

security model in Universal Composability (UC) framework, since it gives a clear view on the security of the protocol in real world scenario.

2 Cryptographic Primitives

2.1 Security Model: Universal Composability (UC)

Security in the UC framework implies that any adversary in the real-life model can be emulated by an adversary in the ideal model. The advantage of this paradigm is that it is possible to show that anything learned by the real-life adversary during the protocol execution is computationally indistinguishable from what is learned by an ideal model adversary. Since in the ideal model, any adversary can learn at most the final result and what is implied by the final result, proving that the real-life model adversary could be simulated by an ideal model adversary implies that real-life adversary could not learn anything more than the ideal model adversary. In other words, the real protocol execution reveal no more information to an adversary than what is revealed to an ideal model adversary. A detailed discussion on the UC framework can be found in [3].

2.2 Homomorphic encryption:

Let $E_{pk}(.)$ denote the encryption function with public key pk and $D_{sk}(.)$ denote the decryption function with private key sk. A public key cryptosystem is called additive homomorphic if it satisfies the following requirements:
(1) given the encryption of plaintexts m_1 and m_2, $E_{pk}(m_1)$ and $E_{pk}(m_2)$, there exists an efficient algorithm to compute the public key encryption of $m_1 + m_2$, such that $E_{pk}(m_1 + m_2) := E_{pk}(m_1) +_h E_{pk}(m_2)$.
(2) given a constant k and the encryption of m_1, $E_{pk}(m_1)$, there exists an efficient algorithm to compute the public key encryption of $k \cdot m_1$, such that $E_{pk}(k \cdot m_1) := k \times_h E_{pk}(m_1)$.

Paillier cryptosystem [15] based on composite residuosity assumption captures the homomorphic property. A detailed description can be found in [15].

2.3 Threshold Two-party Computation

Cramer et. al. proposed multi-party computation with threshold homomorphic cryptosystem in [3]. Given the common public key pk, the private key sk corresponding to pk is divided into two pieces sk_0 and sk_1. There exists an efficient, secure protocol $D_{sk_i}(E_{pk}(m))$ that outputs the random share of the decryption result s_i along with the NIZK proof showing that sk_i is used correctly. Those shares can be combined to calculate the decryption result. Also any single share of the private key sk_i cannot be used to decrypt the ciphertext alone. In other words, s_i does not reveal anything about the final decryption result. We use the same special version of a threshold decryption such that only one party learns the decryption result, as shown in [10]. Such a protocol could be easily implemented exploiting the fact that for any given $E_{pk}(m)$, the party that

needs to learn the decryption result could generate $E_{pk}(r_1)$ and then both parties jointly decrypt the $E_{pk}(m) +_h E_{pk}(r_1)$. Since only one party knows the r_1, only that party can learn the correct decryption result. To prove that a party P_i knows a plaintext, the party can compute the Proof of Plaintext Knowledge PPK(e_m) if he knows an element m in the domain of valid plaintexts such that $D_{sk}(e_m) = m$. Similarly, to prove a multiplication is correct, a party P_i is given an encryption $E_{pk}(m)$, and it chooses constant c and calculates $E_{pk}(m \cdot c)$. Later on, P_i can give the Proof of Correct Multiplication PCM($e_m, e_c, e_{m \cdot c}$) such that $D_{sk}(e_{m \cdot c}) = D_{sk}(e_c) \cdot D_{sk}(e_m)$.

We use the simulators in the security proofs for the above NIZK proofs due to their security properties. The notion of security is such that the state of the adversary returned by those simulators is statistically indistinguishable from the state of the adversary in the real-life model. A different kind of scheme for proof of knowledge named Public Key Encryption with Non-interactive Openning (PKENO) in standard model [4–6, 11] has been proposed, which is more efficient than the known NIZK proofs in standard model. For proving the knowledge, a verifier needs to know the (ciphertext, proof, plaintext)-tuple. But for data mining applications, the plaintext can not be revealed to the verifier. In other words, PKENO is not suitable for our purpose. We use the NIZK proof of [3, 10] in our scheme for fair comparison with the existing works in data mining area.

3 Our Protocols

3.1 Underlying Idea

Our protocol differs from that of [10] in the process of computing the proof of knowledge. After a party completes all the encryptions of its plaintexts, these ciphertexts are multiplied to get a common ciphertext. Then the NIZK proof is constructed for this common ciphertext.

- Let us assume that we have a set of ciphertexts such that $\{c_1 = E_{pk}(m_1), c_2 = E_{pk}(m_2), \ldots, c_n = E_{pk}(m_n)\}$ corresponding to the set of messages $\{m_1, m_2, \ldots, m_n\}$, where n is the number of messages (the ciphertexts are constructed using Paillier's encryption).

- We compute the product of all the ciphertexts such that $C = \prod_{i=1}^{n} c_i = \prod_{i=1}^{n} E_{pk}(m_i)$, and compute the NIZK proof for the product C as $\mathsf{PPK}(C)$. Instead of computing NIZK proofs for each ciphertext c_i where $1 \leq i \leq n$, we have now reduced the number of NIZK proof to one.

- To evaluate $\mathsf{PPK}(C)$, the verifier must compute $C = \prod_{i=1}^{n} c_i = \prod_{i=1}^{n} E_{pk}(m_i)$ first. Then the verifier should verify the $\mathsf{PPK}(C)$. Note that, the plaintext of C is equal to $M := \sum_{i=1}^{n} m_i = (m_1 + m_2 + \ldots + m_n)$, due to the homomorphic property of Paillier's cryptosystem discussed above. The success probability of an adversary A of cheating is $\frac{1}{N}$,where N is a composite number of two big primes, and the probability is negligible (for Paillier encryption, we assume N is 2048-bit (2^{2048}), so the success probability 2^{-2048} is negligible) (See Lemma 1 for proof).

- Note that, we do not avoid the ciphertext computation and communication cost ($O(n)$) to output all of the n ciphertexts, since all of them are necessary for homomorphic encryption to compute the dot product or intersection. However, for practical implementation, the additional costs (i.e., PPK or PCM) must be reduced as much as possible. In our protocol, this additional cost does not depend on the number of items. Due to the reduced number (constant) of proof of knowledge, the following protocols can offer huge savings when the vectors used for the dot product and the set-intersection have many components.

3.2 Efficient and Secure Dot Product Protocol

In a secure dot product protocol, it is required to check whether the final result is correct. This is possible since at least one of the parties will behave semi-honestly. If both parties are malicious, we do not care whether the privacy of any party is protected or parties get correct results. Assuming that at least one party will behave in a semi-honest fashion (the other party can do any malicious act), an efficient protocol can be constructed. Assuming that at least one party is semi-honest is consistent with the definitions of the malicious model. This assumption was also made in [10]. Such an assumption does not reduce the security guarantees provided by the malicious model.

The result of data mining algorithm $res = \sum_{j=1}^{n}(x_{0j} \cdot x_{1j}) + r_1$ is evaluated correctly by at least one party, assuming that at least one party is semi-honest. Both P_0 and P_1 have enough information to calculate res. If both P_0 and P_1 compute the same res value, then computations must be correct, because at least one of them is semi-honest and calculates correct res. Therefore, if we securely make sure that both parties calculate the same value, then either of the local calculations could be decrypted to reveal res to P_0. In our protocol, each party sends the encrypted inputs along with the PPK to each other, then each party P_i locally computes its respective $e_{res^i} = E_{pk}(res^i)$. Both parties jointly decrypt one of those values to reveal res to P_0, given that those two values are equal. We do not need to send PCM for every multiplication. We provide the details of the efficient secure dot product protocol in fig.1.

A note on secure equality protocol: The whole premise of our constructions is that it leads to bigger increases in efficiency when there are a huge number of data items to be processed. However, a secure equality protocol requires to compute whether two data items are equal or not without revealing these items. Therefore, the efficiency of a secure equality protocol under our approach remains same as that in Kantarcioglu et al.'s work [10]. It is straight forward to construct a secure equality protocol in malicious model, due to [10]. For this reason and due to lack of space, we do not include the equality protocol here.

3.3 Efficient and Secure Set-Intersection Protocol

The main idea of this protocol construction is that we can represent the sets owned by each party as a bit vector of size D, and use secure multiplication

Require: Two parties P_0 and P_1 with the shares pr_0 and pr_1 of the private key pr and n bit vectors x_{ij} where x_{ij} belongs to P_i, and $1 \leq j \leq n$.
Ensure: Return $res = \sum_{j=1}^{n}(x_{0j} \cdot x_{1j}) + r_1)$ to P_0 and r_1 to P_1

for all P_i do
  $\forall j$, set $c_{ij} \leftarrow e_{x_{ij}} = E_{pk}(x_{ij})$; $C_i = \prod_{j=1}^{n} c_{ij} = \prod_{j=1}^{n} E_{pk}(x_{ij})$;
  Create $\mathsf{PPK}(C_i)$
  if $P_i = P_1$ then
    pick rand $r_1 \in \{0,1\}^*$, set $C_{r_1} \leftarrow e_{r_1} = E_{pk}(r_1)$, $\mathsf{PPK}(C_{r_1})$
  end if
  Send all encryptions c_{ij} and $\mathsf{PPK}(C_i)$ to P_{1-i}; when $P_i = P_1$, send C_{r_1} and and $\mathsf{PPK}(C_{r_1})$ as well
end for
for P_0 do
  Compute $C_1 = \prod_{j=1}^{n} c_{1j}$; Check $\mathsf{PPK}(C_1)$, $\mathsf{PPK}(C_{r_1})$ are correct else ABORT
  Calculate $e_{res^0} = (E_{pk}(x_{11}) \times_h x_{01}) +_h \ldots +_h (E_{pk}(x_{1n}) \times_h x_{0n}) +_h E_{pk}(r_1)$
  $= e_{x_{01} \cdot x_{11}} +_h e_{x_{02} \cdot x_{12}} +_h \ldots +_h e_{x_{0n} \cdot x_{1n}} +_h e_{r_1}$
  (/$\star$ due to homomorphic property of the encryption $\star$/)
end for
for P_1 do
  Compute $C_0 = \prod_{j=1}^{n} c_{0j}$; Check $\mathsf{PPK}(C_0)$ is correct else ABORT
  Calculate $e_{res^1} = (E_{pk}(x_{01}) \times_h x_{11}) +_h \ldots +_h (E_{pk}(x_{0n}) \times_h x_{1n}) +_h E_{pk}(r_1)$
  $= e_{x_{01} \cdot x_{11}} +_h e_{x_{02} \cdot x_{12}} +_h \ldots +_h e_{x_{0n} \cdot x_{1n}} +_h e_{r_1}$
  (/$\star$ due to homomorphic property of the encryption $\star$/)
end for
Jointly call decrypt equality protocol to check $D_{pr}(e_{res^1}) = D_{pr}(e_{res^0})$
for all P_i do
  If Secure equality protocol returns true for $D_{pr}(e_{res^1}) = D_{pr}(e_{res^0})$ then
    Jointly call private decrypt function s.t. P_0 learns $D_{pr}(e_{res^1})$
  (/$\star$ using threshold two-party computation $\star$/)
  end If
end for

Fig. 1. Our proposed protocol that uses threshold multiparty computation with homomorphic encryption to compute secure and efficient dot product

property of the homomorphic encryption, and we can then associate PPK/PCM proofs to give secure set protocols in the malicious model. Let us assume that x_{0j} is set to 1 if P_0 has item j in its private set else it is set to 0 (similarly for x_{1j} for P_1). Clearly for calculating set-intersection, we need to calculate $x_{0j} \wedge x_{1j}$ for each j. Similarly, for set union, we need to calculate $x_{0j} \vee x_{1j}$ for all j. This can be rewritten as $\neg(\neg x_{0j} \wedge \neg x_{1j})$. Therefore, the dot product protocol for set union can be used, too. The details of the set-intersection protocol is shown in fig. 2. The same protocol can be used for two-party set union using $\neg x_{0j}$ and $\neg x_{1j}$ as the input values and negating the output bits.

Require: Two parties P_0 and P_1 with the shares pr_0 and pr_1 of the private key pr and input bit vectors of size D where x_{ij} is set to 1 if P_i has item j.
Ensure: Return D bit vector I representing the set-intersection where I_j is set to 1 if item j is in the set-intersection.

for P_0 do
  $\forall j$, set $c_{0j} \leftarrow E_{pk}(x_{0j})$; $C_0 = \prod_{j=1}^{n} c_{0j} = \prod_{j=1}^{n} E_{pk}(x_{0j})$;
  Create $\mathsf{PPK}(C_0)$ to prove that each x_{0j} is either 0 or 1.
  Send all encryptions c_{0j} and $\mathsf{PPK}(C_0)$ to P_1
end for
for P_1 do
  Compute $C_0 = \prod_{j=1}^{n} c_{0j}$; Check $\mathsf{PPK}(C_0)$ is correct else ABORT
  $\forall j$ Calculate $c_{1j} \leftarrow E_{pk}(x_{1j})$; $C_1 = \prod_{j=1}^{n} c_{1j} = \prod_{j=1}^{n} E_{pk}(x_{1j})$;
  Create $\mathsf{PPK}(C_1)$
  $\forall j$ Calculate $c_{01j} \leftarrow e_{x_{0j}} \times_h x_{1j}$; Compute $C_{01} = \prod_{j=1}^{n} c_{01j}$;
  Create $\mathsf{PCM}(C_{01})$
  Send all ciphertexts c_{1j}, c_{01j}, $\mathsf{PPK}(C_1)$, $\mathsf{PCM}(C_{01})$ to P_0;
end for
for P_0 do
  Compute $C_1 = \prod_{j=1}^{n} c_{1j}$, $C_{01} = \prod_{j=1}^{n} c_{01j}$;
  Check $\mathsf{PPK}(C_1)$, $\mathsf{PCM}(C_{01})$ are correct else ABORT;
end for
Jointly call private decrypt function to learn $D_{pr}(c_{01j})$
Set I_j to $D_{pr}(c_{01j})$

Fig. 2. Our proposed protocol that uses threshold multiparty computation with homomorphic encryption to compute secure and efficient set intersection

4 Security Analysis

Lemma 1. An adversary who does not know the input plaintexts of the dot product or set-intersection protocols can successfully cheat using its own input strings with negligible probability.

Proof: Note that we do not have to consider the permutation of messages (e.g., $M = m_1 + m_2 = m_2 + m_1$) in the following proof, since such factorization values are reduced by division process of the probability computations. The number of solution vectors $(t_1, \ldots, t_n)$ satisfying $M = \sum_{j=1}^{n} t_j$ is N^{n-1}, since we can randomly choose $t_j \in \mathbb{Z}_p (j = 1, 2, \ldots, n-1)$, and set $t_n = M - \sum_{j=1}^{n-1} t_j$. Cheating means that an adversary $\mathcal{A}$ can compute a PPK which is accepted by the verifier although there exists a message $m \notin \{m_1, \ldots, m_n\}$. Therefore, the probability of the randomly chosen vector $(t_1, \ldots, t_n)$ satisfying $M = \sum_{j=1}^{n} t_j$ is $N^{n-1}/N^n = 1/N$. This implies that, the probability of a proof of false message satisfying $M = \sum_{j=1}^{n} m_j$ is $N^{n-1}/N^n = 1/N$. This implies that, the probability of a proof of false messages satisfying $M := \sum_{j=1}^{n} m_j$ is $\frac{N^{n-1}-1}{N^n} = \frac{1}{N}(1 - \frac{1}{N^{n-1}})$. Next, we consider $t_j = m_j (j = 1, 2, \ldots, \ell)$, where $\ell < n$ is the number of real messages $\mathcal{A}$ can capture. Let $\ell = n - 1$, this means $\mathcal{A}$ has all the messages of

M except m_n . Then $t_n = m_n := M - \sum_{j=1}^{n-1} m_j$ holds. This means that $\mathcal{A}$ has valid messages of M, and the proof of M is not a forged proof. Therefore, we set $\ell < n-1$. Then there exist $N^{n-\ell-1} - 1$ number of pairs of $(t_{\ell+1}, \ldots, t_n)$ such that $(m_1, \ldots, m_\ell, t_{\ell+1}, \ldots, t_n)$ satisfies $M = (\sum_{j=1}^{\ell} m_j) + (\sum_{j=\ell+1}^{n} t_j)$ and $(t_{\ell+1}, \ldots, t_n) \neq (m_{\ell+1}, \ldots, m_n)$. The number of vectors $(t_{\ell+1}, \ldots, t_n)$ is $N^{n-\ell}$. Therefore, the probability of a proof made by valid messages of $(m_1, \ldots, m_\ell)$ and forged messages of $(t_{\ell+1}, \ldots, t_n)$ satisfying $M = (\sum_{j=1}^{\ell} m_j) + (\sum_{j=\ell+1}^{n} t_j)$ is $\frac{N^{n-\ell-1}-1}{N^{n-\ell}} = \frac{1}{N}(1 - \frac{1}{N^{n-\ell-1}}) \leq \frac{1}{N}$. □

Note that, in the malicious model, authors may lie about their input. For example, if the inputs are from $\{0, 1\}$ domain, malicious attacker can replace it with 2. In many cases, zero knowledge proofs are needed to protect against such attacks. Completeness and soundness are the two properties that should be achieved by the zero-knowledge proof to prevent the parties from behaving in such malicious way. Completeness requires that an honest verifier is satisfied with the prover, while soundness says that a dishonest party cannot convince an honest party with incorrect input. By using the Lemma 1, we can achieve both the properties considering that achieving such properties will fail with negligible probability. In other words, if the statement (proof of knowledge) is true, the honest verifier (that is, one following the protocol properly) will be convinced of this fact by an honest prover, and if the statement (proof of knowledge) is false, no cheating prover can convince the honest verifier that it is true, except with some small probability. This gives us a sketch on why our approach of ZKP construction works (of course with a negligible probability of failure).

The following two theorems state that our constructions are secure in UC framework. We omit the proofs due to lack of space.

Theorem 1. *The Dot Product protocol is secure in (secure equality, private decrypt)-hybrid model assuming that the non-interactive zero knowledge proofs (PPK) are secure in the presence of malicious adversaries.*

Theorem 2. *The Set-Intersection protocol is secure in (threshold decryption)-hybrid model assuming that the non-interactive zero knowledge proofs are secure in the presence of malicious adversaries.*

5 Efficiency

Secure Dot Product: The complexity of secure dot product protocol proposed in [10] can be expressed as $O(n)$ where n is the size of the input dataset, in other words the size of data items to be processed. Therefore, the efficiency of the protocol is highly dependent on the size of the input dataset. The complexity displayed in Table 1 involves both communication and computation times, and the difference can be explained with the impact of communication and computation overhead that the proof of knowledge brings. While the overhead of ciphertext computation and communication in our protocol are similar to that in [10], the overhead of computation and communication of proof of knowledge

are constant in our protocol. This is a drastic reduction in computation and communication.

Table 1. Secure Dot Product: Performance Comparison between [10] and our proposed protocol

Schemes	Computation		Communication	
	ciphertext	proof-of-knowledge	ciphertext	proof-of-knowledge
[10]	$O(n)$	$O(n)$	$O(n)$	$O(n)$
Ours	$O(n)$	$O(1)$	$O(n)$	$O(1)$

Secure Set-Intersection: Given that P_0 has n items and P_1 has m items, where both m and n are bigger than $O(\sqrt{D})$ or where n or m equal to total item domain size D, we (as well as [10]) suggest using the simple secure set-intersection and set union protocols that are secure in the malicious model. According to Table 2, the complexity (for proof of knowledge) in our protocol does not depend on the size of data items, rather it is constant. Note that, we need PCM besides PPK in set-intersection computation. PCM requires 6 exponentiations in $\mathbb{Z}_N$. Considering the library example, [10] will require to compute 750 million exponentiations, whereas our protocol requires only 6 exponentiations .

Table 2. Secure Set-Intersection: Performance Comparison between [10] and our proposed protocol

Schemes	Computation		Communication	
	ciphertext	proof-of-knowledge	ciphertext	proof-of-knowledge
[10]	$O(D)$	$O(D)$	$O(D)$	$O(D)$
Ours	$O(D)$	$O(1)$	$O(D)$	$O(1)$

6 Conclusion

In this paper, we have proposed efficient and secure dot product and set-intersection protocols in malicious model which are useful for many practical applications. These protocols can be used in various data mining algorithms as building blocks. We provide sophisticated modifications that lead to bigger increases in efficiency of the privacy-preserving data mining algorithms. Although the constructions of our protocols do not deviate a lot from that of [10], the effect of our construction for efficient implementation is huge. We do not avoid the ciphertext computation and communication cost ($O(n)$) to output all of the n ciphertexts that are necessary to compute the dot product or intersection. Our approach drastically reduces the computational and communication complexity of the proof of knowledge. In our work, computation and communication for proof of knowledge is always constant (independent of number of data items), while computation and communication for encrypted messages remains linear with the number of data items. We also provide the security model in UC framework.

References

1. Boneh, D., Goh,E.G., and Nissim, K.: Evaluating 2-DNF Formulas on Ciphertexts. TCC'05, LNCS, pp. 325–341. (2005)
2. Bunn, P., and Ostrovsky, R.: Secure Two-Party k-Means Clustering. ACM CCS'07, pp. 486–497. (2007)
3. Cramer, R., Damgard, I., and Nielsen, J.B.: Multi-party computation from threshold homomorphic encryption. EUROCRYPT'01, LNCS, pp. 280–299. (2001)
4. Damgard, I., Hofheinz, D., Kiltz, E., and Thorbek, R.: Public-Key Encryption with Non-interactive Opening. CT-RSA'08, LNCS, pp. 239–255. (2008)
5. Damgard, I., Thorbek, R.: Non-interactive proofs for integer multiplication. EUROCRYPT'07, LNCS, pp. 412–429. (2007)
6. Galindo, D., Libert, B., Fischlin, M., Fuchsbauer, G., Lehmann, A., Manulis, M., and Schroder, D.: Public-Key Encryption with Non-interactive Opening: New Constructions and Stronger Definitions. AFRICACRYPT'10, LNCS, pp. 333–350. (2010)
7. Hazay, C., and Nissim, K.: Efficient Set Operations in the Presence of Malicious Adversaries. Public Key Cryptography - PKC'10, LNCS, pp. 312–331. (2010)
8. Jagannathan, G. and Wright, R.N.: Privacy-preserving distributed k-means clustering over arbitrarily partitioned data. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 593–599. (2005)
9. Kantarcioglu, M. and Clifton, C.: Privately computing a distributed k-nn classifier. PKDD'04, LNCS, pp. 279–290. (2004)
10. Kantarcioglu, M., and Kardes, O.: Privacy-preserving data mining in the malicious model. International Journal of Information and Computer Security, Vol. 2, No. 4, pp. 353–375. (2008)
11. Lai, J., Deng, R.H., Liu, S., and Kou, W.: Efficient CCA-Secure PKE from Identity-Based Techniques. CT-RSA'10, LNCS, pp. 132–147. (2010)
12. Lin, X., Clifton, C. and Zhu, M.: Privacy-preserving clustering with distributed EM mixture modeling. Knowledge and Information Systems, July, Vol. 8, No. 1, pp. 68–81. (2005)
13. Lindell, Y. and Pinkas, B.: Privacy preserving data mining, CRYPTO'00, LNCS, pp. 36–54. (2000)
14. Okamoto,T., and Takashima,K.: Homomorphic Encryption and Signatures from Vector Decomposition. Pairing-Based Cryptography - Pairing'08, LNCS, pp. 57–74. (2008)
15. Paillier, P.: Public-key cryptosystems based on composite degree residue classes. EuroCrypt'99, LNCS, pp. 223–238. (1999)
16. Su, C., Bao, F., Zhou, J., Takagi, T., Sakurai, K.: Security and Correctness Analysis on Privacy-Preserving k-Means Clustering Schemes. IEICE Trans. Fundamentals, Vol.E92-A, No.4, pp. 1246–1250. (2009)
17. Top 10 Largest Databases in the World. http://www.worldsbiggests.com/2010/02/top-10-largest-databases-in-world.html
18. Vaidya, J. and Clifton, C.: Privacy preserving association rule mining in vertically partitioned data. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 639–644. (2002)
19. Yang, Z. and Wright, R.N. Privacy-preserving computation of Bayesian networks on vertically partitioned data. IEEE Transactions on Knowledge and Data Engineering, Vol. 18, No. 9, pp. 1253–1264.(2006)