Privacy-Preserving Data Mining in Presence of Covert Adversaries

Atsuko Miyaji and Mohammad Shahriar Rahman

School of Information Science, Japan Advanced Institute of Science and Technology
1-1 Asahidai, Nomi, Ishikawa, Japan 923-1292,
{miyaji,mohammad}@jaist.ac.jp

Abstract. Disclosure of the original data sets is not acceptable due to privacy concerns in many distributed data mining settings. To address such concerns, privacy-preserving data mining has been an active research area in recent years. All the recent works on privacy-preserving data mining have considered either semi-honest or malicious adversarial models, whereby an adversary is assumed to follow or arbitrarily deviate from the protocol, respectively. While semi-honest model provides weak security requiring small amount of computation and malicious model provides strong security requiring expensive computations like Non-Interactive Zero Knowledge proofs, we envisage the need for 'covert' adversarial model that performs in between the semi-honest and malicious models, both in terms of security guarantee and computational cost. In this paper, for the first time in data-mining area, we build efficient and secure dot product and set-intersection protocols in covert adversarial model. We use homomorphic property of Paillier encryption scheme and two-party computation of Aumann et al. to construct our protocols. Furthermore, our protocols are secure in Universal Composability framework.

KeyWords: Privacy-preserving Data Mining, Covert Adversary, Efficiency, Multi Party Computation

1 Introduction

1.1 Background

Recent advances in information technology has empowered many organizations to collect large volumes of data through data mining. A key utility of large databases today is scientific or economic research. However, this data is not useful if worthwhile information cannot be extracted from it. Privacy is a key issue that arises in any huge collection of data. The need for privacy is sometimes due to personal interests, law (e.g., for medical databases), or business interests. However, despite the potential gain, this is often not possible to utilize large databases for scientific or economic research due to the 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 can be categorized into two main classes of adversaries:

Malicious adversaries: These adversaries may behave arbitrarily and are not assumed 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.

The assumption of semi-honest behavior may be unrealistic in some settings. In such cases, participating parties may prefer to use a protocol that is secure against malicious behavior. It is clear that the protocols secure in the malicious model offer more security. Regarding malicious adversaries, it has been shown that, under suitable cryptographic assumptions, any multiparty probabilistic polynomial time functionality can be securely computed for any number of malicious corrupted parties. However, these are not efficient enough to be used in practice. Most of these constructions use general zero-knowledge proofs for fully malicious multi party computation protocols. The zero-knowledge proofs have inefficient constructions. These constructions make a non-black-box use of the underlying cryptographic primitives through the use of trapdoor permutations[8]. Assuming a trapdoor permutation takes one second to compute, its circuit implementation contains trillions of gates, thereby requiring the protocol trillions of second to run. Some middle type of adversary model that accurately models adversarial behavior in the real world efficiently are thus to be introduced in data mining.

In this work, we introduce a new adversary model that lies between the semi-honest and malicious models. To the best of our knowledge, this is the first attempt to introduce such a model for data mining applications. In many real-world settings, parties are willing to actively cheat (not semi-honest), but only if they are not caught (not arbitrarily malicious). This is natural in many business, financial, and diplomatic settings, where honest behavior cannot be assumed, but where the companies, institutions and individuals involved cannot afford the embarrassment, loss of reputation associated with being caught cheating. This type of covert adversarial behavior explicitly models the probability of catching adversarial behavior. In particular, it is not assumed that adversaries are only willing to risk being caught with negligible probability, but rather allow for much higher probabilities.

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 [10, 20, 3]. K-means clustering uses dot product and equality protocols as building blocks. Some recent studies [22, 23] 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 scure 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 [17], k-means clustering [15], k-nn classifiers [12]. 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 [22], and k-means clusters [10, 20]. 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 [13], 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. They use threshold homomorphic encryption for malicious adversaries, presented by Cramer et.al. [4]. [13] shows that there is a positive linear relationship between the overall complexity of the subprotocols and the input size. There has been some other works related to secure two-party computation [2, 18]. In [2], 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, [18] proposed a two-party protocol to securely evaluate a 2DNF formula using homomorphic encryption from vector decomposition. But this protocol has been shown secure only in the semi-honest adversarial model. Recently, [9] 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 multiparty computation, thus increasing the communication overhead.

1.3 Our Contribution

Considering the problems mentioned above, for the first time in data mining area, we provide efficient dot product and set-intersection protocols that are secure in presence of the covert adversaries. While semi-honest model provides weak security requiring small amount of computation and malicious model provides strong security requiring expensive computations like Non-Interactive Zero Knowledge proofs, we envisage the need for 'covert' adversarial model that performs in between the semi-honest and malicious models, both in terms of security guarantee and computational cost. We use homoprphic property of Paillier encryption scheme and two-party computation of Aumann et al. to construct our protocols. Furthermore, our protocols are secure in Universal Composability framework.

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. We briefly visit the universal composability model of [4], a detailed description can be found there.

Ideal Model: Let the set of parties be P_1, P_2 and $I \in \{0, 1\}$ denote the indices of the corrupted parties, controlled by an adversary A. An ideal execution proceeds as follows:

Each party obtains an input; the ith partys input is denoted x_i. The adversary A receives an auxiliary input denoted z. Any honest party P_j sends its received input x_j to the trusted party. The corrupted parties controlled by A may either abort, send their received input, or send some other input of the same length to the trusted party. This decision is made by A and may depend on the values x_i for $i \in I$ and its auxiliary input z. Denote the vector of inputs sent to the trusted party by w. If the trusted party does not receive 2 valid inputs, it replies to all parties with a special symbol and the ideal execution terminates. Otherwise, The trusted party computes $(f_1(w), f_2(w))$ and sends $f_i(w)$ to party P_i, for all $i \in I$ (i.e., to all corrupted parties). A sends either continue or halt to the trusted party. If it sends continue, the trusted party sends $f_j(w)$ to party P_j , for all $j \notin I$ (i.e., to all honest parties). Otherwise, the trusted party sends to all parties. An honest party always outputs the message it obtained from the trusted party. The ideal execution of f on inputs x, auxiliary input z to A and security parameter

n, denoted $\text{IDEAL}_{f,A(z),I}(x,n)$, is defined as the output vector of the honest parties and the adversary A from the above ideal execution.
Real-life Model: The adversary A sends all messages in place of the corrupted parties, and may follow an arbitrary polynomial-time strategy. In contrast, the honest parties follow the instructions of π.

Let f be as above and let π be an two-party protocol for computing f. Furthermore, let A be a non-uniform PPT machine and let I be the set of corrupted parties. Then, the real execution of π on inputs x, auxiliary input z to A and security parameter n, denoted $\text{REAL}= \pi, A(z), I(x,n)$, is defined as the output vector of the honest parties and the adversary A from the real execution of pi.

Definition 1. *Let f and π be as above. Protocol π is said to securely compute f with abort in the presence of malicious adversaries if for every non-uniform PPT adversary A for the real model, there exists a non-uniform PPT adversary S for the ideal model, such that for every $I \subseteq [m]$, every balanced vector $x \in (\{0,1\}^*)^2$, and every auxiliary input $z \in \{0,1\}^*$: $\{\mathsf{IDEAL}_{f,S(z),I}(x,n)\}_{n\in\mathbb{N}} \overset{c}{\approx} \{\mathsf{REAL}_{\pi,A(z),I}(x,n)\}_{n\in\mathbb{N}}$, where $\overset{c}{\approx}$ indicates computational indistinguishability.*

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 $E_{pk}(k \cdot m_1) := k \times_h E_{pk}(m_1)$.

We briefly state the Paillier cryptosystem [19] based on composite residuosity assumption here. A more detailed description can be found in [19].

- **Key generation:** Let p and q be prime numbers where $p < q$ and $p \nmid q - 1$. Set the public key pk to N where $N = p \cdot q$, and private key sk to (λ, N), where $\lambda = LCM(p-1, q-1)$.
- **Encryption with the public key:** Given n, plaintext m, and a random number $r \in [1, \ldots, N-1]$, encryption of the message m: $E_{pk}(m) = (1 + N)^m \cdot r^N \pmod{N^2}$. Given any encrypted message, a different encryption can be computed by multiplying it with some random r^N.
- **Decryption with the private key:** Given N, the ciphertext $c = E_{pk}(m)$, decrypt as follows: $m = \frac{(c^\lambda \pmod{N^2})-1}{N}\lambda^{-1}$, where λ^{-1} is the inverse of λ in modulo N.
- **Adding two ciphertexts:** Given the encryptions $E_{pk}(m_1)$ and $E_{pk}(m_2)$ where $\forall m_1, m_2 \in \mathbb{Z}_N$, $E_{pk}(m_1 + m_2)$ can be computed as follows: $E_{pk}(m_1) \cdot$

$E_{pk}(m_2) \mod N^2 = ((1+N)^{m_1} r_1^N) \cdot ((1+N)^{m_2} r_2^N) \mod N^2 = ((1+N)^{m_1+m_2} \cdot (r_1 r_2)^N) \mod N^2 = E_{pk}(m_1 + m_2) \mod N$

- **Multiplying a ciphertext with a constant:** Given a constant $k \in \mathbb{N}$ and the encrypted value $E_{pk}(m_1)$, $k \times E_pk(m_1)$ can be computed as: $k \times E_{pk}(m_1) = E_{pk}(m_1)^k \mod N^2 = ((1+N)^{m_1} \cdot r^N)^k \mod N^2 = (1+N)^{k \cdot m_1} \cdot r_1^{kN} \mod N^2 = E_{pk}(k \cdot m_1)$

Definition 2. *Assume the existence of semantically secure Paillier encryption scheme with errorless decryption. Then, for any $k = poly(n)$ there exists a secure protocol for computing the parallel string oblivious transfer functionality $((x_1^0, x_1^1), \ldots, (x_n^0, x_1^n), (\sigma_1, \ldots, \sigma_n)) \mapsto (\lambda, (x_1^{\sigma_1}, \ldots, x_n^{\sigma_n}))$ in the presence of covert adversaries with ϵ-deterrent for $\epsilon = 1 - 1/k$.*

2.3 Two-party Computation

We use secure multi-party protocol in covert adversarial model proposed in [1]. A two-party protocol problem is cast by specifying a random process that maps sets of inputs to sets of outputs (one for each party). We refer to such a process as a functionality and denote it $f : (\{0,1\}^*)^2 \to (\{0,1\}^*)^2$, where $f = (f_1, f_2)$. The oblivious transfer functionality is denoted by $((x_0, x_1), \sigma \to (\lambda, x_\sigma)$, where (x_0, x_1) is the first party's input, σ is the second party's input, and λ denotes the empty string (meaning that the first party has no output).

We use the simulators in the security 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. In the case of an attempted cheat, if the trusted party sends $corrupted_i$ to the honest party and the adversary (an event which happens with probability ϵ), then the adversary does not obtain the honest party's inputs. Thus, if cheating is detected, the adversary does not learn anything and the result is essentially the same as a regular abort. In other words, the adversary learns nothing when it is detected. Since it is always detected, this means that full security is achieved. We denote the resultant ideal model by $\mathsf{IDEALSC}^{\epsilon}_{f,S(z),I}(x,n)$ and have the following definition:

Definition 3. *Let f, π be as in Definition 1, and $\epsilon : \mathbb{N} \to [0,1]$. Protocol π is said to securely compute f in the presence of covert adversaries with ϵ-deterrent if for every non-uniform PPT adversary A for the real model, there exists a non-uniform PPT adversary S for the ideal model such that for every $I \subseteq [2]$, every balanced vector $x \in (\{0,1\}^*)^2$, and every auxiliary input $z \in \{0,1\}^*$:* $\{\mathsf{IDEALSC}^{\epsilon}_{f,S(z),I}(x,n)\}_{n\in\mathbb{N}} \overset{c}{\approx} \{\mathsf{REAL}_{\pi,A(z),I}(x,n)\}_{n\in\mathbb{N}}$*, where $\overset{c}{\approx}$ indicates computational indistinguishability.*

A detailed discussion on the relationship among semi-honest model, malicious model, and and ϵ-deterrent covert adversarial model can be found in [1].

3 Our Protocols

3.1 Underlying Idea

In the protocol, P_1 sends P_2 a number of garbled circuits; denote this number by l. Then, P_2 asks P_1 to open all but one of the circuits (chosen at random) in order to check that they are correctly constructed. This opening takes place before P_1 sends the keys corresponding to its input, so nothing is revealed by opening the circuits. If the unopened circuit is correct, then this will constitute a secure execution that can be simulated. With probability $1 - 1/l$ party P_1 will have been caught cheating and so P_2 will output $corrupted_1$ if the unopened circuit is not correct. To do so, it is crucial that the oblivious transfers are run before the garbled circuits are sent by P_1 to P_2. This is due to the fact that the simulator may send a corrupted P_2 a fake garbled circuit that evaluates to the exact output received from the trusted party (and only this output). However, in order for the simulator to receive the output from the trusted party, it must first send it the input used by the corrupted P_2. This is achieved by first running the oblivious transfers, from which the simulator is able to extract the corrupted P_2's input. Moreover, let us consider a corrupted P_1 that behaves exactly like an honest P_1 except that in the oblivious transfers, it inputs an invalid key in the place of the key associated with 0 as the first bit of P_2. The result is that if the first bit of P_2's input is 1, then the protocol succeeds and no problem arises. However, if the first bit of P_2's input is 0, then the protocol will always fail and P_2 will always detect cheating. Thus, P_1's decision to cheat may depend on P_2's private input. In order to solve this problem, a circuit that computes the function $g(x_1, x_2^1, \ldots, x_2^m) = f(x_1; \oplus_{i=1}^m x_2^i)$, instead of a circuit that directly computes f. This makes every bit of P_2's input uniform when considering any proper subset of $x_2^1, \ldots x_2^m$. This helps because as long as P_1 does not provide invalid keys for all m shares of x_2, the probability of failure is independent of P_2's actual input. This method has been used in [1].

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. If both parties are trying to cheat, we do not care whether the privacy of any party is protected or parties get correct results. Assuming that any party may want to actively cheat without being caught, an efficient protocol can be constructed.

Require: Two parties P_1 and P_2 with n bit vector inputs x_{lj} where x_{lj} belongs to P_l, and $1 \leq j \leq n$, $l \in \{1, 2\}$.

Auxiliary input: Both parties have the description of a circuit C for inputs of length n that computes the function f. The input wires associated with x_1 and x_2 are $w_1, \ldots, w_n$ and $w_{n+1}, \ldots, w_{2n}$, respectively.

Ensure: Return $res = \sum_{j=1}^{n}(x_{1j} \cdot x_{2j}))$ to P_1

– Parties P_1 and P_2 define a new circuit C' that receives $n+1$ inputs $x_1, x_2^1, \ldots, x_2^n$ each of length n, and computes the function $f(x_1, \oplus_{i=1}^n x_2^i)$. Denote the input wires associated with x_1 by $w_1, \ldots, w_n$, and the input wires associated with x_2^i by $w_{in+1}, \ldots, w_{(i+1)n}$

– Party P_2 chooses $n-1$ random strings $x_2^1, \ldots, x_2^{n-1} \in_R \{0,1\}^n$ and defines $x_2^n = (\oplus_{i=1}^{n-1} x_2^i) \oplus x_2$. The value $z_2 = x_2^1, \ldots, x_2^n$ serves as P_2's new input of length n^2 to C'

– Party P_1 chooses two sets of $2n^2$ random keys by running $G(1^n)$, the key generator for the encryption scheme: $(\hat{k}_{n+1}^0, \ldots, \hat{k}_{n^2+n}^0)$,$(\tilde{k}_{n+1}^0, \ldots, \tilde{k}_{n^2+n}^0)$, $(\hat{k}_{n+1}^1, \ldots, \hat{k}_{n^2+n}^1)$,$(\tilde{k}_{n+1}^1, \ldots, \tilde{k}_{n^2+n}^1)$

– P_1 and P_2 run n^2 executions of an oblivious transfer protocol, as follows. In the ith execution, party P_1 inputs the pair $((\hat{k}_{n+1}^0, \tilde{k}_{n+1}^0), (\hat{k}_{n+1}^1, \tilde{k}_{n+1}^1))$ and party P_2 inputs the bit z_2^i. The executions are run using a parallel oblivious transfer functionality, as in Definition 2. If a party receives a $corrupted_i$ or $abort_i$ message as output from the oblivious transfer, it outputs it and halts.

– Party P_1 constructs two garbled circuits $G(C')_0$ and $G(C')_1$ using independent randomness. The keys to the input wires $w_{n+1}, \ldots, w_{n^2+n}$ in the garbled circuits are taken from above. Let $\hat{k}_1^0, \hat{k}_1^1, \ldots, \hat{k}_n^0, \hat{k}_n^1$ be the keys associated with P_1's input in $G(C')_0$ and $G(C')_1$ has analogous keys. Then, for every $i \in \{1, \ldots, n\}$ and $b \in \{0,1\}$, party P_1 computes $\hat{c}_i^b = Com(\hat{k}_i^b; \hat{r}_i^b)$ and $\tilde{c}_i^b = Com(\tilde{k}_i^b; \tilde{r}_i^b)$, where Com is a perfectly-binding commitment scheme. P_1 sends the garbled circuits to P_2 together with all of the above commitments. The commitments are sent as two vectors of pairs; in the first vector the ith pair is $\{\hat{c}_i^0, \hat{c}_i^1\}$ in a random order, and in the second vector the ith pair is $\{\tilde{c}_i^0, \tilde{c}_i^1\}$ in a random order.

– Party P_2 chooses a random bit $b \in_R \{0,1\}$ and sends b to P_1. The following steps are a single step: P_1 sends a message to P_2, and P_2 carries out a computation.

– P_1 sends P_2 all of the keys for the inputs wires $w_1, \ldots, w_{n^2+n}$ of the garbled circuit $G(C')_b$, together with the associated mappings and the decommitment values.

– P_2 checks the decommitments to the keys associated with $w_1, \ldots, w_n$, decrypts the entire circuit using the keys and mappings that it received, and checks that it is exactly the circuit C' derived from the auxiliary input circuit C. In addition, it checks that the keys that it received in the oblivious transfers match the correct keys that it received in the opening. If all the checks pass, it proceeds to the next step. If not or it does not get any message, it outputs $corrupted_1$ and halts.

- If $b = 0$, then P_1 sends P_2 the keys and decommitment values $(\tilde{k}_1^{x_1^1}, \tilde{r}_1^{x_1^1}), \ldots, (\tilde{k}_1^{x_1^n}, \tilde{r}_1^{x_1^n})$ to P_2. Otherwise, P_2 sends the $(\hat{k}_1^{x_1^1}, \hat{r}_1^{x_1^1}), \ldots, (\hat{k}_1^{x_1^n}, \hat{r}_1^{x_1^n})$.

- P_2 checks that the values received are valid decommitments to the commitments received above. If not, it outputs $abort_1$. If yes, it uses the keys to compute $C'(x_1, z_2) = C'(x_1, x_2^1, \ldots, x_2^n) = C(x_1, x_2)$, such that $C(x_1, x_2) = f = e_{res^2} = (E_{pk}(x_{21}) \times_h x_{21}) +_h \ldots +_h (E_{pk}(x_{1n}) \times_h x_{2n})$

- P_1 calls decrypt protocol to learn $D_{sk}(e_{res^2}) = res = \sum_{j=1}^{n}(x_{1j} \cdot x_{2j}))$

The result of data mining algorithm $res = \sum_{j=1}^{n}(x_{1j} \cdot x_{2j})$ is evaluated correctly by at least one party, assuming that at least one party tries to cheat. Both P_1 and P_2 have enough information to calculate res. If P_2 computes the same res value as P_1 expects, then computations must be correct, because none of them calculates incorrect res. Therefore, if we securely make sure that any party fails to cheat successfully with incorrect output value, then the local calculation can be decrypted to reveal res to P_1 correctly. In our protocol, party P_1 sends the encrypted inputs along with necessary keys to P_2, then party P_2 locally computes its respective $e_{res^2} = E_{pk}(res^2)$. For example, P_1 generates a homomorphic key pair and sends the pk to P_2 along with the encrypted vector $E_{pk}(x_{1j}) = E_{pk}(x_{11}), (E_{pk}(x_{12}), \ldots, E_{pk}(x_{1n}))$. Given pk, P_2 calculates $e_{x_{1j} \cdot x_{2j}} = (E_{pk}(x_{11}) \times_h x_{21}) +_h (E_{pk}(x_{12}) \times_h x_{22}) +_h \ldots +_h (E_{pk}(x_{1n}) \times_h x_{2n})$ where x_{1j} is the P_1's string's j-th component, and sends e_{res^2} to P_1. For the decryption, the P_1 and P_2 jointly decrypts to reveal res to P_1, given that the computated values are correct.

A note on secure equality protocol: A secure equality protocol requires to compute whether two data items are equal or not without revealing these items. It is thus straight forward to construct a secure equality protocol in covert model. 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 property of the homomorphic encryption to give secure set protocols in the covert 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. One important thing here to note that, unlike the dot product protocol, each parties need to perform the whole protocol. In other words, each party needs to send its garbled circuits and the necessary keys to the other party before they compute the joint function.

We put the detailes of the protocol in the full version due to lack of space. The same protocol can be used for two-party set union negating the input and output bits.

4 Security Analysis

Theorem 1. *Let l and m be parameters in the protocol that are both upper-bound by poly(n), and set $\epsilon = (1 - 1/l)(1 - 2^{m+1})$. Let f be any PPT function. Assume that the Paillier encryption scheme used to generate the garbled circuits has indistinguishable encryptions, and that the oblivious transfer protocol used is secure in the presence of covert adversaries with ϵ-deterrent according to Definition 2. Then, our protocols securely compute dot product and set-intersection in the presence of covert adversaries with ϵ-deterrent.*

Proof (Sketch): We breifly sketch the idea on the proof due to page limitation. Our analysis of the security of the protocol is in the (OT, ϵ)-hybrid model, where the parties are assumed to have access to a trusted party computing the oblivious transfer functionality following the ideal model of our definition. Thus the simulator will play the trusted party in the oblivious transfer, when simulating for the adversary. We separately consider the different corruption cases (when no parties are corrupted, and when either one of the parties is corrupted). In the case that no parties are corrupted, the security reduces to the semi-honest case.

Party P2 is corrupted: Intuitively, the security in this case relies on the fact that P_2 can only learn a single set of keys in the oblivious transfers and thus can decrypt the garbled circuit to only a single value as required. In other words, A is the PPT adversary who controls P_2. The simulator S fixes A's random-tape to a uniformly distributed tape. S meets the requirements of our Definition 3. S only needs to send $cheat_2$ due to the oblivious transfer. Thus, if a"fully secure" oblivious transfer protocol were to be used, the protocol would meet the standard definition of security for malicious adversaries for the case that P_2 is corrupted.

Party P1 is corrupted: The proof of security in this corruption case is considerably more complex. Intuitively, security relies on the fact that if P_1 does not construct the circuits correctly or does not provide the same keys in the oblivious transfers and circuit openings, then it will be caught with probability at least ϵ. In contrast, if it does construct the circuits correctly and provide the same keys, then its behavior is effectively the same as an honest party and so security is preserved. □

5 Efficiency

The efficiency of our protocol is better compared to the best known results for the malicious adversary model. Our protocol requires only a constant number of rounds, a single oblivious transfer for each input bit, and has communication complexity $O(n|C|)$ where n is the security parameter and $|C|$ is the size of the

circuit being computed. Two efficient protocols for general two-party computation in the presence of malicious adversaries has been presented in [11, 16] recently. The protocol of [11] achieves universal composability under the decisional composite residuosity and strong RSA assumptions under a common reference string. The protocol of [16] has been constructed under more general assumptions and is secure in the plain model. [11] requires $O(|C|)$ public-key operations and bandwidth of $O(n \cdot |C|)$. [16] requires symmetric operations and bandwidth of the order of $O(sn|C|+s^2k)$ where k is the input length, n is the computational security parameter, and s is a statistical security parameter. Thus, our protocol in covert adversarial model is much more efficient for circuits that are not very small. On the other hand, it is sufficient for the oblivious transfer protocol to be secure in the presence of covert adversaries. Hence, a protocol for general two-party computation with $\epsilon = 1/2$ is only a constant factor slower than the original protocol of Yao that is only secure for semi-honest adversaries.

6 Conclusion

In this paper, we have proposed efficient and secure dot product and set-intersection protocols in covert adversarial model for the first time 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 greater efficiency of the privacy-preserving data mining algorithms in more realistic settings. The effect of our construction for efficient implementation is huge. Our protocols are much efiicient than the protocols in malicious models without requiring expensive zero knowledge proofs, and are slightly expensinve than the semi-honest models. However, the security of our protocol is much stronger than that in semi-honest models. We also provide the security model in UC framework. Applying the covert adversarial model in a more efficient way for specific data mining applications under reasonable assumptions are the open problems.

References

1. Aumann, Y. and Lindell, Y.: Security Against Covert Adversaries: Efficient Protocols for Realistic Adversaries, TCC'07, LNCS, pp. 137-156. (2007)
2. Boneh, D., Goh,E.G., and Nissim, K.: Evaluating 2-DNF Formulas on Ciphertexts. TCC'05, LNCS, pp. 325-341. (2005)
3. Bunn, P., and Ostrovsky, R.: Secure Two-Party k-Means Clustering. ACM CCS'07, pp. 486-497. (2007)
4. Cramer, R., Damgard, I., and Nielsen, J.B.: Multi-party computation from threshold homomorphic encryption. EUROCRYPT'01, LNCS, pp. 280-299. (2001)
5. Damgard, I., Hofheinz, D., Kiltz, E., and Thorbek, R.: Public-Key Encryption with Non-interactive Opening. CT-RSA'08, LNCS, pp. 239-255. (2008)
6. Damgard, I., Thorbek, R.: Non-interactive proofs for integer multiplication. EUROCRYPT'07, LNCS, pp. 412-429. (2007)

7. 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)
8. Goyal, V., Mohassel, P., and Smith, A.: Efficient Two Party and Multi Party Computation Against Covert Adversaries. EUROCRYPT'08, LNCS, pp. 289-306. (2008)
9. Hazay, C., and Nissim, K.: Efficient Set Operations in the Presence of Malicious Adversaries. Public Key Cryptography - PKC'10, LNCS, pp. 312-331. (2010)
10. 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)
11. Jarecki, S. and Shmatikov, V.: Efficient Two-Party Secure Computation on Committed Inputs. EUROCRYPT'07, LNCS, pp. 97-114. (2007)
12. Kantarcioglu, M. and Clifton, C.: Privately computing a distributed k-nn classifier. PKDD'04, LNCS, pp. 279-290. (2004)
13. 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)
14. 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)
15. 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)
16. Lindell, Y. and Pinkas, B.: An Efficient Protocol for Secure Two-Party Computation in the Presence of Malicious Adversaries. EUROCRYPT'07, LNCS, pp. 52-78. (2007)
17. Lindell, Y. and Pinkas, B.: Privacy preserving data mining, CRYPTO'00, LNCS, pp. 36-54. (2000)
18. Okamoto,T., and Takashima,K.: Homomorphic Encryption and Signatures from Vector Decomposition. Pairing-Based Cryptography - Pairing'08, LNCS, pp. 57-74. (2008)
19. Paillier, P.: Public-key cryptosystems based on composite degree residue classes. EuroCrypt'99, LNCS, pp. 223-238. (1999)
20. 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)
21. Top 10 Largest Databases in the World. http://www.worldsbiggests.com/2010/02/top-10-largest-databases-in-world.html
22. 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)
23. 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)
24. Yao, A.: How to Generate and Exchange Secrets. In FOCS'86, pp. 162-167. (1986)