Weighted Average Problem revisited under Hybrid and Malicious Model

H. Xiong, Echo P. Zhang, T.W. Chim, S.M. Yiu, Lucas C.K. Hui
Department of Computer Science
The University of Hong Kong
Pokfulam Road, Hong Kong
Email: {hxiong, pzhang2, twchim, smyiu, hui}@cs.hku.hk

***Abstract*—It is getting more common for two or more parties to jointly compute some statistics, say for marketing, by combining information on their private databases without disclosing the private data to the others. The core problem is usually known as secure multi-party computation (SMC). A number of solutions have been proposed. However, almost all of them assume a semi-honest model which is unrealistic. On the other hand, protocols that work under the malicious model (all participating parties can be malicious) are usually complicated and expensive in terms of communication and computation. In this paper, we try to consider a more reasonable model, the hybrid security model, in which at least one party is semi-honest. We want to make sure that the malicious parties will not get the correct final result if they perform malicious behaviors. We propose a scheme to solve the two-party weighted average problem (WAP) under this hybrid security model. We also show that the scheme can be extended to work under the malicious model using any fair exchange scheme. We formally show that our schemes are secure. We also implemented the schemes and showed that our scheme under the hybrid security model is reasonably fast and efficient for practical use.**

***Index Terms*—Secure multi-party computation, weighted average problem, hybrid security model, privacy preservation, homomorphic encryption.**

I. Introduction

With the development of information and communications technology, it is feasible to share data which is owned by different parties and stored in different locations. Sharing data for marketing or research purposes is getting more common. However, privacy is still a main concern. The following shows an example. A number of banks wish to jointly compute some statistics based on their transaction records to conduct a study for possible credit fraud behaviors. Although the banks are willing to cooperate with one another, they do not want to reveal the transaction records to other parties due to privacy policies, legal constrains or their own benefits. What we need is the ability to compute the desired statistics based on all transaction records for mining purposes without having actually sharing or disclosing any record [1].

Secure multi-party computation (SMC) provides a solution to this problem. SMC is first proposed by Yao [2] in 1986 and mainly concerns about the problem of evaluating a function based on the secret inputs from two or more parties. Roughly speaking, a protocol can be regarded as a SMC protocol if it satisfies the requirement that participating parties get no information about the inputs of other parties only based on the final result and the intermediate messages collected during the execution of the protocol. A formal definition will be given in Section 3. Generally speaking, there are two types of security models in SMC: semi-honest model and malicious model. In the semi-honest model, it is assumed that each party follows each step of the protocol, however, the adversary will attempt to infer additional information from the final result and the messages collected during execution. In the malicious model, the adversary can diverge arbitrarily from normal execution of the protocol. It has been proven that for any polynomial-time algorithm, there exists a polynomial-time secure protocol that achieves the same functionality under either semi-honest or malicious model [3]. Most protocols such as [4] and [5] are designed under the semi-honest model. Protocols under semi-honest model are usually quite efficient but it is not realistic to assume that all adversaries are semi-honest [1] in practice. On the other hand, protocols designed for the malicious model are usually more complicated and with significant overheads on communications and computation.

Our contributions: Meanwhile, it is also not necessary to assume that all adversaries are malicious. If all parties are malicious, it is not necessary to guarantee any of them to get the correct result. On the other hand, if there are some semi-honest parties, the protocol should guarantee that they are protected no matter how many malicious parties there are. There are other examples showing that this model is more realistic. Let us consider the common server-client model, usually the server is semi-honest if not being hacked but the clients are either semi-honest or malicious. In this paper, we propose to study this new model, the hybrid security model, under which we have two types of parties (both semi-honest and malicious). The formal definition of this hybrid security model is given in Section 3.

Based on this hybrid security model, we study a fundamental problem in SMC, the two-party weighted average problem (WAP), in which the two parties want to compute the weighted sum of a set of values with each one holds a subset of these values and the corresponding weights. WAP was firstly discussed in [4] and is being applied in a variety of areas, such as clustering and decision tree building. Early works only discuss WAP under the semi-honest model. In

this paper, we propose a novel protocol to solve WAP under the hybrid security model. We also show how to extend the protocol to work under the malicious model using any fair exchange scheme. We implement the schemes and show that the performance of our scheme under the hybrid security model is efficient and can be used in practical cases.

The rest of our paper is organized as follows. In Section II, we first describe a few exceptional behavior of participating parties which are not considered by our model, then followed by some basic concepts of homomorphic encryption which will be used in our protocol. The hybrid security model is formally defined in Section III. Our proposed protocol to solve WAP is described in details in Section IV. The analysis and evaluation of our schemes are given in Sections V and VI, respectively. Finally, Section VII concludes the paper.

II. Preliminaries

A. Secure Multi-Party Computation

Secure Multi-Party Computation (SMC) was first proposed by Yao [2] as a Millionaire Problem and extended by [6]–[9]. [10] even proved that SMC is equivalent to requiring a secure computation. SMC mainly concerns about the problem of evaluating a function with two or more parties' private inputs such that after running the protocol, each party holds a share of the output and no additional information is revealed except what is implied directly by the parties' own input and output. As mentioned earlier, there are generally two main security models discussed in earlier works. They are the semi-honest model and the malicious model. We note that the followings cannot be handled by both models.

1) Parties refuse to participate in the protocol.
2) Parties using invalid input instead of their actual data.
3) Parties abort the protocol prematurely.

Without loss of generality, our proposed hybrid security model also does not take these into consideration.

B. Homomorphic Encryption

Homomorphic encryption is a special type of encryption in which the result of applying a special algebraic operation on the plain text(s) can be obtained by applying another (can be different or the same) algebraic operation on the corresponding cipher text(s). Thus, even the user does not know the plain text(s), he/she can still obtain the result of applying that algebraic operation on the plain text(s). The formal definition of homomorphic encryption is as follows.

Let $E : R \times X \rightarrow Y$ be a probabilistic public key encryption function, where R, X and Y are finite domains identified with an initial subset of integers and $D : Y \rightarrow X$ be a private decryption function, such that $\forall(r, x) \in R \times X$, $D(E(r, x)) = x$.

In this paper, we adapt Paillier's public key homomorphic encryption function [11], which has the following properties:

1) The encryption function is additive homomorphic, which means that $\forall(r_1, x_1)$, $(r_2, x_2) \in R \times X$, $\prod(E(r_1, x_1), E(r_2, x_2)) = E(r_3, x_1 + x_2)$, where r_3 can be computed from x_1, r_1, x_2 and r_2 in polynomial time.
2) The encryption function is semantically secure, which means that a set of cipher texts do not provide extra information about the plain texts to the polynomial-bounded computing power adversary.

III. Hybrid Security Model

In this section, we provide the formal definition of the proposed hybrid security model.

A. Definition 3.1

Let α represent the set of parties who are defined in the semi-honest model and β represent the set of parties who are defined in the malicious model. The hybrid security model consists of at least one party belonging to α. Thus, the hybrid security model includes the semi-honest model as a subset. That is, protocol designed for the hybrid security model can be executed correctly even if all parties are semi-honest.

Our hybrid security model can be considered as a trade-off between security and efficiency. This model can be used in several application scenarios such as the server-client scenario. Like the malicious model, there are several behaviors this model cannot handle. Let us recall them again here.

1) Parties belonging to β refuse to participate in the protocol.
2) Parties belonging to β use other input instead of their actual data.
3) Parties belonging to β abort the protocol.

B. Definition 3.2

A protocol ϕ under the hybrid security model must satisfy the following requirements.

1) Privacy: ϕ satisfies the security requirement of a semi-honest secure multi-party (SSMC) protocol.
2) Fairness: Parities who perform malicious behaviors cannot get the correct final result.

The formal definition of privacy is as follows:

Let x and y be the inputs of the two parties respectively and both parties want to compute $f(x, y)$. Further let Π be a two-party protocol to compute f. The view of the first party after executing Π can be defined as $VIEW_1^{\Pi}(x, y) = (x, r, m_1, \ldots, m_t)$, where r are the random bits generated by Party 1 and $m_1, \ldots, m_t$ is the sequence of messages received by Party 1. The view of Party 2 can be defined in the same way.

Π is said to privately compute f if there exists probabilistic polynomial-time algorithms S_1 and S_2 such that

$S_1(x, f(x, y))_{x,y} \equiv^C VIEW_1^{\Pi}(x, y)_{x,y}$

$S_2(x, f(x, y))_{x,y} \equiv^C VIEW_2^{\Pi}(x, y)_{x,y}$

Here, $\equiv^C$ denotes statistically indistinguishable.

IV. Our Solutions

In this section, we firstly explain what the weighted average problem (WAP) is. Then we review the solution for solving the WAP under the semi-honest model as proposed in [4]. Next we discuss our proposed solutions for solving the same problem under the hybrid security and malicious models.

1) The Weighted Average Problem: Weighted average, or weighted mean, is the average value of weighted data points where different data points contribute non-equally to the final average. We define it formally as follows.

Input: $\{x_1, \ldots, x_n\}$ and the weight of x_i is w_i where $i \in \{1, \ldots, n\}$
Output: $\bar{x} = \frac{\sum_{i=1}^{n}(w_i \cdot x_i)}{\sum_{i=1}^{n} w_i}$

Here, if $\{x_1, \ldots, x_n\}$ belong to two separate parties. They need to cooperate to calculate $\bar{x}$, meanwhile each party still wants to keep their private data. In particular, we need to calculate the mean value of Alice's and Bob's inputs (see below) and at the same time without disclosing their private data to the counterpart.

Input:
Alice: $\{x_1, \ldots, x_a\}$ and the weight of x_i is $w1_i$ where $i \in \{1, \ldots, a\}$
Bob: $\{y_1, \ldots, y_b\}$ and the weight of y_j is $w2_j$ where $j \in \{1, \ldots, b\}$
Output:
$\bar{xy} = \frac{\sum_{i=1}^{a} w1_i \cdot x_i + \sum_{j=1}^{b} w2_j \cdot y_j}{\sum_{i=1}^{a} w1_i + \sum_{j=1}^{b} w2_j}$

Here, we only focus on the situation of the same weighted value of x_i and y_j, that is, $w1_i = w2_j$ where $i \in \{1, \ldots, a\}$ and $j \in \{1, \ldots, b\}$. Then, we can simplify the above formula of multi-party weighted average into $\bar{xy} = \frac{X+Y}{a+b}$ where X is the sum value of $\{x_1, \ldots, x_a\}$ and Y is the sum value of $\{y_1, \ldots, y_b\}$. This is one kind of Multi-party Computation (MC). Furthermore, each participant here requires his / her private data not disclosed during the collaboration. Both of them want to get the value of $\frac{X+Y}{a+b}$ without disclosing the knowledge of X, a and Y, b, respectively, to the other party. This problem belongs to the Secure Multi-party Computation (SMC) category.

2) Solving WAP under the semi-honest model: The problem of computing $\frac{X+Y}{a+b}$ has been proposed by [4]. However, their solution is not perfect since it can only guarantee its security under the semi-honest model, which assumes that participants follow the protocol from the beginning until the end. Thus, their scheme is open to attack by a malicious party easily. For completeness, we restate their solution as follows.

1) Alice sends Bob: $E_A(x)$, $E_A(a)$ where $E_A(\cdot)$ is an encryption function which is encryptable by both Alice and Bob but is only decryptable by Alice;
2) Bob generates Z_1 and Z_2 (suppose $Z_1 = Z_2$), computes $E_A(x)^{Z_1}$, $E_A(a)^{Z_2}$, and sends $E_A(Z_1x + Z_1y)$, $E_A(Z_2a + Z_2b)$ to Alice;
3) Alice decrypts them and performs the division $E_A(Z_1x + Z_1y)$ in order to get $(x+y)/(a+b)$. This is possible because Z_1 can be cancelled by Z_2.

In this solution, if Bob does not follow the protocol honestly, Alice cannot get the precise final answer after the execution of the protocol. Consider that Bob uses two different values, Z_1 and Z_2, during Step 2. This will lead to an unfair consequence that Alice gets a wrong value of $Z_1(x + y)/(Z_2(a + b))$, but Bob gets a correct one by simply multiplying the answer by Z_2/Z_1. Therefore, we aim at improving this solution and propose a novel solution to guarantee that it is secure under the hybrid security model and under the malicious model.

3) Solving WAP under hybrid security model: We propose a novel solution to solve WAP under the hybrid security model. The details of the solution is given below.

1) Alice generates Z_1 and sends to Bob: $E_A(Z_1x + a)$, $E_A(Z_1)$ where $E_A(\cdot)$ is an encryption function which is encryptable by both Alice and Bob but is only decryptable by Alice;
2) Bob generates Z_2, computes $E_A(Z_1Z_2(x+y)+Z_2(a+b))$ and sends to Alice;
3) Alice decrypts to obtain $Z_1Z_2(x + y) + Z_2(a + b)$ and sends $E_A(a)$ to Bob;
4) Bob sends $E_A(Z_2(a + b))$ to Alice;
5) Alice calculates $A = Z_1Z_2(x + y)$, $B = Z_2(a + b)$ and $(x + y)/(a + b) = A/(BZ_1)$.

Here we assume that Alice is semi-honest and she will send Bob the final result $(x+y)/(a+b)$. Bob can be malicious but if he performs malicious behavior he can not get the correct final result from Alice. For example, if Bob puts another value Z_2', instead of Z_2 into $E_A(Z_2(a + b))$, he cannot recover the final value on his own since he does not know the value of Z_1.

4) Solving WAP under the malicious model: Next we extend our solution above to work under the malicious model. Our solution consists of two phases.

1) Computation phase: Compute the prescribed function $f(x, y, a, b) = (x + y)/(a + b)$ separately and privately.
2) Verification phase: Verify whether other parties have put correct values during the computation phase.

We aim at meeting two requirements:

1) Privacy: Each party cannot compute additional information (other than its own private data) from the messages received during execution.
2) Verifiable: Parties putting fabricated value will be detected during the verification phase.

To enhance readability, we summarize our solution in Table I and Table II and we will analyze it in details in the next section. In the table, we denote $E_A(\cdot)$ as an encryption function which is encryptable by both Alice and Bob but is only decryptable by Alice and denote $E_B(\cdot)$ as an encryption function which is encryptable by both Alice and Bob but is only decryptable by Bob. Basically the computation phase includes a total of five steps. In the first four steps, each of the

two parties includes one option for the other party. Meanwhile, the order of these two options will not affect the fairness between participants and no one can disclose any private data from the information obtained.

In this protocol, Alice chooses random numbers Z_1 and Z_3 while Bob chooses random numbers Z_2 and Z_4. Since Bob has no idea about Z_1, he cannot change the coefficients of x and y. Otherwise, he cannot eliminate the effect of this change. So does Alice. Also if Bob intends to change the coefficients of a and b, he still cannot succeed. It is because in the fourth round, Alice will use whatever Bob sends back as the denominator after decryption. For the fair exchange scheme in the verification phase, any existing gradual release timed commitments scheme [12], [13] can be adopted to ensure that both Alice and Bob obtains the value $\frac{x+y}{a+b}$ from the other party at about the same time.

V. Security Analysis

In Sections V-A and V-B, we provide the security analysis for the protocol of WAP under the hybrid security model. In Section V-C, we show the security analysis under the malicious model.

A. Privacy preservation

Intuitively, if neither of Alice nor Bob can infer the private data of another party (y, b for Alice and x, a for Bob), we can conclude that the protocol for solving the weighted average problem (WAP), P_{WAP}, is privacy preserving.

The sequence of messages which Alice gets is $Z_1Z_2(x+y)+Z_2(a+b)$ and $Z_2(a+b)$, and Alice knows the values x, a and Z_1. Therefore, we can form two equations:

$$\begin{cases} Z_1Z_2(x+y)+Z_2(a+b)=C_1 \\ Z_2(a+b)=C_2 \end{cases}$$

Alice cannot infer the value of the three variables y, b and Z_2 from the two equations due to the theory of linear algebra.

The sequence of messages which Bob gets is $E(Z_1x+a)$, $E(Z_1)$, $E(a)$ and Bob knows the values y, b and Z_2. Bob cannot infer any knowledge about Z_1, x and a since the encryption algorithm is assumed to be semantically secure.

According to Definition 3.2,

$$\begin{cases} VIEW_{Alice}^{P_{WAP}}(x,a) \\ =(x,a,Z_1Z_2(x+y)+Z_2(a+b),Z_2(a+b)) \\ \\ VIEW_{Bob}^{P_{WAP}}(y,b) \\ =(y,b,E_A(Z_1(x+a)),E_A(Z_1),E_A(a)) \end{cases}$$

Let Z' be a random number, therefore, we define that $S_1(x,a,\frac{x+y}{a+b})$ as follows:

$(x,a,Z'Z_1\frac{x+y}{a+b}+Z',Z')$

$VIEW_{Alice}^{P_{WAP}}(x,a)$ and $S_1(x,a,\frac{x+y}{a+b})$ are statistically indistinguishable. The reason is that Z_2 is a random number, therefore, $Z_2(a+b)$ is a random number as well. Since the encryption scheme is semantically secure, Bob cannot gain extra information from the encrypted values $E(Z_1*(x+a))$, $E(Z_1)$ and $E(a)$. In other words, with the randomly chosen messages x', a' and Z_1', Bob cannot distinguish between $VIEW_{Bob}^{P_{WAP}}(y,b)$ and $(y,b,E(Z_1'(x'+a')),E(Z_1'),E(a'))$ with more than negligible probability.

Therefore, we can conclude that P_{WAP} privately computes $(x+y)/(a+b)$.

B. Fairness

In this hybrid model, the behavior of Alice is defined in semi-honest model, which means that Alice will follow each step of the protocol correctly but want to infer the private values of Bob from the message she receives. The behavior of Bob is defined in malicious model, which means that Bob will not only try to infer the private values of Alice, but also will abort the protocol or put several wrong values during the protocol so as to gain some advantage.

A protocol will work under hybrid model if:

1) Neither of the parties participated in the protocol can gain any knowledge about the private value of another party.
2) The malicious party cannot get the correct final result if he performs malicious behavior.

P_{WAP} has already been proven to meet the first condition, and we will analyze whether it meets the second condition as well.

If Bob aborts the protocol in any step, Alice will not send the final value $(x+y)/(a+b)$. As a result, Bob cannot compute it from the encrypted values he receives since the encryption scheme is assumed to be semantically secure.

If Bob sends y' or b' instead of y or b to Alice, he cannot recover the correct value $(x+y)/(a+b)$ from $(x+y')/(a+b')$ since Bob is lack of the knowledge about x and a.

If Bob sends $E(a+b)^{Z'}$ instead of $E(a+b)^{Z}$ to Alice, then in the last step, he will receive $\frac{Z}{Z'}\frac{x+y}{a+b}+\frac{Z_2-Z'}{Z_1Z'}$ instead of $\frac{x+y}{a+b}$. Bob cannot recover this value since he is lack of the knowledge about Z_1.

Therefore, we can conclude that Bob will not get the final correct result under the condition that he performs any malicious behavior. P_{WAP} will work under the hybrid security model according to the analysis above.

C. Security of our solution under malicious model

The malicious protocol can be viewed as the two way protocols under hybrid security model. Therefore, the proof of the privacy preservation property is quite similar to that in Section V-A. We mainly concern about the problem that whether the verification phase works.

In this protocol, the inputs of Alice are x, a, Z_1, Z_3 and that of Bob are y, b, Z_2, Z_4. We guarantee that:

1) Malicious parties will not get the correct final result if he puts a wrong value during the protocol, or
2) The malicious behavior that lead another party to get an incorrect final result can be detected in the verification phase.

Let us take Alice as an example. If Alice puts incorrect values x', a' into $E_A(Z_1x'+a')$ or $E_A(a')$, then obviously, she will not get the correct final result.

TABLE I
COMPUTATION PHASE

Alice	Bob
sends $E_A(Z_1 \cdot x + a)$, and $E_A(Z_1) \rightarrow$	
	$\leftarrow$ sends $E_B(Z_2 \cdot y + b)$, and $E_B(Z_2)$
sends $E_B(Z_2(x+y) + (a+b))^{Z_3} \rightarrow$	
	$\leftarrow$ sends $E_A(Z_1(x+y) + (a+b))^{Z_4}$
sends $E_A(a) \rightarrow$	
	$\leftarrow$ sends $E_B(b)$
sends $E_B(a+b)^{Z_3} \rightarrow$	
	$\leftarrow$ sends $E_A(a+b)^{Z_4}$
gets $WAP_A = \frac{x+y}{a+b}$	gets $WAP_B = \frac{x+y}{a+b}$

TABLE II
VERIFICATION PHASE

Alice	Bob
Fair Exchange Scheme	
sends $M = \frac{x+y}{a+b} \rightarrow$	$\leftarrow$ sends $N = \frac{x+y}{a+b}$
compares M and N	compares M and N

Recall that in the verification phase, if Alice follows each step correctly, then she can send the correct value of $\frac{x+y}{a+b}$ to Bob in the verification phase. For Alice, she cannot cheat in the verification phase. Since if she performs malicious behavior in the computation phase, she cannot send the right value to Bob.

We analyze this protocol step by step to prove that this verification phase will work. If Alice puts x', a' into $E_B(Z_2 Z_3(x' + y) + Z_3(a' + b)$ or $E_B(Z_3(a' + b))$ to Bob, Bob will compute a wrong final value in the computation phase. However, since the encryption function is assumed to be semantically secure, Alice has no knowledge about the wrong final value of Bob. Therefore, she cannot fabricate the value to convince that they have the same final result and Bob can verify this type of behavior in the verification phase.

If Alice puts a different value for Z_3, say Z_3' into $E_B(Z_3'(a' + b))$, the final value Bob computes becomes $\frac{x+y}{a+b} + \frac{Z_3 - Z_3'}{Z_3 Z_2}$. However, since Alice has no knowledge about Z_2, she cannot fabricate and send the value to convince Bob that the final results of them are the same.

The verification phase will work for Bob under the same reason.

VI. EXPERIMENTAL RESULTS

We implement all the protocols discussed using Java which is executed on a desktop computer with 2 GHz processor and 3 GB memory. For the homomorphic encryption functions, we adopt an open source Paillier library and we set the key size for key generation to 1,024 bits. We vary the data size of x, y, a and b from 100 bits to 1,000 bits in steps of 100 bits. For each data size N (bits), we random pick 4 integers in the range $[0, 2^N)$ and assign them to x, y, a and b respectively. We then input these 4 values into each of the three protocols (under the semi-honest, the hybrid security and the malicious models) and measure the total processing time. We then repeat this process by 100 times to obtain an average value. We ignore any communication time in this experiment because it should be small enough as compared to processing time in today's fast network connections. The results are summarized in Table III and Figure 1.

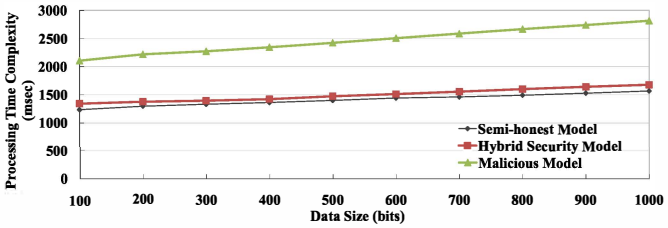

Fig. 1. Processing Time Complexity vs. Data Size

From Table III and Figure 1, we can see that longer processing time is required as the data size increases. If we compare the processing time for semi-honest model and hybrid security model, we find that the processing time for hybrid security model is 8.6% to 7.1% more than that for semi-honest model as the data size increasing from 100 bits to 1000 bits. This is reasonable as our solution under hybrid security model involves about the same number of cryptographic functions as that under the semi-honest model.

If we further compare the processing time for the hybrid security model and the malicious model, we find that the processing time for the malicious model is 56.9% to 67.6% more than that for the hybrid security model as the data size increasing from 100 bits to 1000 bits. This is because our solution under the malicious model involves almost a double of cryptographic functions as that under the hybrid security model.

To conclude, our solution under the hybrid security model provides a more secure environment for solving the WAP but yet the increase in processing time is only marginal (at most 8.6%). Our solution under malicious model demonstrates that such a fully secure environment is possible. However, there is still room to improve the complexity of our solutions and we will leave it as our future work.

TABLE III
EXPERIMENTAL RESULTS

Data Size (bits)	Complexity (msec) (Semi-honest Model)	Complexity (msec) (Hybrid Security Model)	Complexity (msec) (Malicious Model)
100	1236.7	1343.4	2107.2
200	1298.2	1378.0	2219.6
300	1333.6	1397.1	2274.2
400	1364.5	1423.6	2344.7
500	1401.3	1475.3	2423.8
600	1444.2	1514.2	2507.8
700	1463.4	1557.1	2586.1
800	1492.9	1601.9	2667.7
900	1529.8	1643.5	2739.4
1000	1568.7	1679.7	2815.8

VII. CONCLUSIONS

In this paper, we revisit the weighted average problem (WAP). This problem is known as the secure multi-party computation (SMC) problem and a number of solutions have been proposed in the research community. However, almost all of them assume a semi-honest model which is unreasonable in general. While not a proof, protocols under malicious model are more complicated and expensive. In view of this, we propose a novel hybrid security model in this paper. We also propose schemes to solve the WAP under hybrid security and malicious models. We formally prove that our scheme can achieve privacy preservation. Through implementation, we show that our novel scheme under hybrid security model is comparable to that under semi-honest model in terms of processing time complexity. In the future, we will enhance our solution under the malicious model and will adapt our solutions to some interesting problems in the community such as privacy preserving decision tree construction.

ACKNOWLEDGMENT

The work described in this paper was partially supported by the General Research Fund from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. RGC GRF HKU 713009E), the NSFC/RGC Joint Research Scheme (Project No. N_HKU 722/09), HKU Seed Fundings for Applied Research 201102160014, and HKU Seed Fundings for Basic Research 201011159162 and 200911159149.

REFERENCES

[1] W. Jiang and C. Clifton, "AC-Framework for Privacy-Preserving Collaboration," in *Proceedings of the SIAM International Conference on Data Mining*, 2007.

[2] A. Yao, "How to Generate and Exchange Secrets," in *Proceedings of the IEEE 27th Annual Symposium on Foundations of Computer Science*, 1986, pp. 162 – 167.

[3] O. Goldreich, S. Micali, and A. Wigderson, "How to Play any Mental Game or a Completeness Theorem for protocols with Honest Majority," in *Proceedings of the 19th STOC*, 1987, pp. 208 – 229.

[4] S. Jha, L. Kruger, and P. McDaniel, "Privacy Preserving Clustering," in *Proceedings of the 10th European Symposium In Computer Security*, 2005.

[5] Y. Lindell and B. Pinkas, "Privacy Preserving Data Mining," *Springer CRYPTO 2000, LNCS 1880*, pp. 36 – 54, 2000.

[6] O. Goldreich, S. Micali, and A. Wigderson, "Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems," in *Proceedings of the 27th Annual IEEE Symposium on Foundations of Computer Science*, 1986, pp. 174 – 187.

[7] J. Kilian, "Founding Cryptography on Oblivious Transfer," in *Proceedings of the 20th Annual ACM Symposium on the Theory of Computation (STOC)*, 1988, pp. 20 – 31.

[8] O. Goldreich, S. Micali, and A. Wigderson, "Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems," *Journal of the ACM*, pp. 691 – 729, 1991.

[9] R. Canetti, "Security and Composition of Multiparty Cryptographic Protocols," *Journal of Cryptology*, pp. 143 – 202, 1999.

[10] M. Bellare, O. Goldreich, and A. Mityagin, "The Power of Verification Queries in Message Authentication and Authenticated Encryption," *IACR Eprint archive*, 2004.

[11] P. Paillier, "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes," in *Proceedings of the EUROCRYPT'99*, 1999, pp. 223 – 238.

[12] B. Pinkas, "Fair secure two-party computation," in *Proceedings of IACR Eurocrypt (EUROCRYPT03), Warsaw, Poland*, 2003, pp. 87 – 105.

[13] D. Boneh and M. Naor, "Timed commitments," in *Advances in Cryptology and Crypto 2000, LNCS, vol. 1880, Springer-Verlag*, 2000, pp. 236 – 254.