This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2837665, IEEE Access

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Secure Comparison under Ideal/Real Simulation Paradigm

CHUAN ZHAO[1,2], SHENGNAN ZHAO[3], BO ZHANG[1,2], ZHONGTIAN JIA[1,2], ZHENXIANG CHEN*[1,2], MAURO CONTI[4]

[1]Shandong Provincial Key Laboratory of Network Based Intelligent Computing, University of Jinan, Jinan 250022, China
[2]School of Information Science and Engineering, University of Jinan, Jinan 250022, China
[3]School of Computer Science and Technology, Shandong University, Jinan, China
[4]Department of Mathematics, University of Padova, Padua, Italy

Corresponding author: Zhenxiang Chen (e-mail: czx@ujn.edu.cn).

This work is supported by National Natural Science Foundation of China (No. 61702218, 61672262), Natural Science Foundation of Shandong Province (No. ZR2014JL042, ZR2014FL011, ZR2015FL023), Shandong Provincial Key R&D Program (No. 2016GGX101001), CERNET Next Generation Internet Technology Innovation Project (No. NGII20160404), Doctoral Program of University of Jinan (No.160100224), Science and Technology Program of University of Jinan (No. XKY1709), and International Joint Training Program for Young and Middle-aged Scholar of Shandong Province.

ABSTRACT Secure comparison problem, also known as Yao's Millionaires' problem, was introduced by Andrew Yao in 1982. It is a fundamental problem in secure multi-party computation. In this problem, two millionaires are interested in determining the richer one between them without revealing their actual wealth. Yao's millionaires' problem is a classic and fundamental problem in cryptography. The design of secure and efficient solutions to this problem provides effective building blocks for secure multi-party computation. However, only a few of the solutions in the literature have succeeded in resisting attacks of malicious adversaries, and none of these solutions has been proven secure in malicious model under ideal/real simulation paradigm.

In this paper, we propose two secure solutions to Yao's millionaires' problem in malicious model. One solution has full simulation security, and the other solution achieves one-sided simulation security. Both protocols are only based on symmetric cryptography. Experimental results indicate that our protocols can securely solve Yao's millionaires' problem with high efficiency and scalability. Furthermore, our solutions show better performance than state-of-the-art solutions in terms of complexity and security. Specifically, our solutions only require $O(|U|)$ symmetric operations at most to achieve simulation-based security against malicious adversaries, where U denotes the universal set and $|U|$ denotes the size of U.

INDEX TERMS Ideal/Real Simulation Paradigm, Malicious Model, Secure Comparison, Secure Multi-Party Computation, Simulation-based Security, Yao's Millionaires' Problem

I. INTRODUCTION

Joint computation among various organizations or individuals through the Internet is becoming increasingly frequent given the development of big data and distributed computing technologies. Data owners can obtain valuable information by conducting cooperative computation with others. However, a computation may not only be performed among mutually trusted parties but also among competitors. In the latter case, participants are likely to behave dishonestly and attempt to obtain useful information about the private data of the other participants. Therefore, several security properties, such as privacy of individual inputs, correctness of computation output, and independence of inputs, should be guaranteed during the joint computation among different participants. This type of computation is called secure multi-party computation. It is aimed at constructing secure protocols for multiple participants to compute an objective function over their inputs jointly, while ensuring output correctness and maintaining input privacy against dishonest behaviors.

Yao first proposed secure multi-party computation in FOCS 1982 [1]. An interesting problem is presented in his seminal work. Two millionaires, namely, Alice and Bob, aim to determine the richer one between them while keeping their wealth a secret from each other. This problem is known as Yao's millionaires' problem. Alice and Bob aim to compute the inequality $x \leq y$ without disclosing anything other than the result, where x and y are the private inputs of Alice and Bob, respectively.

Secure multi-party computation [2] is a rapidly developing research area. It has significant influence in both theory and practice of cryptography. As a special case of secure multi-party computation, Yao's millionaires' problem discusses a basic operation in computation and thus has extensive applications in various fields of information security. On the one hand, in many cases, people must at times compare private numbers which are confidential and should not be revealed. Solutions to this problem are widely used in data privacy [3] [4] [5] [6] [7] and cloud security [8] [9] [10]. Specific applications include secure bidding and auction [11], privacy-preserving cooperative statistical analysis [12], secure outsourcing computation and cloud storage [13] [14] [15] [16] [17], and privacy-preserving machine learning [18] [19] [20]. On the other hand, Yao's millionaires' problem provides building blocks for many theoretical problems in secure multi-party computation, such as private information retrieval [21] [22], private set intersection [23] [24] [25], oblivious transfer and its variant [26] [27] [28] [29] [30], and oblivious RAM [31] [32] [33]. Yao's millionaires' problem, as a building block, has significantly influenced the security and efficiency of the protocols that invoke this problem. The study of secure and efficient solutions to Yao's millionaires' problem is crucial.

A. RELATED WORK

Yao's millionaires' problem has attracted considerable attention from cryptographic research community since its proposal. Many solutions to Yao's millionaires' problem have been introduced. The first solution, which was presented by Yao himself, was exponential in time and space [1]. Researchers have focused on decreasing the computation and communication costs of protocol execution to improve its efficiency. Specifically, Cachin used a partially trusted third party to reduce complexities in computation and communication [34]. Fischlin constructed a protocol in semi-honest model using Goldwasser-Micali cryptosystem [35]. Ioannidis and Grama proposed an efficient protocol with sub-optimal time and communication complexities [36]. Blake and Kolesnikov presented a protocol using additive homomorphism of the Paillier cryptosystem [37], whereas Lin and Tzeng suggested a protocol using multiplicative homomorphism of the ElGamal cryptosystem [38]. Blake and Kolesnikov proposed and applied efficient solutions to practical settings, such as secure auctions, using a new primitive conditional encrypted mapping [39]. Recently, Hezaveh and Adams investigated the socialist millionaires' problem and proposed a secure protocol against active adversaries based on Goldwasser-Micali cryptosystem [40]. Liu *et al.* extended Yao's millionaires' problem and aimed to determine $x < y, x > y, x = y$ in one execution. They presented a secure solution to the extended problem using a vectorization method and Paillier encryption scheme [41].

However, all the aforementioned solutions require asymmetric cryptographic operations, and thus remaining inefficient and impractical. Li *et al.* presented a solution to Yao's millionaires' problem based on symmetric cryptography. The key point of their solution is invoking a new efficient protocol for set-inclusion problem [42] . Futhermore, Li *et al.* presented two secure protocols for extended millionaires' problem based on only symmetric cryptographic operations [43].

In addition to improving execution efficiency, several works have focused on achieving fairness in the millionaires' problem [34] [39] [44] [45]. Several researchers have investigated the multi-party version of the millionaires' problem [46] [47] [41], and certain works have considered computationally unbounded participants [48] [49] to achieve information-theoretical security.

B. MOTIVATION AND CONTRIBUTIONS

Yao's millionaires' problem is an important problem in cryptography and secure multi-party computation. Efficiency and security of solutions to this problem significantly influence the outer protocols that invoke it. However, to the best of our knowledge, none of the solutions to Yao's millionaires' problem in the literature is verified to be secure in malicious model with simulation-based security.

Malicious model assumes stronger attacks from the adversary and reflects the reality better than semi-honest model. Participants may arbitrarily deviate from protocol specification according to the instruction of the adversary when these participants are corrupted by a malicious adversary, thereby complicating the case. In most cases, protocols that are proven secure against semi-honest adversaries are not secure in malicious model. Providing security in the presence of malicious adversaries is preferred because privacy, correctness, and other security properties are ensured, even when participants are corrupted by an active adversary with arbitrary attack policy. However, it is costly to compile a protocol secure in semi-honest model to one that is secure against malicious adversaries. Most existing works preserve security against malicious adversaries at the expense of heavy computation or communication costs.

Simulation-based security model is the simplest but the most rigorous among the security models for malicious adversaries. This model measures security by comparing the effect of executing objective protocol with the effect of an ideal world, where a trusted third party helps the participants to complete the objective computation task. Theoretically, simulation-based security in malicious model provides the strongest security level in reality. However, protocols that achieve this level of security are typically difficult to construct and inefficient. To the best of our knowledge, none of the state-of-the-art solutions to Yao's millionaires' problem has achieved simulation-based security. Although several generic protocols have been designed for performing any computation task securely [50] [51] [52] [53] [54] [55] [56] [57], the investiga-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2837665, IEEE Access

tion of specialized solutions to Yao's millionaires' problem is necessary to achieve high efficiency.

In this paper, we focus on exploring novel methods for securing Yao's millionaires' problem efficiently against malicious adversaries with simulation-based security. The main contributions of our work are summarized as follows:

- We propose two novel solutions to Yao's millionaires' problem. Both solutions are constructed in malicious model with strong security, that is, simulation-based security. In particular, one solution achieves full simulation security, whereas the other solution attains one-sided simulation security.
- We present a formal proof of security for both solutions under ideal/real simulation paradigm, which provides the simplest but most effective and rigorous method for evaluating the security of cryptographic protocols.
- Our solutions are more efficient than previous works in terms of computation and round complexities. Specifically, our protocols are constructed only through symmetric cryptographic operations and only one round of interaction between the participants in the online phase.
- We conduct experiments on our protocols. The experimental results indicate that both protocols are efficient. In particular, the second protocol is proven sufficiently efficient and scalable to be used in practice.

C. OUTLINE OF THE PAPER

The rest of this paper is organized as follows. We first review the preliminaries in Section II, including related building blocks and security definitions. Then we present a detailed description of the proposed solutions, provide rigorous security proofs in malicious model under ideal/real simulation paradigm, and analyze the efficiency in computation and round complexities in Section III. Next, experimental results on efficiency and scalability are described in Section IV, and comparison results with related work are presented in Section V. Lastly, we conclude this paper and indicate future work in Section VI.

II. PRELIMINARIES

In this section, we review several fundamental techniques and basic tools required in this paper, including negligible function, pseudorandom permutation, message authentication code, standard smart card, and security definition.

A. NEGLIGIBLE FUNCTION

A negligible function is one that is asymptotically smaller than any inverse polynomial function. Thus, we present the following definition:

Definition 1 (Negligible Function). *A function f from the natural numbers to the non-negative real numbers is* negligible *if for every positive polynomial p, there is an N such that for all integers $n > N$, it holds that $f(n) < \frac{1}{p(n)}$.*

In this paper, we denote a negligible function by negl.

B. PSEUDORANDOM PERMUTATION

A pseudorandom permutation is a bijective function that cannot be distinguished from a truly random permutation by any polynomial-time observer with practical effort.

We first introduce a keyed function [58] before describing the formal definition of pseudorandom permutation. A keyed function $F : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^n$ is a two-input function, where n is the security parameter. The first input is called the *key*, which is denoted as k. In typical usage, a key k is selected and fixed. Subsequently, F can be transformed into a single-input function $F_k : \{0,1\}^n \rightarrow \{0,1\}^n$ defined by $F_k(x) = F(k, x)$.

The formal definition of pseudorandom permutation based on the definition of keyed function is presented as follows:

Definition 2 (Pseudorandom Permutation). *Let* PRP *be a keyed function $P_k : \{0,1\}^n \rightarrow \{0,1\}^n$, where $k \in \{0,1\}^n$ is the key and n is the security parameter. We say* PRP *is a pseudorandom permutation if*

- *For any key $k \in \{0,1\}^n$, P_k is a bijection from $\{0,1\}^n$ to $\{0,1\}^n$.*
- *For any key $k \in \{0,1\}^n$ and any input $x \in \{0,1\}^n$, there is a polynomial-time algorithm to evaluate $P_k(x)$.*
- *For any probabilistic polynomial-time distinguisher D, there is a negligible function* negl *such that:*

$$|Pr[D^{P_k(\cdot)}(1^n) = 1] - Pr[D^{f_n(\cdot)}(1^n) = 1]| \leq negl(n),$$

where $k \leftarrow \{0,1\}^n$ is chosen uniformly at random, and f_n is chosen uniformly at random from the set of permutations on n-bit string.

Any polynomial-time observer without knowledge of the key cannot distinguish the objective pseudorandom permutation from a truly random permutation. However, an individual who knows the key can efficiently compute the corresponding pseudorandom permutation and its inverse operation. Secure instantiations of pseudorandom permutations include modern block ciphers, such as 3DES and AES.

C. MESSAGE AUTHENTICATION CODE (MAC)

A message authentication code is a brief piece of information used to authenticate a message. Specifically, this code helps in confirming that the message comes from the stated sender and has been unchanged [59]. The MAC value protects data integrity and authenticity of a message by allowing verifiers, who also possess the secret key, to detect any changes to the message content. Formally, we provide the following definition:

Definition 3 (Message Authentication Code). *A message authentication code is a triple of efficient algorithms (*Gen, Mac, Vrfy*), where* Gen *denotes key-generation algorithm,* Mac *denotes tag-generation algorithm and* Vrfy *denotes verification algorithm. Specifically,*

- Gen *takes as input the security parameter 1^n and outputs a secret key $k \leftarrow_R$* Gen(1^n).

- Mac *takes as input a key* k *and a message* m*, and outputs a tag* $t \leftarrow \mathsf{Mac}_k(m)$.
- Vrfy *takes as input a key* k*, a message* m *and a tag* t*, and outputs a bit* $b := \mathsf{Vrfy}_k(m, t)$*, with* $b = 1$ *meaning* valid *and* $b = 0$ *meaning* invalid.

Correctness requirement. For every n*, every* k *output by* $\mathsf{Gen}(1^n)$ *and every* m *in the message space, the following equality should be satisfied:*

$$Pr[\mathsf{Vrfy}_k(m, \mathsf{Mac}_k(m)) = 1] = 1.$$

The following application scenario is considered. First, the sender of a message m, which is denoted as SENDER, runs the key-generation algorithm Gen, obtains a key k, and shares the key with the receiver, which is denoted as RECEIVER. Second, SENDER runs the tag-generation algorithm Mac to produce a MAC tag t. Subsequently, SENDER transmits t and the message m, which may be tampered with during the transmission, to RECEIVER. We denote the message and tag that RECEIVER obtains as m' and t', respectively. RECEIVER runs the verification algorithm Vrfy using key k, message m' and tag t', and outputs a bit b after receiving m' and t', thereby indicating whether the message was tampered with or not during transmission.

D. STANDARD SMART CARD

A smart card is a kind of pocket-sized card with an embedded integrated circuit. Smart card is a powerful tool that supports numerous functionalities, such as authentication, encryption, data storage, and data processing. In this paper, we consider standard smart cards rather than the special purpose ones due to reliability issues. If a special purpose smart card is used for a secure protocol, then we must believe that the vendor did not construct the functionality incorrectly or leave any backdoors on the card. By contrast, standard smart cards have been tested for many years. Thus, the possibility of malicious implementation and unintentional errors is minimal. Hazay and Lindell introduced standard smart cards in secure set intersection and oblivious database search to construct truly practical secure protocols in malicious model [60].

The standard smart cards used in this work must provide the following functionalities:

- *Symmetric cryptographic operations*. An important functionality used in this paper is symmetric cryptography, including pseudorandom permutation and message authentication code. The keys of these cryptographic schemes are generated outside of the smart cards. The keys can no longer be exported once imported unless deleted.
- *Usage counter*. A usage counter which indicates how many times this key can be used before it is deleted, will be defined once a key is imported.
- *Access control*. A challenge/response test is required for users to perform cryptographic operations and other functions supported by the smart cards to protect smart cards from unauthorized accesses.
- *Data storage*. Data storage is supported by standard smart cards. Nearly all data stored in a smart card, regardless whether private or public, can be read out of the smart card, except for the keys.

E. SECURITY DEFINITION

In this paper, we aim to achieve the strongest security level, that is, simulation-based security against malicious adversaries. We describe adversarial model and ideal/real simulation paradigm to formalize this security level.

1) Adversarial Model

Yao's millionaires' problem is a specific problem in secure two-party computation, a two-party case of secure multi-party computation. Secure two-party computation enables two mutually distrusted participants to complete a cooperative computation task securely on their private inputs, even if one of the participants is corrupted by an adversary. The power of the adversary is defined in adversarial model. This model includes details on whether the adversary is deterministic or randomized, uniform or non-uniform, static or adaptive, and how it interacts with the security game. In this work, we consider a randomized, non-uniform, static adversary with malicious behaviors, which is known as malicious adversarial model. Compared with corrupted parties in semi-honest model, participants in malicious model may arbitrarily deviate from the protocol specification according to the adversary's instructions, thereby complicating the case. In most scenarios, providing security in malicious model is preferred because it ensures that no adversarial attack can succeed. However, protocols that achieve this level of security are typically difficult to construct and less efficient.

2) Ideal/Real Simulation Paradigm

Protocols for secure two-party computation should preserve many security properties, such as correctness, privacy, and independence of inputs. However, the list of these required properties is not a formal definition of security. Ideal/real simulation paradigm, which is an effective method for defining security in secure two-party computation, is proposed to formalize security definition for secure two-party computation [59]. Ideal/real simulation paradigm is a standard and rigorous method for evaluating the security level of the objective protocols. This method involves an "ideal world" and a "real world". In the ideal world, a trusted third party assists two parties in accomplishing the joint computation task. Each participant is only required to transfer his/her own private input via a secure channel to the trusted third party, who is absolutely trustworthy and honest. The trusted third party computes the objective computation task honestly and sends back respective results to each participant upon receiving the inputs (See Fig. 1). From this perspective, the ideal world is considered a model that can achieve the highest level of security. In the real world, a protocol that computes the objective functionality is executed between two parties without any assistance from others (See Fig. 2).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2837665, IEEE Access

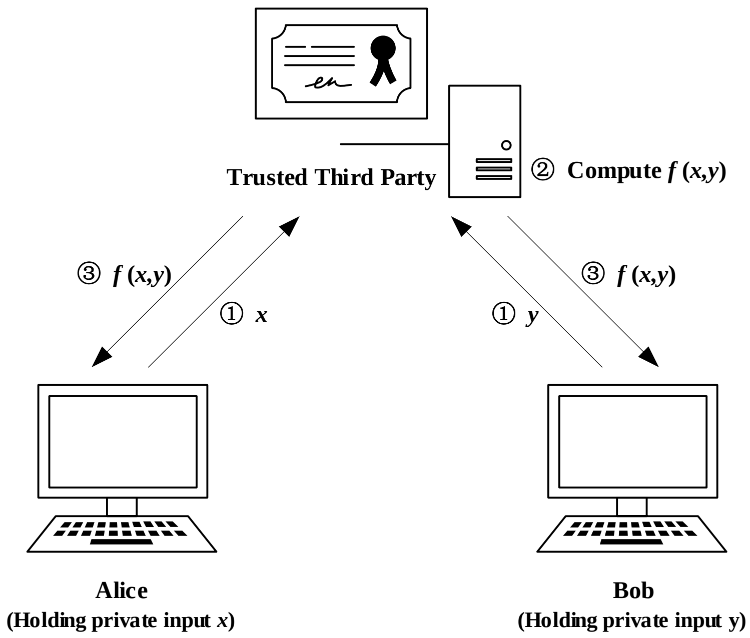

FIGURE 1. Ideal World

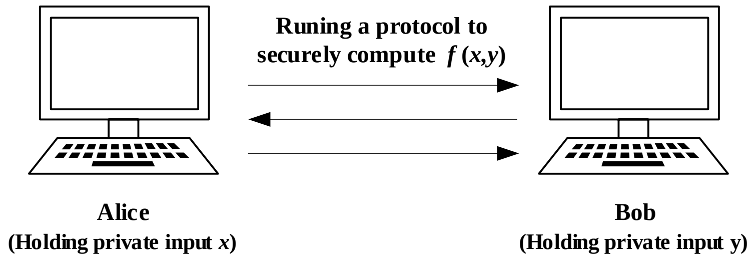

FIGURE 2. Real World

A protocol is considered secure under ideal/real simulation paradigm if the real world where the objective protocol is executed emulates the effect of the ideal world. Formally speaking, the objective protocol is considered secure under ideal/real simulation paradigm, if for every adversary in the real world, there exists an adversary in the ideal world that can simulate all the actions of the real adversary. This scenario ensures that the joint output distribution of the honest party and the adversary in a real protocol execution is indistinguishable with that in an ideal execution.

3) Formal Security Definition

We consider the security definition of secure two-party computation presented in [61] to formalize security definition for Yao's millionaires' problem. Specifically, denote $\mathrm{IDEAL}_{f,\mathcal{S}(z),i}(x,y,n)$ as the output pair of the honest party and an ideal adversary $\mathcal{S}$ in the ideal world, and denote $\mathrm{REAL}_{\pi,\mathcal{A}(z),i}(x,y,n)$ as the output pair of the honest party and a malicious adversary $\mathcal{A}$ in the real world, where f is the objective functionality, π is a two-party protocol for computing f, z is an auxiliary input to the adversary, $i \in \{\text{Alice},\text{Bob}\}$ is the index of the corrupted party, x is the input of Alice to f, y is the input of Bob to f and n is the security parameter. The formal security definition under ideal/real simulation paradigm with full simulation in malicious model is presented as follows:

Definition 4 (Full Simulation Security in Malicious Model)**.** *Let $f : \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^* \times \{0,1\}^*$ be a polynomial-time functionality and π is a two-party protocol for computing f. Protocol π is said to securely compute f in malicious model with full simulation if for every non-uniform probabilistic polynomial-time adversary $\mathcal{A}$ in the real world,* *there exists a non-uniform polynomial-time adversary $\mathcal{S}$ in the ideal world, such that for every $i \in \{\mathsf{Alice,Bob}\}$,*

$$\{\mathrm{IDEAL}_{f,\mathcal{S}(z),i}(x,y,n)\}_{x,y,z,n}$$

$$\stackrel{c}{\equiv} \{\mathrm{REAL}_{\pi,\mathcal{A}(z),i}(x,y,n)\}_{x,y,z,n},$$

where $\stackrel{c}{\equiv}$ denotes computational indistinguishability, $x, y, z \in \{0,1\}^*$ and $n \in \mathbb{N}$.

Full simulation security in malicious model provides a strong security level. It contains all the aforementioned security properties, including privacy, correctness, and independence of inputs. However, several cases exhibit that full simulation security is difficult or costly to be achieved. In these cases, a relaxed level of security, namely, one-sided simulation security, is helpful in constructing highly efficient protocols against malicious adversaries. In this security definition, only one participant, Bob, for example, has the output. Ideal/real simulation is achievable when Bob is corrupted and only privacy is ensured when Alice is corrupted. The privacy property ensures that Alice learns nothing about the private input y of Bob. We formalize this property by comparing the protocol view of the adversary that corrupts Alice. Specifically, we say Bob's input is private if the adversary that corrupts Alice cannot distinguish the case that Bob used input y with the case that Bob used another input y'. The formal definition is described as follows:

Definition 5 (One-Sided Simulation in Malicious Model)**.** *Let $f : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ be a polynomial-time functionality where only Bob receives output, and π is a two-party protocol for computing f. Protocol π is said to securely compute f in malicious model with one-sided simulation if the following holds:*

1. For every non-uniform probabilistic polynomial-time adversary $\mathcal{A}$ corrupting Bob in the real world, there exists a non-uniform polynomial-time adversary $\mathcal{S}$ in the ideal world, such that

$$\{\mathrm{IDEAL}_{f,\mathcal{S}(z),\mathsf{Bob}}(x,y,n)\}_{x,y,z,n}$$

$$\stackrel{c}{\equiv} \{\mathrm{REAL}_{\pi,\mathcal{A}(z),\mathsf{Bob}}(x,y,n)\}_{x,y,z,n},$$

where $\stackrel{c}{\equiv}$ denotes computational indistinguishability, $x, y, z \in \{0,1\}^*$ and $n \in \mathbb{N}$.

2. For every non-uniform probabilistic polynomial-time adversary $\mathcal{A}$ corrupting Alice, it satisfies that

$$\{\mathrm{VIEW}^{\mathcal{A}}_{\pi,\mathcal{A}(z),\mathsf{Alice}}(x,y,n)\}_{x,y,z,n}$$

$$\stackrel{c}{\equiv} \{\mathrm{VIEW}^{\mathcal{A}}_{\pi,\mathcal{A}(z),\mathsf{Alice}}(x,y',n)\}_{x,y',z,n},$$

where $\mathrm{VIEW}^{\mathcal{A}}_{\pi,\mathcal{A}(z),\mathsf{Alice}}(x,y,n)$ denotes the view of the adversary after a real execution of π, $x, y, y', z \in \{0,1\}^*$ and $n \in \mathbb{N}$.

III. PROPOSED SOLUTIONS

Now we introduce our solutions to Yao's millionaires' problem in detail. Without loss of generality, a universal set $U = \{0, 1, 2, \cdots, 2^l - 1\}$ exists, where l is a parameter that indicates the size of U. Two millionaires, Alice and Bob, want to jointly compute the functionality $f(x,y) = x \leq y?$, where $x \in U$ is the private wealth value of Alice and $y \in U$ is the private wealth value of Bob.

In this section, we propose two efficient solutions to the aforementioned problem. Both solutions can achieve simulation-based security against malicious adversaries. The first solution, called Full Simulatable Protocol, denoted as $\mathcal{P}_{Full}$, achieves full simulation security in malicious model under Definition 4. The second solution, called One-sided Simulatable Protocol, denoted as $\mathcal{P}_{One-sided}$, is secure against malicious adversaries with rigorous security proof under Definition 5.

A. FULLY SIMULATABLE PROTOCOL $\mathcal{P}_{FULL}$

We first present the protocol $\mathcal{P}_{Full}$, which achieves full simulation security.

Protocol 1. $\mathcal{P}_{Full}$***: Protocol for Computing*** $f(x,y) = x \leq y?$ ***with Full Simulation Security***

Inputs: *Alice inputs a private input $x \in U$ and Bob inputs a private input $y \in U$.*

Output: *Alice outputs nothing; Bob outputs 1 if $x \leq y$ and 0 otherwise.*

Initialization:

Step 1. *Alice chooses two keys $k, k_{\mathsf{MAC}} \leftarrow \{0,1\}^n$ in random, where k is the key of a pseudorandom permutation PRP and k_{MAC} is the key of a message authentication code MAC. Both PRP and MAC are embedded in a standard smart card $\mathsf{SC}_{\mathsf{Alice}}$. After obtaining this smart card, Alice imports k, k_{MAC} into it and sets the parameter Count as 1, which indicates that the total number of queries supported by this smart card is 1.*

Step 2. *Alice sends $\mathsf{SC}_{\mathsf{Alice}}$ to Bob via offline channel.*

Online Interaction:

Step 1. *Upon receiving the smart card $\mathsf{SC}_{\mathsf{Alice}}$, Bob acts as follows:*

a) *Inputs his private value y to $\mathsf{SC}_{\mathsf{Alice}}$, and obtains a set $\{\mathsf{PRP}_k(0), \mathsf{PRP}_k(1), \cdots, \mathsf{PRP}_k(y)\}$ where the elements in it are randomly permuted, denoted as $\psi(\mathsf{S})$;*

b) *Issues a $\mathsf{Complete}$ command to $\mathsf{SC}_{\mathsf{Alice}}$ and receives back a confirmation message Done and its MAC tag, denoted as $(\mathsf{Done}, \mathsf{Mac}_{k_{\mathsf{MAC}}}(\mathsf{Done}))$, where Mac is the tag generation algorithm;*

c) *Sends the MACed confirmation to Alice.*

Step 2. *Upon receiving the confirmation, Alice verifies its validity with the verification algorithm. If valid, Alice computes $\mathsf{PRP}_k(x)$ with key k and sends it to Bob.*

Step 3. *Bob determines whether $x \le y$ as follow. If $\mathsf{PRP}_k(x) \in \psi(\mathsf{S})$, then $x \le y$, Bob outputs 1; otherwise, $x > y$, Bob outputs 0.*

Please refer to Fig. 3 for a clear diagram of the proposed protocol.

1) Security Analysis

We now analyze the security of Protocol $\mathcal{P}_{Full}$. Formally, we have the following theorem.

Theorem 1. *If PRP is a pseudorandom permutation and MAC is message authentication code, then $\mathcal{P}_{Full}$ securely computes function $f(x, y) = x \le y?$ under Definition 4.*

Proof. Let f be the objective function and π be the two-party protocol presented above. Then π securely computes f in the presence of malicious adversaries under ideal/real simulation paradigm with full simulation security if the following satisfies:

For every non-uniform probabilistic polynomial-time adversary $\mathcal{A}$ in the real world, there exists a non-uniform polynomial-time adversary $\mathcal{S}$ in the ideal world, such that for each $i \in \{\mathsf{Alice}, \mathsf{Bob}\}$,

$$\{\mathrm{IDEAL}_{f,\mathcal{S}(z),i}(x, y, n)\} \stackrel{c}{\equiv} \{\mathrm{REAL}_{\pi,\mathcal{A}(z),i}(x, y, n)\},$$

where $x, y \in U$, $z \in \{0, 1\}^*$ and $n \in \mathbb{N}$.

We prove the above equation separately for the case that Alice is corrupted (i.e., $i = \mathsf{Alice}$) and the case that Bob is corrupted (i.e., $i = \mathsf{Bob}$). In each case, we construct an ideal, non-uniform polynomial-time adversary $\mathcal{S}$, also known as the simulator, to simulate the output distribution of the real execution. The simulator can internally invoke and interacts with the real adversary $\mathcal{A}$, thereby reading all the contents on $\mathcal{A}$'s output tape and writing on $\mathcal{A}$'s input tape. The simulator should satisfy two basic requirements:

- The simulator $\mathcal{S}$ should ensure that the real adversary $\mathcal{A}$ cannot distinguish whether it is interacting with an honest party or with the simulator to validate the invocation of $\mathcal{A}$.
- The simulator $\mathcal{S}$ should extract and send the real input of $\mathcal{A}$ to the trusted third party in the ideal world to ensure that the output distribution in the ideal world is the same as that in the real world.

The constructed simulator can be considered valid given that the aforementioned requirements are satisfied, and the indistinguishability of the two output distributions remains to be proven. Now we formally prove Theorem 1 separately for two cases.

Alice is corrupted. Suppose that the adversary attacking Protocol $\mathcal{P}_{Full}$ corrupts and controls Alice. Denote the adversary as $\mathcal{A}_{\mathsf{Alice}}$. We construct a simulator $\mathcal{S}_{\mathsf{Alice}}$ in the ideal world. The simulator internally invokes $\mathcal{A}_{\mathsf{Alice}}$ and interacts with it as Bob and $\mathsf{SC}_{\mathsf{Alice}}$. Besides, $\mathcal{S}_{\mathsf{Alice}}$ externally interacts with the trust third party computing f as the corrupted Alice. The simulator we construct is described as follows:

- $\mathcal{S}_{\mathsf{Alice}}$ invokes $\mathcal{A}_{\mathsf{Alice}}$ with its initial input, interacts with $\mathcal{A}_{\mathsf{Alice}}$ as Bob and $\mathsf{SC}_{\mathsf{Alice}}$. If any cheating of $\mathcal{A}_{\mathsf{Alice}}$ is detected, $\mathcal{S}_{\mathsf{Alice}}$ sends $\perp$ to the trusted third party as the simulation of Bob aborting the protocol, and outputs whatever $\mathcal{A}_{\mathsf{Alice}}$ outputs. Otherwise, $\mathcal{S}_{\mathsf{Alice}}$ continues.
- $\mathcal{S}_{\mathsf{Alice}}$ plays as $\mathsf{SC}_{\mathsf{Alice}}$. It obtains the keys k and k_{MAC} from $\mathcal{A}_{\mathsf{Alice}}$'s output tape. Both keys are supposed to be imported into $\mathsf{SC}_{\mathsf{Alice}}$ by $\mathcal{A}_{\mathsf{Alice}}$.
- $\mathcal{S}_{\mathsf{Alice}}$ plays as Bob. It computes a confirmation message $(\mathsf{Done}, \mathsf{MAC}_{k_{\mathsf{MAC}}}(\mathsf{Done}))$ with the key k_{MAC} and sends it to $\mathcal{A}_{\mathsf{Alice}}$.
- $\mathcal{S}_{\mathsf{Alice}}$ plays as Bob. It receives from $\mathcal{A}_{\mathsf{Alice}}$ a value $\mathsf{PRP}_k(x)$, denoted as e. $\mathcal{S}_{\mathsf{Alice}}$ computes $\mathsf{PRP}_k^{-1}(e)$ with k, and obtains the input value x of $\mathcal{A}_{\mathsf{Alice}}$.
- $\mathcal{S}_{\mathsf{Alice}}$ plays as corrupted Alice. It sends the extracted input x of $\mathcal{A}_{\mathsf{Alice}}$ to the trusted third party externally, and outputs whatever $\mathcal{A}_{\mathsf{Alice}}$ outputs and halts.

For a legible description of the constructed simulator $\mathcal{S}_{\mathsf{Alice}}$, please refer to the diagram shown in Fig. 4 (a).

First, we prove that the simulator $\mathcal{S}_{\mathsf{Alice}}$ constructed above is valid:

- As $\mathcal{S}_{\mathsf{Alice}}$ can read $\mathcal{A}_{\mathsf{Alice}}$'s output tape, it can easily obtain k_{MAC} and then computes the MACed message $(\mathsf{Done}, \mathsf{MAC}_{k_{\mathsf{MAC}}}(\mathsf{Done}))$ with the key k_{MAC}. Consequently, the real adversary $\mathcal{A}_{\mathsf{Alice}}$ cannot distinguish whether it is interacting with honest Bob or with the simulator, because messages sent by Bob and constructed by $\mathcal{S}_{\mathsf{Alice}}$ are identical;
- As $\mathcal{S}_{\mathsf{Alice}}$ can read $\mathcal{A}_{\mathsf{Alice}}$'s output tape, it can also obtain k and the encrypted value $\mathsf{PRP}_k(x)$, accordingly computes $\mathsf{PRP}_k^{-1}(\mathsf{PRP}_k(x))$ with k, and extracts the input value x of $\mathcal{A}_{\mathsf{Alice}}$.

Secondly, we prove that the joint output distribution of honest Bob and the adversary $\mathcal{A}_{\mathsf{Alice}}$ is indistinguishable with the joint output distribution of honest Bob and the simulator $\mathcal{S}_{\mathsf{Alice}}$:

- As the view of $\mathcal{A}_{\mathsf{Alice}}$ in the real protocol is indistinguishable with that in the simulation with $\mathcal{S}_{\mathsf{Alice}}$, the output distribution of the adversary $\mathcal{A}_{\mathsf{Alice}}$ is indistinguishable with that of the simulator $\mathcal{S}_{\mathsf{Alice}}$.
- In the real protocol execution, suppose $\mathcal{A}_{\mathsf{Alice}}$ uses x as its real input, and Bob uses y as his input. If $\mathsf{PRP}_k(x) \in \psi(\mathsf{S})$, it means $x \in \{0, 1, \cdots, y\}$, i.e., $x \le y$, $f(x, y)$ is 1; otherwise, $x > y$, $f(x, y)$ is 0. Therefore, the protocol result obtained by Bob is exactly $f(x, y)$. In the ideal world, as $\mathcal{S}_{\mathsf{Alice}}$ can successfully extract $\mathcal{A}_{\mathsf{Alice}}$'s input, and honest Bob uses the same input as in the real world, the output of Bob is just $f(x, y)$. Therefore, the output distribution of honest Bob in the ideal world is indistinguishable with that in the real world.

Protocol $\mathcal{P}_{Full}$ is secure in the case that Alice is corrupted.

Bob is corrupted. Suppose that the adversary attacking Protocol $\mathcal{P}_{Full}$ corrupts and controls Bob. Denote the adversary as $\mathcal{A}_{\mathsf{Bob}}$. We construct a simulator $\mathcal{S}_{\mathsf{Bob}}$ in the ideal world. The simulator internally invokes $\mathcal{A}_{\mathsf{Bob}}$ and interacts

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2837665, IEEE Access

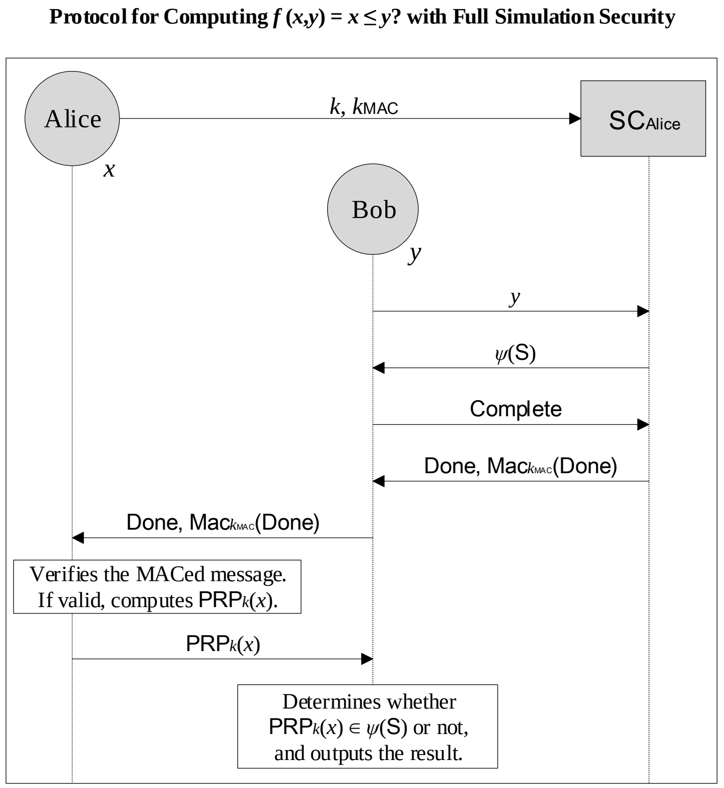

FIGURE 3. Diagram of $\mathcal{P}_{Full}$

with it as Alice and $\mathsf{SC}_{\mathsf{Alice}}$. Besides, $\mathcal{S}_{\mathsf{Bob}}$ externally interacts with the trust third party computing f as the corrupted Bob. The simulator we construct is described as follows:

- $\mathcal{S}_{\mathsf{Bob}}$ invokes $\mathcal{A}_{\mathsf{Bob}}$ with its initial input, interacts with $\mathcal{A}_{\mathsf{Bob}}$ as Alice and $\mathsf{SC}_{\mathsf{Alice}}$. If any cheating of $\mathcal{A}_{\mathsf{Bob}}$ is detected, $\mathcal{S}_{\mathsf{Bob}}$ sends $\perp$ to the trusted third party as the simulation of Alice aborting the protocol, and outputs whatever $\mathcal{A}_{\mathsf{Bob}}$ outputs. Otherwise, $\mathcal{S}_{\mathsf{Bob}}$ continues.
- $\mathcal{S}_{\mathsf{Bob}}$ plays as $\mathsf{SC}_{\mathsf{Alice}}$. It obtains $\mathcal{A}_{\mathsf{Bob}}$'s input y from $\mathcal{A}_{\mathsf{Bob}}$'s output tape. This value is supposed to be sent to $\mathsf{SC}_{\mathsf{Alice}}$ by $\mathcal{A}_{\mathsf{Bob}}$.
- $\mathcal{S}_{\mathsf{Bob}}$ plays as $\mathsf{SC}_{\mathsf{Alice}}$. It randomly chooses a pseudo-random permutation key $k \leftarrow \{0,1\}^n$, computes a set $\mathsf{S} = \{\mathsf{PRP}_k(0), \mathsf{PRP}_k(1), \cdots, \mathsf{PRP}_k(y)\}$ with the key k, and sends $\mathcal{A}_{\mathsf{Bob}}$ a random permutation version of S, denoted as $\psi(\mathsf{S})$.
- $\mathcal{S}_{\mathsf{Bob}}$ plays as corrupted Bob. It sends the extracted input y of $\mathcal{A}_{\mathsf{Bob}}$ to the trusted third party externally, and receives back the computation result 1 or 0, indicating whether $x \leq y$ or not, where x is the input of honest Alice in the real world.
- $\mathcal{S}_{\mathsf{Bob}}$ plays as $\mathsf{SC}_{\mathsf{Alice}}$. After receiving a $\mathsf{Complete}$ command from $\mathcal{A}_{\mathsf{Bob}}$, $\mathcal{S}_{\mathsf{Bob}}$ randomly chooses a MAC key $k_{\mathsf{MAC}} \leftarrow \{0,1\}^n$ and sends back a confirmation message MACed with k_{MAC}.
- $\mathcal{S}_{\mathsf{Bob}}$ plays as Alice. After receiving the MACed confirmation from $\mathcal{A}_{\mathsf{Bob}}$, $\mathcal{S}_{\mathsf{Bob}}$ sends back to $\mathcal{A}_{\mathsf{Bob}}$ a value x', which is computed as follows:
  - If the result obtained from the trusted third party is 1, $\mathcal{S}_{\mathsf{Bob}}$ sets x' as $\mathsf{PRP}_k(a)$, where a is a randomly chosen value from the set $\{0,1,2,\cdots,y\}$;
  - Otherwise, $\mathcal{S}_{\mathsf{Bob}}$ sets x' as $\mathsf{PRP}_k(a)$, satisfying $a \leftarrow_R \{0,1\}^n$ and $a \notin \{0,1,2,\cdots,y\}$.
- $\mathcal{S}_{\mathsf{Bob}}$ plays as Alice. It sends x' to $\mathcal{A}_{\mathsf{Bob}}$, outputs whatever $\mathcal{A}_{\mathsf{Bob}}$ outputs and halts.

For a legible description of the constructed simulator $\mathcal{S}_{\mathsf{Bob}}$, please refer to the diagram shown in Fig. 4 (b).

First, we prove that the simulator $\mathcal{S}_{\mathsf{Bob}}$ constructed above is valid:

- The view of $\mathcal{A}_{\mathsf{Bob}}$ in the real protocol consists of a set $\psi(\mathsf{S})$, a MACed message $(\mathsf{Done}, \mathsf{Mac}_{k_{\mathsf{MAC}}}(\mathsf{Done}))$ and a pseudorandom permutation $\mathsf{PRP}_k(x)$. In the simulation of the simulator $\mathcal{S}_{\mathsf{Bob}}$, both $\psi(\mathsf{S})$ and $(\mathsf{Done}, \mathsf{Mac}_{k_{\mathsf{MAC}}}(\mathsf{Done}))$ are computed with random keys, which is the same as in the real protocol execution. $\mathsf{PRP}_k(x)$ is pseudo-random, and is computed according to the output result in the ideal world, which is indistinguishable with that in the real protocol and results in the identical output distribution. Consequently, the real adversary $\mathcal{A}_{\mathsf{Bob}}$ cannot distinguish whether it is interacting with honest Alice/$\mathsf{SC}_{\mathsf{Alice}}$ or with the simulator $\mathcal{S}_{\mathsf{Bob}}$;

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2837665, IEEE Access

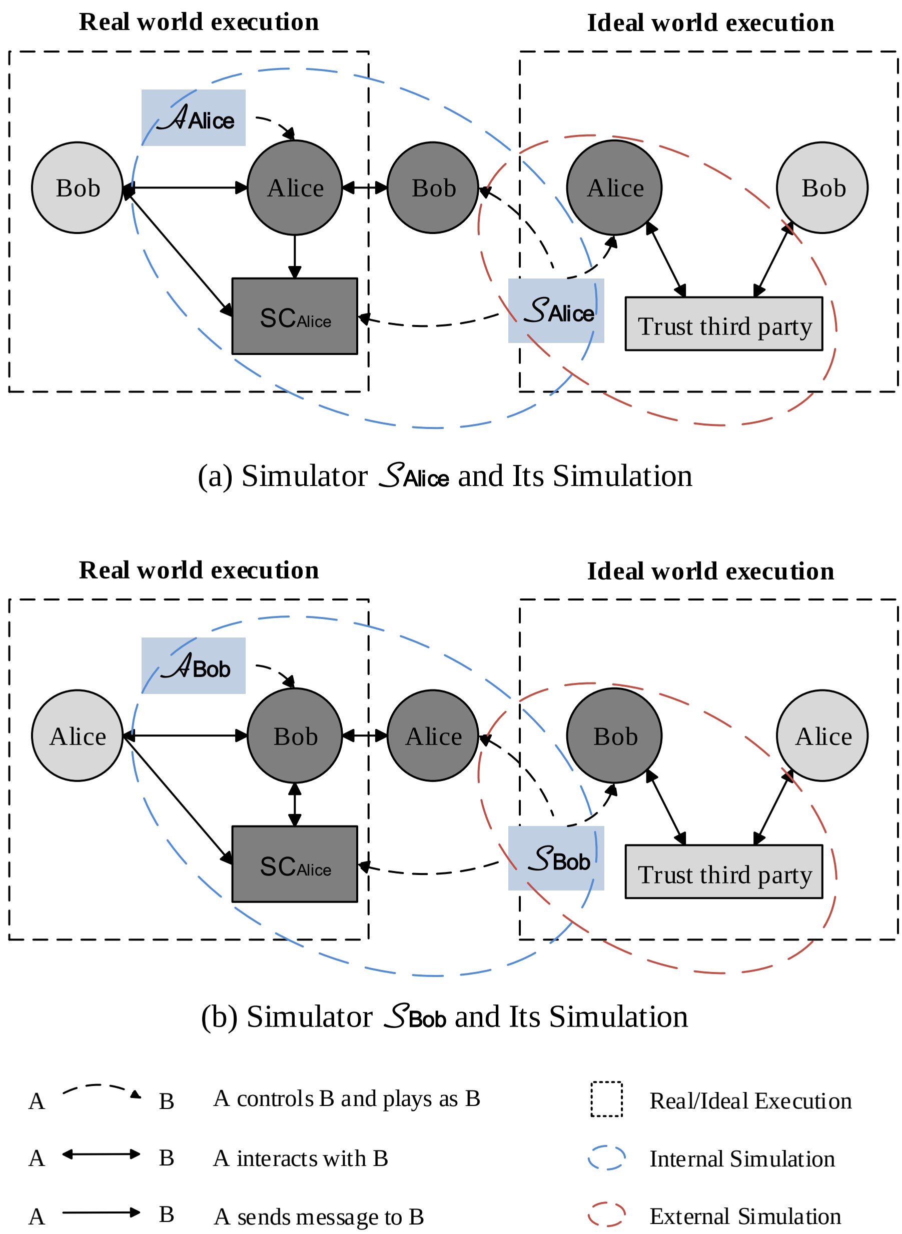

FIGURE 4. Diagram of Simulator $\mathcal{S}_{\text{Alice}}$ and $\mathcal{S}_{\text{Bob}}$, and Their Simulations

- As $\mathcal{S}_{\text{Bob}}$ can read $\mathcal{A}_{\text{Bob}}$'s output tape, it can directly obtain $\mathcal{A}_{\text{Bob}}$'s real input y, which is supposed to be sent to SC_{Alice} by $\mathcal{A}_{\text{Bob}}$.

Secondly, we prove that the joint output distribution of honest Alice and the adversary $\mathcal{A}_{\text{Bob}}$ is indistinguishable with the joint output distribution of honest Alice and the simulator $\mathcal{S}_{\text{Bob}}$:

- As the view of $\mathcal{A}_{\text{Bob}}$ in the real protocol is indistinguishable with that in the simulation with $\mathcal{S}_{\text{Bob}}$, the output distribution of the adversary $\mathcal{A}_{\text{Bob}}$ is indistinguishable with that of the simulator $\mathcal{S}_{\text{Bob}}$.
- As Alice has no output in the objective functionality, there is no need to analyze the output distribution of honest Alice.

Protocol $\mathcal{P}_{Full}$ is secure in the case that Bob is corrupted. This concludes our proof.

□

2) Efficiency Analysis

We analyze the complexity of Protocol $\mathcal{P}_{Full}$. We first analyze the computation complexity. The proposed protocol contains only symmetric cryptographic operations, including pseudorandom permutation (denoted as PRP), message authentication code (denoted as MAC), and string matching (denoted as SM). The concrete complexity of Alice, Bob, and $\mathsf{SC}_{\mathsf{Alice}}$ are summarized in Table 1.

In terms of round efficiency, Protocol $\mathcal{P}_{Full}$ has a constant number of rounds. As the interaction between Bob and $\mathsf{SC}_{\mathsf{Alice}}$ is carried out locally, we only consider the interaction between Alice and Bob, which requires only one round to be specific.

TABLE 1. Efficiency Analysis of Protocol $\mathcal{P}_{Full}$

Computation Complexity				Round Complexity
Paricipants	PRP	MAC	SM	
Alice	1	1	0	1
Bob	0	0	y+1	
$\mathsf{SC}_{\mathsf{Alice}}$	y+1	1	0	

B. ONE-SIDED SIMULATABLE PROTOCOL $\mathcal{P}_{ONE-SIDED}$

Protocol $\mathcal{P}_{Full}$ requires the smart card to perform heavy computation tasks, that is , $y+1$ pseudorandom permutations, although this protocol achieves full simulation security in malicious model. Smart cards are generally computation-bounded devices, thereby reducing the efficiency of protocol execution. We consider a relaxed level of security, that is, one-sided simulation security, to design an efficient protocol. In this security model, simulation is only required when Bob is corrupted. We only guarantee the privacy of Bob's input when Alice is corrupted.

Protocol 2. $\mathcal{P}_{One-sided}$***: Protocol for Computing*** $f(x, y) = x \leq y$***? with One-Sided Simulation Security***

Inputs: *Alice inputs a private set* $x \in U$ *and Bob inputs a private value* $y \in U$.

Output: *Alice outputs nothing; Bob outputs 1 if* $x \leq y$ *and 0 otherwise.*

Initialization:

Step 1. *Alice chooses two keys* $k, k_{\mathsf{MAC}} \leftarrow \{0,1\}^n$ *in random, where* k *is the key of a pseudorandom permutation* PRP *and* k_{MAC} *is the key of a message authentication code* MAC*. Both* PRP *and* MAC *are embedded in the smart card* $\mathsf{SC}_{\mathsf{Alice}}$*. After obtaining this smart card, Alice imports* k, k_{MAC} *into it and sets the parameter* Count *as 1, which indicates that the total number of queries supported by this smart card is 1.*

Step 2. *Alice sends* $\mathsf{SC}_{\mathsf{Alice}}$ *to Bob via offline channel.*

Online Interaction:

Step 1. *Upon receiving the smart card* $\mathsf{SC}_{\mathsf{Alice}}$*, Bob acts as follows:*

a) *Sends his input* y *to* $\mathsf{SC}_{\mathsf{Alice}}$ *and obtains* $\mathsf{PRP}_k(y)$*;*
b) *Issues a* $\mathsf{Complete}$ *command to* $\mathsf{SC}_{\mathsf{Alice}}$ *and receives back a confirmation message* Done *and its MAC tag, denoted as* $(\mathsf{Done}, \mathsf{Mac}_{k_{\mathsf{MAC}}}(\mathsf{Done}))$*, where* Mac *is the tag generation algorithm;*
c) *Sends the MACed confirmation to Alice.*

Step 2. *Upon receiving the confirmation, Alice acts as follows:*

a) *Verifies the validity of the MACed confirmation with the verification algorithm. If valid, computes the set* $X_{\mathsf{PRP}} = \{\mathsf{PRP}_k(x)\}_{x \in X}$ *with* k*, where* $X = \{0, 1, \cdots, x-1\}$*;*
b) *Chooses a set* $R = \{r_i | r_i \leftarrow \{0,1\}^n, r_i \notin U, i = 1, 2, \cdots, |\bar{X}|\}$ *at random, where* $\bar{X}$ *is the complementary set of* X*, i.e.,* $\bar{X} = U - X$*, and computes the set* $R_{\mathsf{PRP}} = \{\mathsf{PRP}_k(x)\}_{x \in R}$ *with* k*;*
c) *Computes* $\mathsf{S} = X_{\mathsf{PRP}} \cup R_{\mathsf{PRP}}$*, and generates a random permutation of* S*, denoted as* $\psi(\mathsf{S})$*;*
d) *Sends* $\psi(\mathsf{S})$ *to Bob.*

Step 3. *Bob determines whether* $x \leq y$ *as follow. If* $\mathsf{PRP}_k(y) \in \psi(\mathsf{S})$*, then* $x > y$*, Bob outputs 0; otherwise,* $x \leq y$*, Bob outputs 1.*

Please refer to Fig. 5 for a clear diagram of the proposed protocol.

1) Security Analysis

We now analyze the security of Protocol $\mathcal{P}_{One-sided}$. Formally, we have the following theorem.

Theorem 2. *If* PRP *is a pseudorandom permutation and* MAC *is message authentication code, then Protocol* $\mathcal{P}_{One-sided}$ *securely computes function* $f(x, y) = x \leq y$*? under Definition 5.*

Proof. Let f be the objective function and π be the two-party protocol presented above. Then π securely computes f in the presence of malicious adversaries under ideal/real simulation paradigm with one-sided simulation security if the following satisfies:

1. For every non-uniform probabilistic polynomial-time adversary $\mathcal{A}$ corrupting Bob in the real world, there exists a non-uniform polynomial-time adversary $\mathcal{S}$ in the ideal world, such that
$$\{\text{IDEAL}_{f,\mathcal{S}(z),\mathsf{Bob}}(x, y, n)\} \stackrel{c}{\equiv} \{\text{REAL}_{\pi,\mathcal{A}(z),\mathsf{Bob}}(x, y, n)\},$$
where $x, y \in U$, $z \in \{0,1\}^*$ and $n \in \mathbb{N}$.
2. For every non-uniform probabilistic polynomial-time adversary $\mathcal{A}$ corrupting Alice, it satisfies that
$$\{\text{VIEW}^{\mathcal{A}}_{\pi,\mathcal{A}(z),\mathsf{Alice}}(x, y, n)\} \stackrel{c}{\equiv} \{\text{VIEW}^{\mathcal{A}}_{\pi,\mathcal{A}(z),\mathsf{Alice}}(x, y', n)\},$$
where $\text{VIEW}^{\mathcal{A}}_{\pi,\mathcal{A}(z),\mathsf{Alice}}(x, y, n)$ denotes the view of the adversary after a real execution of π, $x, y, y' \in U$, $z \in \{0,1\}^*$ and $n \in \mathbb{N}$.

We formally prove Theorem 2 separately for the case Bob is corrupted and the case Alice is corrupted.

Bob is corrupted. Suppose that the adversary attacking Protocol $\mathcal{P}_{One-sided}$ corrupts and controls Bob. Denote the adversary as $\mathcal{A}_{\mathsf{Bob}}$. We construct a simulator $\mathcal{S}_{\mathsf{Bob}}$ in the ideal world. The simulator internally invokes $\mathcal{A}_{\mathsf{Bob}}$ and interacts with it as Alice and $\mathsf{SC}_{\mathsf{Alice}}$. Besides, $\mathcal{S}_{\mathsf{Bob}}$ externally interacts with the trust third party computing f as the corrupted Bob. The simulator we construct is described as follows:

- $\mathcal{S}_{\mathsf{Bob}}$ invokes $\mathcal{A}_{\mathsf{Bob}}$ with its initial input, interacts with $\mathcal{A}_{\mathsf{Bob}}$ as Alice and $\mathsf{SC}_{\mathsf{Alice}}$. If any cheating of $\mathcal{A}_{\mathsf{Bob}}$ is detected, $\mathcal{S}_{\mathsf{Bob}}$ sends $\perp$ to the trusted third party as the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2837665, IEEE Access

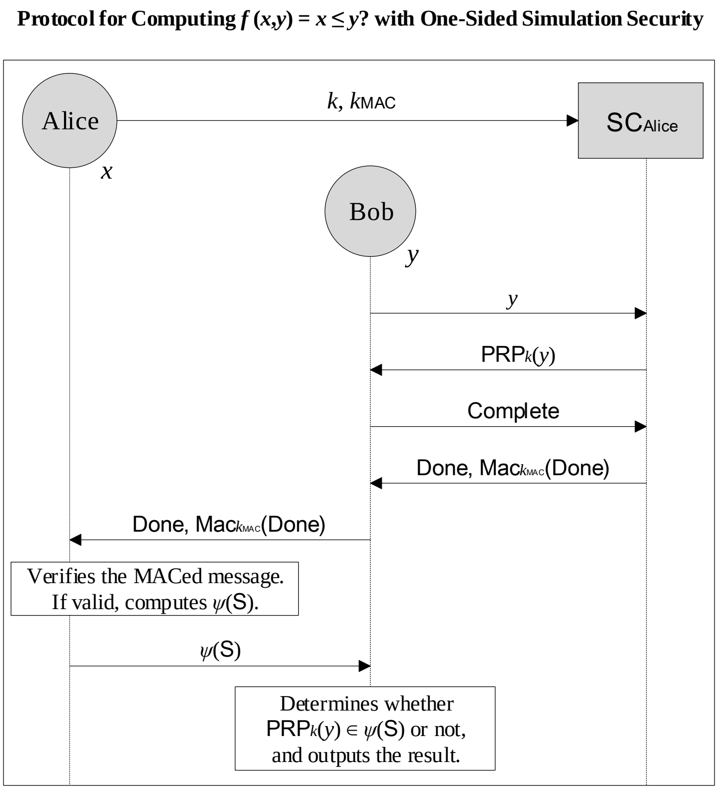

FIGURE 5. Diagram of $\mathcal{P}_{One-sided}$

simulation of Alice aborting the protocol, and outputs whatever $\mathcal{A}_{\mathsf{Bob}}$ outputs. Otherwise, $\mathcal{S}_{\mathsf{Bob}}$ continues.

- $\mathcal{S}_{\mathsf{Bob}}$ plays as $\mathsf{SC}_{\mathsf{Alice}}$. It obtains $\mathcal{A}_{\mathsf{Bob}}$'s input y from $\mathcal{A}_{\mathsf{Bob}}$'s output tape. This value is supposed to be sent to $\mathsf{SC}_{\mathsf{Alice}}$ by $\mathcal{A}_{\mathsf{Bob}}$.
- $\mathcal{S}_{\mathsf{Bob}}$ plays as $\mathsf{SC}_{\mathsf{Alice}}$. It randomly chooses a pseudo-random permutation key $k \leftarrow \{0,1\}^n$ and sends $\mathcal{A}_{\mathsf{Bob}}$ a value $\mathsf{PRP}_k(y)$ computed with the key k.
- $\mathcal{S}_{\mathsf{Bob}}$ plays as corrupted Bob. It sends the extracted input y of $\mathcal{A}_{\mathsf{Bob}}$ to the trusted third party externally, and receives back the computation result 1 or 0, indicating whether $x \leq y$ or not, where x is the input of honest Alice in the real world.
- $\mathcal{S}_{\mathsf{Bob}}$ plays as $\mathsf{SC}_{\mathsf{Alice}}$. After receiving a $\mathsf{Complete}$ command from $\mathcal{A}_{\mathsf{Bob}}$, $\mathcal{S}_{\mathsf{Bob}}$ randomly chooses a MAC key $k_{\mathsf{MAC}} \leftarrow \{0,1\}^n$ and sends back a confirmation message MACed with k_{MAC}.
- $\mathcal{S}_{\mathsf{Bob}}$ plays as Alice. After receiving the MACed confirmation from $\mathcal{A}_{\mathsf{Bob}}$, $\mathcal{S}_{\mathsf{Bob}}$ sends back to $\mathcal{A}_{\mathsf{Bob}}$ a set $\psi(\mathsf{S})$, which is constructed as follows:
  - If the result obtained from the trusted third party is 1, $\mathcal{S}_{\mathsf{Bob}}$ takes $\mathsf{PRP}_k(y)$ as one element of $\psi(\mathsf{S})$, and sets the other $|U|-1$ elements as randomly chosen values from the domain of PRP_k, where $|U|$ denotes the size of U;
  - Otherwise, $\mathcal{S}_{\mathsf{Bob}}$ sets all elements in $\psi(\mathsf{S})$ as randomly chosen values from the domain of PRP_k under the condition that $\mathsf{PRP}_k(y)$ is not chosen.
- $\mathcal{S}_{\mathsf{Bob}}$ plays as Alice. It sends $\psi(\mathsf{S})$ to $\mathcal{A}_{\mathsf{Bob}}$, outputs whatever $\mathcal{A}_{\mathsf{Bob}}$ outputs and halts.

For a legible description of the constructed simulator $\mathcal{S}_{\mathsf{Bob}}$, please refer to the diagram shown in Fig. 6.

First, we prove that the simulator $\mathcal{S}_{\mathsf{Bob}}$ constructed above is valid:

- The view of $\mathcal{A}_{\mathsf{Bob}}$ in the real protocol consists of a pseudorandom permutation $\mathsf{PRP}_k(y)$, a MACed message $(\mathsf{Done}, \mathsf{Mac}_{k_{\mathsf{MAC}}}(\mathsf{Done}))$ and a set $\psi(\mathsf{S})$. In the simulation of the simulator $\mathcal{S}_{\mathsf{Bob}}$, both $\mathsf{PRP}_k(y)$ and $(\mathsf{Done}, \mathsf{Mac}_{k_{\mathsf{MAC}}}(\mathsf{Done}))$ are computed with random keys, which is the same as in the real protocol execution. $\psi(\mathsf{S})$ is computed according to the output result in the ideal world, which is indistinguishable with that in the real protocol and results in the identical output distribution. Consequently, the real adversary $\mathcal{A}_{\mathsf{Bob}}$ cannot distinguish whether it is interacting with honest Alice/$\mathsf{SC}_{\mathsf{Alice}}$ or with the simulator $\mathcal{S}_{\mathsf{Bob}}$;
- As $\mathcal{S}_{\mathsf{Bob}}$ can read $\mathcal{A}_{\mathsf{Bob}}$'s output tape, it can directly obtain $\mathcal{A}_{\mathsf{Bob}}$'s real input y, which is supposed to be sent to $\mathsf{SC}_{\mathsf{Alice}}$ by $\mathcal{A}_{\mathsf{Bob}}$.

Secondly, we prove that the joint output distribution of honest Alice and the adversary $\mathcal{A}_{\mathsf{Bob}}$ is indistinguishable with

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2837665, IEEE Access

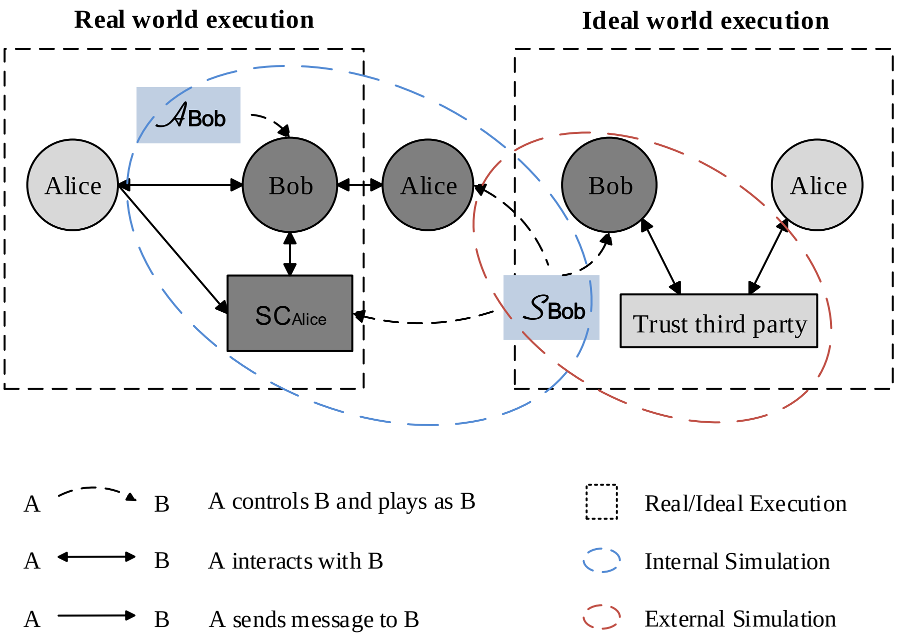

FIGURE 6. Diagram of Simulator $\mathcal{S}_{\mathsf{Bob}}$ and Its Simulation

the joint output distribution of honest Alice and the simulator $\mathcal{S}_{\mathsf{Bob}}$:

- As the view of $\mathcal{A}_{\mathsf{Bob}}$ in the real protocol is indistinguishable with that in the simulation with $\mathcal{S}_{\mathsf{Bob}}$, the output distribution of the adversary $\mathcal{A}_{\mathsf{Bob}}$ is indistinguishable with that of the simulator $\mathcal{S}_{\mathsf{Bob}}$.
- As Alice has no output in the objective functionality, there is no need to analyze the output distribution of honest Alice.

Protocol $\mathcal{P}_{One-sided}$ is secure in the case that Bob is corrupted.

Alice is corrupted. The protocol transcript received by Alice in Protocol $\mathcal{P}_{One-sided}$ contains only a MACed confirmation message sent by Bob. This message consists of a confirmation message Done and its MAC tag $\mathsf{Mac}_{k_{\mathsf{MAC}}}(\mathsf{Done})$, which are both independent with Bob's input. Specifically, the view of Alice can be written as follows:

$$\mathrm{VIEW}^{\mathcal{A}}_{\pi,\mathcal{A}(z),\mathsf{Alice}}(x,y,n) = \{x, r, (\mathsf{Done}, \mathsf{Mac}_{k_{\mathsf{MAC}}}(\mathsf{Done}))\},$$

where x is Alice's input, r is the randomness Alice used in protocol execution and $(\mathsf{Done}, \mathsf{Mac}_{k_{\mathsf{MAC}}}(\mathsf{Done}))$ is the message received from Bob. Therefore, the following equality holds no matter what Bob's input is:

$$\{\mathrm{VIEW}^{\mathcal{A}}_{\pi,\mathcal{A}(z),\mathsf{Alice}}(x,y,n)\} \stackrel{c}{\equiv} \{\mathrm{VIEW}^{\mathcal{A}}_{\pi,\mathcal{A}(z),\mathsf{Alice}}(x,y',n)\},$$

where $x, y, y' \in U$, $z \in \{0,1\}^*$ and $n \in \mathbb{N}$.

Protocol $\mathcal{P}_{One-sided}$ is secure in the case that Alice is corrupted.

This concludes our proof. □

2) Efficiency Analysis

We analyze the complexity of Protocol $\mathcal{P}_{One-sided}$. We first analyze the computation complexity. The proposed protocol contains only symmetric cryptographic operations, including pseudorandom permutation (denoted as PRP), message authentication code (denoted as MAC), and string matching (denoted as SM). The concrete complexity of Alice, Bob, and $\mathsf{SC}_{\mathsf{Alice}}$ are summarized in Table 2.

In terms of round efficiency, Protocol $\mathcal{P}_{One-sided}$ also requires only one round in the online phase.

TABLE 2. Efficiency Analysis of Protocol $\mathcal{P}_{One-sided}$

Computation Complexity				Round Complexity
Paricipants	PRP	MAC	SM	
Alice	$\|U\|$	1	0	1
Bob	0	0	$\|U\|$	
$\mathsf{SC}_{\mathsf{Alice}}$	1	1	0	

IV. EXPERIMENTAL RESULTS

The aforementioned efficiency analysis for each protocol indicates the concrete efficiency of our protocols in theory. However, a smart card is a resource-bounded device. Its efficiency significantly influences the proposed protocols. Thus, experiments on the performance of smart cards in practice should be conducted. We ran $\mathsf{SC}_{\mathsf{Alice}}$ on a standard smart card produced by FEITIAN Technologies Co., Ltd. and tested the performance of our protocols. In the experiments, AES 128 is taken as an instantiation of pseudorandom permutations. Tables 3 and 4 summarize the experimental results of Protocols $\mathcal{P}_{Full}$ and $\mathcal{P}_{One-sided}$, respectively.

The experiments show that both protocols exhibit favorable execution performance in terms of the run time of smart cards. Protocol $\mathcal{P}_{One-sided}$ performs better than Protocol $\mathcal{P}_{Full}$, and the run time of smart cards in Protocol $\mathcal{P}_{One-sided}$ is independent of the size of the universal set. From this point, Protocol $\mathcal{P}_{One-sided}$ is sufficiently scalable and efficient to be used in practice.

The introduction of smart cards enhances the security of the protocols but leads to inefficiency simultaneously. The reduction of the computation task run on smart cards with a sacrifice of security level is a practical choice to improve efficiency. In most circumstances, a trade-off between security and efficiency should be made.

V. COMPARISON WITH RELATED WORK

In this section, we compare our protocols with state-of-the-art solutions to millionaires' problem in terms of computation complexity, round complexity and security level. The comparison results are listed in Table 5.

This table displays that our protocols exceed other methods in terms of efficiency and security. Specifically, our solutions achieve simulation-based security against malicious adversaries and require symmetric cryptography only. Our methods obtain a high security level with low computation costs because symmetric cryptographic operation is more efficient than asymmetric cryptography.

VI. CONCLUSIONS AND FUTURE WORK

In this work, we reviewed a classical problem in secure two-party computation, namely, Yao's millionaires' problem. We used a standard smart card as our building block to solve this problem in malicious model with strong security and high efficiency. Specifically, we proposed two novel, efficient solutions with simulation-based security. The proposed protocols were built upon only symmetric cryptography, which was more efficient than asymmetric cryptography. The experimental results indicated that our solutions securely solved Yao's millionaires' problem with high efficiency and scalability. Comparison with related work showed that our protocols are better than other state-of-the-art methods in terms of efficiency and security.

In the future, we plan to extend our study on general problems in secure two-party computation, such as efficient constructions of generic protocols based on garbled circuit or homomorphic encryption.

TABLE 3. Experimental Result of Protocol $\mathcal{P}_{Full}$

Size of y	Run time of $\mathrm{SC}_{\mathrm{Alice}}$
10	9319 µs
50	45195 µs
100	92931 µs
500	449873 µs
1000	916556 µs

TABLE 4. Experimental Result of Protocol $\mathcal{P}_{One-sided}$

Size of U	Run time of $\mathrm{SC}_{\mathrm{Alice}}$
10	1214 µs
50	1485 µs
100	1191 µs
500	1268 µs
1000	1397 µs
2000	1303 µs
5000	1275 µs
10000	1586 µs

REFERENCES

[1] A. C. Yao, "Protocols for secure computations," in Proceedings of the 23th Annual Symposium on Foundations of Computer Science. IEEE, 1982, pp. 160–164.
[2] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game," in Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing. ACM, 1987, pp. 218–229.
[3] R. J. Bayardo and R. Agrawal, "Data privacy through optimal k-anonymization," in Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on. IEEE, 2005, pp. 217–228.
[4] Z. Huang, S. Liu, X. Mao, K. Chen, and J. Li, "Insight of the protection for data security under selective opening attacks," Information Sciences, vol. 412, pp. 223–241, 2017.
[5] J. Xu, L. Wei, Y. Zhang, A. Wang, F. Zhou, and C.-z. Gao, "Dynamic fully homomorphic encryption-based merkle tree for lightweight streaming authenticated data structures," Journal of Network and Computer Applications, vol. 107, pp. 113–124, 2018.
[6] X. Zhang, Y.-A. Tan, C. Liang, Y. Li, and J. Li, "A covert channel over volte via adjusting silence periods," IEEE Access, vol. 6, pp. 9292–9302, 2018.
[7] Q. Lin, H. Yan, Z. Huang, W. Chen, J. Shen, and Y. Tang, "An id-based linearly homomorphic signature scheme and its application in blockchain," IEEE Access, 2018.
[8] B. R. Kandukuri, A. Rakshit et al., "Cloud security issues," in Services Computing, 2009. SCC'09. IEEE International Conference on. IEEE, 2009, pp. 517–520.
[9] Z. Cai, H. Yan, P. Li, Z.-a. Huang, and C. Gao, "Towards secure and flexible ehr sharing in mobile health cloud under static assumptions," Cluster Computing, vol. 20, no. 3, pp. 2415–2422, 2017.
[10] J. Shen, Z. Gui, S. Ji, J. Shen, H. Tan, and Y. Tang, "Cloud-aided lightweight certificateless authentication protocol with anonymity for wireless body area networks," Journal of Network and Computer Applications, 2018.
[11] M. K. Franklin and M. K. Reiter, "The design and implementation of a secure auction service," IEEE Transactions on Software Engineering, vol. 22, no. 5, pp. 302–312, 1996.
[12] W. Du and M. J. Atallah, "Privacy-preserving cooperative statistical analysis," in Computer Security Applications Conference, 2001. ACSAC 2001. Proceedings 17th Annual. IEEE, 2001, pp. 102–110.
[13] C. Wang, S. S. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for secure cloud storage," IEEE transactions on computers, vol. 62, no. 2, pp. 362–375, 2013.
[14] X. Chen, J. Li, J. Weng, J. Ma, and W. Lou, "Verifiable computation over large database with incremental updates," IEEE transactions on Computers, vol. 65, no. 10, pp. 3184–3195, 2016.
[15] B. Li, Y. Huang, Z. Liu, J. Li, Z. Tian, and S.-M. Yiu, "Hybridoram: practical oblivious cloud storage with constant bandwidth," Information Sciences, 2018.
[16] J. Li, Y. Zhang, X. Chen, and Y. Xiang, "Secure attribute-based data sharing for resource-limited users in cloud computing," Computers & Security, vol. 72, pp. 1–12, 2018.
[17] P. Li, J. Li, Z. Huang, C.-Z. Gao, W.-B. Chen, and K. Chen, "Privacy-preserving outsourced classification in cloud computing," Cluster Computing, pp. 1–10, 2017.
[18] Y. Lindell and B. Pinkas, "Privacy preserving data mining," in Annual International Cryptology Conference. Springer, 2000, pp. 36–54.
[19] C.-z. Gao, Q. Cheng, P. He, W. Susilo, and J. Li, "Privacy-preserving naive bayes classifiers secure against the substitution-then-comparison attack," Information Sciences, vol. 444, pp. 72–88, 2018.
[20] T. Li, J. Li, Z. Liu, P. Li, and C. Jia, "Differentially private naive bayes learning over multiple data sources," Information Sciences, vol. 444, pp. 89–104, 2018.

TABLE 5. Comparison with related work

Related Schemes	Computation Complexity		Rounds	Security Level
	Symmetric Operations	Asymmetric Operations		
Reference [42]	$O(\|U\|)$	—	2	Semi-honest Model
Reference [43]	$O(\|U\|)$	—	1	Semi-honest Model
Reference [40]	—	$O(s\log N)$	1	Semi-honest Model (IND-CCA2 and NM-CCA2 Security)
Reference [41]	—	$O(s\log N)$	1	Semi-honest Model
Protocol $\mathcal{P}_{Full}$	$O(y)$	—	1	Malicious Model (Full Simulation Security)
Protocol $\mathcal{P}_{One-sided}$	$O(\|U\|)$	—	1	Malicious Model (One-sided Simulation Security)

Note: $|U|$ denotes the size of the universal set, s denotes the dimension of the encoding vector, N denotes the modulus of the public-key encryption scheme, y denotes the input value of Bob, IND-CCA2 denotes indistinguishability under adaptive chosen ciphertext attack and NM-CCA2 denotes non-malleability under adaptive chosen ciphertext attack.

[21] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," in Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on. IEEE, 1995, pp. 41–50.
[22] S. Yekhanin, "Private information retrieval," Communications of the ACM, vol. 53, no. 4, pp. 68–73, 2010.
[23] M. J. Freedman, K. Nissim, and B. Pinkas, "Efficient private matching and set intersection," in International conference on the theory and applications of cryptographic techniques. Springer, 2004, pp. 1–19.
[24] E. De Cristofaro and G. Tsudik, "Practical private set intersection protocols with linear complexity," in International Conference on Financial Cryptography and Data Security. Springer, 2010, pp. 143–159.
[25] P. Rindal and M. Rosulek, "Improved private set intersection against malicious adversaries," in Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, 2017, pp. 235–259.
[26] M. Naor and B. Pinkas, "Efficient oblivious transfer protocols," in Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics, 2001, pp. 448–457.
[27] C. Peikert, V. Vaikuntanathan, and B. Waters, "A framework for efficient and composable oblivious transfer," in Advances in Cryptology-CRYPTO 2008. Springer, 2008, pp. 554–571.
[28] C. Zhao, H. Jiang, Q. Xu, X. Wei, and H. Wang, "Several oblivious transfer variants in cut-and-choose scenario," International Journal of Information Security and Privacy (IJISP), vol. 9, no. 2, pp. 1–12, 2015.
[29] C. Zhao, H. Jiang, X. Wei, Q. Xu, and M. Zhao, "Cut-and-choose bilateral oblivious transfer and its application," in Trustcom/BigDataSE/ISPA, 2015 IEEE, vol. 1. IEEE, 2015, pp. 384–391.
[30] X. Wei, H. Jiang, C. Zhao, M. Zhao, and Q. Xu, "Fast cut-and-choose bilateral oblivious transfer for malicious adversaries," in Trustcom/BigDataSE/IâĂŃ SPA, 2016 IEEE. IEEE, 2016, pp. 418–425.
[31] E. Stefanov, M. Van Dijk, E. Shi, C. Fletcher, L. Ren, X. Yu, and S. Devadas, "Path oram: an extremely simple oblivious ram protocol," in Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. ACM, 2013, pp. 299–310.
[32] S. Lu and R. Ostrovsky, "Distributed oblivious ram for secure two-party computation," in Theory of Cryptography. Springer, 2013, pp. 377–396.
[33] Z. Liu, Y. Huang, J. Li, X. Cheng, and C. Shen, "Divoram: Towards a practical oblivious ram with variable block size," Information Sciences, vol. 447, pp. 1–11, 2018.
[34] C. Cachin, "Efficient private bidding and auctions with an oblivious third party," in Proceedings of the 6th ACM conference on Computer and communications security. ACM, 1999, pp. 120–127.
[35] M. Fischlin, "A cost-effective pay-per-multiplication comparison method for millionaires," Topics in CryptologyâĂŤCT-RSA 2001, pp. 457–471, 2001.
[36] I. Ioannidis and A. Grama, "An efficient protocol for yao's millionaires' problem," in System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on. IEEE, 2003, pp. 6–pp.
[37] I. F. Blake and V. Kolesnikov, "Strong conditional oblivious transfer and computing on intervals." in ASIACRYPT, vol. 4, no. 3329. Springer, 2004, pp. 515–529.
[38] H.-Y. Lin and W.-G. Tzeng, "An efficient solution to the millionaires' problem based on homomorphic encryption," in ACNS, vol. 5. Springer, 2005, pp. 456–466.
[39] I. F. Blake and V. Kolesnikov, "Conditional encrypted mapping and comparing encrypted numbers," in Financial Cryptography, vol. 4107. Springer, 2006, pp. 206–220.
[40] M. Hezaveh and C. Adams, "An efficient solution to the socialist millionaires' problem," in Electrical and Computer Engineering (CCECE), 2017 IEEE 30th Canadian Conference on. IEEE, 2017, pp. 1–4.
[41] X. Liu, S. Li, X. Chen, G. Xu, X. Zhang, and Y. Zhou, "Efficient solutions to two-party and multiparty millionaires' problem," Security and Communication Networks, vol. 2017, 2017.
[42] S. Li, D. Wang, Y. Dai, and P. Luo, "Symmetric cryptographic solution to yao's millionaires' problem and an evaluation of secure multiparty computations," Information Sciences, vol. 178, no. 1, pp. 244–255, 2008.
[43] S. Li, D. Wang, and Y. Dai, "Symmetric cryptographic protocols for extended millionaires' problem," Science in China Series F: Information Sciences, vol. 52, no. 6, pp. 974–982, 2009.
[44] R. Li, C. Wu, and Y. Zhang, "A fair and efficient protocol for the millionaires' problem," Chin. J. Electron, vol. 18, no. 2, pp. 249–254, 2009.
[45] A. Maitra, G. Paul, and A. K. Pal, "Millionaires' problem with rational players: a unified approach in classical and quantum paradigms," arXiv preprint arXiv:1504.01974, 2015.
[46] I. Damgård, M. Fitzi, E. Kiltz, J. B. Nielsen, and T. Toft, "Unconditionally secure constant-rounds multi-party computation for equality, comparison, bits and exponentiation," in TCC, vol. 3876. Springer, 2006, pp. 285–304.
[47] T. Nishide and K. Ohta, "Multiparty computation for interval, equality, and comparison without bit-decomposition protocol," in Public Key Cryptography, vol. 4450. Springer, 2007, pp. 343–360.
[48] D. Grigoriev and V. Shpilrain, "Yao's millionaires'problem and decoy-based public key encryption by classical physics," International Journal of Foundations of Computer Science, vol. 25, no. 04, pp. 409–417, 2014.
[49] D. Grigoriev, L. B. Kish, and V. Shpilrain, "Yao's millionaires' problem and public-key encryption without computational assumptions," IACR Cryptology ePrint Archive, vol. 2016, p. 189, 2016, https://eprint.iacr.org/2016/189.
[50] Y. Lindell and B. Pinkas, "An efficient protocol for secure two-party computation in the presence of malicious adversaries," in Advances in Cryptology-EUROCRYPT 2007. Springer, 2007, pp. 52–78.
[51] ——, "Secure two-party computation via cut-and-choose oblivious transfer," in Theory of Cryptography. Springer, 2011, pp. 329–346.
[52] Y. Huang, D. Evans, J. Katz, and L. Malka, "Faster secure two-party computation using garbled circuits." in USENIX Security Symposium, vol. 201, no. 1, 2011, pp. 331–335.
[53] Y. Huang, J. Katz, and D. Evans, "Efficient secure two-party computation using symmetric cut-and-choose," in Advances in Cryptology-CRYPTO 2013. Springer, 2013, pp. 18–35.
[54] Y. Lindell, "Fast cut-and-choose based protocols for malicious and covert adversaries," in Advances in Cryptology-CRYPTO 2013. Springer, 2013, pp. 1–17.
[55] C. Zhao, H. Jiang, Q. Xu, Y. Wang, X. Wei, and S. Cui, "Fast two-output secure computation with optimal error probability," Chinese Journal of Electronics, vol. 26, no. 5, pp. 933–941, 2017.
[56] J. Katz, S. Ranellucci, and X. Wang, "Authenticated garbling and communication-efficient, constant-round, secure two-party computation," IACR Cryptology ePrint Archive, vol. 2017, p. 030, 2017, https://eprint.iacr.org/2017/030.
[57] X. Wang, A. J. Malozemoff, and J. Katz, "Faster secure two-party computation in the single-execution setting," in Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, 2017, pp. 399–424.
[58] J. Katz and Y. Lindell, Introduction to modern cryptography. CRC press, 2014.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2837665, IEEE Access

[59] O. Goldreich, Foundations of cryptography: volume 2, basic applications. Cambridge university press, 2004.

[60] C. Hazay and Y. Lindell, "Constructions of truly practical secure protocols using standardsmartcards," in Proceedings of the 15th ACM conference on Computer and communications security. ACM, 2008, pp. 491–500.

[61] ——, "Efficient secure two-party protocols," Information Security and Cryptography. Springer, Heidelberg, 2010.

CHUAN ZHAO is currently a Lecturer at School of Information Science and Engineering, University of Jinan, China. He received his Ph.D. degree in Computer Science and Technology from Shandong University, China, in 2016. His research interests include cryptography, information security and privacy. He is currently focusing on practical secure multi-party computation, privacy-preserving technologies, and mobile security.

SHENGNAN ZHAO is currently a Ph.D. candidate at School of Computer Science and Technology, Shandong University, China. He received his bachelor degree at School of Control and Computer Engineering, North China Electric Power University, China, in 2016. His research interests include secure multiparty computation and search on encrypted data.

BO ZHANG is currently a Lecturer at School of Information Science and Engineering, University of Jinan, China. He received his Ph.D. degree in Computer Application Technology from Shandong University, China, in 2010. His research interests include public key cryptography and information security. He is currently focusing on certificateless signcryption, privacy-preserving data processing and aggregation protocol in healthcare wireless sensor networks.

ZHONGTIAN JIA is currently an associate professor at School of Information Science and Engineering, University of Jinan, China. He received his Ph.D. degree in cryptography from Beijing University of Posts and Telecommunications, China, in 2012. His research interests include cryptography, information security and privacy. He is currently focusing on biometric based identity authentication, DDoS and its' prevention.

ZHENXIANG CHEN is a Professor in the School of Information Science and Engineering, University of Jinan, China. He received his B.S and M.S. degrees from University of Jinan, Jinan, Shandong, China, in 2001 and 2004. He received his Ph.D. degree in the School of Computer science and technology, Shandong University, Jinan, China, in 2008. His research interests include network behavior analysis, mobile security and privacy, hybrid computational intelligence.

MAURO CONTI is an Associate Professor at the University of Padua, Italy. He obtained his Ph.D. from Sapienza University of Rome, Italy, in 2009. After his Ph.D., he was a Post-Doc Researcher at Vrije Universiteit Amsterdam, The Netherlands. In 2011 he joined as Assistant Professor the University of Padua, where he became Associate Professor in 2015. In 2017, he obtained the national habilitation as Full Professor for Computer Science and Computer Engineering. He has been Visiting Researcher at GMU (2008, 2016), UCLA (2010), UCI (2012, 2013, 2014, 2017), TU Darmstadt (2013), UF (2015), and FIU (2015, 2016). He has been awarded with a Marie Curie Fellowship (2012) by the European Commission, and with a Fellowship by the German DAAD (2013). His research is also funded by companies, including Cisco and Intel. His main research interest is in the area of security and privacy. In this area, he published more than 200 papers in topmost international peer-reviewed journals and conference. He is Associate Editor for several journals, including IEEE Communications Surveys & Tutorials, IEEE Transactions on Information Forensics and Security, and IEEE Transactions on Network and Service Management. He was Program Chair for TRUST 2015, ICISS 2016, WiSec 2017, and General Chair for SecureComm 2012 and ACM SACMAT 2013. He is Senior Member of the IEEE.